ENVELOPT: CONSTRAINED CONVEX COMPOSITE OPTIMIZATION

ALBERTO DE MARCHI* AND DOMINIQUE ORBAN†

Abstract. We introduce Envelopt, a globally convergent iterative framework for a broad class of structured optimization problems where a smooth objective is augmented by a nonsmooth convex regularizer composed with a smooth mapping, and the variables are subject to general smooth constraints. All smooth functions may be nonconvex. The method is akin to an augmented-Lagrangian method in which partial minimization with respect to a lifting variable results in smooth subproblems involving the Moreau envelope of the nonsmooth regularizer, and the original constraints are retained explicitly. Only the proximal operator of the regularizer is required. Subproblems may be solved with off-the-shelf smooth optimization solvers. We state global convergence properties, establish that feasible limit points are asymptotically stationary, and develop an infeasibility detection mechanism. We derive worst-case iteration complexity bounds when the penalty parameter is and is not bounded away from zero. The framework subsumes the classical augmented Lagrangian method and accommodates important extensions, including stabilized formulations for degenerate problems, exact penalty methods, and conic constraints. We provide a Julia implementation, `Envelopt.jl`, as part of the JuliaSmoothOptimizers ecosystem. Numerical experiments with low-rank matrix completion, semidefinite programming, complementarity-constrained optimization, and nonconvex regularizers demonstrate the effectiveness and versatility of Envelopt.

Key words. Convex composite constrained optimization, nonlinear programming, augmented Lagrangian method, proximal algorithms, Moreau envelope.

AMS subject classifications. 49M37, 49J53, 65K05, 65K10, 90C06

1. Introduction.

We consider the problem

$$\underset{x\in\mathbb{R}^n}{\text{minimize}} \quad f(x) + h(F(x)) \quad \text{subject to} \quad c(x) \in C, \tag{1.1}$$

where $f : \mathbb{R}^n \to \mathbb{R}$, $F : \mathbb{R}^n \to \mathbb{R}^p$, and $c : \mathbb{R}^n \to \mathbb{R}^m$ are C^1, $h : \mathbb{R}^p \to \mathbb{R} \cup \{+\infty\}$ is proper, lower semicontinuous (lsc), and convex, and $C \subseteq \mathbb{R}^m$ is nonempty, closed, and convex. Even though f, F and c may be nonconvex, we call (1.1) a *constrained convex composite problem* because of the crucial role played by the convexity of h [6].

Our approach consists in solving (1.1) by solving a sequence of smooth problems

$$\underset{x\in\mathbb{R}^n}{\text{minimize}} \quad f(x) + h_\mu(F(x) + \mu\hat{y}) \quad \text{subject to} \quad c(x) \in C, \tag{1.2}$$

where $h_\mu : \mathbb{R}^n \to \mathbb{R}$ is the C^1 Moreau envelope of h, $\mu > 0$, $\hat{y} \in \mathbb{R}^p$, and c is as in (1.1). Our main working assumptions are that an efficient solver for such smooth problems is available, and that the proximal operator of h is available. Some or all components of $F(x)$ could originate from constraints that were moved to the objective via an indicator. In our approach, such constraints will, in general, not be satisfied along the iterations. In that sense, constraints in c may be viewed as *hard* or *unrelaxable*. If any constraint is *soft*, or *relaxable*, the user has the option of encoding it into h by way of an indicator provided that the proximal operator remains available. Keeping a constraint explicit means our solver can, at the user's option, enforce it throughout,

*Institute of Applied Mathematics and Scientific Computing, Department of Aerospace Engineering, University of the Bundeswehr Munich, Germany. E-mail: alberto.demarchi@unibw.de. ORCID: 0000-0002-3545-6898.

†GERAD and Department of Mathematics and Industrial Engineering, Polytechnique Montréal, Canada. E-mail: dominique.orban@gerad.ca. ORCID: 0000-0002-8017-7687.

Funding: Research partially supported by NSERC Discovery Grant RGPIN–2020–06535.

but requires a subproblem solver for (1.2) capable of handling that constraint type; encoding it into h removes that requirement but forces the solver to discover feasibility through penalization.

Numerical experiments on low-rank matrix completion, semidefinite programming, and complementarity-constrained optimization confirm that Envelopt is competitive with or outperforms state-of-the-art solvers, including on degenerate problems where standard methods fail.

We provide a Julia implementation of Envelopt with examples named `Envelopt.jl` that is available from `github.com/JuliaSmoothOptimizers/Envelopt.jl` as part of the JSO ecosystem.

Contributions. Our main contributions are as follows:

1. we provide a globally-convergent general framework for constrained convex composite optimization based on the Moreau envelope that entirely relies on smooth subproblems—see Section 3;
2. our convergence analysis is based on asymptotic satisfaction of stationarity conditions and does not assume any constraint qualification–see Section 4;
3. we extract the features of a concrete algorithm that ensure convergence and discuss whether classic augmented-Lagrangian methods have those features–see Algorithm 4.1 and Assumption 4.1;
4. we provide a worst-case complexity analysis whether the penalty parameter remains bounded away from zero or not—see Theorem 4.5 and Theorem 4.6;
5. we review important extensions and applications of our framework, particularly NCL, exact penalty methods, and the proximal operator of composite sums—see Section 5;
6. we provide a complete Julia implementation of an instance of an algorithm that has the required features, named Envelopt—see Sections 6.1 and 6.2;

Related work. The augmented Lagrangian method has been a powerful optimization tool for many decades, starting with Hestenes [29], Powell [44] and Rockafellar [47] for smooth problems. Several efficient numerical schemes have been proposed, including those of Conn et al. [10] and Birgin and Martínez [3].

In the context of nonlinear programming, Andreani et al. [2] investigated augmented Lagrangian methods in which the lower-level constraints of subproblems need not be simple bounds, proving convergence to approximate KKT points under weak constraint qualifications. In a related spirit, Birgin et al. [5] proposed SECO, in which subproblems are equality-constrained smooth NLPs while bound constraints are handled by an outer augmented Lagrangian loop. Envelopt shares this philosophy of splitting constraints between those kept explicit in subproblems and those penalized in outer iterations.

Extensions to nonsmooth problems include those of Rockafellar [48] with h convex, and De Marchi and Themelis [18], De Marchi et al. [19], Hallak and Teboulle [28] with h nonconvex. The latter works hinge around unconstrained or proximal subproblems. The combination of nonconvex h with general hard constraints $c(x) \in C$ remains an open problem.

Dhingra et al. [21] focus on (1.1) with convex h, without explicit constraints, and assume that F is a bounded linear operator. However, they already observed that, by constraining the augmented Lagrangian to the manifold that corresponds to the explicit minimization over the auxiliary variable, the subproblem involves the Moreau envelope of the nonsmooth regularizer and is continuously differentiable. Dhingra et al. [22] augment their approach to use a generalized Hessian for the Moreau

envelope. Rockafellar [48] refers to the objective of (1.2) as the *generalized* augmented Lagrangian. These works lay the groundwork for the smooth-subproblem approach that Envelopt builds upon, while being limited to the unconstrained setting $C = \mathbb{R}^m$.

De Marchi [15] addresses (1.1) with h possibly nonconvex, but without hard constraints $c(x) \in C$, using the same partial-minimization approach over the lifting variable. Envelopt removes this restriction by retaining $c(x) \in C$ explicitly in every subproblem, which can be accounted for by any off-the-shelf constrained solver.

A different line of work avoids the augmented Lagrangian altogether. The *prox-linear* scheme of Lewis and Wright [32] addresses the minimization of functions that are composition of a prox-regular h with a smooth F. Their proximal linearized subproblem involves the composition of h with a linearization of F, which can be easily evaluated for well-structured problems only.

When $F = \mathbb{I}$, Chouzenoux et al. [8] introduce a proximal interior point algorithm able to handle problems with convex f, h, and inequality constraints. Leconte and Orban [31] develop an interior-point method for bound-constrained problems in which both f and h are allowed to be nonconvex, and De Marchi and Themelis [17] further consider nonconvex inequality constraints.

Dai et al. [13] investigate a numerical scheme for the special case where $F = \mathbb{I}$, h real- and nonnegative-valued, and only equality constraints are present. Curtis et al. [12] cover nonlinear inequality constraints, but their decomposition procedure accesses the regularizer by solving a constrained subproblem, which is tractable in practice only for special cases such as $h := \|\cdot\|_1$. Envelopt sidesteps these limitations by accessing h only through its proximal operator, inheriting the full generality of the augmented Lagrangian framework while retaining hard constraints explicitly.

Notation. We use lowercase Latin letters for vectors in $\mathbb{R}^n$ and lowercase Greek letters for scalars. Exceptions are that f, g and h are functions. Uppercase Latin letters are sets, except F and L, which are functions. For $x \in \mathbb{R}^n$, we let $\nabla F(x)$ and $\nabla c(x)$ be the *gradients* of F and c at x, i.e., the $n \times p$ and $n \times m$ matrices whose i-th column is $\nabla F_i(x)$ and $\nabla c_i(x)$, respectively, where $F(x) = (F_1(x), \dots, F_p(x))$ and $c(x) = (c_1(x), \dots, c_m(x))$. The identity function is denoted $\mathbb{I}$.

2. Background. The domain of h is $\operatorname{dom} h := \{x \in \mathbb{R}^n \mid h(x) < +\infty\}$. h is proper if $\operatorname{dom} h \neq \varnothing$ and lower semicontinuous if $h(x) \leq \liminf_{y \to x} h(y)$ for all $x \in \mathbb{R}^n$.

If $C \subseteq \mathbb{R}^n$, the indicator of C is $\chi(x \mid C) := 0$ if $x \in C$ and $+\infty$ otherwise. When C is convex, $\chi(\cdot \mid C)$ is convex as well, and, when C is closed, $\chi(\cdot \mid C)$ is lsc. If C is closed, the Euclidean distance function to C is $\operatorname{dist}(x \mid C) := \min\{\|x - y\|_2 \mid y \in C\}$.

For f_1, $f_2 : \mathbb{R}^n \to \mathbb{R} \cup \{\pm\infty\}$, the infimal convolution of f_1 and f_2 is the function denoted $f_1 \square f_2$ defined by

$$(f_1 \square f_2)(x) := \inf\{f_1(y) + f_2(x - y) dy \in \mathbb{R}^n\}.$$

A special case is $\operatorname{dist}(\cdot \mid C) = \chi(\cdot \mid C) \square \|\cdot\|_2$.

The Fenchel conjugate of h is $h^* : \mathbb{R}^n \to \mathbb{R} \cup \{+\infty\}$ defined by

$$h^*(y) := \sup\{y^\top x - h(x) dx \in \mathbb{R}^n\}.$$

The only function that coincides with its Fenchel conjugate is $\frac{1}{2}\|\cdot\|_2^2$.

The epigraph of h is the set $\{(x, \alpha) \mid h(x) \leq \alpha\} \subseteq \mathbb{R}^n \times (\mathbb{R} \cup \{\pm\infty\})$. If $\{h_k\}$ is a sequence of functions, we say that h_k epi-converges to h if the epigraphs of h_k converge to the epigraph of h in the Painlevé–Kuratowski sense.

For $u \in \mathbb{R}^n$ and $\mu > 0$, the *proximal operator* of h is

$$\operatorname*{prox}_{\mu h}(u) := \operatorname*{argmin}_{v \in \mathbb{R}^n} \left\{ \tfrac{1}{2}\|v-u\|_2^2 + \mu h(v) \right\}. \tag{2.1}$$

The *Moreau envelope* of h with parameter μ, introduced by Moreau [39] and denoted h_μ, is

$$h_\mu(u) := \inf_{v \in \mathbb{R}^n} \left\{ \tfrac{1}{2}\mu^{-1}\|v-u\|_2^2 + h(v) \right\} = \left(h \square (\tfrac{1}{2}\mu^{-1}\|\cdot\|_2^2) \right)(u), \tag{2.2}$$

and is the optimal value function in (2.1).

Because h is convex, (2.1) and (2.2) enjoy strong properties.

PROPOSITION 2.1 (49, Theorem 2.26). *Assume that $h : \mathbb{R}^n \to \mathbb{R} \cup \{+\infty\}$ is proper, lsc and convex. For every $\mu > 0$, $\operatorname{prox}_{\mu h}$ is single valued and continuous, and h_μ is convex and C^1 with*

$$\nabla h_\mu(u) = \frac{u - \operatorname{prox}_{\mu h}(u)}{\mu}. \tag{2.3}$$

Though (2.1) is a set, Proposition 2.1 indicates that it is single-valued under our assumptions, and we will abuse notation and write $v = \operatorname{prox}_{\mu h}(u)$ to mean $\{v\} = \operatorname{prox}_{\mu h}(u)$. In general, h_μ is not twice differentiable, but its gradient is Lipschitz continuous with constant μ^{-1} [49, Exercise 12.23].

In addition, there is no need to appeal to the general concept of limiting subdifferential of h [49, Definition 8.3b] for the latter coincides with the Fréchet subdifferential [49, Definition 8.3a], and both coincide with the subdifferential of convex analysis [49, Proposition 8.12], i.e.,

$$\partial h(x) := \{v \in \mathbb{R}^n \mid h(y) \geq h(x) + v^\top (y-x) \text{ for all } y \in \mathbb{R}^n\} \quad (x \in \operatorname{dom} h). \tag{2.4}$$

For $\bar{y} \in C$, the normal cone to C at $\bar{y}$ is

$$N_C(\bar{y}) := \{v \in \mathbb{R}^m \mid v^\top (y - \bar{y}) \leq 0 \text{ for all } y \in C\}. \tag{2.5}$$

We say that $x \in \mathbb{R}^n$ is *feasible* for (1.1) if $F(x) \in \operatorname{dom} h$ and $c(x) \in C$. Such feasible x is *stationary* for (1.1) if there exist $y \in \mathbb{R}^p$ and $z \in \mathbb{R}^m$ such that

$$0 = \nabla f(x) + \nabla F(x) y + \nabla c(x) z \tag{2.6a}$$

$$y \in \partial h(F(x)) \tag{2.6b}$$

$$z \in N_C(c(x)). \tag{2.6c}$$

A local minimizer of (1.1) is only guaranteed to be stationary under a constraint qualification. However, an asymptotic concept of stationarity holds without constraint qualification.

Let $x \in \mathbb{R}^n$ be feasible for (1.1). We say that x is *asymptotically stationary* if there exist sequences $\{x^k\} \to x$, $\{u^k\} \subseteq \operatorname{dom} h$, $\{y^k\} \subset \mathbb{R}^n$, $\{w^k\} \subset \mathbb{R}^m$ and $\{z^k\} \subset \mathbb{R}^m$ with $y^k \in \partial h(u^k)$ and $z^k \in N_C(w^k)$ for all $k \in \mathbb{N}$ such that

$$\nabla f(x^k) + \nabla F(x^k) y^k + \nabla c(x^k) z^k \to 0, \quad F(x^k) - u^k \to 0, \quad \text{and} \quad c(x^k) - w^k \to 0. \tag{2.7}$$

In (2.7), the slack variables u^k and w^k allow for the constraints $F(x) \in \operatorname{dom} h$ and $c(x) \in C$ to be violated along the iterations, yet satisfied in the limit. The following result connects local optimality and asymptotic stationarity.

PROPOSITION 2.2 (19, Proposition 2.5). *If $\bar{x} \in \mathbb{R}^n$ is a local minimizer of (1.1), it is asymptotically stationary.*

Let $\epsilon > 0$. For the purpose of designing a stopping condition in an iterative scheme, we say that $x \in \mathbb{R}^n$ is *ϵ-stationary* if there exist $u \in \operatorname{dom} h$, $w \in C$, $y \in \partial h(u)$ and $z \in N_C(w)$ such that

$$\|\nabla f(x) + \nabla F(x)y + \nabla c(x)z\| \leq \epsilon, \quad \|F(x) - u\| \leq \epsilon \quad \text{and} \quad \|c(x) - w\| \leq \epsilon. \tag{2.8}$$

3. Methodology. At any $x \in \mathbb{R}^n$, we assume that it is possible to evaluate f and ∇f, the proximal operator of h (with the value attained at the proximal point), and F and gradient-vector products $\nabla F(x)v$ for $v \in \mathbb{R}^p$.

A major obstacle if one wishes to apply a proximal method to (1.1) is that the proximal operator of $h \circ F$ is not available in general, even if that of h is.

We begin by introducing a lifting variable $u \in \mathbb{R}^p$ and reformulating (1.1) as

$$\underset{x \in \mathbb{R}^n, u \in \mathbb{R}^p}{\text{minimize}} \quad f(x) + h(u) \quad \text{subject to} \quad F(x) - u = 0, \quad c(x) \in C. \tag{3.1}$$

As may be expected, (3.1) is related to (1.1) in the following way.

LEMMA 3.1 (19, Lemma 3.1). *A feasible point $\bar{x} \in \mathbb{R}^n$ of (1.1) is asymptotically stationary for (1.1) if and only if $(\bar{x}, F(\bar{x}))$ is asymptotically stationary for (3.1).*

At this point, a nonsmooth augmented-Lagrangian method such as ALPS [19] could be applied to (3.1). However, (3.1) has p more variables and constraints than (1.1). Fortunately, the augmented-Lagrangian subproblem associated with (3.1) is amenable to partial minimization with respect to u, and that results in a smooth problem in x alone involving the Moreau envelope of h. We now detail the procedure.

Let $\mu > 0$ be a penalty parameter and $\hat{y} \in \mathbb{R}^p$ be a vector of Lagrange multiplier estimates for the equality constraint $F(x) - u = 0$. The (partial) augmented Lagrangian function for (3.1) is

$$\begin{aligned} \mathcal{L}(x, u; \hat{y}, \mu) &:= f(x) + h(u) + \hat{y}^\top (F(x) - u) + \tfrac{1}{2}\mu^{-1}\|F(x) - u\|_2^2 \\ &= f(x) + h(u) + \tfrac{1}{2}\mu^{-1}\|F(x) - u + \mu\hat{y}\|_2^2 - \tfrac{1}{2}\mu\|\hat{y}\|_2^2. \end{aligned} \tag{3.2}$$

For fixed $\hat{y}$ and $\mu > 0$, the augmented-Lagrangian subproblem is

$$\underset{x \in \mathbb{R}^n, u \in \mathbb{R}^p}{\text{minimize}} \quad \mathcal{L}(x, u; \hat{y}, \mu) \quad \text{subject to} \quad c(x) \in C. \tag{3.3}$$

In (3.3), we kept the constraints $c(x) \in C$ explicit, while we relaxed $F(x) = u$ and penalized its violation. Moreover, u is unconstrained, and the objective is strongly convex in u. Hence, partially minimizing (3.2) with respect to u is appealing. By (2.2), the optimal value of (3.2) with respect to u is

$$\begin{aligned} L(x; \hat{y}, \mu) &:= \min_{u \in \mathbb{R}^p} \mathcal{L}(x, u; \hat{y}, \mu) = f(x) + h_\mu(F(x) + \mu\hat{y}) - \tfrac{1}{2}\mu\|\hat{y}\|_2^2 \\ &= \mathcal{L}(x, u(x; \hat{y}, \mu); \hat{y}, \mu), \end{aligned} \tag{3.4}$$

where $u(x; \hat{y}, \mu) = \operatorname{prox}_{\mu h}(F(x) + \mu\hat{y}) \in \operatorname{dom} h$ realizes the minimum. Thus, we replace (3.3) with the partially-eliminated augmented-Lagrangian subproblem (1.2), which has a decision space of dimension n instead of $n + p$, and whose objective and constraints are C^1 by Proposition 2.1 and our assumptions on f, F and c.

For reference, Proposition 2.1 implies that the gradient of the objective in (1.2) is

$$\nabla f(x) + \nabla F(x) \nabla h_\mu(F(x) + \mu\hat{y}) = \nabla f(x) + \nabla F(x) y(x; \hat{y}, \mu).$$

where

$$y(x; \hat{y}, \mu) \coloneqq \hat{y} + \frac{F(x) - u(x; \hat{y}, \mu)}{\mu}. \tag{3.5}$$

Thus, $x \in \mathbb{R}^n$ is stationary for (1.2) if there exists $z \in \mathbb{R}^m$ such that

$$\nabla f(x) + \nabla F(x) y(x; \hat{y}, \mu) + \nabla c(x) z = 0, \quad \text{and} \quad z \in N_C(c(x)). \tag{3.6}$$

For $\epsilon > 0$, we say that $x \in \mathbb{R}^n$ is ϵ-stationary for (1.2) if there exist $w \in C$ and $z \in N_C(w)$ such that

$$\|\nabla f(x) + \nabla F(x) y(x; \hat{y}, \mu) + \nabla c(x) z\| \leq \epsilon, \quad \text{and} \quad \|c(x) - w\| \leq \epsilon. \tag{3.7}$$

LEMMA 3.2. *Let $\mu > 0$, $\hat{y} \in \mathbb{R}^p$, $\epsilon > 0$, and $x \in \mathbb{R}^n$ be ϵ-stationary for (1.2). If $\|F(x) - u(x; \hat{y}, \mu)\| \leq \epsilon$, then x is ϵ-stationary for (1.1).*

Proof. By assumption, there exist $w \in C$ and $z \in N_C(w)$ such that (3.7) holds. By definition of $u(x; \hat{y}, \mu)$, $y(x; \hat{y}, \mu)$ and (2.1), $y(x; \hat{y}, \mu) \in \partial h(u(x; \hat{y}, \mu))$ holds, and so does (2.8). ∎

4. Algorithm and convergence analysis. We state a generic template algorithm to solve (1.1) by solving a sequence of problems of the form (1.2) as Algorithm 4.1. Several concrete algorithms fit the template and will be detailed in Section 6.1. For the purposes of the convergence analysis, we state the features that such a concrete algorithm should possess.

Algorithm 4.1 Envelopt: template algorithm

1: Choose a stopping tolerance $\epsilon > 0$.
2: **for** $k = 0, 1, \ldots$ **do**
3: Select suitable $\epsilon_k > 0$, $\mu_k > 0$, and $\hat{y}_k \in \mathbb{R}^p$.
4: Find an ϵ_k-stationary point x_k of (1.2).
5: Compute $u_k = u(x_k; \hat{y}_k, \mu_k)$ and $y_k = y(x_k; \hat{y}_k, \mu_k)$.
6: If x_k is ϵ-stationary for (1.1), return x_k.
7: **end for**

Let x_k and u_k be generated by Algorithm 4.1 at iteration k. By Lemma 3.2, if $\epsilon_k \leq \epsilon$ and $\|F(x_k) - u_k\| \leq \epsilon$, x_k is an ϵ-stationary point of (1.1), and Algorithm 4.1 terminates at step 6.

4.1. Convergence analysis. We make the following assumption that a concrete implementation of Algorithm 4.1 should satisfy.

ASSUMPTION 4.1. *The sequences $\{\mu_k\}$, $\{x_k\}$, $\{u_k\}$ and $\{\hat{y}_k\}$ generated by Algorithm 4.1 are such that*

(i) if there exists $\mu > 0$ such that $\mu_k \geq \mu$ for all k, then, $\{F(x_k) - u_k\} \to 0$;
(ii) if there is an index set $K \subseteq \mathbb{N}$ such that $\{\mu_k\}_K \to 0$, then $\{\mu_k \hat{y}_k\}_K \to 0$.

The two conditions in Assumption 4.1 are related to mutually exclusive scenarios: if the penalty parameter remains bounded away from zero, then the iterates approach feasibility, otherwise the Lagrange multiplier estimates "do not behave too badly," as suggested by Conn et al. [10, §3].

THEOREM 4.1. *Let $\{x_k\}$ be a sequence generated by Algorithm 4.1 with $\{\epsilon_k\} \to 0$ and let Assumption 4.1 be satisfied. Then, any limit point of $\{x_k\}$ that is feasible for (1.1) is asymptotically stationary for (1.1).*

Proof. By construction, each x_k is ϵ_k-stationary for (1.2). Let $\bar{x}$ be a limit point of $\{x_k\}$ that is feasible for (1.1), and let $K \subseteq \mathbb{N}$ be an index set such that $\{x_k\}_K \to \bar{x}$. Because $\{\epsilon_k\}_K \to 0$, Lemma 3.2 implies that, if $\{F(x_k)-u_k\}_K \to 0$, $\bar{x}$ is asymptotically stationary for (1.1).

There are two cases. In the first case, there exists $\mu > 0$ such that $\mu_k \geq \mu$ for all k. Clearly, Assumption 4.1 then implies that $\{F(x_k) - u_k\}_K \to 0$.

In the second case, $\{\mu_k\}_K \to 0$. By definition of u_k, for all $u \in \mathbb{R}^p$ and all k,

$$\tfrac{1}{2}\|F(x_k) + \mu_k\hat{y}_k - u_k\|_2^2 + \mu_k h(u_k) \leq \tfrac{1}{2}\|F(x_k) + \mu_k\hat{y}_k - u\|_2^2 + \mu_k h(u).$$

In particular, for $u = F(\bar{x}) \in \operatorname{dom} h$,

$$\tfrac{1}{2}\|F(x_k) + \mu_k\hat{y}_k - u_k\|_2^2 + \mu_k h(u_k) \leq \tfrac{1}{2}\|F(x_k) + \mu_k\hat{y}_k - F(\bar{x})\|_2^2 + \mu_k h(F(\bar{x})).$$

We take the limit inferior over $k \in K$ on both sides and use Assumption 4.1 to obtain

$$\liminf_{k\in K} \tfrac{1}{2}\|F(\bar{x}) - u_k\|_2^2 + \mu_k h(u_k) \leq 0. \tag{4.1}$$

We now wish to establish that $\liminf_{k\in K} \|F(\bar{x}) - u_k\|_2 = 0$.

We first show that $\{u_k\}_K$ has a bounded subsequence. If it were not the case, $\liminf_{k\in K} \|F(\bar{x}) - u_k\|_2 = +\infty$. By convexity, h is bounded from below by an affine function, i.e., there exist $a \in \mathbb{R}^p$ and $b \in \mathbb{R}$ such that $h(u) \geq a^\top u + b$ for all $u \in \mathbb{R}^p$. Thus,

$$\liminf_{k\in K} \tfrac{1}{2}\|F(\bar{x}) - u_k\|_2^2 + \mu_k(a^\top u_k + b) = \liminf_{k\in K} \tfrac{1}{2}\|F(\bar{x}) - u_k\|_2^2 + \mu_k a^\top u_k \leq 0,$$

which can be written

$$\liminf_{k\in K} \tfrac{1}{2}\|u_k\|^2 + (\mu_k a - F(\bar{x}))^\top u_k + \tfrac{1}{2}\|F(\bar{x})\|_2^2 \leq 0.$$

But the above is impossible as the value of the left-hand side is $+\infty$. Thus, reducing to a further subsequence if necessary, we may assume without loss of generality that $\{u_k\}_K$ is bounded. Consequently, $\{F(\bar{x}) - u_k\}_K$ is bounded as well.

Assume by contradiction that $\liminf_{k\in K} \|F(\bar{x}) - u_k\|_2 := \delta > 0$. Without loss of generality, we may assume that $\{u_k\}_K \to \bar{u}$. We now obtain from (4.1) that $\frac{1}{2}\|F(\bar{x}) - \bar{u}\|_2^2 \leq 0$, which contradicts the assumption that $\delta > 0$. The above establishes that $\liminf_{k\in K} \|F(\bar{x}) - u_k\|_2 = 0$. Again, reducing to a further subsequence if necessary, we obtain that $\bar{x}$ is asymptotically stationary for (1.1). ∎

4.2. Infeasibility detection. There are two ways in which (1.1) can be infeasible. The first is that there is no $x \in \mathbb{R}^n$ such that $c(x) \in C$. In order to detect such situation, we rest upon the subproblem solver since those constraints are kept explicitly in (1.2). Thus, in this section, we assume that $c(x) \in C$ is feasible. The second way is that there is no $x \in \mathbb{R}^n$ such that $c(x) \in C$ and $F(x) \in \operatorname{dom} h$. In order to detect this second situation, we formulate the feasibility problem

$$\underset{x\in\mathbb{R}^n}{\text{minimize}} \quad \tfrac{1}{2}\operatorname{dist}(F(x) \mid \overline{\operatorname{dom} h})^2 \quad \text{subject to} \quad c(x) \in C. \tag{4.2}$$

In general, (4.2) is impractical as we may not know $\operatorname{dom} h$ explicitly, let alone how to compute the distance to its closure. However, (4.2) captures the concept of infeasibility for (1.1) because, as in the proof of Theorem 4.1, if $\{x_k\}$ has a feasible limit point, $\{F(x_k) - u_k\} \to 0$ with $u_k \in \operatorname{dom} h$ for all k. Thus, if (1.1) is feasible, we may hope that $F(x_k) \in \overline{\operatorname{dom} h}$ in the limit while the subproblem solver ensures that $\operatorname{dist}(c(x_k) \mid C) \to 0$.

To identify cases where the optimal value of (4.2) is positive, we now argue that an appropriate proxy for (4.2) is

$$\underset{x \in \mathbb{R}^n}{\text{minimize}} \quad \mu h_\mu(F(x)) \quad \text{subject to} \quad c(x) \in C. \tag{4.3}$$

The next result shows that, as $\{\mu_k\} \to 0$, $\mu_k h$ flattens to the zero function over the closure of its domain.

LEMMA 4.2. *For $h : \mathbb{R}^n \to \mathbb{R} \cup \{+\infty\}$ proper, lsc and convex, and any positive nonincreasing sequence $\{\mu_k\} \to 0$,*

$$\operatorname*{e-lim}_{k \to \infty} \mu_k h = \chi(\cdot \mid \overline{\operatorname{dom} h}).$$

Proof. By [49, Proposition 7.2], we must show that, for all $x \in \mathbb{R}^n$,

1. $\liminf_{k \to \infty} \mu_k h(x_k) \geq \chi(x \mid \overline{\operatorname{dom} h})$ for every sequence $\{x_k\} \to x$, and
2. $\limsup_{k \to \infty} \mu_k h(x_k) \leq \chi(x \mid \overline{\operatorname{dom} h})$ for at least one sequence $\{x_k\} \to x$.

Let $x \in \mathbb{R}^n$ and consider an arbitrary sequence $\{x_k\} \to x$. Two situations can occur. Consider first the case where $x \notin \overline{\operatorname{dom} h}$. For all sufficiently large k, it must be that $x_k \notin \operatorname{dom} h$, and thus $h(x_k) = \mu_k h(x_k) = +\infty$. Hence, $\liminf_{k \to \infty} \mu_k h(x_k) = +\infty = \chi(x \mid \overline{\operatorname{dom} h})$.

Consider now the case where $x \in \overline{\operatorname{dom} h}$. Because h is proper and convex, it is bounded below by an affine function, i.e., there exist $a \in \mathbb{R}^n$ and $b \in \mathbb{R}$ such that $h(x) \geq a^\top x + b$ for all $x \in \mathbb{R}^n$. Thus, $\mu_k h(x) \geq \mu_k a^\top x + \mu_k b$ for all k, and $\liminf_{k \to \infty} \mu_k h(x_k) \geq 0 = \chi(x \mid \overline{\operatorname{dom} h})$. We have established condition 1.

We now establish condition 2. Let $x \in \mathbb{R}^n$. Three situations can occur. Consider first the case where $x \notin \overline{\operatorname{dom} h}$, and let $x_k := x$ for all k. Then, $\limsup_{k \to \infty} \mu_k h(x_k) = +\infty = \chi(x \mid \overline{\operatorname{dom} h})$.

Consider next the case where $x \in \operatorname{dom} h$, and let $x_k := x$ for all k. Then, $\limsup_{k \to \infty} \mu_k h(x_k) = 0 = \chi(x \mid \overline{\operatorname{dom} h})$.

Finally, consider the case where $x \in \overline{\operatorname{dom} h} \setminus \operatorname{dom} h$. There exists a sequence $\{w_k\} \subseteq \operatorname{dom} h$ such that $\{w_k\} \to x$. Let $C_k := \{x \in \mathbb{R}^n \mid |h(x)| \leq 1/\sqrt{\mu_k}\} \subseteq \operatorname{dom} h$ for all k. We have $\operatorname{dom} h = \cup_k C_k$. Let $q_0 \geq 0$ be the smallest integer such that $w_{q_0} \in C_0$. Because $\{\mu_k\}$ is nonincreasing, for each $k \geq 1$, there exists a smallest integer $q_k > q_{k-1}$ such that $w_{q_k} \in C_k$. By construction, $\{q_k\}$ is increasing, and therefore, $\{w_{q_k}\} \to x$. Let $x_k := w_{q_k}$ for all k. We obtain $\mu_k |h(x_k)| \leq \sqrt{\mu_k} \to 0$, and hence, $\limsup_k \mu_k h(x_k) = \lim_k \mu_k h(x_k) = 0 = \chi(x \mid \overline{\operatorname{dom} h})$. ∎

The next result shows that epi-convergence is preserved under infimal convolution with the half squared Euclidean norm.

LEMMA 4.3. *Let h_k and $h : \mathbb{R}^n \to \mathbb{R} \cup \{+\infty\}$ be proper, lsc and convex, and let $g := \frac{1}{2}\|\cdot\|_2^2$. If $\operatorname{e-lim} h_k = h$, then $\operatorname{e-lim}(h_k \square g) = h \square g$.*

Proof. Under our assumptions, [49, Theorem 11.34] implies that $\operatorname{e-lim} h_k^* = h^*$. Because $g = g^*$, [49, Exercise 7.8*a*] yields $\operatorname{e-lim} h_k^* + g^* = h^* + g^*$. Note that h_k^*, h^* and g^* are proper, lsc and convex as well. The sum of Fenchel conjugates is

related to the inf-convolution via $h^* + g^* = (h\square g)^*$ [49, Theorem 11.23a], and hence e-lim$(h_k\square g)^* = (h\square g)^*$. Thus, [49, Theorem 11.34] again implies e-lim$(h_k\square g) = h\square g$.□

We are now in position to establish the main result that links (4.2) and (4.3).

THEOREM 4.4. *For any positive nonincreasing sequence* $\{\mu_k\} \to 0$,

$$\operatorname*{e-lim}_{k\to\infty} \mu_k h_{\mu_k} = \tfrac{1}{2}\ dist(\cdot \mid \overline{\operatorname{dom} h})^2.$$

Proof. Apply Lemma 4.2 and Lemma 4.3 to $h_k := \mu_k h$ to obtain

$$\operatorname*{e-lim}_{k\to\infty}((\mu_k h)\square g) = \chi(\cdot \mid \overline{\operatorname{dom} h})\square g = \tfrac{1}{2}\operatorname{dist}(\cdot \mid \overline{\operatorname{dom} h})^2.$$

The result follows from the observation that $(\mu_k h)\square g = \mu_k(h\square(\mu_k^{-1} g)) = \mu_k h_{\mu_k}$. □

The first-order optimality conditions for (4.3) are that there exist $w \in C$ and $z \in N_C(w)$ such that

$$\nabla F(x)(F(x) - \operatorname*{prox}_{\mu h}(F(x))) + \nabla c(x) z = 0, \quad c(x) - w = 0. \tag{4.4}$$

Then, in analogy with (2.8), it seems reasonable to declare local infeasibility in Algorithm 4.1 if $\{\mu_k\} \to 0$ and

$$\|\nabla F(x_k)(F(x_k) - u_k) + \nabla c(x_k) z\| \le \epsilon^{\text{inf}}, \quad \|c(x_k) - w\| \le \epsilon^{\text{inf}} \tag{4.5a}$$

$$\text{and} \quad \|F(x_k) - u_k\| > \epsilon, \tag{4.5b}$$

for some $w \in C$, $z \in N_C(w)$, and prescribed tolerances $\epsilon, \epsilon^{\text{inf}} > 0$. Criterion (4.5) is easily computable along the iterations and provides a well-defined termination condition, as we now demonstrate.

By (3.7), each iteration of Algorithm 4.1 produces iterates x_k, $u_k = u(x_k; \hat{y}_k, \mu_k)$, $y_k = y(x_k; \hat{y}_k, \mu_k) \in \partial h(u_k)$, $w_k \in C$, and $z_k \in N_C(w_k)$ such that

$$\|\nabla f(x_k) + \nabla F(x_k) y_k + \nabla c(x_k) z_k\| \le \epsilon_k, d\|c(x_k) - w_k\| \le \epsilon_k.$$

The first condition can be written

$$\|\mu_k \nabla f(x_k) + \nabla F(x_k)(\mu_k \hat{y}_k + F(x_k) - u_k) + \nabla c(x_k)(\mu_k z_k)\| \le \mu_k \epsilon_k. \tag{4.6}$$

Assume there exists an index set $K \subseteq N$ such that $\{\mu_k\}_K \to 0$. If $\nabla f(x_k)$ remains bounded, $\{\mu_k \nabla f(x_k)\}_K \to 0$. By Assumption 4.1, $\{\mu_k \hat{y}_k\}_K \to 0$. Because $N_C(w_k)$ is a cone, $\mu_k z_k \in N_C(w_k)$. Therefore, (4.5) will be satisfied for sufficiently large $k \in K$ if (1.1) is infeasible.

4.3. Complexity. Let $\epsilon > 0$. We first wish to bound the maximum number of iterations for Algorithm 4.1 to return an ϵ-stationary point of (1.1) in the event that $\{\mu_k\}$ remains bounded away from zero. We require the following assumption.

ASSUMPTION 4.2. *A concrete implementation of Algorithm 4.1 is such that there exists a positive sequence* $\{\eta_k\}$ *and constants* $0 < \kappa_\mu < 1$, $0 < \kappa_\eta < 1$ *and* $0 < \kappa_\epsilon < 1$ *satisfying the following conditions for all* k*:*

(i) $\|F(x_k) - u_k\| > \eta_k \implies \mu_{k+1} \le \kappa_\mu \mu_k$*;*

(ii) $\|F(x_k) - u_k\| \le \eta_k \implies \mu_{k+1} = \mu_k$, $\eta_{k+1} \le \kappa_\eta \eta_k$, *and* $\epsilon_{k+1} \le \kappa_\epsilon \epsilon_k$.

The following result is inspired from [4, Theorem 3.1].

THEOREM 4.5. *Let Assumption 4.2 be satisfied, and let $\epsilon_0 \geq \epsilon$. Without loss of generality, assume that $\bar{\eta} \geq \epsilon$. Assume that each iteration of Algorithm 4.1 succeeds, and that*

(i) there exists $\bar{\mu} > 0$ such that $\mu_k \geq \bar{\mu}$ for all k, and

(ii) there exists $\bar{\eta} \geq \epsilon$ such that $\|F(x_k) - u_k\| \leq \bar{\eta}$ for all k.

Algorithm 4.1 returns an ϵ-stationary point of (1.1) in at most

$$\left\lceil \frac{\log(\bar{\mu}/\mu_0)}{\log(\kappa_\mu)} \right\rceil \max\left(\left\lceil \frac{\log(\epsilon/\epsilon_0)}{\log(\kappa_\epsilon)} \right\rceil, \left\lceil \frac{\log(\epsilon/\bar{\eta})}{\log(\kappa_\eta)} \right\rceil \right) \tag{4.7}$$

iterations.

Proof. Because μ_k never increases and is bounded away from zero, Assumption 4.2 ensures that it eventually remains constant. When that occurs, η_k continues to decrease, so that $F(x_k) - u_k \to 0$. In addition, there are at most $\lceil \log(\bar{\mu}/\mu_0)/\log(\kappa_\mu) \rceil$ iterations k such that $\|F(x_k) - u_k\| > \eta_k$.

On iterations for which $\|F(x_k) - u_k\| \leq \eta_k$, ϵ_k decreases by a factor of at least κ_ϵ, and η_k decreases by a factor of at least κ_η. Thus, there are at most $\lceil \log(\epsilon/\epsilon_0)/\log(\kappa_\epsilon) \rceil$ such iterations until $\epsilon_k \leq \epsilon$, and at most $\lceil \log(\epsilon/\bar{\eta})/\log(\kappa_\eta) \rceil$ such iterations until $\eta_k \leq \epsilon$.

In the worst case, primal feasibility $\|F(x_k) - u_k\|$ returns to its largest value $\bar{\eta}$ at each subproblem. Thus, the total number of iterations is at most (4.7). ∎

In Theorem 4.5, we assumed without loss of generality that $\bar{\eta} \geq \epsilon$. If it were not the case, approximate primal feasibility would be satisfied throughout the iterations and the second term in the maximum in (4.7) would simply not be present. In all cases, the maximum number of iterations is $O(\log(\epsilon))$.

In general, though, Algorithm 4.1 may stop at an iterate x_k that appears to be infeasible and, at the same time, a local minimizer of the infeasibility problem (4.3). The possibility that $\mu_k \to 0$ must be considered, since it necessarily takes place, for example, when (1.1) is infeasible. The following theorem establishes an upper bound on the number of iterations before Envelopt finds an approximate stationary point (in the sense of (2.8)) or an infeasible point that is approximately stationary for the infeasibility problem (4.3), according to (4.5).

Let $\omega(k) \in \mathbb{N}$ denote the number of penalty updates up to the k-th iteration. The update rules in Assumption 4.2 give $\mu_k \leq \kappa_\mu^{\omega(k)} \mu_0$ and $\omega(k) \leq \ln(\mu_k/\mu_0)/\ln \kappa_\mu$.

THEOREM 4.6. *Let $\epsilon > 0$ and $\epsilon^{\text{inf}} \in (0, \epsilon]$ be given. Let Assumption 4.2 be satisfied, and let $\epsilon_0 \geq \epsilon$. Assume that*

(i) there exists $\bar{\eta} \geq \epsilon$ such that $\|F(x_k) - u_k\| \leq \bar{\eta}$ for all k,

(ii) there exist $c_f, c_F \in \mathbb{R}$ such that $\|\nabla f(x_k)\| \leq c_f$ and $\|\nabla F(x_k)\| \leq c_F$ for all k,

(iii) there exists $N(\epsilon^{\text{inf}}, \epsilon) \in \mathbb{N}$ such that $\epsilon_k \leq \min\{\epsilon, \epsilon^{\text{inf}}/4\}$ for all $k \geq N(\epsilon^{\text{inf}}, \epsilon)$,

(iv) there exist $\varrho_0 > 0$ and $\kappa_\varrho \in (0, 1)$ such that $\mu_k \|\hat{y}_k\| \leq \varrho_0 \kappa_\varrho^{\omega(k)}$ for all k.

Then, Algorithm 4.1 returns either an ϵ-stationary point of (1.1), according to (2.8), or an ϵ-infeasible ϵ^{inf}-stationary point of (4.3), according to (4.5), in at most

$$\max\{N(\epsilon^{\text{inf}}, \epsilon), P(\epsilon) U(\epsilon^{\text{inf}})\}$$

iterations, where $P(\epsilon) := \left\lceil \frac{\ln(\epsilon/\bar{\eta})}{\ln(\kappa_\eta)} \right\rceil$ and

$$U(\epsilon^{\text{inf}}) := \left\lceil \max\left\{ 1, \frac{\ln(\epsilon^{\text{inf}}/(4 c_F \varrho_0))}{\ln \kappa_\varrho}, \frac{\ln(\min\{1, \epsilon^{\text{inf}}/(4 c_f)\}/\mu_0)}{\ln \kappa_\mu} \right\} \right\rceil .$$

Some comments on the assumptions are in order, particularly in relation to [4, Theorem 3.5]. Similarly to the boundedness requirement (i) on the constraint violation, the assumption (ii) on the derivatives appears in [4, Lemma 3.3] as a result of the boundedness of $\{x_k\}$ (which they assume directly). In contrast, requirements (iii) and (iv) are algorithmic: they pose restrictions on the sequences of hyperparameters selected by Algorithm 4.1, and not on the problem at hand. While $N(\epsilon^{\text{inf}}, \epsilon)$ in (iii) is the same as in [4, Thm 3.5], the requirement (iv) is weaker than the (rigid) safeguarding condition adopted for the AL method in [4]. Indeed, the BCL scheme of Conn et al. [10] is also covered; see [10, Lemma 4.1].

Before proving Theorem 4.6 we give an intermediate result concerning stationarity for the infeasibility problem (4.3).

LEMMA 4.7. *Let $\epsilon > 0$ and $\epsilon^{\text{inf}} \in (0, \epsilon]$ be given. Let Assumption 4.2 be satisfied. Suppose (ii)–(iv) from Theorem 4.6 are valid for the iterates generated by Algorithm 4.1. Then, $k \geq N(\epsilon^{\text{inf}}, \epsilon)$ and $\omega(k) \geq U(\epsilon^{\text{inf}})$ together imply (4.5a).*

Proof. Fix $k \in \mathbb{N}$ and assume $k \geq N(\epsilon^{\text{inf}}, \epsilon)$ and $\omega(k) \geq U(\epsilon^{\text{inf}})$ hold. Owing to the definition of $U(\epsilon^{\text{inf}})$, the latter condition implies

$$\varrho_0 \kappa_\varrho^{\omega(k)} \leq \epsilon^{\text{inf}}/(4c_F) \qquad \text{and} \qquad \mu_k \leq \min\{1, \epsilon^{\text{inf}}/(4c_f)\}. \tag{4.8}$$

For convenience, define the C^1 functions

$$P_k(x) := \mu_k h_{\mu_k}(F(x)) \qquad \text{and} \qquad \widehat{P}_k(x) := \mu_k h_{\mu_k}(F(x) + \mu_k \hat{y}_k)$$

to write more compactly the objective functions of (4.3) and (1.2), respectively.

We proceed in three steps, (a) showing that $\|c(x_k) - w\| \leq \epsilon^{\text{inf}}$ holds for some $w \in C$, (b) providing expressions for $\nabla P_k(x_k)$ and $\nabla \widehat{P}_k(x_k)$, and (c) establishing that $\|\nabla P_k(x_k) + \nabla c(x_k) z\| \leq \epsilon^{\text{inf}}$ holds for some $z \in N_C(w)$. Together, these observations prove (4.5a).

(a) By (iii), $k \geq N(\epsilon^{\text{inf}}, \epsilon)$ implies $\epsilon_k \leq \epsilon^{\text{inf}}$, hence subproblem (1.2) delivers $\|c(x_k) - w_k\| \leq \epsilon^{\text{inf}}$ for some $w_k \in C$, by (3.7).

(b) Direct derivation of P_k yields $\nabla P_k(x_k) = \nabla F(x_k)(F(x_k) - \operatorname{prox}_{\mu_k h}(F(x_k)))$, using the differentiation rule for the Moreau envelope (2.3). Similarly, using also the definition of $u_k = u(x_k; \hat{y}_k, \mu_k)$, it is $\nabla \widehat{P}_k(x_k) = \nabla F(x_k)(F(x_k) - u_k + \mu_k \hat{y}_k)$.

(c) We find first a bound for $\|\nabla \widehat{P}_k(x_k) + \nabla c(x_k) z\|$, then for the difference $\|\nabla \widehat{P}_k(x_k) - \nabla P_k(x_k)\|$, and finally for $\|\nabla P_k(x_k) + \nabla c(x_k) z\|$ by combining the first two. It will be natural to consider the multiplier $z = \mu_k z_k \in N_C(w_k)$. From (4.6) and (2.3) we have that

$$\begin{aligned} \mu_k \epsilon_k &\geq \|\mu_k \nabla f(x_k) + \nabla F(x_k)(F(x_k) - u_k + \mu_k \hat{y}_k) + \nabla c(x_k)(\mu_k z_k)\| \\ &= \|\mu_k \nabla f(x_k) + \nabla \widehat{P}_k(x_k) + \nabla c(x_k)(\mu_k z_k)\| \end{aligned}$$

and therefore

$$\begin{aligned} \|\nabla \widehat{P}_k(x_k) + \nabla c(x_k)(\mu_k z_k)\| &\leq \|\mu_k \nabla f(x_k)\| + \mu_k \epsilon_k \\ &\leq \mu_k c_f + \mu_k \epsilon_k && [\text{with } c_f \text{ from (ii)}] \\ &\leq \mu_k c_f + \mu_k \epsilon^{\text{inf}}/4 && [\text{due to } k \geq N(\epsilon^{\text{inf}}, \epsilon)] \\ &\leq \epsilon^{\text{inf}}/2. && [\text{by (4.8)}] \end{aligned}$$

Now we bound the difference between $\nabla \widehat{P}_k$ and ∇P_k, obtaining

$$\begin{aligned}
&\|\nabla \widehat{P}_k(x_k) - \nabla P_k(x_k)\| \\
&= \left\| \nabla F(x_k) \left[\operatorname*{prox}_{\mu_k h}(F(x_k)) + \mu_k \hat{y}_k - \operatorname*{prox}_{\mu_k h}(F(x_k) + \mu_k \hat{y}_k) \right] \right\| \\
&\leq c_F \left\| \operatorname*{prox}_{\mu_k h}(F(x_k)) + \mu_k \hat{y}_k - \operatorname*{prox}_{\mu_k h}(F(x_k) + \mu_k \hat{y}_k) \right\| && \text{[with } c_F \text{ from (ii)]} \\
&\leq 2 c_F \mu_k \|\hat{y}_k\| \\
&\leq 2 c_F \varrho_0 \kappa_\varrho^{\omega(k)} \leq \epsilon^{\inf}/2, && \text{[by (iv) and (4.8)]}
\end{aligned}$$

where the second inequality follows from the nonexpansiveness of $\operatorname{prox}_{\mu_k h}$ (unit Lipschitz constant by [49, Thms 12.12, 12.17]). Therefore, combining the bounds above we obtain

$$\begin{aligned}
\|\nabla P_k(x_k) + \nabla c(x_k)(\mu_k z_k)\| &= \|\nabla P_k(x_k) - \nabla \widehat{P}_k(x_k) + \nabla \widehat{P}_k(x_k) + \nabla c(x_k)(\mu_k z_k)\| \\
&\leq \|\nabla P_k(x_k) - \nabla \widehat{P}_k(x_k)\| + \|\nabla \widehat{P}_k(x_k) + \nabla c(x_k)(\mu_k z_k)\| \\
&\leq \epsilon^{\inf},
\end{aligned}$$

concluding the proof. ∎

The main complexity result can now be established.

Proof of Theorem 4.6. We proceed in two steps, always considering iterations $k \geq N(\epsilon^{\inf}, \epsilon)$. First, we show that if (2.8) is never satisfied, then (4.5b) holds and the number of iterations between two consecutive penalty updates is bounded above by $P(\epsilon)$. Second, we invoke Lemma 4.7: if the number of penalty updates exceeds $U(\epsilon^{\inf})$, then (4.5a) holds.

First step: After the first $N(\epsilon^{\inf}, \epsilon)$ iterations, $\epsilon_k \leq \epsilon$ by (iii) guarantees that

$$\|\nabla f(x_k) + \nabla F(x_k) y_k + \nabla c(x_k) z_k\| \leq \epsilon d \text{and} \quad \|c(x_k) - w_k\| \leq \epsilon$$

are satisfied for some suitable $w_k \in C$ and $z_k \in N_C(w_k)$. Therefore, (4.5b) must hold if (2.8) fails.

Because $\|F(x_k) - u_k\| \leq \bar{\eta}$ for all k by assumption (i), the conditions in Assumption 4.2 indicate that it takes at most $P(\epsilon) := \lceil \ln(\epsilon/\bar{\eta})/\ln(\kappa_\eta) \rceil$ consecutive iterations without penalty updates to reach $\|F(x_k) - u_k\| \leq \epsilon$. This means that, for $k \geq N(\epsilon^{\inf}, \epsilon)$, penalty updates are at most $P(\epsilon)$ iterations apart as long as (2.8) fails.

Second step: We now invoke Lemma 4.7, which guarantees that condition (4.5a) is satisfied after at most $N(\epsilon^{\inf}, \epsilon)$ iterations and $U(\epsilon^{\inf})$ penalty updates.

Combining these observations, the total number of iterations to satisfy either (2.8) or (4.5) is at most $N(\epsilon^{\inf}, \epsilon) + P(\epsilon) U(\epsilon^{\inf})$. ∎

5. Extensions and applications.

5.1. Standard augmented Lagrangian method. A simple observation that is worth pointing out is that the standard augmented Lagrangian method is a special case of Algorithm 4.1. Suppose indeed that $h = 0$ in (1.1), and that we reformulate it equivalently as

$$\operatorname*{minimize}_{x \in \mathbb{R}^n} df(x) + \chi(c(x) \mid C).$$

The latter has the form (1.1) with $h = \chi(\cdot \mid C)$, $F = c$ and no explicit constraints. Each Envelopt subproblem involves the Moreau envelope of the indicator of C, i.e.,

$$\chi_\mu = \chi(\cdot \mid C) \square (\tfrac{1}{2}\mu^{-1}\|\cdot\|_2^2) = \tfrac{1}{2}\mu^{-1}\operatorname{dist}(\cdot \mid C)^2.$$

Hence, each Envelopt subproblem has the familiar form

$$\underset{x \in \mathbb{R}^n}{\text{minimize}}\; df(x) + \tfrac{1}{2}\mu^{-1}\operatorname{dist}(c(x) + \mu\hat{y} \mid C)^2.$$

5.2. NCL extension. As a strategy to solve constrained problems that do not satisfy a constraint qualification, Ma et al. [34] proposed the NCL formulation of the augmented Lagrangian method. The main idea is to relax the hard constraints $c(x) \in C$ by introducing *residual* variables r such that $c(x) - r \in C$, and to penalize the constraint $r = 0$ via an augmented Lagrangian. The same idea applied to (1.1) yields augmented Lagrangian subproblems of the form

$$\underset{x \in \mathbb{R}^n, r \in \mathbb{R}^m}{\text{minimize}}\; df(x) + h(F(x)) + \tfrac{1}{2}\rho^{-1}\|r + \rho\hat{z}\|_2^2 \quad \text{subject to} \quad c(x) - r \in C,$$

where $\rho > 0$ is a penalty parameter and $\hat{z} \in \mathbb{R}^m$ is a vector of Lagrange multiplier estimates. Advantages of the formulation above are that the subproblems are always feasible, always satisfy LICQ, and that solvers are better able to exploit sparsity of ∇c than if $c(x) \in C$ were penalized explicitly in the objective via an augmented Lagrangian. We refer the interested reader to [34, 35] for more details on the NCL formulation and its implementation.

The smoothing strategy of $h(F(x))$ described in Section 3 can be applied to the above subproblem to yield subproblems of the form

$$\underset{x \in \mathbb{R}^n, r \in \mathbb{R}^m}{\text{minimize}}\; df(x) + h_\mu(F(x) + \mu\hat{y}) + \tfrac{1}{2}\rho^{-1}\|r + \rho\hat{z}\|_2^2 \quad \text{subject to} \quad c(x) - r \in C.$$

One may choose $\rho = \mu$ in each subproblem, or to manage the two penalty parameters separately.

The combined Envelopt/NCL formulation may be obtained by applying Envelopt to the formulation

$$\underset{x \in \mathbb{R}^n}{\text{minimize}}\; df(x) + h(F(x)) + \chi(c(x) \mid C),$$

without explicit constraints and the combined proper, lsc and convex nonsmooth term $(u, v) \mapsto h(u) + \chi(v \mid C)$. In that sense, Envelopt generalizes NCL.

5.3. An exact penalty method. Consider the smooth constrained problem

$$\underset{x \in \mathbb{R}^n}{\text{minimize}}\; f(x) \quad \text{subject to} \quad F(x) = 0, \quad \underline{x} \le x \le \overline{x}. \tag{5.1}$$

By way of introducing slack variables, any smooth constrained problem with equality and inequality constraints can be cast in the form (5.1). The exact penalty method [43] consists in solving a sequence of subproblems of the form

$$\underset{x \in \mathbb{R}^n}{\text{minimize}}\; f(x) + \tau\|F(x)\| \quad \text{subject to} \quad \underline{x} \le x \le \overline{x}, \tag{5.2}$$

where $\|\cdot\|$ is a norm and $\tau > 0$ is a penalty parameter. The latter subproblem has the general form (1.1) with $h = \|\cdot\|$, $c = \mathbb{I}$ and $C = [\underline{x}, \overline{x}]$. The proximal operator of

h, and hence its Moreau envelope, is known and is related to the projection into the unit ball in the dual norm. Most often, h is the ℓ_1 or Euclidean norm. Algorithm 4.1 provides a way to address the exact penalty subproblem (5.2) based on a smoothing of h via its Moreau envelope. Algorithm 5.1 summarizes the resulting algorithm to solve (5.1) via exact penalty.

Algorithm 5.1 Exact penalty method based on Algorithm 4.1.

Choose a stopping tolerance $\epsilon > 0$, $0 < \kappa_\epsilon < 1$, and $\kappa_\tau > 1$.
Choose $\epsilon_0 \geq \epsilon$, and $\tau_0 > 0$.
for $k = 0, 1, \ldots$ **do**
    Find an ϵ_k-stationary point x_k of (5.2) with $\tau = \tau_k$ using Algorithm 4.1.
    If x_k is ϵ-stationary for (5.1), return x_k.
    Set $\tau_{k+1} := \kappa_\tau \tau_k$ and $\epsilon_{k+1} = \kappa_\epsilon \epsilon_k$.
end for

Each subproblem (5.2) is bound constrained and can be solved with any appropriate solver for bound-constrained optimization. Had bound constraints not been present in (5.1), the subproblems would have been unconstrained. Similarly, any constraint that was part of $F(x) = 0$ could have been kept explicit provided a general constrained solver is available. In this sense, the approach above is more general than that described by Diouane et al. [23], who consider only equality-constrained problems and must appeal to a solver for nonsmooth problems to solve the subproblems. When $\|\cdot\|$ is a polyhedral norm, (5.2) can be reformulated as a smooth problem, though one with general inequality constraints. The approach above yields smooth subproblems with either simple bounds or no constraints at all.

5.4. Proximal operator of a composite sum. Let $g : \mathbb{R}^n \to \mathbb{R} \cup \{+\infty\}$ be proper, lsc, and have a known proximal operator. To evaluate the proximal operator of $(h \circ F) + g$, with parameter $\lambda > 0$, we must solve a problem of the form

$$\underset{x \in \mathbb{R}^n}{\text{minimize}} \quad \tfrac{1}{2}\lambda^{-1}\|x - u\|_2^2 + h(F(x)) + g(x), \tag{5.3}$$

whose solutions are generally not known explicitly. However, Algorithm 4.1 can be used to approximate one. Indeed, each subproblem (1.2) takes the form

$$\underset{x \in \mathbb{R}^n}{\text{minimize}}\, d\, \tfrac{1}{2}\lambda^{-1}\|x - u\|_2^2 + h_\mu(F(x) + \mu\hat{y}) + g(x),$$

and can be solved with, e.g., the proximal-gradient method.

6. Implementation and numerical experiments.

6.1. Realizations of Envelopt. Two concrete implementations of Algorithm 4.1 inspired directly by popular augmented Lagrangian methods in the literature satisfy Assumption 4.1. The first is the algorithm of Conn et al. [10], which we restate as Algorithm 6.1.

PROPOSITION 6.1. *Algorithm 6.1 satisfies Assumption 4.1.*

Proof. To establish the first condition in Assumption 4.1, consider $\epsilon = 0$. If there exists $\bar{\mu} > 0$ such that $\mu_k \geq \bar{\mu}$, μ_k can only be updated finitely many times, and the updates of Line 10 are performed infinitely many times for all sufficiently large k. Therefore, since $\alpha_{k+1} \in (0, 1)$ for each k, we obtain $\eta_k \to 0$, and hence, $F(x_k) - u_k \to 0$. The second condition in Assumption 4.1 was established in [10, Lemma 4.1]. ∎

Algorithm 6.1 Envelopt algorithm modeled after [10].

Choose stopping tolerance $\epsilon > 0$.
Choose $0 < \kappa_\mu < 1$, $0 < \gamma < 1$, $\alpha_\epsilon > 0$, $\alpha_\eta > 0$, $\beta_\epsilon > 0$, and $\beta_\eta > 0$.
Choose $\epsilon_1 \geq \epsilon$, $\eta_1 \geq \epsilon$, $\mu_0 > 0$ and $\hat{y}_0 \in \mathbb{R}^p$.
for $k = 0, 1, \ldots$ **do**
    Find an ϵ_k-stationary point x_k of (1.2).
    Compute $u_k = u(x_k; \hat{y}_k, \mu_k)$ and $y_k = y(x_k; \hat{y}_k, \mu_k)$.
    If x_k is ϵ-stationary for (1.1), return x_k.
    if $\|F(x_k) - u_k\| \leq \max(\epsilon, \eta_k)$ **then**
        Set $\mu_{k+1} \leftarrow \mu_k$ and $\hat{y}_{k+1} \leftarrow y_k$.
        Set $\alpha_{k+1} \leftarrow \min(\gamma, \mu_{k+1})$, $\epsilon_{k+1} \leftarrow \epsilon_k \alpha_{k+1}^{\beta_\epsilon}$, and $\eta_{k+1} \leftarrow \eta_k \alpha_{k+1}^{\beta_\eta}$.
    else
        Set $\mu_{k+1} \leftarrow \kappa_\mu \mu_k$ and $\hat{y}_{k+1} \leftarrow \hat{y}_k$.
        Set $\alpha_{k+1} \leftarrow \min(\gamma, \mu_{k+1})$, $\epsilon_{k+1} \leftarrow \epsilon_1 \alpha_{k+1}^{\alpha_\epsilon}$, and $\eta_{k+1} \leftarrow \eta_1 \alpha_{k+1}^{\alpha_\eta}$.
    end if
end for

Another widespread scheme that complies with Assumption 4.1 is that of Birgin and Martínez [3], see also [2, 19, 20]. Following this approach the multiplier estimates $\hat{y}_{k+1}$ are always updated, but *safeguarded* by projecting y_k onto a suitable compact set, so that the product $\mu_k \hat{y}_k$ vanishes as μ_k approaches zero.

6.2. Implementation details. In our implementation, several subproblem solvers are available depending on m, the number and types of hard constraints. If $m = 0$, subproblems are unconstrained and are solved by way of a traditional trust-region method [11, Chapter 6] in which steps are computed by the truncated conjugate gradient (CG) method [52]. Our implementation uses the `trunk` solver [36].

If $m > 0$ but the hard constraints are simple bounds, subproblems are bound constrained. We solve them with our own implementation of the trust-region method for bound-constrained problems of Lin and Moré [33], named `tron`, where steps are computed using a combination of projected-gradient steps and the truncated CG method. Whereas the original code of [33] requires explicit exact second-derivatives and uses a limited-memory factorization as preconditioner for CG, our implementation, also found in [36], does not require exact second derivatives, but does not use a preconditioner. In particular, the latter can be used with quasi-Newton approximations.

When $m > 0$ and c represents constraints that are more general than simple bounds, we appeal to a general constrained solver and currently support three of them: the filter interior-point method implemented in Ipopt [42, 55] and MadNLP [51], and the various solvers available as part of the KNITRO library [7, 37]. Currently, we use the *direct* interior-point method implemented in KNITRO [56]. The main reason for relying on interior-point methods is that our implementation warm starts them as the outer iterations progress in a manner similar to what is done in the NCL solver [34, 35, 38]. Naturally, it is also possible to use Ipopt, MadNLP or KNITRO when the subproblems are bound constrained or unconstrained.

An important feature of (1.2) is that h_μ is not twice differentiable in general. Thus, we rely on quasi-Newton approximations. Ipopt, MadNLP and KNITRO all support quasi-Newton approximations, but their management is hard-coded in those libraries in such a way that is crucial for the efficiency of the step computation—see, e.g., [56]. However, `trunk` and `tron` let users supply their own approximations. With

those two solvers, we implemented structured quasi-Newton approximations in the sense that we supply the Hessian approximation

$$\nabla^2 f(x_{k,j}) + B_{k,j},$$

where $B_{k,j} = B_{k,j}^\top$ is a limited-memory BFGS or SR1 approximation constructed from pairs

$$s_{k,j} := x_{k,j+1} - x_{k,j}, dy_{k,j} := \nabla h_\mu(F(x_{k,j}) + \mu_k \hat{y}_k) - \nabla h_\mu(F(x_{k,j-1}) + \mu_k \hat{y}_k),$$

where k denotes an iteration of Algorithm 4.1 and j one of the subproblem solver.

6.3. Numerical experiments. The broad applicability of Envelopt is demonstrated in this section with illustrative and real-world problems. We also compare the performance of our Envelopt[1] implementation with relevant solvers, such as MadNLP[2] [51], Ipopt [42, 55], SCS [40], Clarabel [26], NCL [34, 35], and ALPS [19].

6.3.1. Degenerate SDP. We consider an illustrative linear conic problem by Ramana [45, Example 4], which reads

$$\underset{x\in\mathbb{R}^2}{\text{minimize}} \quad -x_2 \qquad \text{subject to} \quad F(x) := \begin{bmatrix} 1-x_2 & 0 & 0 \\ 0 & -x_1 & -x_2 \\ 0 & -x_2 & 0 \end{bmatrix} \in \mathcal{S}_+^3 \tag{6.1}$$

where $\mathcal{S}_+^3$ denotes the cone of 3×3 positive semidefinite matrices. Due to lack of constraint qualifications, (6.1) and its dual problem exhibit a finite and positive duality gap, which makes the problem degenerate and hard to solve. Indeed, SCS [40], an operator-splitting conic solver, and Clarabel [26], an interior-point solver tailored to conic problems, struggle with (6.1).

For all tolerances $\epsilon \in \{10^{-3}, 10^{-4}, 10^{-5}, 10^{-6}\}$, SCS[3] hits the maximum number of iterations (10^5 by default) and returns with status `solved (inaccurate - reached max_iters)`, while Clarabel[4] stops after 38 iterations because of `insufficient progress`.

Problem (6.1) fits (1.1) with the linear cost in f, h as the indicator of $\mathcal{S}_+^3$, and F the affine matrix-valued operator appearing in (6.1). Then, subproblems (1.2) take the unconstrained form

$$\underset{x\in\mathbb{R}^2}{\text{minimize}}\, d - x_2 + \tfrac{1}{2}\mu^{-1} \operatorname{dist}(F(x) + \mu\hat{y} \mid \mathcal{S}_+^3)^2.$$

Envelopt is able to find ϵ-stationary points for (6.1), even with $\epsilon = 10^{-6}$, as reported in Table 6.1. MadNLP performs better than Ipopt and KNITRO in this case, tackling the subproblems down to tolerance $\epsilon = 10^{-6}$ and with fewer total inner iterations.

6.3.2. Simple MPCC. We showcase the benefits of integrating Envelopt and NCL, as discussed in section 5.2, by considering the following two-dimensional problem with a complementarity constraint, inspired by Scholtes [50]:

$$\underset{x\in\mathbb{R}^2}{\text{minimize}} \quad \|x-1\|^2 + \|x\|_1 \qquad \text{subject to} \quad x \geq 0, \quad x_1 x_2 \leq 0. \tag{6.2}$$

[1] `Envelopt v0.1.0`, `github.com/JuliaSmoothOptimizers/Envelopt.jl`.
[2] `MadNLP v0.9.2`, `github.com/MadNLP/MadNLP.jl`.
[3] `SCS v2.6.3`, `github.com/jump-dev/SCS.jl`.
[4] `Clarabel v0.11.1`, `github.com/oxfordcontrol/Clarabel.jl`.

TABLE 6.1
Solving (6.1) with Envelopt. Comparison for different subsolvers and tolerances ϵ in terms of number of iterations, and total number of subsolver iterations. A dash indicates solver failure.

subsolver	MadNLP			Ipopt			KNITRO		
tolerance ϵ	10^{-4}	10^{-5}	10^{-6}	10^{-4}	10^{-5}	10^{-6}	10^{-4}	10^{-5}	10^{-6}
iterations	9	7	7	6	19	-	6	4	-
subsolver iter.	61	72	69	61	125	-	111	75	-

This example also allows us to highlight the flexibility offered by Envelopt in the problem formulation, particularly the encoding of constraints as soft or hard.

The objective of (6.2) is convex, and the feasible set is connected but nonconvex. Moreover, standard constraint qualifications fail at the origin, which is feasible. There are two global minimizers, $(0.5, 0)$ and $(0, 0.5)$, with objective value 1.75. The origin is a local maximizer, with objective value 2.

Problem (6.2) fits the template (1.1) with $f(x) := \|x - 1\|^2$, $F := \mathbb{I}$, $h := \|\cdot\|_1$, nonnegativity bounds, and a nonlinear inequality constraint. We run Envelopt and its NCL extension for (6.2), along with Ipopt, MadNLP and NCL [35, 41] for the equivalent reformulation of (6.2) as the smooth nonlinear program

$$\underset{x \in \mathbb{R}^2}{\text{minimize}} \quad x_1^2 + x_2^2 - x_1 - x_2 + 2 \qquad \text{subject to} \quad x \geq 0, \quad x_1 x_2 \leq 0. \tag{6.3}$$

Starting from 100 random initial points, each solver terminated with a successful status and approximately at $(0, 0.5)$, $(0.5, 0)$, or $(0, 0)$ in at least 96 cases. A summary of the numerical performance is reported in Table 6.2. Only the NCL variants and Envelopt with Ipopt consistently find the global minimum; the other solvers returned the local maximizer $(0, 0)$ in 24–27% of cases.

TABLE 6.2
Solving a simple MPCC from 100 random initial points with Envelopt, NCL-Envelopt, and NLP solvers. Comparison for different problem formulations and subsolvers in terms of success rate in finding the global minimum and total number of subsolver iterations. Tolerance $\epsilon = 10^{-5}$.

problem	solver (subsolver)	global min rate	local min rate	subsolver iterations (median)	subsolver iterations (max)
(6.2)	Envelopt (MadNLP)	71%	27%	62.5	154
	Envelopt (Ipopt)	97%	0%	25	106
	NCL-Envelopt (MadNLP)	100%	0%	154	679
	NCL-Envelopt (Ipopt)	100%	0%	54	104
(6.3)	MadNLP	69%	27%	24	31
	Ipopt	76%	24%	26	48
	NCL	100%	0%	13	81
(6.4)	Envelopt (MadNLP)	73%	27%	142.5	194
	Envelopt (Ipopt)	73%	27%	57.5	98
	Envelopt (`tron`)	54%	46%	97.5	334
(6.5)	Envelopt (MadNLP)	100%	0%	79	121
	Envelopt (Ipopt)	100%	0%	67.5	91
	Envelopt (`tron`)	100%	0%	3553.5	4112
	Envelopt (`trunk`)	74%	7%	537	5503

We use (6.2) to inspect the effect of formulating constraints as hard or soft. As

discussed in Section 1, Envelopt offers the option of dealing with constraints directly in the subsolver, in contrast with other schemes that relax all constraints. Let us rewrite (6.2) as the equivalent problem

$$\text{(6.4)} \qquad \underset{x\in\mathbb{R}^2}{\text{minimize}} \quad \|x-1\|^2 + \|x\|_1 + \chi(x_1x_2 \mid \mathbb{R}_-) \qquad \text{subject to} \quad x \geq 0.$$

With this formulation, the nonlinear inequality constraint $x_1x_2 \leq 0$ is not passed to the subsolver but instead relaxed and treated with an augmented Lagrangian penalty in the Envelopt iterations; the corresponding subproblems (1.2) have only bound constraints. Another equivalent formulation of (6.2) is

$$\text{(6.5)} \qquad \underset{x\in\mathbb{R}^2}{\text{minimize}} \quad \|x-1\|^2 + \|x\|_1 + \chi(x \mid \mathbb{R}^2_+) + \chi(x_1x_2 \mid \mathbb{R}_-),$$

which leads to unconstrained subproblems. Note that there is no NCL counterpart for (6.4) and (6.5), since they lack nonlinear explicit constraints.

The performance of Envelopt on formulations (6.4) and (6.5) is summarized in Table 6.2, for comparison with (6.2). For each Envelopt subsolver, the formulation (6.2) with hard constraints requires significantly fewer iterations than (6.4) and (6.5). A tentative explanation is that simple constraints are better handled directly by the subsolver than by the Moreau envelope relaxation and smoothing in the outer iterations.

6.3.3. Low-rank matrix completion. Consider N points $p_1, \ldots, p_N \in \mathbb{R}^\ell$ and $P := \begin{bmatrix} p_1 & p_2 & \cdots & p_N \end{bmatrix} \in \mathbb{R}^{\ell\times N}$. Let $\Delta \in \mathbb{R}^{N\times N}$ denote the associated Euclidean distance matrix, i.e., $\Delta_{i,j} := \|p_i - p_j\|^2$ for all $i, j = 1, \ldots, N$. We wish to recover information on P based on partial knowledge of Δ. In particular, we assume that $\Omega \subset \{1, \ldots, N\}^2$ is a set of pairs such that only the entries $\Delta_{i,j}$, $(i, j) \in \Omega$, of Δ are known.

We lift the problem by introducing $B := P^\top P$ whose rank is, by construction, less than or equal to ℓ. We seek a symmetric matrix $B \in \mathbb{R}^{N\times N}$ that satisfies the distance constraints associated with the observations. Among these admissible matrices, those with minimum rank are preferred. We formulate the matrix completion task as De Marchi et al. [19, §4.5], for comparison with ALPS:

$$\text{(6.6)} \qquad \begin{aligned} \underset{B\in\mathbb{R}^{N\times N}}{\text{minimize}} \quad & \|B\|_* \\ \text{subject to} \quad & B_{i,i} - B_{i,j} - B_{j,i} + B_{j,j} = \Delta_{i,j} && \forall (i,j) \in \Omega \\ & B_{i,j} = B_{j,i} && \forall i, j \in \{1, \ldots, N\} \end{aligned}$$

where the nuclear norm $\|\cdot\|_* := \sum_i \sigma_i(\cdot)$ encourages B to have sparse singular values $\sigma_i(B)$, hence low rank [46].

The problem has the form (1.1) with $x = \text{vec}(B)$, $f = 0$, $h(x) = \|\text{mat}(x)\|_*$, $F(x) = x$, and $c(x)$ represents the explicit constraints above, where we denoted vec the operation that stacks the columns of a matrix into a vector, and mat its inverse. The problem has $n = N^2$ variables and $m = |\Omega| + N(N-1)/2$ constraints.

We generated random instances as in [19, §4.5], with $N = 10$, $\ell = 5$, and $|\Omega| = \lfloor N(N+1)/6 \rfloor$ observations, leading to problems (1.1) with 100 variables and 63 (linear equality) constraints. Table 6.3 summarizes the results obtained by invoking Envelopt on 100 random instances, each with a random initialization, for different tolerance levels.

Across all tolerances ϵ, the failures of Envelopt arise from the subsolver returning prematurely with an error message, always for subproblems (1.2) with $\mu > 10^{-3}$.

TABLE 6.3
Solving 100 random instances of (6.6) with Envelopt. Comparison for different subsolvers and tolerances ϵ in terms of success rate, number of iterations, and total number of subsolver iterations.

Envelopt subsolver tolerance ϵ		MadNLP 10^{-3}	MadNLP 10^{-4}	MadNLP 10^{-5}	MadNLP 10^{-6}	Ipopt 10^{-3}	Ipopt 10^{-4}	Ipopt 10^{-5}	Ipopt 10^{-6}
success rate		97%	97%	93%	90%	89%	87%	76%	60%
iterations	(median)	6	7	7	8	5	7	8	9
	(max)	9	12	14	16	8	8	9	10
subsolver iterations	(median)	14	14	14	13	113	167	217.5	266
	(max)	444	801	1068	854	592	1451	1019	1538

Rather than structural limitations of the Envelopt approach, this indicates the need for a better integration and tuning of subsolvers in our implementation. Conversely, Envelopt is significantly more iteration-efficient than ALPS, an augmented Lagrangian scheme without the partial minimization step, which required several thousands of projected-gradient steps, see [19, Fig. 5].

6.3.4. Exact penalty method. An interesting application of Envelopt is the solution of *exact penalty subproblems*, as mentioned in section 5.3. Envelopt can address the subproblem (5.2) for different choices of penalty function h. We illustrate this by invoking Algorithm 5.1 on two problems from the CUTEst benchmark set [27]. The small problem `zangwil3` has $n = 3$ variables and $m = 3$ equality constraints. The medium-sized problem `hs118` has $n = 32$ and $m = 17$ in the form (5.1).

The results reported in Table 6.4 indicate that Envelopt can effectively address the subproblems arising from the exact penalty method. Comparing subsolvers and penalty norms, it appears that Ipopt and $h := \|\cdot\|_1$ perform slightly better than other configurations.

TABLE 6.4
Solving CUTEst problems with Algorithm 5.1 and Envelopt for (5.2). Comparison for different choices of penalty norms and subsolvers, in terms of iterations for Algorithm 5.1 on (5.1), for Envelopt on (5.2), and the total number of subsolver iterations for each call to Envelopt.

Envelopt subsolver	number of iterations		problem `hs118` penalty norm h $\|\cdot\|_1$	$\|\cdot\|_2$	$\|\cdot\|_\infty$	problem `zangwil3` penalty norm h $\|\cdot\|_1$	$\|\cdot\|_2$	$\|\cdot\|_\infty$
MadNLP	penalty method		6	12	7	4	3	5
	Envelopt	(median)	6	9.5	6	2	3	3
		(max)	8	13	10	3	4	4
	subsolver	(median)	428.5	1435	439	4	6	7
		(max)	696	4134	2201	19	12	17
Ipopt	penalty method		7	7	6	4	6	2
	Envelopt	(median)	6	6	6	3	3	3.5
		(max)	8	14	9	4	3	4
	subsolver	(median)	356	359	338	8.5	7	14.5
		(max)	503	1550	719	16	13	19

6.3.5. Structured composite sum. We test Envelopt on the use case mentioned in Section 5.4. Given a convex regularizer h, a matrix $A \in \mathbb{R}^{p\times n}$, and a prox-friendly regularizer g, the proximal operator of $\varphi := g + h \circ A$ corresponds to solving the

problem (5.3), with $F(x) := Ax$, for some prescribed $\xi \in \mathbb{R}^n$ and $\lambda > 0$. Since the quadratic $f(x) := \frac{1}{2\lambda}\|x - \xi\|^2$ is convex smooth and h is convex prox-friendly, problem (5.3) can be tackled with the three-term splitting schemes Vũ–Condat [9, 54] and AFBA [30] when also g is convex prox-friendly. In contrast, Envelopt can address (5.3) also with nonconvex g, nonconvex f, and h composed with a nonlinear operator.

We choose $h \in \{\|\cdot\|_1, \|\cdot\|_2\}$, $g \in \left\{\|\cdot\|_1, \|\cdot\|_{1/2}^{1/2}, \|\cdot\|_0\right\}$, set $n = p = 10$, sample A, ξ from a normal distribution, and take values $\lambda \in \{10^{-3}, \dots, 10^3\}$ equidistant on a logarithmic scale. For the Envelopt subproblems we use the proximal-gradient methods R2DH [24, §7] and NMPG [14] through their implementation `R2DH` and `NMPG` in `RegularizedOptimization.jl` [25]. While both methods use a backtracking strategy for globalization, R2DH requires at least a fraction of Cauchy decrease and NMPG adopts an Armijo-type condition. They also employ different merit values for nonmonotone globalization, with max and average formulas, respectively. Envelopt with these subsolvers is compared against the primal-dual methods implemented as `VuCondat` and `AFBA` in `ProximalAlgorithms.jl` [53]. All solvers are given a tolerance $\epsilon = 10^{-5}$, maximum 10^5 iterations, and initialized with a random $x_0 \in \mathbb{R}^n$ (and zero dual). Although the solvers have different stopping conditions, our objective here is to show that Envelopt can solve problems that the other two solvers cannot.

Envelopt always found an ϵ-stationary point, with both subsolvers. Vũ–Condat and AFBA solved all instances with $g := \|\cdot\|_1$, and otherwise failed on about 40% of instances, regardless of h. In the convex setting, where Vũ–Condat and AFBA have convergence guarantees, Envelopt requires fewer iterations for small values λ, and slightly more for large values. The statistics reported in Table 6.5 indicate that Envelopt behaves consistently across different regularizers g and h, regardless of the subsolver.

TABLE 6.5

Computing the proximal operator of a composite sum (5.3). Comparison for different choices of functions h and g, in terms of total number of iterations. While Envelopt solved all instances, Vũ–Condat and AFBA failed on about 40% of instances with g nonconvex; their statistics are omitted in this case.

		$g = \|\cdot\|_1$		$g = \|\cdot\|_{1/2}^{1/2}$		$g = \|\cdot\|_0$	
	iterations	median	max	median	max	median	max
$\|\cdot\|_1$	Vũ–Condat	12	34403	-	-	-	-
$\|\cdot\|$	AFBA	59	1219	-	-	-	-
h	Envelopt (R2DH)	18	1363	13	953	13	789
h	Envelopt (NMPG)	10	362	7	864	5.5	625
$\|\cdot\|_2$	Vũ–Condat	12	4538	-	-	-	-
$\|\cdot\|$	AFBA	59	2817	-	-	-	-
h	Envelopt (R2DH)	16	140	9	127	7.5	138
h	Envelopt (NMPG)	7	134	4	97	5	93

7. Discussion and perspectives. Our implementation of Algorithm 4.1 could clearly be improved in a number of ways. One such way is to use generalized Hessians of the Moreau envelope as discussed by Dhingra et al. [22]. Whether generalized Hessians yield better performance in practice than quasi-Newton approximations remains to be seen. A second way is for cases where the proximal operator of h is not known in analytic form, but can be approximated by an iterative procedure that can be interrupted early. Inexact evaluations of the proximal operator imply inexact

evaluations of the Moreau envelope and of its gradient, but recent research indicates that such inexact evaluations can be incorporated into proximal algorithms while preserving their convergence and worst-case complexity properties [1, 16].

An obvious open question is whether any of the above generalizes to nonconvex h. In view of [49, Proposition 13.37], if h is *prox regular*, its proximal operator is locally single-valued and Lipschitz continuous for sufficiently small values of μ. Moreover, its Moreau envelope remains differentiable with gradient (2.3). Thus, borrowing ideas from [15], a generalization may be possible though it appears difficult in general to determine values of μ for which those properties occur.

REFERENCES

[1] N. Allaire, S. Le Digabel, and D. Orban. An inexact modified quasi-Newton method for nonsmooth regularized optimization. Cahier G-2025-73, GERAD, Montréal, Canada, 2025. cited on p. 21

[2] R. Andreani, E. G. Birgin, J. M. Martínez, and M. L. Schuverdt. On augmented Lagrangian methods with general lower–level constraints. *SIAM J. Optim.*, 18(4):1286–1309, 2008. cited on pp. 2 and 15

[3] E. G. Birgin and J. M. Martínez. *Practical Augmented Lagrangian Methods for Constrained Optimization*. SIAM, Philadelphia, USA, 4 2014. cited on pp. 2 and 15

[4] E. G. Birgin and J. M. Martínez. Complexity and performance of an augmented Lagrangian algorithm. *Optim. Method Softw.*, 35(5):885–920, 2020. cited on pp. 9 and 11

[5] E. G. Birgin, L. F. Bueno, and J. M. Martínez. Sequential equality-constrained optimization for nonlinear programming. *Comput. Optim. Appl.*, 65(3):699–721, 2016. cited on p. 2

[6] J. V. Burke. Descent methods for composite nondifferentiable optimization problems. *Math. Program.*, 33(3):260–279, 1985. cited on p. 1

[7] R. H. Byrd, J. Nocedal, and R. A. Waltz. KNITRO: An integrated package for nonlinear optimization. In G. di Pillo and M. Roma, editors, *Large-Scale Nonlinear Optimization*, pages 35–59. Springer Verlag, 2006. cited on p. 15

[8] E. Chouzenoux, M.-C. Corbineau, and J.-C. Pesquet. A proximal interior point algorithm with applications to image processing. *J. Math. Imaging Vis.*, 62(6):919–940, 2020. cited on p. 3

[9] L. Condat. A primal–dual splitting method for convex optimization involving Lipschitzian, proximable and linear composite terms. *J. Optim. Theory Appl.*, 158(2):460–479, 2013. cited on p. 20

[10] A. R. Conn, N. I. M. Gould, and Ph. L. Toint. A globally convergent augmented Lagrangian algorithm for optimization with general constraints and simple bounds. *SIAM J. Numer. Anal.*, 28(2):545–572, 4 1991. cited on pp. 2, 6, 11, 14, and 15

[11] A. R. Conn, N. I. M. Gould, and Ph. L. Toint. *Trust-Region Methods*. Number 1 in MPS-SIAM Series on Optimization. SIAM, Philadelphia, USA, 2000. cited on p. 15

[12] F. E. Curtis, X. Qu, and D. P. Robinson. A proximal-gradient method for solving regularized optimization problems with general constraints. Preliminary Report arXiv:2512.23166v1, 2025. cited on p. 3

[13] Y. Dai, X. Qu, and D. P. Robinson. A proximal-gradient method for equality constrained optimization. *SIAM J. Optim.*, 35(4):2654–2683, 2025. cited on p. 3

[14] A. De Marchi. Proximal gradient methods beyond monotony. *J. Nonsmooth Anal. Optim.*, 4 (10290), 2023. cited on p. 20

[15] A. De Marchi. Implicit augmented Lagrangian and generalized optimization. *J. Appl. Numer. Optim.*, 6(2):291–320, 2024. cited on p. 20

[16] A. De Marchi and A. Themelis. Proximal gradient algorithms under local Lipschitz gradient continuity. *J. Optim. Theory Appl.*, 194(3):771–794, 2022. cited on pp. 3 and 21

[17] A. De Marchi and A. Themelis. An interior proximal gradient method for nonconvex optimization. *Open J. Math. Optim.*, 5:1–22, 2024. cited on p. 21

[18] A. De Marchi and A. Themelis. A penalty barrier framework for nonconvex constrained optimization. *J. Nonsmooth Anal. Optim.*, 5(14585), 2025. cited on p. 3

[19] A. De Marchi, X. Jia, C. Kanzow, and P. Mehlitz. Constrained composite optimization and augmented Lagrangian methods. *Math. Program.*, 201(1):863–896, 2023. cited on p. 2

[20] A. De Marchi, T. Hoheisel, and P. Mehlitz. Augmented Lagrangian methods for fully convex composite optimization. Preliminary Report arXiv:2511.07117, 2025. cited on p. 2

[21] N. K. Dhingra, S. Z. Khong, and M. R. Jovanović. The proximal augmented Lagrangian method for nonsmooth composite optimization. *IEEE T. Autom. Control*, 64(7):2861–2868, 2019. cited on pp. 2, 5, 15, 16, 18, and 19

[22] N. K. Dhingra, S. Z. Khong, and M. R. Jovanović. A second order primal-dual method for nonsmooth convex composite optimization. *IEEE T. Autom. Control*, 67(8):4061–4076, 2022. cited on p. 15

[23] Y. Diouane, M. Gollier, and D. Orban. A nonsmooth exact penalty method for equality- cited on p. 2

constrained optimization: complexity and implementation. *SIAM J. Optim.*, 36(2):626–650, 2026. cited on p. 14

[24] Y. Diouane, M. L. Habiboullah, and D. Orban. A proximal modified quasi-Newton method for nonsmooth regularized optimization. *SIAM J. Optim.*, 36(2):534–563, 2026. cited on p. 20

[25] M. Gollier, M. L. Habiboullah, G. Leconte, R. Baraldi, A. De Marchi, D. Orban, and Y. Diouane. RegularizedOptimization.jl: A Julia framework for regularized and nonsmooth optimization. *J. Open Source Softw.*, 11(118):9344, 2026. cited on p. 20

[26] P. J. Goulart and Y. Chen. Clarabel: An interior-point solver for conic programs with quadratic objectives. *Math. Program. Comp.*, 2026. cited on p. 16

[27] N. I. M. Gould, D. Orban, and Ph. L. Toint. CUTEst: a constrained and unconstrained testing environment with safe threads for mathematical optimization. *Comput. Optim. Appl.*, 60(3): 545–557, 2015. cited on p. 19

[28] N. Hallak and M. Teboulle. An adaptive Lagrangian-based scheme for nonconvex composite optimization. *Math. Oper. Res.*, 48(4):2337–2352, 2023. cited on p. 2

[29] M. R. Hestenes. Multiplier and gradient methods. *J. Optim. Theory Appl.*, 4:303–320, 1969. cited on p. 2

[30] P. Latafat and P. Patrinos. Asymmetric forward–backward–adjoint splitting for solving monotone inclusions involving three operators. *Comput. Optim. Appl.*, 68(1):57–93, 2017. cited on p. 20

[31] G. Leconte and D. Orban. An interior-point trust-region method for nonsmooth regularized bound-constrained optimization. Cahier G-2024-17, GERAD, Montréal, Canada, 2024. cited on p. 3

[32] A. S. Lewis and S. J. Wright. A proximal method for composite minimization. *Math. Program.*, 158(1):501–546, 2016. cited on p. 3

[33] C. Lin and J. J. Moré. Newton's method for large bound-constrained optimization problems. *SIAM J. Optim.*, 9(4):1100–1127, 1999. cited on p. 15

[34] D. Ma, K. L. Judd, D. P. Orban, and M. A. Saunders. Stabilized optimization via an NCL algorithm. In M. Al-Baali, L. Grandinetti, and A. Purnama, editors, *Numerical Analysis and Optimization*, pages 173–191. Springer International Publishing, 2018. cited on pp. 13, 15, and 16

[35] D. Ma, D. Orban, and M. A. Saunders. A Julia implementation of Algorithm NCL for constrained optimization. In M. Al-Baali, A. Purnama, and L. Grandinetti, editors, *Numerical Analysis and Optimization, NAO-V, Muscat, Oman, January 2020*, volume 354 of *Springer Proceedings in Mathematics and Statistics*, pages 133–182. Springer, 2021. cited on pp. 13, 15, 16, and 17

[36] T. Migot, D. Monnet, D. Orban, and A. S. Siqueira. JSOSolvers.jl: Unconstrained and bound-constrained optimization solvers. *J. Open Source Softw.*, 11(117):9467, Jan. 2026. cited on p. 15

[37] T. Migot, D. Orban, A. Soares Siqueira, and contributors. NLPModelsKnitro.jl: A thin KNITRO wrapper for NLPModels, May 2026. cited on p. 15

[38] A. Montoison, F. Pacaud, M. Saunders, S. Shin, and D. Orban. MadNCL: A GPU implementation of algorithm NCL for large-scale, degenerate nonlinear programs. Preliminary Report arXiv:2510.05885, 2025. cited on p. 15

[39] J. J. Moreau. Proximité et dualité dans un espace hilbertien. *Bull. Soc. Math. Fr.*, 93:273–299, 1965. cited on p. 4

[40] B. O'Donoghue, E. Chu, N. Parikh, and S. Boyd. Conic optimization via operator splitting and homogeneous self-dual embedding. *J. Optim. Theory Appl.*, 169(3):1042–1068, 2016. cited on p. 16

[41] D. Orban, D. Ma, M. A. Saunders, A. Kapoor, and P.-E. Personnaz. NCL.jl: A Julia Implementation of Algorithm NCL for Constrained Optimization, May 2026. cited on p. 17

[42] D. Orban, A. Soares Siqueira, and contributors. NLPModelsIpopt.jl: A thin IPOPT wrapper for NLPModels, Apr. 2026. cited on

[43] T. Pietrzykowski. An exact potential method for constrained maxima. *SIAM J. Numer. Anal.*, 6(2):299–304, 1969. cited on pp. 15 and 16

[44] M. J. D. Powell. A method for nonlinear constraints in minimization problems. In R. Fletcher, editor, *Optimization*. Academic Press, London, 1969. cited on p. 13

[45] M. V. Ramana. An exact duality theory for semidefinite programming and its complexity implications. *Math. Program.*, 77(1):129–162, 1997. cited on p. 2

[46] B. Recht, M. Fazel, and P. A. Parrilo. Guaranteed minimum-rank solutions of linear matrix equations via nuclear norm minimization. *SIAM Rev.*, 52(3):471–501, 2010. cited on p. 16

[47] R. T. Rockafellar. Augmented Lagrangians and applications of the proximal point algorithm in convex programming. *Math. Oper. Res.*, 1(2):97–116, 1976. cited on p. 18

[48] R. T. Rockafellar. Convergence of augmented Lagrangian methods in extensions beyond nonlinear programming. *Math. Program.*, 199(1):375–420, 2023. cited on p. 2

[49] R. T. Rockafellar and R. J. Wets. *Variational Analysis*, volume 317. Springer Verlag, Berlin, 1998. cited on pp. 2 and 3

[50] S. Scholtes. Convergence properties of a regularization scheme for mathematical programs with complementarity constraints. *SIAM J. Optim.*, 11(4):918–936, 2001. cited on pp. 4, 8, 9, 12, and 21

[51] S. Shin, M. Anitescu, and F. Pacaud. Accelerating optimal power flow with GPUs: SIMD cited on p. 16

abstraction of nonlinear programs and condensed-space interior-point methods. *Electric Power Systems Research*, 236:110651, 2024. cited on pp. 15 and 16

[52] T. Steihaug. The conjugate gradient method and trust regions in large scale optimization. *SIAM J. Numer. Anal.*, 20(3):626–637, 1983. cited on p. 15

[53] L. Stella and contributors. ProximalAlgorithms.jl: Proximal algorithms for nonsmooth optimization in Julia, Jan. 2025. cited on p. 20

[54] B. C. Vũ. A splitting algorithm for dual monotone inclusions involving cocoercive operators. *Advances in Computational Mathematics*, 38(3):667–681, 2013. cited on p. 20

[55] A. Wächter and L. T. Biegler. On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming. *Math. Program.*, 106(1):25–57, 2006. cited on pp. 15 and 16

[56] R. Waltz, J. Morales, J. Nocedal, and D. Orban. An interior algorithm for nonlinear optimization that combines line search and trust region steps. *Math. Program.*, 107(3):391–408, 2006. cited on p. 15

Contents.