Decision-focused predictions via pessimistic bilevel optimization: complexity and algorithms

Víctor Bucarey[1,4][0000−0002−3043−8404], Sophia Calderón[2][0009−0002−8677−6733], Gonzalo Muñoz[3,4][0000−0002−9003−441X], and Frédéric Semet[5][0000−0002−1334−5417]

[1] Engineering Sciences Institute, Universidad de O'Higgins, Chile
[2] Engineering School, Universidad de O'Higgins, Chile
[3] Industrial Engineering Department, Universidad de Chile, Chile
[4] Instituto Sistemas Complejos de Ingeniería (ISCI), Chile
[5] Univ. Lille, CNRS, Inria, Centrale Lille, UMR 9189 CRIStAL, F-59000 Lille, France

Abstract. Dealing with uncertainty in optimization parameters is an important and longstanding challenge. Typically, uncertain parameters are predicted accurately, and then a deterministic optimization problem is solved. However, the decisions produced by this so-called *predict-then-optimize* procedure can be highly sensitive to uncertain parameters. In this work, we contribute to recent efforts in producing *decision-focused* predictions, i.e., to build predictive models that are constructed with the goal of minimizing a *regret* measure on the decisions taken with them. We begin by formulating the exact expected regret minimization as a pessimistic bilevel optimization model. Then, we establish NP-completeness of this problem in a heavily restricted case. Using duality arguments, we reformulate it as a non-convex quadratic optimization problem. Finally, leveraging the quadratic reformulation, we show various computational techniques to achieve empirical tractability. We report extensive computational results on shortest-path and bipartite matching instances with uncertain cost vectors. Our results indicate that our approach can improve training performance over the approach of Elmachtoub and Grigas (2022), a state-of-the-art method for decision-focused learning.

Keywords: predict and optimize · pessimistic bilevel optimization · non-convex quadratics

1 Introduction

Decision-making processes often involve uncertainty in input parameters, which is an important and longstanding challenge. Commonly, a two-stage approach is employed: firstly, training a machine learning (ML) model to estimate the uncertain input accurately, and secondly, using this estimate to tackle the decision task. This decision task is typically an optimization problem. Classical machine learning methods focus mainly on minimizing prediction errors on the parameters, disregarding the impact these errors might have on the subsequent optimization task. This approach overlooks how inaccurate predictions can negatively influence the optimization solution, potentially leading to decisions of poor quality.

In recent years, *decision-focused learning* (DFL) (also known as smart predict-then-optimize (SPO)) approaches, in which prediction and optimization tasks are integrated into the learning process, have received significant attention. In this approach, a machine learning model is specifically trained to enhance the effectiveness of the whole decision-making process. This involves combining, during the training phase, the prediction and the optimization in one single model.

In this work, we focus on linear optimization problems where the coefficients of the cost vector $c \in \mathbb{R}^n$ are unknown, but we have at hand a vector of correlated features x. Our goal is to train a parametric machine learning model $m(\omega, x)$, where ω is the vector of parameters of the machine learning model, so that the impact of the prediction error on the whole decision process is minimal. The typical measure of how a prediction performs in the decision process is the *regret*: the excess of cost in the optimization task caused by prediction errors.

As we will see in the following sections, finding the model $m(\omega, x)$ that minimizes the regret can be formulated as a pessimistic bilevel optimization problem. In fact, authors in Elmachtoub and Grigas (2022) define the unambiguous-SPO loss function as a pessimistic regret: among all the solutions that minimize predicted objective function, it penalizes the one that hurts the most when it is evaluated with the true cost vector.

Unfortunately, the complexity of finding these models and scalability are two major roadblocks to this DFL approach. Consequently, the literature has mainly focused on stochastic gradient-based approaches via approximations of the loss function through a surrogate convex loss function and/or solving a relaxed optimization problem (see Mandi et al. (2024)). Here, we follow a different path, take a step back, and focus on carefully studying the mathematical object behind the *exact* expected regret minimization. Our main hypothesis is that by understanding the mathematical object and designing new methods for solving the pessimistic bilevel optimization problem, better predictions and better algorithms can be developed.

The contributions of this work are the following: i. we formulate the expected regret minimization problem as a pessimistic bilevel optimization problem; ii. we prove that the problem is NP-complete in a heavily restricted setting, which improves on the known hardness results; iii. we reformulate the bilevel pessimistic formulation as a non-convex quadratically-constrained quadratic program (QCQP), which can be tackled by current optimization technology for moderate sizes; iv. we propose heuristics to improve the solution procedure based on the quadratic reformulation; 5) we conduct a comprehensive computational study on shortest path and bipartite matching instances. An early version of this work was published as a short paper in the conference proceedings of CPAIOR 2024 (Bucarey et al., 2024).

1.1 Problem setting

We consider a nominal optimization problem with a linear objective function:

$$P(c): \quad z^*(c) := \min_{v \in V} c^\top v \tag{1}$$

In this work, we restrict V to be a non-empty polytope, i.e. a non-empty bounded polyhedron. For a given c, we define $V^*(c)$ as the set of optimal solutions to (1).

In our setting, the value of $c \in \mathbb{R}^n$ is not known, but we have access to a dataset $\mathcal{D} = \{(x^i, c^i)\}_{i=1}^N$ with historical observations of c and correlated feature vectors $x \in \mathbb{R}^K$. Given these observations, and for a fixed set of parameters ω, we can empirically measure the sensitivity of the decisions given by the predictions using an average *regret*:

$$\begin{aligned} &\text{Regret}(\mathcal{D}, \omega) := \\ \max_{v} \quad & \frac{1}{N} \sum_{i \in [N]} \left(c^{i\top} v^i - z^*(c^i) \right), && (2) \\ \text{s.t.} \quad & v^i \in V^*(m(\omega, x^i)) \quad \forall i \in [N] && (3) \end{aligned}$$

Here, we are comparing the *true* optimal value $z^*(c^i)$ with the value $c^{i\top}v^i$, which is the "true cost" of a solution that is optimal for the prediction $m(\omega, x^i)$. To find the values of ω that minimize the regret (2), we must solve the following pessimistic bilevel optimization problem.

$$\min_{\omega} \max_{v^i \in V^*(m(\omega,x^i))} \quad \frac{1}{N} \sum_{i\in[N]} \left(c^{i\top}v^i - z^*(c^i)\right) \tag{4}$$

In this formulation, there are three optimization problems involved: i. the lower-level problem optimizing $m(\omega, x^i)^\top v$ over V; ii. over all the possible optimal solutions of the latter, take the one with the maximum (worst) regret. This corresponds to the pessimistic version of the bilevel formulation, defining the pessimistic regret. iii. Minimize the pessimistic regret using ω as a variable. In what follows, we use the notation $\hat{c}^i(\omega) := m(\omega, x^i)$.

1.2 Importance of the pessimistic approach

To illustrate why the pessimistic approach is relevant (as opposed to taking an optimistic approach), we consider the following example of a linear optimization problem with two variables:

$$\begin{aligned} P(c): \quad \min_{v} \quad & c_1v_1 + c_2v_2 \\ \text{s.t.} \quad & v_1 + v_2 \leq 1 \\ & v_1, v_2 \geq 0. \end{aligned}$$

Suppose that we have the two following observations of the true cost vectors with one single feature:

$$\begin{aligned} (x^1, c^1) &= \left(\begin{pmatrix}1\\0\end{pmatrix}, \begin{pmatrix}-4\\-3.5\end{pmatrix}\right), \\ (x^2, c^2) &= \left(\begin{pmatrix}0\\-2\end{pmatrix}, \begin{pmatrix}-2\\-3\end{pmatrix}\right), \end{aligned}$$

where each feature vector corresponds to the attribute regarding one coordinate. The goal is to estimate a simple regression model $m(\omega, x_j) = \hat{c}_j(\omega) = \omega_0 + \omega_1 x_j$, one per coordinate. Note that we can assume, without loss of generality, $|\omega_i| \leq 1$ for $i = 0, 1$.

The optimistic version is obtained dropping the "$\max_{v\in V^*(\hat{c}^i(\omega))}$" in (4); this effect can be equivalently achieved by setting $\omega_0 = \omega_1 = 0$, since $V^*((0,0)) = V$ (i.e., every $v \in V$ is optimal). The optimistic version then "guesses" the right optimal solution, and thus, the optimistic regret equals 0. However, the values for this solution evaluated in the pessimistic version are 4 for cost vector c^1 and 3 for c^2, leading to a regret of 3.5. Thus, this "optimistic regret" does not adequately measure the sensitivity of the decision with respect to the predictive model.

Note that the classical linear regression performs better than the optimistic approach in terms of pessimistic regret. Actually, the minimum squared error estimator of this regression is $\omega_0 = -3.156$, $\omega_1 = 0.253$, producing a regret of 0.5.

By solving the bilevel pessimistic formulation exactly, we obtain a regret of 0.25. An optimal solution here is $\omega_0 = -1$, $\omega_1 = 0.125$: with these values, the predicted costs are $\hat{c}^1 = (-0.875, -1)^\top$, $\hat{c}^2 = (-1, -1.125)^\top$. Note that this is not the only optimal solution for this example. If we consider $\omega_0 = -3.125$, $\omega_1 = 0.1$ also returns a pessimistic regret equal

to 0.25. We depict the different solutions and the observations in Figure 1, where the green stars correspond to each coordinate of observation 1 (x^1, c^1) and black squares correspond to observation 2 (x^2, c^2).

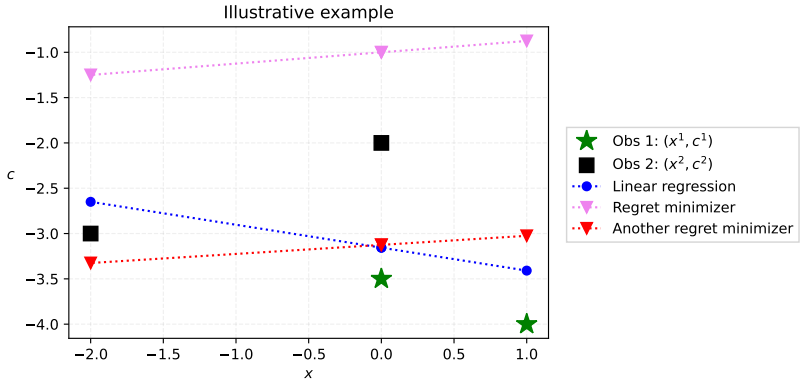

Fig. 1. Linear regression and regret minimizers in the numerical example of Section 1.2. Black squares and green stars correspond to the observations. Dotted lines correspond to the different solutions for the linear model $m(\omega, x_j)$.

1.3 Related work

Bilevel optimization. Bilevel optimization problems are hierarchical ones: they consist of an optimization problem (called upper-level or leader problem) that contains in its constraints optimality conditions of other problems, called lower-level or follower problems (Dempe and Zemkoho (2020)). In the presence of several optimal solutions at the lower-level problems, the way that the solution is chosen opens two approaches: the optimistic approach, also known as the strong solution, in which the solution chosen is the one that favors the upper-level optimization problem; and the pessimistic, also known as the weak solution, that chooses the one that worsens the most the upper-level objective.

Optimistic bilevel optimization is NP-hard even when the upper and lower-level problems are linear. Even having compact/efficient formulations can be a difficult task (Kleinert et al. (2020)). See Kleinert et al. (2021) for a survey on mixed-integer programming techniques in bilevel optimization. The necessity of establishing the pessimistic solution for bilevel optimization problems was first raised by Leitmann (1978) and studied in several articles (e.g., Loridan and Morgan (1996); Aboussoror and Mansouri (2005)). It is a common belief that the pessimistic approach is much more difficult than the optimistic one (Zeng (2020)). We refer to readers to Wiesemann et al. (2013); Liu et al. (2018, 2020) for comprehensive surveys of theoretical background and methods for bilevel pessimistic optimization problems. We note that the hardness results in bilevel optimization rule out the existence of *efficient general-purpose* techniques that could be applied in our setting.

Decision-focused learning. As mentioned earlier, and to the best of our knowledge, the existing methods developed for training decision-focused predictions are primarily based on estimating the gradient of how predictions impact a specific loss function. The main

challenge in these approaches is to estimate the changes in the optimal solution with respect to the model parameters, known as optimization mapping. According to the recent survey by Mandi et al. (2024), these methods can be classified into four categories: i. those that compute the gradient of the optimization mappings analytically, as in Amos and Kolter (2017); or ii. solving a smooth version of the optimization mapping, as seen in Wilder et al. (2019); Mandi and Guns (2020); Ferber et al. (2020); iii. ; or smoothing the optimization mapping by random perturbations Pogancic et al. (2020); Niepert et al. (2021) ; or iv. those solving a surrogate loss function that approximates regret, as discussed in Elmachtoub and Grigas (2022); Mulamba et al. (2021); Mandi et al. (2022).

Our study diverges from these approaches. Instead, to find regret-minimizing models, we leverage the pessimistic bilevel optimization formulation (4), and by employing duality arguments, we formulate it as a single-level non-convex quadratic model. A related method for minimizing the expected regret was proposed by Jeong et al. (2022): they cast the problem, not explicitly, as an optimistic bilevel optimization problem and use a symbolic reduction to solve this problem. However, as mentioned in Elmachtoub and Grigas (2022) and discussed in Section 1.2, casting this problem as an optimistic bilevel optimization problem may lead to undesirable predictions.

In terms of the complexity of (4), the results of Elmachtoub and Grigas (2022) imply that this problem is NP-hard. They show that minimizing the regret function generalizes the minimization of the 0-1 loss function, which is known to be NP-hard (Ben-David et al., 2003). Here, we improve on this hardness result by showing that in a very restrictive case, the problem remains NP-hard. We also complement this by showing membership in NP of a decision version of (4).

2 Complexity results

In this section, we focus on studying the complexity of (4) in a basic setting: when the predictive model $m(\omega, x)$ is linear. Technically, x are feature *vectors*, but slightly abusing notation, we will treat them as matrices in this section. This allows to write $m(\omega, x) = x\omega$ for a linear predictive model.

As mentioned in the previous section, the results of Elmachtoub and Grigas (2022) imply that (4) is NP-hard since it generalizes the minimization of the 0-1 loss function, which is known to be NP-hard (Ben-David et al., 2003).

In this section, we provide a sharper hardness result by showing that (4) is NP-hard even if we restrict to a linear nominal problem with a bounded feasible region and only *one* observation. We also show membership in NP, thus establishing NP-completeness of the problem.

Remark 1. The reduction in Elmachtoub and Grigas (2022) of the 0-1 loss minimization to (4) does not provide NP-hardness in the case we consider here since the former can be solved in polynomial time when there is only one observation (Blumer et al., 1989). This provides an interesting separation (complexity-wise) of problem (4) and the minimization of the 0-1 loss.

Remark 2. While many NP-hardness results are known in bilevel optimization (e.g. Audet et al. (1997); Hansen et al. (1992); Vicente et al. (1994); Buchheim (2023)), they are not directly applicable to (4) due to the following reasons:

- Problem (4) is a *pessimistic* bilevel optimization problem.
- The leader variables ω are unrestricted.

– The leader and follower variables interact non-linearly in the follower's problem, even when $m(\omega, x)$ is a linear function.

Adapting the existing bilevel optimization reductions to our setting seems like a highly non-trivial task, which is why we opted for a new custom proof.

Let us consider the following decision problem, associated with a simplified version of the optimization problem (4).

Definition 1. *The decision problem SIMPLE-REGRET(N) is defined as follows. Given $(c^i, x^i)_{i=1}^N$ a collection of rational vectors and matrices, a polytope V, and a rational number M decide if there exists ω such that*

$$\max_{v^i \in V^*(x^i\omega)} \quad \frac{1}{N} \sum_{i\in[N]} c^{i\top} v^i \quad \leq \quad M \tag{6}$$

We note that (6) is obtained from (4) by restricting $m(\omega, x) = x\omega$, and removing the constant part of the objective function $z^*(c^i)$; ; these terms can be computed efficiently, as they are only linear programs..

Theorem 1. *For every fixed N, SIMPLE-REGRET(N) is NP-hard.*

Proof. It suffices to show SIMPLE-REGRET(1) is NP-hard. To do so, we show that 3SAT can be reduced in polynomial time to SIMPLE-REGRET(1).

Fix an instance of 3SAT and take $\{C_j\}_{j=1}^m$ the set of clauses and u_k, $k \in [n]$, the set of variables. For each clause j we denote j_1, j_2, j_3 the indices of the 3 variables appearing in C_j. We also define $f_j(u_{j_\ell})$ as

$$f_j(u_{j_\ell}) = \begin{cases} u_{j_\ell} & \text{if } C_j \text{ contains } u_{j_\ell} \text{ as a literal} \\ 1 - u_{j_\ell} & \text{if } C_j \text{ contains } \overline{u_{j_\ell}} \text{ as a literal} \end{cases}$$

Using this, we let $V \subseteq \mathbb{R}^{n+m}$ be the polytope defined by

$$\begin{aligned} y_j &\geq f_j(u_{j_\ell}) \quad \forall j \in [m],\, \ell = 1, 2, 3 \\ y &\in [0,1]^m \\ u &\in [0,1]^n \end{aligned}$$

We note that the projection of this polytope onto the u variables is simply $[0,1]^n$. To define the instance of SIMPLE-REGRET(1) we set

$$c^1 = (\underbrace{0, \ldots, 0}_{n}, \underbrace{-1, \ldots, -1}_{m}) \in \mathbb{R}^{n+m}.$$

Additionally, we define x^1 as the following $(n+m) \times (n+m)$ matrix

$$x^1 = \begin{bmatrix} I_n & 0 \\ 0 & 0 \end{bmatrix}$$

where I_n is the $n \times n$ identity matrix and the zeros are stand-ins for all-zero matrices of appropriate dimensions. This way, $x^1\omega = (\omega_u, 0)$ where ω_u is the sub-vector of ω consisting of its first n entries. Finally, we set $M = -m$.

Using these definitions, this instance of SIMPLE-REGRET(1) outputs "Yes" if and only if there exists ω such that the value of the following bilevel problem is at most $-m$:

$$\max \quad -\sum_{j=1}^{m} y_j \tag{7a}$$

$$\text{s.t} \quad (u, y) \in \arg\min \left\{ \omega_u^\top u : \right. \tag{7b}$$

$$y_j \geq f_j(u_{j_\ell}) \quad \forall j \in [m],\, \ell = 1, 2, 3, \tag{7c}$$

$$y \in [0,1]^m, \tag{7d}$$

$$\left. u \in [0,1]^n \right\} \tag{7e}$$

Clearly, (7) can be constructed in polynomial time. We now show that 3SAT outputs "Yes" if and only there exists ω such that the value of (7) is at most $-m$.

If the 3SAT formula is satisfiable, we can take $\hat{u}$ an assignment of the u variables that makes all clauses true. With this, we define

$$(\omega_u)_k = \begin{cases} -1 \text{ if } \hat{u}_k = 1 \\ 1 \quad \text{if } \hat{u}_k = 0 \end{cases}$$

and we observe that (u, y) satisfies (7b) if and only if $u = \hat{u}$ and y satisfies that $y_j = 1$ when clause C_j is satisfied by $\hat{u}$, or $y_j \in [0, 1]$ otherwise. Since $\hat{u}$ was a satisfiable assignment, this means that all m entries of y will be 1, and thus

$$-\sum_{j=1}^{m} y_j \leq -m.$$

for any (u, y) satisfying (7b). Thus, SIMPLE-REGRET(1) outputs "Yes".

Let us now assume SIMPLE-REGRET(1) outputs "Yes" and take the corresponding ω. Define the following binary vector:

$$\hat{u}_k = \begin{cases} 1 \text{ if } (\omega_u)_k \leq 0 \\ 0 \text{ if } (\omega_u)_k > 0 \end{cases}$$

and accordingly, set

$$\hat{y}_j = \begin{cases} 1 & \text{if } f_j(u_{j_\ell}) = 1 \text{ for some } \ell = 1, 2, 3 \\ 0 & \text{otherwise} \end{cases}$$

Note that $\hat{y}_j$ is 1 if and only if the assignment $\hat{u}$ satisfies clause C_j. It is not hard to see that $(\hat{u}, \hat{y})$ satisfies (7b) (i.e., it is optimal for the lower-level problem), and thus it is feasible for the maximization problem. But since we know the maximum is at most $-m$, we must have

$$-\sum_{j=1}^{m} \hat{y}_j \leq -m.$$

This means there are at least m clauses satisfied, i.e., the 3SAT formula is satisfiable.

To finalize our proof, we note that in our reduction we construct instances satisfying $x^1\omega = (\omega_u, 0)$ and $c^1 = (0, \ldots, 0, -1, \ldots, -1)$. Hence, the 0-1 loss is always 1, regardless of ω_u. In particular, any ω_u is optimal for the 0-1 loss minimization problem.

Theorem 2. *For every fixed N, SIMPLE-REGRET(N) $\in$ NP*

Proof. It suffices to show that, for any "Yes" instance of SIMPLE-REGRET(N), there is a polynomially-sized ω such that (6) holds. Indeed, if we have such an ω, we can simply compute the optimal faces of each of the N lower-level problems (which can be done in polynomial time), and then solve the resulting maximization problem.

Consider an arbitrary instance of SIMPLE-REGRET(N), with $V = \{v : Av \leq b\}$, and suppose there exists $\hat{\omega}$ such that the value of

$$\begin{aligned} \max \quad & \frac{1}{N}\sum_{i=1}^{N}(c^i)^\top v^i \\ \text{s.t} \quad & v^i \in \arg\min \left\{(x^i\hat{\omega})^\top v \,:\, Av \leq b\right\} \end{aligned}$$

is less or equal than M. We will show we can modify $\hat{\omega}$ to have polynomial size and not change any of the argmins. Since we assume V is always non-empty and bounded (which can be verified in polynomial time), strong duality always holds. The dual of the i-th lower-level problem reads

$$\begin{aligned} \max \quad & (\rho^i)^\top b \\ \text{s.t} \quad & (\rho^i)^\top A = x^i\hat{\omega} \\ & \rho^i \leq 0. \end{aligned}$$

For each i, let us consider $\hat{\rho}^i$ in the relative interior of the optimal face of the dual. This means that the optimal face of the primal can be described as

$$F^i(\hat{\omega}) := \{v \in V \,:\, a_j^\top v = b_j,\, j \,:\, \hat{\rho}^i_j < 0\}$$

We claim that any $(\tilde{\omega}, \tilde{\rho})$ satisfying

$$(\tilde{\rho}^i)^\top A = x^i\tilde{\omega} \qquad \forall i \in [N] \tag{8a}$$

$$\tilde{\rho}^i_j \leq -1 \qquad \hat{\rho}^i_j < 0,\, \forall i \in [N] \tag{8b}$$

$$\tilde{\rho}^i_j = 0 \qquad \hat{\rho}^i_j = 0,\, \forall i \in [N] \tag{8c}$$

is such that $\arg\min\{(x^i\tilde{\omega})^\top v \,:\, Av \leq b\} = F^i(\hat{\omega})$ for every $i \in [N]$. Indeed, if there are such $\tilde{\omega}$ and $\tilde{\rho}$, then

$$\begin{aligned} & \arg\min\{(x^i\tilde{\omega})^\top v \,:\, Av \leq b\} \\ =\, & \arg\min\{(\tilde{\rho}^i)^\top Av \,:\, Av \leq b\} \\ =\, & \arg\min\left\{\sum_{j:\hat{\rho}^i_j<0} \tilde{\rho}^i_j a_j^\top v \,:\, Av \leq b\right\} \end{aligned}$$

The last objective function is lower bounded by $\sum_{j:\hat{\rho}^i_j<0} \tilde{\rho}^i_j b_j$ (recall that the $\tilde{\rho}$ values are non-positive). Moreover, this lower bound is met if and only if

$$a_j^\top v = b_j \qquad \forall j \,:\, \hat{\rho}^i_j < 0.$$

Therefore, we obtain that each $\tilde{\rho}^i$ is dual optimal for $\min\{(x^i\tilde{\omega})^\top v : Av \leq b\}$ and $\arg\min\{(x^i\tilde{\omega})^\top v : Av \leq b\} = F^i(\hat{\omega})$.

To conclude, note that (8) is always feasible, as a rescaled version of $(\hat{\omega}, \hat{\rho})$ satisfies the system. Since the coefficients of (8) are given by the entries of A, x^i, zeros, and ones, we conclude that there must be a $(\tilde{\omega}, \tilde{\rho})$ of *polynomial size* that satisfies the system.

Corollary 1. *For every fixed N, SIMPLE-REGRET(N) is NP-complete.*

As a last comment of this section, we note that membership in NP makes it unlikely for SIMPLE-REGRET(N) to be higher on the complexity hierarchy.

3 A non-convex quadratic reformulation

While the previous section establishes that finding a worst-case efficient method for solving (4) is unlikely, we can still hope to find methods with good practical performance.

In this section, we will apply duality arguments, supported by the assumption that V in problem (1) is non-empty and bounded, in order to obtain a more manageable formulation of the problem.

Under our assumptions, we can assume (1) has the following form:

$$\begin{aligned} \min_{v} \quad & c^\top v && \text{(9a)} \\ \text{s.t.} \quad & Av \geq b && \text{(9b)} \\ & v \leq 1 && \text{(9c)} \\ & v \geq 0. && \text{(9d)} \end{aligned}$$

Our predicted costs have the form $m(\omega, x)$ for some feature vector x and parameters ω (to be determined), and the terms $z^*(c^i)$ are constant. Thus, an equivalent formulation of our (pessimistic) regret-minimization problem is:

$$\begin{aligned} \min_{\omega} \max_{v} \quad & \frac{1}{N} \sum_{i \in [N]} (c^i)^\top v^i \\ \text{s.t.} \quad & v^i \in \arg\min_{\tilde{v}^i} \quad m(\omega, x^i)^\top \tilde{v}^i \\ & \qquad \text{s.t.} \quad A\tilde{v}^i \geq b \\ & \qquad\qquad \tilde{v}^i \leq 1 \\ & \qquad\qquad \tilde{v}^i \geq 0. \end{aligned} \tag{10}$$

3.1 Duality arguments

A common approach to solving optimistic bilevel problems involving convex lower-level problems is to reformulate them by replacing the lower-level problems with their optimality (Karush-Kuhn-Tucker or KKT) conditions (see Kleinert et al. (2021)). In this subsection, we follow the same type of argument twice to achieve a single-level reformulation of the pessimistic bilevel problem (10).

Since the feasible region of the lower-level problem in (10) is a non-empty polytope, which is unaffected by ω, we can apply LP duality and reformulate (10) as:

$$\min_{\omega} \max_{v,\rho,\alpha} \quad \frac{1}{N} \sum_{i\in[N]} (c^i)^\top v^i \tag{11a}$$

$$\text{s.t.} \quad Av^i \geq b \qquad \forall i \in [N] \tag{11b}$$

$$v^i \leq 1 \qquad \forall i \in [N] \tag{11c}$$

$$v^i \geq 0 \qquad \forall i \in [N] \tag{11d}$$

$$A^\top \rho^i + \alpha^i \leq m(\omega, x^i) \qquad \forall i \in [N] \tag{11e}$$

$$\rho^i \geq 0 \qquad \forall i \in [N] \tag{11f}$$

$$\alpha^i \leq 0 \qquad \forall i \in [N] \tag{11g}$$

$$m(\omega, x^i)^\top v^i \leq b^\top \rho^i + \mathbf{1}^\top \alpha^i \qquad \forall i \in [N] \tag{11h}$$

In this formulation (11b)-(11d) impose primal feasibility, (11e)-(11g) impose dual feasibility, and (11h) imposes strong duality[6].

The inner maximization problem of (11) is an LP, which is feasible for every value of ω. This is true because it represents a primal-dual system of a linear problem over a non-empty polytope, meaning it always has a solution. Moreover, since the objective function in (11a) involves only the v variables, which are bounded, we know that strong duality holds, allowing us to take the dual once again. This yields the following reformulation of (10):

$$\min_{\omega,\mu,\theta,\delta,\gamma} \sum_{i\in[N]} \left(b^\top \mu^i + \mathbf{1}^\top \theta^i + m(\omega, x^i)^\top \delta^i\right) \tag{12a}$$

$$\text{s.t.} \quad A^\top \mu^i + m(\omega, x^i)\gamma^i + \theta^i \geq \frac{1}{N} c^i \qquad \forall i \in [N] \tag{12b}$$

$$A\delta^i - b\gamma^i \geq 0 \qquad \forall i \in [N] \tag{12c}$$

$$-\gamma^i \mathbf{1} + \delta^i \leq 0 \qquad \forall i \in [N] \tag{12d}$$

$$\mu^i \leq 0 \qquad \forall i \in [N] \tag{12e}$$

$$\delta^i, \gamma^i, \theta^i \geq 0 \qquad \forall i \in [N] \tag{12f}$$

This is a single-level, non-convex quadratic problem.

3.2 Shortest path as a bounded linear program

In this work, and motivated by Elmachtoub and Grigas (2022), we consider the shortest path problem with unknown cost vectors. It is well known that this problem can be formulated as a linear program using a totally unimodular constraint matrix. However, the feasible region may not be bounded, as the underlying graph may have negative cycles, which correspond to extreme rays of the corresponding polyhedron.

To apply our framework (which relies on duality arguments), we need to assume no negative cycles exist for every possible prediction $m(\omega, x)$. For this reason, we make the following assumption.

[6] Strong duality is typically written as an equality constraint; however, the $\geq$ inequality always holds due to weak duality.

Assumption 1 *The underlying graph G defining the shortest path is directed acyclic.*

Under this assumption, we can safely formulate the shortest path problem as (9), and every extreme point solution will be a binary vector indicating the shortest path. Additionally, the inner maximization problem in (11) always has a binary optimal solution, as its feasible region is an extended formulation of the optimal face of the lower-level problem. From this discussion, we can guarantee that under Assumption 1, formulation (12) is valid for the shortest path problem with uncertain costs.

In our experiments, we also consider weighted bipartite instances. These instances can be directly formulated as a linear program with a bounded and integral feasible region, and thus, we do not need any extra assumption on them.

4 Solution methods

As mentioned earlier, in this work, we focus on the case when $m(\omega, x)$ is a linear model. Given its excellent performance, we will use the SPO+ loss function as a baseline.

4.1 SPO+

Authors in Elmachtoub and Grigas (2022) define the SPO+ loss function as a convex surrogate loss, which serves as an upper bound on the true regret. The authors derive an informative subgradient for this loss, which can be utilized in any subgradient descent-based approach. Additionally, they present a linear programming formulation for cases where the nominal problem is linear, and the model for estimation is also linear. This surrogate loss is defined as

$$\ell_{SPO}(c, m(\omega, x)) = \max_{v \in V} \left\{ (c - 2m(\omega, x))^\top v \right\} + 2m(\omega, x))^\top v^*(c) - z^*(c).$$

By using (9) as a nominal problem, the problem that minimizes the expected SPO+ loss can be cast as the following LP:

$$\begin{aligned}
\min_{\omega, \rho} \quad & \frac{1}{N} \sum_{i \in [N]} -b^\top \rho_i + \mathbf{1}^\top \theta^i + 2m(\omega, x^i)^\top v^*(c^i) && && \text{(13a)} \\
\text{s.t.} \quad & -\rho_i^\top A + \theta^i \geq c^i - 2m(\omega, x^i) && i \in [N] && \text{(13b)} \\
& \rho^i, \theta^i \geq 0 && i \in [N]. && \text{(13c)}
\end{aligned}$$

Variables ρ and θ are dual variables associated to constraints (9b) and (9c), respectively. We consider this formulation as a starting point and as a benchmark against which we will compare our methods.

4.2 Local Search

The intermediate reformulation presented in (11) can be seen as an unrestricted optimization problem of the form

$$\min_{\omega} \Lambda(\omega).$$

Here, $\Lambda(\omega)$ is a function that for each ω returns the optimal value of the inner maximization problem in (11). As mentioned above, for each ω, $\Lambda(\omega)$ is a feasible linear problem. We

Algorithm 1 Local-search based algorithm

1: **Input** Training data, Starting model parameters ω_0.
2: **Hyperparameters**: size of neighbourhood ϵ, sample size T, maximum number of iterations L.
3: Solve $\Lambda(\omega_0)$
4: **for** $i = 0, \ldots, N$ **do**
5: Sample T parameters in the neighbourhood of ω_i: $\omega_t \leftarrow \omega_i + \epsilon \cdot \mathcal{N}(0, 1)$
6: Compute $\Lambda(\omega_t) \quad \forall t \in \{1, \ldots, T\}$
7: Update $\omega_{i+1} \leftarrow \arg\min_{t=1,\ldots,T} F(\omega_t)$
8: **end for**

propose the following simple local-search-based heuristic: given an initial incumbent solution ω_0, we randomly generate T new solutions in a neighborhood of ω_0, evaluate the regret for each, and update the incumbent solution with the one with the smallest regret. We repeat these steps during L iterations. The procedure is detailed in Algorithm 1.

4.3 Penalization

Based on the ideas of Aboussoror and Mansouri (2005), we propose the following related formulation: we fix the variables γ^i in (12) to have all the same fixed value κ. This parameter κ is set before optimization and can be seen as a hyperparameter of the optimization problem. This results in the following model.

$$\min_{\omega,\mu,\theta,\delta} \quad \sum_{i\in[N]} \left(b^\top \mu^i + \mathbf{1}^\top \theta^i + m(\omega, x^i)^\top \delta^i\right) \tag{14a}$$

$$\text{s.t.} \quad A^\top \mu^i + m(\omega, x^i)\kappa + \theta^i \geq \frac{1}{N} c^i \qquad \forall i \in [N] \tag{14b}$$

$$A\delta^i - b\kappa \geq 0 \qquad \forall i \in [N] \tag{14c}$$

$$-\kappa \mathbf{1} + \delta^i \leq 0 \qquad \forall i \in [N] \tag{14d}$$

$$\mu^i \leq 0 \qquad \forall i \in [N] \tag{14e}$$

$$\delta^i, \theta^i \geq 0 \qquad \forall i \in [N] \tag{14f}$$

This formulation corresponds to a slice of the formulation (12) which removes all non-linearities of the constraints. Consequently, by adopting this approach, we solve an optimization problem with quadratic non-convex objective and linear constraints.

We remark that this approach results in a problem that is not a traditional penalization (which typically yields relaxations) but rather a restriction of the problem. The name "penalization", which is used in Aboussoror and Mansouri (2005), comes from a derivation of (14), which follows a similar approach to the one described in Section 3. The difference lies in penalizing (11h) using κ before taking the dual a second time.

4.4 Alternating direction method

We note that the non-convexities of formulation (12) come from products between the model parameter ω and dual variables γ or δ. Hence, if we fix either the ω variables, or variables γ and δ, we obtain linear programming problems. Specifically, if we fix ω to $\bar{\omega}$ the resulting

LP reads

$$\min_{\mu,\theta,\delta,\gamma} \sum_{i\in[N]} \left(b^\top \mu^i + \mathbf{1}^\top \theta^i + m(\bar{\omega}, x^i)^\top \delta^i\right) \tag{15a}$$

$$\text{s.t. } A^\top \mu^i + m(\bar{\omega}, x^i)\gamma^i + \theta^i \geq \frac{1}{N} c^i \qquad \forall i \in [N] \tag{15b}$$

$$A\delta^i - b\gamma^i \geq 0 \qquad \forall i \in [N] \tag{15c}$$

$$-\gamma^i \mathbf{1} + \delta^i \leq 0 \qquad \forall i \in [N] \tag{15d}$$

$$\mu^i \leq 0 \qquad \forall i \in [N] \tag{15e}$$

$$\delta^i, \gamma^i, \theta^i \geq 0 \qquad \forall i \in [N]. \tag{15f}$$

Analogously, fixing $\bar{\gamma}$ and $\bar{\delta}$ yields the LP

$$\min_{\mu,\theta,\omega} \sum_{i\in[N]} \left(b^\top \mu^i + \mathbf{1}^\top \theta^i + m(\omega, x^i)^\top \bar{\delta}^i\right) \tag{16a}$$

$$\text{s.t. } A^\top \mu^i + m(\omega, x^i)\bar{\gamma}^i + \theta^i \geq \frac{1}{N} c^i \qquad \forall i \in [N] \tag{16b}$$

$$\mu^i \leq 0,\ \theta^i \geq 0 \qquad \forall i \in [N] \tag{16c}$$

Based on these observations, we propose Algorithm 2 as a heuristic to find high-quality values for ω.

Algorithm 2 Alternating descent algorithm

1: **Input** Training data, Starting model parameters ω_0.
2: **Hyperparameters**: Maximum number of iterations L.
3: **for** $i = 0, \ldots, L$ **do**
4: Solve problem (15) using $\bar{\omega}_i$. Retrieve optimal variables $\bar{\delta}_i$ and $\bar{\gamma}_i$
5: Solve problem (16) using $\bar{\delta}_i$ and $\bar{\gamma}_i$ and retrieve a new vector of parameters $\bar{\omega}_{i+1}$
6: **end for**
7: Return $\bar{\omega}_L$

The following proposition follows directly from the definition of Algorithm 2.

Proposition 1. *The sequence $\Lambda(\bar{\omega}^0), \Lambda(\bar{\omega}^1), \ldots, \Lambda(\bar{\omega}^L)$ produced by Algorithm 2 is non-increasing.*

4.5 Regression bounds and valid inequalities

To prevent the solver from generating solutions with large coefficients in the non-convex QCQPs, and since the lower-level optimization problem is invariant to scalings of the objective, we can impose arbitrary bounds to the values of ω. Any bound is valid, but we avoided small numbers to prevent numerical instabilities.

Additionally, to improve the performance of the optimization solver, we included the following valid inequality to (12):

$$\sum_{i\in[N]} \left(b^\top \mu^i + \mathbf{1}^\top \theta^i + m(\omega, x^i)^\top \delta^i\right) \geq \frac{1}{N} \sum_{i\in[N]} z^*(c^i)$$

Variables		$N\,(\#\text{Nodes} + 2 \cdot \#\text{Edges} + 1) + K\#\text{Edges}$
Constraints	Linear (excluding var. bounds)	$N(\#\text{Nodes} + \#\text{Edges})$
	Non-convex	$N\#\text{Edges}$

Table 1. Description of the size of the exact reformulation.

This is a dual cut-off constraint. Its left hand side takes the same value as $\frac{1}{N}\sum_{i\in N}(c^i)^\top v^i$ in (10), and thus, by optimality of z^*, the inequality holds. This simple inequality provided considerable improvements in solver's performance.

5 Computational experiments

5.1 Computational set-up

Data generation. We consider an adaptation of the data generation process described in Elmachtoub and Grigas (2022) and Tang and Khalil (2024). The training data consists of $\{(x^i, c^i)\}_{i=1}^N$ generated synthetically the following way.

We consider two families of instances: small instances including values of $N = \{50, 100, 200\}$ where all methods can be run in moderate running times, and instances with $N = 1000$ to test the scalability of our approach. Each dataset is separated in 70% for training and 30% for testing. Feature vectors are generated by sampling them from a standard normal distribution (mean zero and standard deviation equals to 1). We generate cost vectors by first generating the parameters ω of the model –representing the true underlying model– and then using the following formula (see Elmachtoub and Grigas (2022) and Tang and Khalil (2024)):

$$c_a^i = \left[\frac{1}{3.5^{\text{Deg}}}\left(\frac{1}{\sqrt{K}}\left(\sum_{k=1}^{K}\omega_k x_{ak}^i\right) + 3\right)^{\text{Deg}} + 1\right]\cdot\varepsilon \tag{17}$$

where c_a^i is the component of the cost vector c^i corresponding to the arc a in the graph. The *Deg* parameter specifies the extent of model misspecification; as a linear model is used as a predictive model in the experiments, the higher the value of *Deg*, the more the relation between the features and cost coefficients deviates from a linear one (and thus, the larger the errors will be). Finally, a multiplicative noise term ε is sampled randomly from a uniform distribution in $[0.5, 1.5]$. We perform our experiments by considering the values of the parameter Deg in $\{2, 8, 16\}$.

We consider two nominal problems: shortest path over a directed acyclic grid graph of 5×5 nodes; a maximum weight matching on a bipartite graph of 13 and 12 nodes. To obtain graphs of similar sizes in both families of instances, we fix the number of edges of the bipartite matching graph to 40.

We remark that the exact reformulation requires solving a challenging non-convex problem, and that the underlying problem is NP-hard even for one observation (as established in Section 2); these are the main reasons why we focus on moderate instance sizes in this work. To provide some perspective, in Table 1 we provide the number of variables and constraints (linear and non-convex) that the non-convex quadratic problem has.

Algorithms. In our experiments, we consider the following sequence of steps to generate decision-focused predictions:

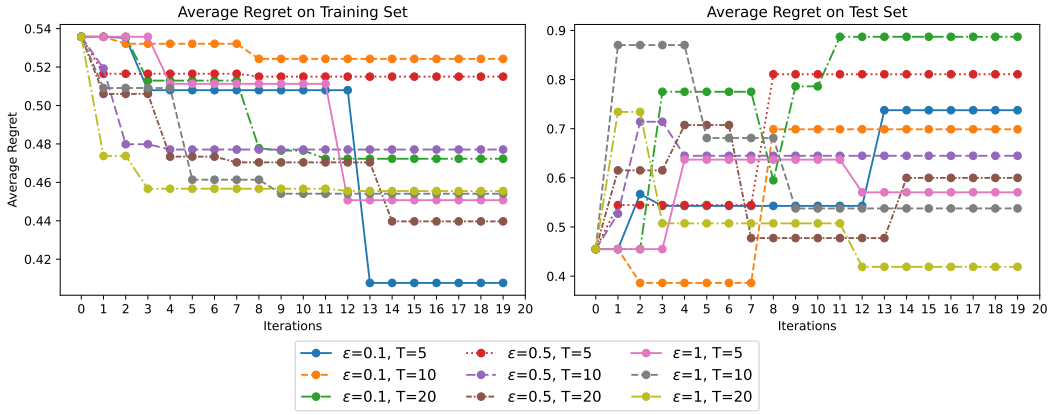

Fig. 2. Execution of local search on a 10×13 grid with different combinations of hyperparameters.

1. Generate an initial solution using the SPO+ method computed by the linear program (13) (SPO)
2. Improve the previous solution using Algorithm 1 (LS)
3. Either:
   (a) Solve (14) (Penalized) using Gurobi with the solution produced by LS as a warm start
   (b) Solve (12) (Exact) using Gurobi with the solution produced by LS as a warm start
   (c) Use the Alternating Method (Algorithm 2) with the solution produced by LS as starting point
   (d) Use the Alternating Method (Algorithm 2) with SPO as starting point

In terms of computational efficiency, Step 1 involves solving one LP; Step 2 solves a fixed number of LPs; Steps 3(a) and 3(b) are non-convex quadratic problems; and Steps 3(c) and 3(d) are sequences of LPs.

This generates six different combinations that we test below, with their names indicating the sequence: SPO, SPO-LS, SPO-LS-EXA, SPO-LS-PEN, SPO-LS-ALT, and SPO-ALT.

In all variants, we limit the entire suite of algorithms to one hour in the following way. We set a maximum time limit of 20 minutes for the SPO-LS part, leaving the remaining time for executing either EXA, PEN, or ALT. In our results, we provide the intermediary performance of SPO-LS. We remark that SPO can be solved in seconds. For instance, in our experiments, a typical instance with $N = 1000$ is solved in less than a minute.

Hyperparameters. In the implementation of Algorithm 1, we fixed the number of iterations to 20. To fix the rest of the hyperparameters in this algorithm, we tested various combinations to choose the best. In Figure 2, we show the regrets on a training and test set instance of the bipartite matching problem, when running the local search algorithm starting from the SPO+ solution. We also ran similar tests for the shortest path instances.

Figure 2 shows that $\epsilon = 1$ (size of the neighborhood) and $T = 20$ (samples per iteration) provide the best trade-off between test and training performance; we thus set these values for experiments involving bipartite matching. In the case of shortest path, we determined that $\epsilon = 0.1$ and $T = 20$ yields the best results.

In the case of the penalization method, to fix the value of κ, we also tested different variations ($\kappa \in \{0.1, 1, 10\}$) and observed that $\kappa = 0.1$ provided the best performance overall.

Hardware and Software We implemented all the aforementioned routines using Python 3.10. All non-convex quadratic models were solved using Gurobi 10.0.3 (Gurobi Optimization, LLC, 2023). All experiments were run single-threaded on a Linux machine with an Intel Xeon Silver 4210 2.2G CPU and 128 GB RAM.

5.2 Computational results

In this section, we compare the decision-focused predictions obtained using the aforementioned methods by evaluating the regrets they achieve. To ensure the regrets are displayed on the same scale, and following the approach in Elmachtoub and Grigas (2022), we use the *normalized* regret instead of reporting the direct regret (as defined in (2)). The normalized regret is defined as

$$\frac{\text{Regret}(\mathcal{D}, \omega)}{\sum_{i \in [N]} z^*(c^i)}.$$

Henceforth, when we reference the regret, we mean this normalized version.

Training set performance Tables 2 and 3 show the performance of the methods on the training set for small instances of shortest paths and maximum weight bipartite matching, respectively. In these tables, we use the regret returned by SPO (model (13)) as the baseline. The subsequent columns display the percentage decrease in regret achieved by our methods. The method with the best performance in terms of normalized regret is highlighted in color.

In the shortest path instances shown in Table 2, our alternating method, either by itself or in tandem with local search, achieves the best performance in terms of regret. We also observe that the penalized method may often improve (decrease) the regret, but in some instances, it can dramatically increase the regret (black bold numbers). This occurs whenever the starting point is not valid for the slice given by the chosen penalty factor in (14), causing the solver to reject the solution. Table 3 shows that for bipartite matching there are cases where the the penalized method was the best. However, the alternating direction method obtained the overall best results.

These results indicate that the performance of the alternating method considerably depends on its starting point. In most cases, starting from the solution provided by the local search algorithm yields a better outcome than starting with SPO. However, since the time for SPO, LS, and ALT is divided within the one-hour time limit, there are cases where starting with SPO and allowing the alternating method to run for the majority of the time results in better performance. Overall, we believe these are highly encouraging results: all methods we develop here are tailored for improving the training performance in moderate running times, to which we succeded for these challenging instances.

In what follows, we discuss two remaining aspects: test set performance and scalability.

Test set performance Regarding the performance on the test set, the results are less conclusive: the performance of SPO can be improved, but there is no clear dominance of one method.

In Table 4 we presents results for shortest path instances. We observe that in 14 out of 16 cases, one of the methods we propose here improves the SPO performance. The two

Table 2. Training set performance on small shortest path instances. The first three columns specify the instance parameters. The column labeled SPO provides the normalized regret achieved by the SPO+ method. Subsequent columns provide the improvements over SPO. Best regrets are highlighted in color.

N	Deg	Noise	SPO	SPO-LS	SPO-LS-EXA	SPO-LS-PEN	SPO-LS-ALT	SPO-ALT
50	2	0.0	0.035	0.0%	-14.3%	-17.1%	-42.9%	**-54.3%**
50	2	0.5	0.171	0.0%	-22.2%	-25.1%	**-43.3%**	**-43.3%**
50	8	0.0	0.133	0.0%	-13.5%	**+53.4%**	**-19.5%**	**-19.5%**
50	8	0.5	0.381	-23.6%	-45.1%	**+9.2%**	**-67.5%**	-64.8%
50	16	0.0	1.254	-66.4%	-70.6%	-6.7%	**-82.0%**	-75.4%
50	16	0.5	0.99	-59.5%	-64.5%	**+253.5%**	-71.2%	**-77.8%**
100	2	0.0	0.078	0.0%	0.0%	0.0%	**-32.1%**	**-32.1%**
100	2	0.5	0.195	0.0%	0.0%	0.0%	-37.9%	**-40.5%**
100	8	0.0	0.399	0.0%	0.0%	0.0%	**-20.8%**	**-20.8%**
100	8	0.5	0.572	-29.0%	-29.0%	-29.0%	**-47.6%**	-44.9%
100	16	0.0	3.243	-61.2%	-61.2%	-23.0%	**-69.1%**	-13.8%
100	16	0.5	1.945	-15.3%	-15.3%	**+76.1%**	**-28.0%**	-5.3%
200	2	0.0	0.089	0.0%	0.0%	0.0%	**-20.2%**	**-20.2%**
200	2	0.5	0.232	-0.9%	-0.9%	-0.9%	-11.2%	**-13.4%**
200	8	0.0	0.693	-16.7%	-16.7%	-16.7%	**-25.4%**	-4.0%
200	8	0.5	0.621	-1.4%	-1.4%	-1.4%	**-22.1%**	-16.1%
200	16	0.0	2.14	-60.5%	-60.5%	-60.5%	**-61.5%**	-2.9%
200	16	0.5	6.194	-79.4%	-79.4%	-79.4%	**-80.4%**	-0.2%

variants of the alternating method, which was the best in training performance, has the best performance in 8 out of 18 test instances, however, it can increase the test regret in comparison to SPO in some cases. The results for bipartite matching, which we show in Table 5, are slightly different. Here, the alternating method, in its various implementations, achieves the best performance in 9 out of 18 instances –one more than in the shortest path instances. On the other hand, in 6 instances, no method was able to improve the test performance of SPO.

Overall, from the test performance we see the following takeaways: (1) SPO can provide a strong solution very quickly, (2) in many instances, the SPO performance can be improved considerably and (3) which method yields the best improvement can vary significantly.

Other performance metrics and scalability To better understand the performance of the different algorithms, in this section we provide more details on their execution.

Both the exact method (EXA) and the penalized method (PEN) often reach the time limit; in Table 6 we report the final gap values reported by Gurobi. The penalized method, in particular, shows a tendency to return exceedingly large optimality gaps. This occurs because PEN sometimes rejects the initial feasible solution and struggles to find either good bounds or high-quality feasible solutions. In one extreme case, the penalized method failed to identify a feasible solution for the shortest path instance entirely. Also note that the bipartite matching instances always finish with larger gaps than their shortest path counterpart.

Table 3. Training set performance on small bipartite matching instances. The first three columns specify the instance parameters. The column labeled SPO provides the normalized regret achieved by the SPO+ method. Subsequent columns provide the improvements over SPO. Best regrets are highlighted in color.

N	Deg	Noise	SPO	SPO-LS	SPO-LS-EXA	SPO-LS-PEN	SPO-LS-ALT	SPO-ALT
50	2	0.0	0.065	0.0%	-12.3%	-46.2%	**-56.9%**	**-56.9%**
50	2	0.5	0.125	0.0%	-15.2%	-35.2%	**-46.4%**	**-46.4%**
50	8	0.0	0.344	0.0%	-12.8%	-24.4%	**-31.7%**	**-31.7%**
50	8	0.5	0.249	0.0%	-15.3%	-11.2%	**-39.4%**	**-39.4%**
50	16	0.0	0.316	-3.5%	-11.4%	**+5.1%**	**-52.2%**	-50.0%
50	16	0.5	0.344	-18.3%	-32.0%	-18.3%	**-74.1%**	-38.4%
100	2	0.0	0.091	0.0%	-4.4%	**-47.3%**	-29.7%	-29.7%
100	2	0.5	0.182	0.0%	0.0%	**-42.3%**	-27.5%	-27.5%
100	8	0.0	0.391	-10.7%	-10.7%	-10.7%	**-40.2%**	-20.2%
100	8	0.5	0.29	0.0%	-5.9%	0.0%	**-27.6%**	**-27.6%**
100	16	0.0	0.415	0.0%	-6.7%	0.0%	**-19.8%**	**-19.8%**
100	16	0.5	0.536	-12.9%	-12.9%	-12.9%	**-38.6%**	-19.8%
200	2	0.0	0.103	0.0%	0.0%	0.0%	**-15.5%**	**-15.5%**
200	2	0.5	0.207	0.0%	0.0%	0.0%	**-11.6%**	**-11.6%**
200	8	0.0	0.361	0.0%	0.0%	0.0%	**-8.9%**	**-8.9%**
200	8	0.5	0.387	0.0%	0.0%	0.0%	**-7.5%**	**-7.5%**
200	16	0.0	0.478	-17.6%	-17.6%	-17.6%	**-28.9%**	-17.6%
200	16	0.5	0.589	-25.3%	-25.3%	-25.3%	**-33.8%**	-28.5%

These results indicate that, even though EXA provides an exact reformulation of (10), the current solver technology is still unable to provide a provably optimal solution in these instances.

We also performed experiments on larger instances: with $N = 1000$. Based on the analysis above, we exclude from further consideration the methods involving the exact method (EXA) and the penalized method (PEN), since their performance is dramatically affected. We report our results in Tables 7 and 8. From these results, we see that our methods remain competitive, improving the performance of SPO in many cases. In one extreme case, we obtained an improvement of 25.9% in the case of the shortest path and 39.1% in the case of bipartite matching. However, overall, we note that these improvements start to become more modest than in the smaller instances. As before, the case of bipartite matching is harder to improve than shortest path.

Finally, to provide a more fleshed-out analysis on the alternating method and shed light on the reduced improvements for large instances, we show per-iteration statistics in Table 9. This table shows the number of iterations and average time-per-iteration of the two versions of the alternating method: starting from SPO directly or from local search.

From Table 9 we note that the number of iterations reduces dramatically as N increases: this can partially explain why we observed more moderate improvements with respect to SPO in $N = 1000$. We also note that, even if running local search reduces the number of

Table 4. Test set performance on small shortest path instances. The first three columns specify the instance parameters. The column labeled SPO provides the normalized regret achieved by the SPO+ method. Subsequent columns provide the improvements over SPO. Best regrets are highlighted in color.

N	Deg	Noise	SPO	SPO-LS	SPO-LS-EXA	SPO-LS-PEN	SPO-LS-ALT	SPO-ALT
50	2	0.0	0.105	0.0%	-2.9%	**-24.8%**	-11.4%	-4.8%
50	2	0.5	0.304	0.0%	**-15.1%**	+6.9%	-6.9%	-7.2%
50	8	0.0	0.45	0.0%	+8.2%	-34.2%	**-52.7%**	-34.4%
50	8	0.5	1.056	**-3.3%**	+8.4%	+7.4%	+16.5%	+5.2%
50	16	0.0	3.762	-74.9%	-73.1%	**-76.2%**	-72.8%	-74.9%
50	16	0.5	3.504	-9.6%	-7.4%	+42.9%	+6.2%	**-13.8%**
100	2	0.0	0.099	0.0%	0.0%	0.0%	**-1.0%**	**-1.0%**
100	2	0.5	**0.227**	0.0%	0.0%	0.0%	+7.0%	+11.0%
100	8	0.0	0.776	0.0%	0.0%	0.0%	+8.9%	**-13.9%**
100	8	0.5	0.707	-3.0%	-3.0%	-3.0%	**-24.9%**	+35.5%
100	16	0.0	3.882	+39.6%	+39.6%	**-12.6%**	-6.1%	-7.5%
100	16	0.5	13.645	+13.9%	+13.9%	**-8.2%**	+14.9%	+5.0%
200	2	0.0	0.099	0.0%	0.0%	0.0%	0.0%	**-1.0%**
200	2	0.5	0.238	+11.3%	+11.3%	+11.3%	+10.9%	**-5.0%**
200	8	0.0	0.591	**-2.7%**	**-2.7%**	**-2.7%**	+1.9%	+14.7%
200	8	0.5	1.023	**-23.3%**	**-23.3%**	**-23.3%**	-19.8%	-2.5%
200	16	0.0	**1.836**	+39.9%	+39.9%	+39.9%	+27.6%	+15.0%
200	16	0.5	3.384	+2.0%	+2.0%	+2.0%	-5.8%	**-12.1%**

iterations that ALT can perform (since the budget time is shared), the results in Tables 7 and 8 suggest that it may be worth running them in tandem.

We believe that these results suggest that a batch version of the alternating method can be a worthwhile for large instances. We strongly believe that this, along with other computational enhancements, can scale the strong result we observer in small instances.

6 Conclusions

In this work, we present an in-depth analysis of the optimization problem behind the training task of decision-focused learning. We improve on known complexity results by showing that producing regret-minimizing models is NP-complete even when only one observation is present. On the flip side, our proof of membership in NP indicates that this problem is not higher in the computational complexity hierarchy, unless the latter collapses. Additionally, we derive a non-convex quadratic optimization reformulation of the problem, whose structure we exploit empirically. Furthermore, by leveraging intermediate steps of the reformulation, we develop algorithms to address the problem at scale, including a local search procedure and an alternating direction method. These two algorithms only require solving linear programs at each iteration. When performed in tandem, they provide predictions with strong performance on both training and test sets for challenging shortest path instances and maximum weight matching problems with unknown cost/weight vectors.

Our results show that improvements (both in training and test instances) over SPO+ –a state-of-the-art method– can be achieved, thus effectively producing better decision-focused predictive models. The main drawback we observe is scalability: when the number of observations is large, the methods we show here, while still competitive, provide more moderate

Table 5. Test set performance on small bipartite matching instances. The first three columns specify the instance parameters. The column labeled SPO provides the normalized regret achieved by the SPO+ method. Subsequent columns provide the improvements over SPO. Best regrets are highlighted in color.

N	Deg	Noise	SPO	SPO-LS	SPO-LS-EXA	SPO-LS-PEN	SPO-LS-ALT	SPO-ALT
50	2	0.0	0.118	0.0%	+39.0%	-2.5%	**-3.4%**	+14.4%
50	2	0.5	0.26	0.0%	0.0%	-5.8%	-4.2%	**-10.8%**
50	8	0.0	0.349	0.0%	+30.4%	-6.9%	-6.3%	**-10.3%**
50	8	0.5	**0.474**	0.0%	+6.1%	+4.0%	+6.5%	+5.3%
50	16	0.0	0.781	-17.2%	-20.5%	-13.3%	+15.1%	**-17.4%**
50	16	0.5	0.596	**-21.8%**	-9.4%	**-21.8%**	-9.9%	+11.4%
100	2	0.0	0.118	0.0%	+51.7%	**-16.1%**	+5.9%	+5.9%
100	2	0.5	0.252	0.0%	0.0%	**-1.2%**	0.0%	-0.8%
100	8	0.0	**0.403**	+6.2%	+6.2%	+6.2%	+5.2%	0.0%
100	8	0.5	0.431	0.0%	-1.6%	0.0%	**-2.6%**	-2.1%
100	16	0.0	0.734	0.0%	0.1%	0.0%	**-18.5%**	-18.4%
100	16	0.5	0.455	+16.9%	+16.9%	+16.9%	+52.1%	**-1.3%**
200	2	0.0	0.126	0.0%	0.0%	0.0%	**-4.0%**	-3.2%
200	2	0.5	**0.229**	0.0%	0.0%	0.0%	+3.1%	+2.6%
200	8	0.0	**0.368**	0.0%	0.0%	0.0%	+6.5%	+6.5%
200	8	0.5	**0.401**	0.0%	0.0%	0.0%	+1.5%	+1.5%
200	16	0.0	0.739	-6.4%	-6.4%	-6.4%	**-19.4%**	+0.1%
200	16	0.5	**0.455**	+51.2%	+51.2%	+51.2%	+52.5%	+13.2%

improvements. However, our results show great potential of our non-convex optimization framework, and we strongly believe that after computational enhancements, such as a batch version of the alternating algorithm, these methods can achieve large-scale tractability.

Acknowledgments The authors would like to thank Paul Grigas for helpful comments on this work. Víctor Bucarey and Sophia Calderón were funded by the Fondecyt project number 11220864 and the ECOS project 230033. Gonzalo Muñoz was funded by Fondecyt project number 1231522. This work was also funded by the Chilean Agency for Research and Development (ANID) through PIA/PUENTE AFB230002. Frédéric Semet was partially funded by the ECOS project C23E08.

Table 6. Gap (percentage) returned by Gurobi at time limit for Exact and Penalized approaches.

			Shortest Path		Bipartite Matching	
N	**Deg**	**Noise**	**SPO-LS-EXA**	**SPO-LS-PEN**	**SPO-LS-EXA**	**SPO-LS-PEN**
50	2	0.0	3.4	2.8	6.0	3.7
50	2	0.5	11.7	11.4	11.9	8.8
50	8	0.0	10.4	17.3	42.9	52.4
50	8	0.5	17.3	46.1	26.7	33.7
50	16	0.0	27.0	10.9	43.8	155.6
50	16	0.5	26.0	18.9	30.6	436.7
100	2	0.0	7.2	8.5	10.0	5.0
100	2	0.5	16.3	18.6	22.3	11.7
100	8	0.0	28.5	42.9	53.6	138.1
100	8	0.5	28.9	42.4	37.5	101.6
100	16	0.0	55.7	43.4	70.8	930.8
100	16	0.5	62.2	91.2	87.8	840.0
200	2	0.0	8.2	9.3	11.5	14.0
200	2	0.5	18.7	18.8	26.2	34.7
200	8	0.0	36.6	45.9	56.4	121.8
200	8	0.5	38.0	49.5	63.1	122.6
200	16	0.0	45.8	-	65.0	1276.6
200	16	0.5	56.0	58.6	78.6	1348.2

Table 7. Training and test performance on large ($N = 1000$) shortest path instances

		Train Set				Test Set			
Deg	**Noise**	**SPO**	**SPO-LS**	**SPO-LS-ALT**	**SPO-ALT**	**SPO**	**SPO-LS**	**SPO-LS-ALT**	**SPO-ALT**
2	0.0	0.097	0.0%	**-2.1%**	**-2.1%**	0.1	0.0%	**-2.0%**	**-2.0%**
2	0.5	0.245	0.0%	**-1.2%**	**-1.2%**	0.268	0.0%	-0.7%	**-1.1%**
8	0.0	0.532	-6.4%	**-9.0%**	-0.8%	0.601	**-5.3%**	+0.7%	-0.7%
8	0.5	0.678	0.0%	**-4.4%**	**-4.4%**	0.699	0.0%	**-0.3%**	-0.1%
16	0.0	3.983	-24.4%	**-25.9%**	-0.5%	13.977	-43.2%	**-43.7%**	-0.0%
16	0.5	3.712	-11.3%	**-14.4%**	-0.2%	4.155	**-2.0%**	-1.8%	-0.4%

Table 8. Training and test performance on large ($N = 1000$) bipartite matching instances

		Train Set				Test Set			
Deg	**Noise**	**SPO**	**SPO-LS**	**SPO-LS-ALT**	**SPO-ALT**	**SPO**	**SPO-LS**	**SPO-LS-ALT**	**SPO-ALT**
2	0.0	0.117	0.0%	**-1.7%**	**-1.7%**	**0.118**	0.0%	0.0%	0.0%
2	0.5	0.225	0.0%	**-0.9%**	**-0.9%**	**0.242**	0.0%	+0.4%	+0.4%
8	0.0	0.406	0.0%	**-0.7%**	**-0.7%**	0.416	0.0%	**-0.5%**	**-0.5%**
8	0.5	0.411	0.0%	**-1.5%**	**-1.5%**	**0.449**	0.0%	0.0%	0.0%
16	0.0	0.673	-11.0%	**-11.6%**	-0.9%	**0.652**	+6.3%	+6.4%	+0.5%
16	0.5	0.604	-38.9%	**-39.1%**	-0.3%	0.697	-4.9%	**-5.0%**	+1.0%

Table 9. Detailed performance metrics of alternating method. The reported times are the average iteration time in the alternating method.

			Shortest Path		**Bipartite Matching**	
N	**Deg**	**Noise**	**SPO-LS-ALT** Iterations (Time)	**SPO-ALT** Iterations (Time)	**SPO-LS-ALT** Iterations (Time)	**SPO-ALT** Iterations (Time)
50	2	0.0	332 (1.60s)	456 (1.62s)	1922 (0.83s)	2308 (0.87s)
50	2	0.5	327 (1.63s)	471 (1.65s)	2197 (0.72s)	2262 (0.72s)
50	8	0.0	353 (1.63s)	448 (1.65s)	1717 (1.22s)	1759 (1.23s)
50	8	0.5	610 (1.09s)	583 (1.11s)	1770 (1.14s)	1812 (1.14s)
50	16	0.0	318 (1.62s)	435 (1.95s)	1792 (1.28s)	1688 (1.20s)
50	16	0.5	333 (1.67s)	449 (1.61s)	1379 (1.24s)	1754 (1.59s)
100	2	0.0	141 (6.15s)	182 (5.86s)	862 (2.39s)	872 (2.33s)
100	2	0.5	114 (6.99s)	198 (7.13s)	880 (2.41s)	852 (2.33s)
100	8	0.0	115 (6.59s)	202 (7.01s)	918 (2.25s)	896 (2.16s)
100	8	0.5	121 (7.09s)	177 (5.94s)	754 (2.85s)	763 (2.79s)
100	16	0.0	182 (4.46s)	208 (4.70s)	741 (2.88s)	802 (2.90s)
100	16	0.5	160 (4.28s)	223 (4.55s)	830 (2.69s)	796 (2.57s)
200	2	0.0	50 (24.01s)	73 (22.85s)	298 (6.34s)	397 (6.53s)
200	2	0.5	50 (16.73s)	84 (21.36s)	266 (7.44s)	348 (7.59s)
200	8	0.0	52 (20.27s)	77 (19.27s)	310 (6.89s)	342 (6.82s)
200	8	0.5	49 (18.96s)	78 (21.53s)	259 (8.07s)	320 (8.09s)
200	16	0.0	59 (13.48s)	116 (14.69s)	326 (6.66s)	349 (6.36s)
200	16	0.5	53 (15.19s)	83 (19.31s)	318 (6.69s)	341 (6.56s)
1000	2	0.0	32 (10.36s)	49 (10.41s)	173 (5.42s)	256 (5.45s)
1000	2	0.5	33 (7.61s)	50 (7.03s)	184 (4.39s)	276 (4.42s)
1000	8	0.0	33 (7.74s)	50 (7.59s)	167 (5.76s)	250 (5.78s)
1000	8	0.5	33 (7.88s)	50 (7.82s)	166 (6.08s)	245 (5.86s)
1000	16	0.0	33 (8.29s)	50 (8.47s)	151 (6.70s)	233 (7.32s)
1000	16	0.5	33 (8.90s)	49 (7.76s)	146 (6.61s)	230 (7.69s)

References

Aboussoror, A. and Mansouri, A. (2005). Weak linear bilevel programming problems: existence of solutions via a penalty method. *Journal of Mathematical Analysis and Applications*, 304(1):399–408.

Amos, B. and Kolter, J. Z. (2017). Optnet: Differentiable optimization as a layer in neural networks. In *International Conference on Machine Learning*, pages 136–145. PMLR.

Audet, C., Hansen, P., Jaumard, B., and Savard, G. (1997). Links between linear bilevel and mixed 0–1 programming problems. *Journal of optimization theory and applications*, 93:273–300.

Ben-David, S., Eiron, N., and Long, P. M. (2003). On the difficulty of approximately maximizing agreements. *Journal of Computer and System Sciences*, 66(3):496–514.

Blumer, A., Ehrenfeucht, A., Haussler, D., and Warmuth, M. K. (1989). Learnability and the Vapnik-Chervonenkis dimension. *Journal of the ACM (JACM)*, 36(4):929–965.

Bucarey, V., Calderón, S., Muñoz, G., and Semet, F. (2024). Decision-focused predictions via pessimistic bilevel optimization: a computational study. In *International Conference on the Integration of Constraint Programming, Artificial Intelligence, and Operations Research*, pages 127–135. Springer.

Buchheim, C. (2023). Bilevel linear optimization belongs to NP and admits polynomial-size KKT-based reformulations. *Operations Research Letters*, 51(6):618–622.

Dempe, S. and Zemkoho, A. (2020). Bilevel optimization. In *Springer optimization and its applications*, volume 161. Springer.

Elmachtoub, A. and Grigas, P. (2022). Smart "predict, then optimize". *Management Science*, 68(1):9–26.

Ferber, A., Wilder, B., Dilkina, B., and Tambe, M. (2020). Mipaal: Mixed integer program as a layer. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 1504–1511.

Gurobi Optimization, LLC (2023). Gurobi Optimizer Reference Manual.

Hansen, P., Jaumard, B., and Savard, G. (1992). New branch-and-bound rules for linear bilevel programming. *SIAM Journal on scientific and Statistical Computing*, 13(5):1194–1217.

Jeong, J., Jaggi, P., Butler, A., and Sanner, S. (2022). An exact symbolic reduction of linear smart Predict+ Optimize to mixed integer linear programming. In *International Conference on Machine Learning*, pages 10053–10067. PMLR.

Kleinert, T., Labbé, M., Ljubić, I., and Schmidt, M. (2021). A survey on mixed-integer programming techniques in bilevel optimization. *EURO Journal on Computational Optimization*, 9:100007.

Kleinert, T., Labbé, M., Plein, F. a., and Schmidt, M. (2020). There's no free lunch: on the hardness of choosing a correct big-M in bilevel optimization. *Operations research*, 68(6):1716–1721.

Leitmann, G. (1978). On generalized Stackelberg strategies. *Journal of optimization theory and applications*, 26(4):637–643.

Liu, J., Fan, Y., Chen, Z., and Zheng, Y. (2018). Pessimistic bilevel optimization: a survey. *International Journal of Computational Intelligence Systems*, 11(1):725–736.

Liu, J., Fan, Y., Chen, Z., and Zheng, Y. (2020). Methods for pessimistic bilevel optimization. In *Bilevel Optimization*, pages 403–420. Springer.

Loridan, P. and Morgan, J. (1996). Weak via strong Stackelberg problem: new results. *Journal of global Optimization*, 8:263–287.

Mandi, J., Bucarey, V., Tchomba, M. M. K., and Guns, T. (2022). Decision-focused learning: Through the lens of learning to rank. In *International Conference on Machine Learning*, pages 14935–14947. PMLR.

Mandi, J. and Guns, T. (2020). Interior point solving for lp-based prediction+ optimisation. *Advances in Neural Information Processing Systems*, 33:7272–7282.

Mandi, J., Kotary, J., Berden, S., Mulamba, M., Bucarey, V., Guns, T., and Fioretto, F. (2024). Decision-focused learning: Foundations, state of the art, benchmark and future opportunities. *Journal of Artificial Intelligence Research*, 80:1623–1701.

Mulamba, M., Mandi, J., Diligenti, M., Lombardi, M., Lopez, V. B., and Guns, T. (2021). Contrastive losses and solution caching for predict-and-optimize. In *30th International Joint Conference on Artificial Intelligence (IJCAI-21): IJCAI-21*, page 2833. International Joint Conferences on Artificial Intelligence.

Niepert, M., Minervini, P., and Franceschi, L. (2021). Implicit mle: backpropagating through discrete exponential family distributions. *Advances in Neural Information Processing Systems*, 34:14567–14579.

Pogancic, M. V., Paulus, A., Musil, V., Martius, G., and Rolinek, M. (2020). Differentiation of blackbox combinatorial solvers. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*.

Tang, B. and Khalil, E. B. (2024). PyEPO: A Pytorch-based end-to-end predict-then-optimize library for linear and integer programming. *Math. Prog. Comp.*

Vicente, L., Savard, G., and Júdice, J. (1994). Descent approaches for quadratic bilevel programming. *Journal of Optimization theory and applications*, 81(2):379–399.

Wiesemann, W., Tsoukalas, A., Kleniati, P.-M., and Rustem, B. (2013). Pessimistic bilevel optimization. *SIAM Journal on Optimization*, 23(1):353–380.

Wilder, B., Dilkina, B., and Tambe, M. (2019). Melding the data-decisions pipeline: Decision-focused learning for combinatorial optimization. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 1658–1665.

Zeng, B. (2020). A practical scheme to compute the pessimistic bilevel optimization problem. *INFORMS Journal on Computing*, 32(4):1128–1142.