The bilevel continuous knapsack problem with uncertain follower’s objective*

Christoph Buchheim, Dorothee Henke
Fakultät für Mathematik, TU Dortmund, Germany
christoph.buchheim,dorothee.henke@math.tu-dortmund.de

We consider a bilevel continuous knapsack problem where the leader controls the capacity of the knapsack and the follower’s profits are uncertain. Adopting the robust optimization approach and assuming that the follower’s profits belong to a given uncertainty set, our aim is to compute a worst case optimal solution for the leader. We show that this problem can be solved in polynomial time for both discrete and interval uncertainty, but that the same problem becomes NP-hard when each coefficient can independently assume only a finite number of values. In particular, this demonstrates that replacing uncertainty sets by their convex hulls may change the problem significantly, in contrast to the situation in classical single-level robust optimization. For general polytopal uncertainty, the problem again turns out to be NP-hard, and the same is true for ellipsoidal uncertainty even in the uncorrelated case. We also address the stochastic problem variant and show that optimizing the leader’s objective function is \#P-hard in the case of independently uniformly distributed profits. All presented hardness results already apply to the evaluation of the leader’s objective function.

Keywords: bilevel optimization, robust optimization, interval order

1 Introduction

Bilevel optimization has received increasing attention in the last decades. The aim is to model situations where certain decisions are taken by a so-called leader, but then one or more followers optimize their own objective functions subject to the choices of the leader. The follower’s decisions in turn influence the leader’s objective, or even the feasibility of her decisions. The objective is to determine an optimal decision from the leader’s perspective. In general, bilevel optimization problems are very hard to solve.

*A short abstract containing some of the results presented here has appeared in the proceedings of the 17th Cologne Twente Workshop (CTW 2019). This work has partially been supported by Deutsche Forschungsgemeinschaft (DFG) under grant no. BU 2313/2.
Even in the case that both objectives and all constraints are linear, the bilevel problem turns out to be strongly NP-hard in general [11]. Several surveys and books on bilevel optimization have been published recently, e.g., [5, 7, 8].

Our research is motivated by complexity questions. We investigate whether – and in which cases – taking uncertainty into account renders the problem significantly harder, where we focus on the robust optimization approach. In this approach, the uncertain parameters are specified by so-called uncertainty sets which contain all possible (or likely) scenarios; the aim is to find a solution that is feasible in each of these scenarios and that optimizes the worst case.

Due to the hardness of deterministic bilevel optimization, it is not surprising that relatively few articles dealing with bilevel optimization problems under uncertainty, in particular using the robust optimization approach, have been published so far. In [4], the authors consider bilevel problems with linear constraints and a linear follower’s objective, while the leader’s objective is a polynomial. The robust counterpart of the problem, with interval uncertainty in the leader’s and the follower’s constraints, is solved via a sequence of semidefinite programming relaxations. In [19], a bilevel linear programming problem with the follower’s objective vector lying in a polytopal uncertainty set is considered. A vertex enumeration approach is combined with a global optimality test based on an inner approximation method. Similar models have also been considered in a game-theoretic context; see, e.g., [15]. More literature has been published on the stochastic optimization approach to bilevel optimization under uncertainty, where (some of) the problem’s parameters are assumed to be random variables and the aim is to determine a solution optimizing the expected objective value; see, e.g., [13] and the references therein.

In the following, we only consider uncertainty in the follower’s objective function. This is a very natural setting in bilevel optimization, as in practice the leader often does not know the follower’s objective function precisely. Assuming, on the contrary, that the leader knows the follower’s objective function precisely, but that the follower does not necessarily choose his optimal solution, we arrive at a closely related mathematical model. The latter interpretation corresponds to situations where the follower acts with so-called bounded rationality; this framework is also known as decision variable uncertainty.

Even in the case of objective uncertainty, in classical single-level robust optimization, some classes of uncertainty sets may lead to substantially harder robust counterparts, e.g., finite uncertainty sets in the context of combinatorial optimization [18]. In other cases, the robust counterparts can be solved by an efficient reduction to the underlying certain problem. This is true in particular for the case of interval uncertainty, where each coefficient may vary independently within some given range. In this case, each interval may be replaced by one of its endpoints, depending on the direction of optimization, so that the robust counterpart is not harder than the underlying certain problem. For an overview of complexity results in robust combinatorial optimization under objective uncertainty, we refer the reader to the recent survey [2] and the references therein.

However, we will show that the situation in case of interval uncertainty is more complicated in bilevel optimization. We concentrate on a bilevel continuous knapsack problem where the leader only controls the capacity. Without uncertainty, this problem is easy
to solve. However, if the follower’s objective is uncertain, the problem becomes much more involved. It turns out that this approach requires to deal with partial orders, more precisely, with the interval orders induced by the relations between the follower’s profit ranges. Adapting an algorithm by Woeginger [21] for some precedence constraint knapsack problem, we show that the problem can still be solved in polynomial time. We also discuss why the case of finite uncertainty sets is tractable as well. For many other types of uncertainty sets, the robust bilevel continuous knapsack problem turns out to be NP-hard, and the same is true even for the problem of evaluating the leader’s objective function, i.e., the adversary’s optimization problem. We are able to show this for polytopal and ellipsoidal uncertainty sets, among others.

Our results also emphasize another significant difference to classical robust optimization: in general, it is not possible anymore to replace the uncertainty set by its convex hull and thus assume convexity without loss of generality. In fact, when restricting the possible scenarios in the interval case to the endpoints of the intervals, we show that the problem turns NP-hard. More precisely, the problem is NP-hard when the input consists of a finite set of realizations for each coefficient and these realizations arise independently.

We also discuss the complexity of the problem in the stochastic optimization setting, where – instead of an uncertainty set – a distribution of the unknown objective coefficient vector is given. The aim is then to optimize the leader’s expected solution value, instead of the worst case. While the problem is again easy to solve as long as the support is finite and given explicitly as part of the input, the problem becomes #P-hard when assuming that every coefficient of the follower’s objective is uniformly distributed either on a set with two elements or on an interval. Since the latter case corresponds to the interval uncertainty case in the robust setting, we conclude that the stochastic problem variant is harder in this case than the robust variant.

The remainder of this paper is organized as follows. In Section 2, we introduce and discuss the certain variant of the problem. We then settle the cases of finite uncertainty sets in Section 3 and interval uncertainty in Section 4. The discrete uncorrelated case, where each coefficient varies in a finite set independently, is discussed in Section 5. Uncertainty sets defined as simplices (Section 6) or by norms (Section 7) are discussed next. Finally, we investigate the stochastic approach in Section 8. Section 9 concludes.

2 Underlying certain problem

2.1 Problem formulation

We first discuss the deterministic variant of the bilevel optimization problem under consideration, in which the follower solves a continuous knapsack problem, while the leader determines the knapsack’s capacity and optimizes another linear objective function than the follower. This problem is also discussed in [8], but we replicate the formulation and the algorithm here for sake of completeness. Note that we only deal with the continuous problem version here, which is solvable in polynomial time in the deterministic variant. For the variant of this problem that contains a binary instead of a continuous knapsack.
sack problem, as well as other binary bilevel knapsack problems, we refer to [3] and the references therein.

First recall that an important issue in bilevel optimization is that the follower’s optimum solution is not necessarily unique, but the choice among the optimum solutions might have an impact on the leader. The two main approaches here are the optimistic and the pessimistic one. In the former case, the follower is assumed to decide in favor of the leader, whereas in the latter case, he chooses the optimum solution that is worst for the leader. For more details, see e.g., [8]. While often only the optimistic approach is considered, we will focus on the pessimistic one in this paper, since it combines more naturally with the concept of robustness that will be added to the problem later on. Indeed, both the pessimistic view of the follower and the robustness force the leader to consider the worst case with respect to some set of choices. However, all our results hold for the optimistic approach as well, making only small changes to the proofs necessary, which will be sketched for every result.

The pessimistic version of the problem, without uncertainty, can be formulated as follows, where the minimization over $x$ represents the leader’s pessimism about the follower’s choice among his optimum solutions:

$$\max_{b \in [b^-, b^+]} \min_x d^\top x - \delta b$$

subject to:

- $x \in \text{argmax } c^\top x$
- $a^\top x \leq b$
- $0 \leq x \leq 1$

(P)

The leader’s only variable is $b \in \mathbb{R}$, which can be considered as the knapsack’s capacity. The follower’s variables are $x \in \mathbb{R}^n$, i.e., the follower fills the knapsack with a subset of the objects, where also fractions are allowed. The item sizes $a \in \mathbb{R}^n$, the follower’s item values $c \in \mathbb{R}^n > 0$, the capacity bounds $b^-, b^+ \in \mathbb{R}$ as well as the leader’s item values $d \in \mathbb{R}^n_{\geq 0}$ and a number $\delta \geq 0$ are given. The latter can be thought of as a price the leader has to pay for providing one unit of knapsack capacity. We may assume $0 \leq b^- \leq b^+ \leq \sum_{i=1}^n a_i$ and $a > 0$.

For a given leader’s choice $b$, due to the assumptions $0 \leq b \leq \sum_{i=1}^n a_i$ and $c > 0$, every optimum solution of the follower’s problem satisfies $a^\top x = b$. This can be used to show that the following three ways to formulate the leader’s objective function are equivalent:

(a) $d \in \mathbb{R}^n$ and $\delta \in \mathbb{R}$, the most general variant,

(b) $d \in \mathbb{R}^n_{\geq 0}$ and $\delta \geq 0$, as in the problem formulation above, and

(c) $d \in \mathbb{R}^n$ and $\delta = 0$, which we will use in the following.

Clearly, (b) and (c) are special cases of (a). An objective function $d^\top x - \delta b$ of type (a) can be reformulated in the form of (c):

$$d^\top x - \delta b = d^\top x - \delta (a^\top x) = (d - \delta a)^\top x$$
and similarly in the form of (b):

\[ d^\top x - \delta b = (d - \delta a)^\top x = (d - \varepsilon a - (\delta - \varepsilon)a)^\top x = (d - \varepsilon a)^\top x - (\delta - \varepsilon)b, \]

where \( \varepsilon := \min\{\delta, \min_{i \in \{1, \ldots, n\}} \frac{d_i}{a_i}\} \). This proves that all three formulations are equivalent. Note that this equivalence still holds for the uncertain problem versions considered later on. As already indicated, we will use the third formulation from now on because it is the most compact one, i.e., we omit \( \delta \) and assume \( d \in \mathbb{R}^n \).

2.2 Solution algorithm

The follower solves a continuous knapsack problem with fixed capacity \( b \). This can be done, for example, using Dantzig’s algorithm [6]: by first sorting the items, we may assume

\[ c_1 a_1 \geq \cdots \geq c_n a_n. \]

The idea is then to pack the items into the knapsack in this order until it is full. More formally, if \( b = \sum_{i=1}^n a_i \), everything can be taken, so the optimum solution is \( x_i = 1 \) for all \( i \in \{1, \ldots, n\} \). Otherwise, we consider the critical item

\[ k := \min\{j \in \{1, \ldots, n\} : \sum_{i=1}^j a_i > b\} \]

and an optimum solution is given by

\[
 x_j := \begin{cases} 
 1 & \text{for } j \in \{1, \ldots, k-1\} \\
 \frac{1}{a_k}(b - \sum_{i=1}^{k-1} a_i) & \text{for } j = k \\
 0 & \text{for } j \in \{k+1, \ldots, n\}.
\end{cases}
\]

We now turn to the leader’s perspective. As only the critical item \( k \), but not the sorting depends on \( b \), the leader can just compute the described order of items by the values \( \frac{c_i}{a_i} \), and then consider the behavior of the follower’s optimum solution \( x \) when \( b \) changes. The leader’s objective function \( f \) is given by the corresponding values \( d^\top x \):

\[
 f(b) := \sum_{i=1}^{j-1} d_i + \frac{d_j}{a_j} \left(b - \sum_{i=1}^{j-1} a_i\right) \text{ for } b \in \left[ \sum_{i=1}^{j-1} a_i, \sum_{i=1}^j a_i \right], j \in \{1, \ldots, n\}
\]

Note that this piecewise linear function is well-defined and continuous with vertices in the points \( b = \sum_{i=1}^j a_i \), in which the critical item changes from \( j \) to \( j + 1 \). The leader has to maximize \( f \) over the range \([b^-, b^+]\). As \( f \) is piecewise linear, it suffices to evaluate it at the boundary points \( b^- \) and \( b^+ \) and at all feasible vertices, i.e., at \( b = \sum_{i=1}^j a_i \) for all \( j \in \{0, \ldots, n\} \) with \( \sum_{i=1}^j a_i \in [b^-, b^+] \). Hence, Problem (P) can be solved in time \( O(n \log n) \), which is the time needed for sorting.

Remark 1. The order of items and hence the follower’s optimum solution is not unique if the profits \( c_i/a_i \) are not all different. In the optimistic approach, a follower would sort the items with the same profit in descending order of the values \( d_i/a_i \), in the pessimistic setting in ascending order. If this is still not unique, there is no difference for the leader either.
3 Discrete uncertainty

Turning to the uncertain problem variant, we first consider the robust version of the problem where the follower’s objective function is uncertain for the leader, and this uncertainty is given by a finite uncertainty set \( U \subset \mathbb{R}_{>0}^n \) containing the possible objective vectors \( c \). We obtain the following problem formulation:

\[
\max_{b \in [b^- , b^+]} \min_{c, x} \quad d^\top x \\
\text{s.t.} \quad c \in U \\
\quad x \in \text{argmax} \quad c^\top x \\
\quad \text{s.t.} \quad a^\top x \leq b \\
\quad 0 \leq x \leq 1
\]

The inner minimization problem can be interpreted as being controlled by an adversary, thus leading to an optimization problem involving three actors: first, the leader takes her decision \( b \), then the adversary chooses a follower’s objective \( c \) leading to a follower’s solution that is worst possible for the leader, and finally the follower optimizes this objective choosing \( x \). In the pessimistic view of the bilevel problem, the adversary can be assumed to also choose among the follower’s optimum solutions if this is not unique, i.e., to minimize over \( c \) as well as \( x \).

Again, we aim at solving this problem from the leader’s perspective, which can be done as follows: for every \( c \in U \), consider the piecewise linear function \( f_c \) as described in Section 2. The vertices of each \( f_c \) can be computed in \( \mathcal{O}(n \log n) \) time, both in the optimistic and the pessimistic approach. The leader’s objective function is then the pointwise minimum \( f := \min_{c \in U} f_c \) and her task is to maximize \( f \) over \([b^- , b^+]\).

Regarding the computation of the pointwise minimum, or lower envelope, of piecewise linear functions, one may use results that are derived from properties of Davenport-Schinzel sequences; see, e.g., [20]. In [12], the authors mention, as an application of their more general result, that the lower envelope of \( m \) linear segments consists of \( \mathcal{O}(m \alpha(m)) \) many linear segments, where \( \alpha \) is the inverse of the Ackermann function, which grows extremely slowly. In our case, this translates to the leader’s objective function being a piecewise linear function with \( \mathcal{O}(|U| n \alpha(|U|)) \) many segments. A divide-and-conquer approach to compute the lower envelope in \( \mathcal{O}(m \alpha(m) \log m) \) time is also proposed in [12]. Its runtime is improved to \( \mathcal{O}(m \log m) \) in [14]. This proves

**Theorem 2.** The robust bilevel continuous knapsack problem with finite uncertainty set \( U \) can be solved in \( \mathcal{O}(|U| n \log(|U|)) \) time.

Exploiting the special structure of the linear segments arising in our context, it might be possible to improve this runtime further, in the spirit of [16], where a generalization of Davenport-Schinzel sequences is applied to envelopes of piecewise linear functions. Since our aim was to devise a polynomial time algorithm, we did not investigate this further.
4 Interval uncertainty

We next address a robust version of the problem having the same structure as in Section 3, but now the uncertainty is given by an interval for each component of $c$. We thus consider $U = [c^-_1, c^+_1] \times \cdots \times [c^-_n, c^+_n]$ and assume $0 < c^- \leq c^+$. In classical robust optimization, one could just replace each uncertain coefficient $c_i$ by an appropriate endpoint $c_i^-$ or $c_i^+$ and obtain a certain problem again. However, such a replacement is not a valid reformulation in the bilevel context. We will show that, in fact, the situation in the bilevel case is more complicated, even though we can still devise an efficient algorithm.

To simplify the notation, we define $p^-_i := -\frac{c^+_i}{a_i}$, $p^+_i := -\frac{c^-_i}{a_i}$ for the remainder of this section. It turns out that interval orders defined by the intervals $[p^-_i, p^+_i]$ play a crucial role in the investigation.

4.1 Interval orders and precedence constraint knapsack problems

For the leader, the exact entries of $c_i$ in their intervals $[c^-_i, c^+_i]$ do not matter, but only the induced sorting that the follower will use. The follower sorts the items by their values $c_i / a_i$ and we therefore have to consider the intervals $[c^-_i / a_i, c^+_i / a_i]$ induced by the uncertainty set. Intuitively speaking, two situations can arise for a pair $(i, j)$ of items: either, their corresponding intervals are disjoint, say, $c^-_i / a_i > c^+_j / a_j$. In this case, $i$ will precede $j$ in every sorting induced by some adversary’s choice $c \in U$. Otherwise, the two intervals intersect. Then, the adversary can decide which of the two items comes first by choosing the values $c_i$ and $c_j$ appropriately.

More formally, given $U$ and $a$, the possible sortings are exactly the linear extensions of the partial order $P$ that is induced by the intervals $[p^-_i, p^+_i]$, in the sense that we set

\[ i <_P j \iff \frac{c^-_i}{a_i} > \frac{c^+_j}{a_j} \iff p^+_i < p^-_j. \]

Such a partial order is called an interval order. Note that the values $c_i / a_i$ are actually sorted in decreasing order by the follower, but it is more common to read an interval order from left to right. Therefore, by the definition of $p^-_i$ and $p^+_i$, the intervals were flipped, so that we can think of the follower sorting the negative values $-c_i / a_i$ increasingly.

Note that, in the pessimistic problem version, intervals intersecting in only one point do not have to be treated differently from intervals intersecting properly, since the adversary choosing the same value $c_i / a_i$ for several items will result in the order of these items that is worst for the leader anyway, due to the pessimism.

To solve the robust bilevel continuous knapsack problem with interval uncertainty, one could compute the partial order $P$ and enumerate all linear extensions of $P$. Every linear extension corresponds to a sorting of the items the follower will use when the adversary has chosen $c$ appropriately. Every sorting corresponds to a piecewise linear function, and the leader’s objective function is the pointwise minimum of all these, as in Section 3. This approach does not have polynomial runtime in general, as there could be exponentially
many linear extensions. However, it turns out that it is not necessary to consider all linear extensions explicitly and that the problem can still be solved in polynomial time. We will see that the adversary’s problem for fixed $b \in [b^-, b^+]$ is closely related to the precedence constraint knapsack problem or partially ordered knapsack problem. This is a 0-1 knapsack problem, where additionally, a partial order on the items is given and it is only allowed to pack an item into the knapsack if all its predecessors are also selected; see, e.g., Section 13.2 in [17].

For the special case of this problem where the partial order is an interval order, Woeginger described a pseudopolynomial algorithm, see Lemma 11 in [21]. There the problem is formulated in a scheduling context and is called good initial set. The algorithm is based on the idea that every initial set (i.e., prefix of a linear extension of the interval order) consists of

- a head, which is the item whose interval has the rightmost left endpoint among the set,
- all predecessors of the head in the interval order, and
- some subset of the items whose intervals contain the left endpoint of the head in their interior,

assuming that all interval endpoints are pairwise distinct.

The algorithm iterates over all items as possible heads, and looks for the optimum subset of the items whose intervals contain the left endpoint of the head in their interior that results in an initial set satisfying the capacity constraint. Since these items are incomparable to each other in the interval order, each subproblem is equivalent to an ordinary 0-1 knapsack problem and can be solved in pseudopolynomial time using dynamic programming; see e.g., [17]. Our algorithm for the adversary’s problem is a variant of this algorithm for the continuous knapsack and uses Dantzig’s algorithm as a subroutine, therefore we will obtain polynomial runtime.

For this, we introduce the notion of a fractional prefix of a partial order $P$, which is a triple $(J, j, \lambda)$ such that $J \subseteq \{1, \ldots, n\}$, $j \in J$, $0 < \lambda \leq 1$, and there is an order of the items in $J$, ending with $j$, that is a prefix of a linear extension of $P$. Every optimum solution of the follower, given some $b$ and $c$, corresponds to a fractional prefix. The follower’s solution corresponding to a fractional prefix $F = (J, j, \lambda)$ is defined by

$$x^F_i := \begin{cases} 1 & \text{for } i \in J \setminus \{j\} \\ 0 & \text{for } i \in \{1, \ldots, n\} \setminus J \\ \lambda & \text{for } i = j \end{cases}$$

Additionally, there is the empty fractional prefix $\emptyset$ with $x^\emptyset = 0$.

Let $F_P$ be the set of all fractional prefixes of the interval order $P$ given by $U$ and $a$, corresponding to the set of all optimum follower’s solutions for some $b \in [0, \sum_{i=1}^n a_i]$ and some $c \in U$. Then the adversary’s task is, for fixed $b$, to choose a fractional prefix $F \in F_P$ satisfying $a^T x^F = b$, which in the original formulation, he does implicitly
by choosing some $c \in U$ and anticipating the follower’s optimum solution under this objective. Therefore the leader’s problem can be reformulated as follows:

$$\max_{b \in [b^-, b^+]} \min_{F \in \mathcal{F}_P} \frac{1}{a^T x} \mathbf{d}^T x^F$$

In the next subsections, we first describe an algorithm to solve the inner minimization problem for fixed $b$, i.e., the adversary’s problem, which will then be generalized to the maximization problem over $b$.

### 4.2 Solving the adversary’s problem

First, consider the special case where the interval order has no relations. This means that all intervals intersect and hence, all permutations are valid linear extensions. Note that a pairwise intersection of intervals implies that all intervals have a common intersection, since for every two intervals $[z^{-1}_i, z^+_i]$ and $[z^{-1}_j, z^+_j]$ holds that $z^{-1}_i \leq z^+_j$ for $i, j \in \{1, 2\}$, so the smallest right endpoint is right of or equal to the largest left endpoint.

Then the adversary’s problem is very similar to the ordinary continuous knapsack problem. The only differences are that the objective is now to minimize (and not maximize), that the objective vector $\mathbf{d}$ may contain positive and negative entries, and that the constraint $a^T x \leq b$ is replaced by $a^T x = b$. But with this changed, the problem can still be solved using Dantzig’s algorithm as described in Section 2, note that the algorithm fills the knapsack completely anyway assuming $b \leq \sum_{i=1}^{n} a_i$ and $c > 0$. Denote this algorithm, returning the corresponding fractional prefix, by DANTZIG. We will also need this algorithm as a subroutine on a subset of the item set (like the pseudopolynomial knapsack algorithm in Woeginger’s algorithm). Therefore, we consider DANTZIG as having input $I \subseteq \{1, \ldots, n\}$, $d \in \mathbb{R}^n$, $a \in \mathbb{R}^{n}_{>0}$, and $b \in [0, \sum_{i \in I} a_i]$, and only choosing a solution among the items in $I$.

We have seen that in case the interval order has no relations the adversary’s problem can be solved by DANTZIG($\{1, \ldots, n\}, d, a, b$). The general adversary’s problem can now be solved by Algorithm 1. In the notation of Woeginger’s algorithm, the $k$-th item is the head in iteration $k$, $I^{-}_k$ is the set of its predecessors, and $I^0_k$ corresponds to the intervals containing the left endpoint of the head – not necessarily in their interior here, so that, in particular, also $k \in I^0_k$.

The basic difference to Woeginger’s algorithm is that here it is important to have a dedicated last item of the prefix, which will be the one possibly taken fractionally by the follower. Apart from that, the order of the items in the prefix is not relevant. In our construction, any element of $I^0_k$ could be this last item, in particular it could be $k$, but it does not have to. Note that, in Algorithm 1, the prefix constructed in iteration $k$ does not necessarily contain the $k$-th item, but still, all prefixes that do contain it as their head are covered by this iteration.

**Lemma 3.** Algorithm 1 is correct.

*Proof.* For $b = 0$, the only feasible and therefore optimum solution is $\emptyset$, so that the result is correct if the algorithm terminates in line 2.
Algorithm 1: Algorithm for the adversary’s problem

\textbf{Input}: \(a \in \mathbb{R}_n^+, 0 \leq b \leq \sum_{i=1}^n a_i, d \in \mathbb{R}^n, p^- \leq p^+, \) inducing a interval order \(P\)

\textbf{Output}: \(F \in \mathcal{F}_P\) with \(a^T x^F = b\) minimizing \(d^T x_F\)

1 \textbf{if} \(b = 0\) \textbf{then}

2 \hspace{1em} \textbf{return} \(\emptyset\)

3 \hspace{1em} \(K := \emptyset\)

4 \textbf{for} \(k = 1, \ldots, n\) \textbf{do}

5 \hspace{1em} \(I_k^+ := \{i \in \{1, \ldots, n\}: p_i < p_k\}\)

6 \hspace{1em} \(I_k^- := \{i \in \{1, \ldots, n\}: p_i \leq p_k \leq p_i^+\}\)

7 \hspace{1em} \textbf{if} \(0 < b - \sum_{i \in I_k^-} a_i \leq \sum_{i \in I_k^0} a_i\) \textbf{then}

8 \hspace{2em} \((J_k^+, j_k, \lambda_k) := \text{DANTZIG}(I_k^0, d, a, b - \sum_{i \in I_k^-} a_i)\)

9 \hspace{2em} \(J_k := J_k^+ \cup I_k^-\)

10 \hspace{2em} \(K := K \cup \{k\}\)

11 \textbf{return} \((J_k, j_k, \lambda_k) \) with \(k = \text{argmin}\{d^T x(J_k^+, j_k, \lambda_k) : k \in K\}\)

So assume \(b \neq 0\) now. The first part of the proof shows that the algorithm returns a feasible solution if \(K \neq \emptyset\). The second part proves the optimality of the returned solution and also that \(K \neq \emptyset\) always holds.

In each iteration \(k\), \(I_k^-\) is the set of predecessors of \(k\) in the interval order \(P\). The set \(I_k^0\) consists of items that are incomparable to \(k\) and to each other in \(P\), since the corresponding intervals all contain the point \(p_k\) by definition. Hence it is valid (with respect to \(P\)) to call Dantzig’s algorithm in line 8 on \(I_k^0\). The condition in line 7 makes sure that we only call the subroutine if the available capacity is in the correct range, i.e., if it is possible to fill the knapsack with the items in \(I_k^-\) and a nonempty subset of the items in \(I_k^0\).

Then \((J_k, j_k, \lambda_k)\) is a fractional prefix of \(P\), as all predecessors of \(k\) and therefore also all predecessors of all \(i \in J_k \subseteq I_k^- \cup I_k^0\) belong to \(J_k\), since \(p_i^- \leq p_k\) holds for them. The item \(j_k\) is a valid last item of a prefix consisting of the items in \(J_k\) because \(j_k \in I_k^0\) by construction and therefore, there are no successors of \(j_k\) in \(J_k\). Moreover, \(a^T x(J_k^+, j_k, \lambda_k) = \sum_{i \in I_k^-} a_i + (b - \sum_{i \in I_k^+} a_i) = b\) by construction and by the correctness of DANTZIG. Therefore, for all \(k \in K\), \((J_k, j_k, \lambda_k)\) is a feasible solution.

Now we prove the optimality of the returned solution. Let \((J, j, \lambda)\) be an optimum solution (if \(\emptyset\) is optimum, then \(b\) must be 0, and this case is trivial). Choose \(k \in J\) with maximal \(p_k^-\) (i.e., a head of the prefix). Then \(I_k^- \subseteq J\) since \(J\) is a prefix and \(k \in J\), so all predecessors of \(k\) must be in \(J\), as well. Moreover, \(j \in J \setminus I_k^-\) as all items in \(I_k^-\) have at least one successor (namely \(k\)) in \(J\). By the choice of \(k\), we have \(J \setminus I_k^- \subseteq I_k^0\) and \((J \setminus I_k^-, j, \lambda)\) is a feasible solution of the subproblem solved by the call of DANTZIG in...
line 8 since $a^\top x^{(J\setminus I_k \setminus J, \lambda)} = a^\top x^{(J, \lambda)} - \sum_{i \in I_k} a_i = b - \sum_{i \in I_k} a_i$. Thus

$$d^\top x^{(J, j_k, \lambda)} = \sum_{i \in I_k} d_i + d^\top x^{(J \setminus I_k \setminus J, \lambda)} \geq \sum_{i \in I_k} d_i + d^\top x^{(J_k, j_k, \lambda)} = d^\top x^{(J_k, j_k, \lambda)},$$

which is at least the cost of any returned solution. The second part of the proof also shows that $K \neq \emptyset$. Thus, the algorithm always returns an optimum solution.

An optimum solution of the adversary’s problem in the original formulation, i.e., a vector $c \in U$, can be derived from the fractional prefix $(J_k, j_k, \lambda_k)$ returned by the algorithm in the following way:

$$c_i := \begin{cases} c_i^+ & \text{for } i \in J_k \setminus \{j_k\} \\ c_i^- & \text{for } i \in \{1, \ldots, n\} \setminus J_k \\ c_k^+/a_k \cdot a_{j_k} & \text{for } i = j_k \end{cases}$$

Note that $c_k^- \leq c_k^+/a_k \cdot a_{j_k} \leq c_k^+$ holds because by construction $p_{j_k}^- \leq p_k^- \leq p_{j_k}^+$ as $j_k \in I_k^0$. Indeed, this definition ensures that the items $i \in J_k \setminus \{j_k\}$ precede $j_k$ in the follower’s sorting, since $p_i^\leq \leq p_k^\leq$ and therefore, $c_i/a_i = c_i^+/a_i = -p_i^- \geq -p_k^- = c_k^+/a_k = c_{j_k}/a_{j_k}$. Analogously, the items $i \in \{1, \ldots, n\} \setminus J_k$ will be packed after $j_k$ by the follower.

Note that this solution sets each variable except for $c_{j_k}$ to an endpoint of its corresponding interval. In general, there is no optimum solution with all variables set to an interval endpoint. This can be seen by the following example.

**Example 4.** Let $n = 3$ and define $a = (1, 1, 1)^\top$, $b = \frac{3}{2}$, $U = \{3\} \times \{2\} \times [1, 4]$ and $d = (-1, 1, 0)^\top$. The optimum solution returned by the algorithm is $((1, 3), 3, \frac{1}{2})$ with value 1. For the follower to select the first item and half of the third item, i.e., for $c_1 \geq c_3 \geq c_2$ to hold, the adversary must choose $c_3 \in [2, 3]$, so it cannot be at one of the endpoints of the interval $[1, 4]$.

### 4.3 Solving the leader’s problem

Next, we describe an algorithm to solve the robust bilevel optimization problem, which performs the maximization over the capacity $b$. For this, we will use the variant of Dantzig’s algorithm which returns a piecewise linear function, as described in Section 2. We call this routine `BilevelDantzig` and assume its input to be $I \subseteq \{1, \ldots, n\}$, $d \in \mathbb{R}^n$, $a \in \mathbb{R}_{>0}^n$, and $0 \leq b^- \leq b^+ \leq \sum_{i \in I} a_i$, since, as before, we need it also to work on a subset of the item set. The output is a piecewise linear function $f$, which can be represented by a list of all its vertices, given as points of the graph of $f$. The leader’s problem can now be solved by Algorithm 2.

**Lemma 5.** Algorithm 3 is correct.
Algorithm 2: Algorithm for the leader’s problem

**Input**: \( a \in \mathbb{R}_n \geq 0, 0 \leq b^- \leq b^+ \leq \sum_{i=1}^n a_i, \) \( d \in \mathbb{R}^n, p^- , p^+ \in \mathbb{R}^n \leq 0 \) with \( p^- \leq p^+ \), inducing an interval order \( P \)

**Output**: value \( b \in [b^-, b^+] \) maximizing the result of Algorithm 1

1. \( K := \emptyset \)
2. if \( b^- = 0 \) then
3. \( f_0 := \{(0, 0)\} \)
4. \( K := K \cup \{0\} \)
5. for \( k = 1, \ldots, n \) do
6. \( I^-_k := \{i \in \mathbb{Z}^+ : p^+_i < p^-_k\} \)
7. \( P^+_k := \{i \in \mathbb{Z}^+ : p^-_k \leq p^+_i \} \)
8. if \( 0 \leq b^+ - \sum_{i \in I^-_k} a_i \) and \( b^- - \sum_{i \in I^-_k} a_i \leq \sum_{i \in P^+_k} a_i \) then
9. \( \hat{b}^- := \max\{0, b^- - \sum_{i \in I^-_k} a_i\} \)
10. \( \hat{b}^+ := \min\{b^+ - \sum_{i \in I^-_k} a_i, \sum_{i \in P^+_k} a_i\} \)
11. \( f'_k := \text{BilevelDantzig}(P^+_k, d, a, \hat{b}^-, \hat{b}^+) \)
12. \( f_k := f'_k + (\sum_{i \in I^-_k} a_i, \sum_{i \in I^-_k} d_i) \) // shift all vertices of \( f'_k \) by \( \sum_{i \in I^-_k} a_i \) in \( x \)
13. \( \) // and by \( \sum_{i \in I^-_k} d_i \) in \( y \) direction
14. \( K := K \cup \{k\} \)
15. \( f := \min\{f_k : k \in K\} \) // pointwise minimum
16. \( \text{return} \ \max\{f(b) : b^- \leq b \leq b^+\} \)
Proof. First note that Algorithm 1 can be considered as the special case of Algorithm 2, where \( b = b^− = b^+ \). For the correctness of Algorithm 2, it is enough to show that the function \( f \) describes the value of the output of Algorithm 1 depending on \( b \in [b^−, b^+] \). For \( b = 0 \), which is only possible if \( b^− = 0 \), this is clearly the case.

The condition in line 8 ensures that \( b^− \leq \tilde{b}^+ \), so the call of BILEVELDANTZIG in line 11 is valid. Let \((J_k^b, \tilde{f}_b^k, \lambda_k^b)\) be the fractional prefix \((J_k, \tilde{f}_k, \lambda_k)\) in Algorithm 1 called for \( b \in [b^−, b^+] \). We claim that, for all \( k \in \{1, \ldots, n\} \) and all \( b \in [b^−, b^+] \) with \( b > 0 \), the function \( f_k \) in Algorithm 2 is defined in the point \( b \) if and only if \((J_k^b, \tilde{f}_k^b, \lambda_k^b)\) is defined in Algorithm 1 and then \( f_k(b) = d^T x(J_k^b, \tilde{f}_k^b, \lambda_k^b) \).

Let \( k \in \{1, \ldots, n\} \) and \( b \in [b^−, b^+] \) with \( b > 0 \). Then \( f_k(b) \) is defined if and only if \( \tilde{b}^− + \sum_{i \in I_k^−} a_i \leq b \leq \tilde{b}^+ + \sum_{i \in I_k^−} a_i \), i.e., if and only if

\[
\sum_{i \in I_k^−} a_i \leq b \leq \sum_{i \in I_k^0} a_i + \sum_{i \in I_k^−} a_i ,
\]

which is (almost) the same condition as the one for defining \((J_k^b, \tilde{f}_k^b, \lambda_k^b)\). Actually, \((J_k^b, \tilde{f}_k^b, \lambda_k^b)\) is not defined if \( b = \sum_{i \in I_k^−} a_i \), but this is only for convenience in the formulation of Algorithm 1. We could define it there as \((I_k^−, j^+, 1)\), where \( j^+ \) is a maximal element in \( I_k^− \). But this is not relevant since this fractional prefix is also considered in iteration \( j^+ \).

In case \( f_k(b) \) is defined, the corresponding values \( f_k(b) \) and \( d^T x(J_k^b, \tilde{f}_k^b, \lambda_k^b) \) agree because the piecewise linear function returned by BILEVELDANTZIG consists of the values of the solutions returned by DANTZIG for given values of \( b \). Since \( f(b) \) is the minimum of all \( f_k(b) \) that are defined, this implies that it is equal to the value of the optimum solution computed in Algorithm 2.

\[\square\]

Theorem 6. The robust bilevel continuous knapsack problem with interval uncertainty can be solved in \( O(n^2 \log n) \) time.

Proof. In every of the \( n \) iterations, Algorithm 2 needs \( O(n) \) time to compute the sets \( I_k^− \) and \( I_k^0 \), and, since \(|I_k^0| \leq n\), \( O(n \log n) \) time for Dantzig’s algorithm. As explained in Section 3, the pointwise minimum of the at most \( n \) piecewise linear functions with at most \( n \) segments each, as well as the maximum of the resulting function (lines 15 and 16) can be computed in \( O(n^2 \log n) \) time.

\[\square\]

Remark 7. In the pessimistic problem version, intersections between intervals that only consist in one point, are not special since the worst possible order of the corresponding items from the leader’s perspective will be chosen anyway. If we use the optimistic approach, we have to distinguish between two cases of one-point intersections. Consider items \( i \) and \( j \) with \( p^+_i = p^-_j \):

1. If \( d_i/a_i > d_j/a_j \), the order where \( i \) precedes \( j \) is the preferred one for the adversary anyway, hence it is safe to allow the adversary to choose any of the two possible orders of \( i \) and \( j \) as in the pessimistic version.
2. If \(d_i/a_i < d_j/a_j\), the adversary would prefer the order where \(j\) precedes \(i\), which can only possibly be achieved by choosing \(p_i = p_j\). But with this choice, the optimism will lead to the follower choosing \(i\) before \(j\). Therefore, the adversary cannot enforce his preferred order, although the two intervals intersect. This can be modelled by shifting one of the endpoints slightly, such that \(p_i^+ < p_j^-\), and therefore enforcing that \(i\) always precedes \(j\) by the interval order.

Note that we do not have to care about the case \(d_i/a_i = d_j/a_j\) since then, any order of the items \(i\) and \(j\) leads to the same objective function value.

This extends in a similar way to more than two intervals intersecting in only one point. Thus, with some preprocessing or a slight change in the algorithm in order to respect these special cases explicitly in the construction of \(I_k^-\) and \(I_k^0\), the optimistic case can be solved in the same way as the pessimistic one.

5 Discrete uncorrelated uncertainty

We now consider the robust bilevel continuous knapsack problem with an uncertainty set of the form \(U = U_1 \times \cdots \times U_n\), where each \(U_i\) is a finite set, i.e., for every component of \(c\), there is a finite number of options the adversary can choose from and these are independent of each other. It turns out that this version of the problem is NP-hard in general, even if we consider the special case \(U = \{c^-_1, c^+_1\} \times \cdots \times \{c^-_n, c^+_n\}\), where there are only two options in every component.

**Theorem 8.** The robust bilevel continuous knapsack problem with an uncertainty set being the product of finite sets, is NP-hard, even if each of these sets has size two.

We show Theorem 8 by a reduction from the well-known NP-hard subset sum problem. Let \(m \in \mathbb{N}\) and \(w_1, \ldots, w_m, W \in \mathbb{N}\) be an instance of subset sum. Without loss of generality, we may assume \(1 \leq W \leq \sum_{i=1}^{m} w_i - 1\). We show that we can decide if there is a subset \(S \subseteq \{1, \ldots, m\}\) with \(\sum_{i \in S} w_i = W\) in polynomial time if the following instance of the robust bilevel continuous knapsack problem can be solved in polynomial time: define \(\varepsilon := \frac{1}{4}\) and \(M := \sum_{i=1}^{m} w_i + \varepsilon\). Let \(n := m + 2\) and set

\[
\begin{align*}
a & := (\varepsilon, w_1, \ldots, w_m, M)^	op \\
d & := (-M, -w_1, \ldots, -w_m, \varepsilon)^	op \\
b^- & := W \\
b^+ & := W + 2\varepsilon.
\end{align*}
\]

The uncertainty set is defined as

\[
U := \{1 \cdot a_1, (n + 1) \cdot a_1\} \times \{2 \cdot a_2, (n + 2) \cdot a_2\} \times \cdots \times \{n \cdot a_n, 2n \cdot a_n\},
\]

which leads to \(c^-_1/a_1 < \cdots < c^-_n/a_n < c^+_1/a_1 < \cdots < c^+_n/a_n\). Note that, since all these values are distinct, the optimistic and pessimistic approach do not have to be distinguished for this instance.
In the following two lemmas, we investigate the structure of optimum follower’s solutions and of the leader’s objective function, respectively. Both is done for a large range \([\varepsilon, M]\) of values for \(b\), which will be useful to understand the two different behaviors the function can have in the actual range \([b^-, b^+]\) afterwards, depending on the subset sum instance being a yes or a no instance.

**Lemma 9.** Let any leader decision \(b \in [\varepsilon, M]\) be given. Then, for every optimal choice of the adversary, the resulting follower’s solution \(x\) satisfies \(x_i \in \{0,1\}\) for all \(i \in \{2,\ldots,n\}\).

**Proof.** First note that the adversary can always enforce a solution with a value of at most \(-M\), e.g., by setting \(c_1 = c_i^+\) and \(c_i = c_i^-\) for all \(i \in \{2,\ldots,n\}\), so that the first item is taken first and therefore completely because \(a_1 = \varepsilon \leq b\). This implies that we always have \(x_1 > 0\), because every solution not taking the first item at all has value greater than \(-M\). Moreover, note that we always have \(x_n < 1\) as \(a_n = M > b\).

Now let \(i \in \{2,\ldots,n-1\}\). If the adversary chooses \(c_i = c_i^+\), it follows that \(x_i = 1\). Indeed, \(x_i < 1\) would imply \(x_1 = 0\) since \(c_1/a_1 < c_i^+ / a_i\) always holds. Analogously, if the adversary chooses \(c_i = c_i^-\), we have \(x_i = 0\), as \(x_i > 0\) would imply \(x_n = 1\), since \(c_i^- / a_i < c_n / a_n\). Therefore, an optimum adversary’s solution always leads to a follower’s solution \(x\) with \(x_i \in \{0,1\}\) for all \(i \in \{2,\ldots,n-1\}\). \(\square\)

For the remainder of the proof, denote the leader’s objective function by \(f\). This function is described for all \(b \in [\varepsilon, M]\) by the following

**Lemma 10.** Let \(V_1, V_2 \in \mathbb{N}_0\) be two consecutive values arising as sums of subsets of \(\{w_1,\ldots,w_m\}\). Then

\[
f(b) = \begin{cases} 
-M - V_1 + \frac{\varepsilon}{M}(b - V_1 - \varepsilon), & \text{if } b \in [V_1 + \varepsilon, V_2) \\
\min \left\{ -M - V_1 + \frac{\varepsilon}{M}(b - V_1 - \varepsilon), -\frac{M}{2}(b - V_2) - V_2 \right\}, & \text{if } b \in [V_2, V_2 + \varepsilon). 
\end{cases}
\]

**Proof.** Let \(\sum_{i \in T} w_i = V_1\) for some \(T \subseteq \{1,\ldots,m\}\). Using Lemma 9 it is easy to see that for all \(b \in [V_1 + \varepsilon, V_2)\), the unique best choice of the adversary is to let the follower set \(x_1 = 1\) and pack \(2,\ldots,n-1\) according to \(T\), the rest is filled by a fraction of item \(n\). Indeed, for \(b < V_2\), no better packing of items \(2,\ldots,n-1\) than \(T\) is possible, and since \(b \geq V_1 + \varepsilon\), the most profitable item 1 (from the adversary’s perspective) can be added entirely without making the packing \(T\) infeasible. The adversary can produce this solution by choosing \(c_1 = c_1^+, \ c_n = c_n^-\) and for \(i \in \{2,\ldots,n-1\}\), \(c_i = c_i^+\) if \(i - 1 \in T\), or \(c_i = c_i^-\) otherwise. This leads to

\[
f(b) = -M - V_1 + \frac{\varepsilon}{M}(b - V_1 - \varepsilon)
\]

for \(b \in [V_1 + \varepsilon, V_2)\). If \(b \in [V_2, V_2 + \varepsilon)\), the same solution, with a larger fraction of item \(n\), is still possible. However, since now a better packing \(S \subseteq \{1,\ldots,m\}\) with \(\sum_{i \in S} w_i = V_2\) is available, we also have to consider to pack according to \(S\) and add as much of item 1
as possible (which the adversary can obtain again by setting $c_1 = c_1^+$, $c_n = c_n^-$ and for $i \in \{2, \ldots, n-1\}$, $c_i = c_i^+$ if $i-1 \in S$, or $c_i = c_i^-$ otherwise). This solution has value

$$-\frac{M}{\epsilon} (b - V_2) - V_2,$$

and the adversary chooses the solution leading to a smaller value of $f$. \hfill \Box$

Let $b^* \in [b^-, b^+] = [W, W + 2\epsilon]$ be an optimal leader’s solution in the original range. We conclude the proof of Theorem 8 by showing that the given instance of subset sum is a yes instance if and only if $b^* \neq b^+$, which follows from the following two lemmas. The two situations arising in Lemma 11 and Lemma 12 are illustrated in Figure 1.

**Lemma 11.** If the given instance $(w_1, \ldots, w_m, W)$ of subset sum is a no instance, then $b^* = b^+$.

**Proof.** Since $W$ does not arise as a subset sum, the same holds for all values in the interval $(W - 1, W + 1)$ by integrality. By Lemma 10, the function $f$ is linear on $[b^-, b^+]$ with slope $\frac{M}{\epsilon} > 0$. \hfill \Box

**Lemma 12.** If the given instance $(w_1, \ldots, w_m, W)$ of subset sum is a yes instance, then $f(b^-) > f(b^+)$ and hence $b^* \neq b^+$.

**Proof.** Let $V$ be the largest subset sum with $V < W$. We obtain $f(b^+) = f(W + 2\epsilon) = -M - W + \frac{\epsilon}{M}$ by applying Lemma 10 to $V_1 = W$. By Lemma 10 applied to $V_1 = V$ and $V_2 = W$, the two possible values of $f$ in $b^- = W$ are $-M - V + \frac{\epsilon}{M}(W - V - \epsilon)$ and $-W$. Both can be easily seen to be strictly larger than $f(b^+)$. \hfill \Box

This concludes the proof of Theorem 8. Note that this also proves the NP-hardness of the adversary’s problem:

**Theorem 13.** Evaluating the objective function value of the robust bilevel continuous knapsack problem with an uncertainty set being the product of finite sets, in some feasible point $b$, is NP-hard, even if each of the sets has size two.
Proof. Looking at the proof of Theorem 8, the two cases in Lemma 11 and Lemma 12 can also be distinguished by computing the adversary’s optimum solution value (or the leader’s objective function value, equivalently) in the point \( b = b^* = W + 2\varepsilon \). In case of a yes instance it is

\[
f(b) = -M - W + \frac{\varepsilon}{M}(b - W - \varepsilon),
\]

while in case of a no instance it is

\[
f(b) = -M - V + \frac{\varepsilon}{M}(b - V - \varepsilon),
\]

where \( V \) is the largest subset sum with \( V < W \), which leads to the second value being larger than the first one. This shows that there is also a reduction from the subset sum problem to the adversary’s problem.

In Section 4, we have seen that the robust bilevel continuous knapsack problem can be solved efficiently when each coefficient is chosen independently from a given interval. Theorem 8 shows that the same problem turns NP-hard when the adversary is only allowed to choose the follower’s objective coefficients from the endpoints of the intervals. In particular, this implies that replacing an uncertainty set by its convex hull may change the problem significantly, in contrast to the situation in single-level robust optimization. This can also be shown by the following explicit example.

**Example 14.** Let \( n = 5 \) and define \( a = (1,1,1,1,1)^\top \), \( b^- = 0 \), \( b^+ = 5 \), \( U = \{5\} \times \{4\} \times \{3\} \times \{2\} \times \{1,6\} \) and \( d = (2,-1,1,-2,0)^\top \). In this instance, the order of the items 1, 2, 3, and 4 is fixed, while item 5 could be in the first or last position with respect to uncertainty set \( U \), but also in every position in between the other items when the uncertainty set is \( \text{conv}(U) = \{5\} \times \{4\} \times \{3\} \times \{2\} \times [1,6] \). The leader’s objective function on \([0,5]\) is depicted in Figure 2 for uncertainty sets \( U \) and \( \text{conv}(U) \).

In the former case, the leader’s unique optimal solution is \( b = \frac{3}{2} \) with objective value \( \frac{5}{2} \), while in the latter case the two optimal solutions are \( b = \frac{5}{3} \) and \( b = \frac{10}{5} \) with objective value \( \frac{17}{3} \).
6 Simplicial uncertainty

We next consider uncertainty sets being simplices and again show that the problem is NP-hard in this case. It can be considered a special case of polytopal and Gamma-uncertainty, but can also be viewed as the convex variant of the discrete case. We will write the uncertainty set in the following as

\[ U_{\hat{c}, \Gamma} = \{ c \in \mathbb{R}^n : c_i \geq \hat{c}_i \text{ for all } i \in \{1, \ldots, n\}, \sum_{i=1}^n (c_i - \hat{c}_i) \leq \Gamma \}, \]

where a vector \( \hat{c} \in \mathbb{R}^n \geq 0 \) and a number \( \Gamma > 0 \) bounding the deviation from \( \hat{c} \) are given.

We have

**Theorem 15.** The robust bilevel continuous knapsack problem with simplicial uncertainty set \( U_{\hat{c}, \Gamma} \), where \( \hat{c} \) and \( \Gamma \) are part of the input, is NP-hard.

**Proof.** We again show this by a reduction from the subset sum problem. Let \( m \in \mathbb{N} \) and \( w_1, \ldots, w_m, W \in \mathbb{N} \) be an instance of subset sum. We show that we can decide if there is a subset \( S \subseteq \{1, \ldots, m\} \) with \( \sum_{i \in S} w_i = W \) in polynomial time if the following instance of the robust bilevel continuous knapsack problem can be solved in polynomial time: define \( n := m + 1 \), \( M := \sum_{i=1}^m w_i + 1 \) and

\[
\begin{align*}
a &:= (w_1, \ldots, w_m, M)^	op \\
d &:= (-w_1, \ldots, -w_m, M)^	op \\
b^- &:= 0 \\
b^+ &:= \sum_{i=1}^n a_i = \sum_{i=1}^m w_i + M \\
\hat{c} &:= ((2M - 1)w_1, \ldots, (2M - 1)w_m, 2M^2)^	op \\
\Gamma &:= W.
\end{align*}
\]

As in the discrete uncertainty case, we may imagine the leader’s objective function \( f \) as the pointwise minimum of all piecewise linear functions \( f_c \) for \( c \in U_{\hat{c}, \Gamma} \). Note that, although there are infinitely many \( c \in U_{\hat{c}, \Gamma} \), the number of distinct functions \( f_c \) is finite since the functions depend only on the follower’s sorting of the items induced by \( c \). Since \( d_i/a_i = -1 \) for all \( i = 1, \ldots, m \) and \( d_{m+1}/a_{m+1} = 1 \), all functions \( f_c \) have the structure shown in Figure 3 where \( V_c := \sum_{i \in I_c} a_i = \sum_{i \in I_c} w_i \) for

\[ I_c := \{ i \in \{1, \ldots, m\} : c_i/a_i \geq c_{m+1}/a_{m+1} \}. \]

The leader’s objective function is the pointwise minimum of the functions \( f_c \). It follows easily that it agrees with \( f^{c^*} \) where \( c^* \in \arg\max_{c \in U_{\hat{c}, \Gamma}} V_c \). In particular, since \( f(V^{c^*} + M) = -V^{c^*} + M > 0 = f(0) \), the leader’s optimal solution is \( V^{c^*} + M \) with value \( -V^{c^*} + M \), so that it remains to show that by computing \( V^{c^*} \) we can decide the subset sum instance.

Since \( \hat{c}_i/a_i = 2M - 1 < 2M = \hat{c}_{m+1}/a_{m+1} \) for all \( i = 1, \ldots, m \), the adversary has two options for every item \( i \):
1. Either he decides to include $i$ in the set $I_c$, which can be achieved by shifting the value $\hat{c}_i/a_i$ to the right by 1 up to $\hat{c}_{m+1}/a_{m+1}$, by choosing $c_i = \hat{c}_i + a_i = \hat{c}_i + w_i$ and thus paying $c_i - \hat{c}_i = w_i$ in the constraint $\sum_{i=1}^m (c_i - \hat{c}_i) \leq \Gamma$ on the uncertainty set. Note that there is no incentive to choose $c_i$ greater than that, as it would not change the leader’s objective function value.

2. Or he decides not to include $i$ in the set $I_c$, for which it is most efficient in terms of the $\Gamma$-constraint to choose $c_i = \hat{c}_i$.

Note that there is no incentive to choose $c_{m+1} > \hat{c}_{m+1}$. Therefore, we may assume that $c_i \in \{\hat{c}_i, \hat{c}_i + w_i\}$ for all $i \in \{1, \ldots, m\}$ and $c_{m+1} = \hat{c}_{m+1}$ hold in an optimum adversary’s solution. By maximizing $V_c = \sum_{i \in I_c} w_i$, we can thus compute the largest subset sum $V$ with $V \leq W$, since $\sum_{i \in I_c} w_i = \sum_{i=1}^m (c_i - \hat{c}_i) \leq \Gamma = W$. Thus, the subset sum instance is a yes instance if and only if $V^* = W$.

Remark 16. In the proof of Theorem 15 we used the assumption of the pessimistic approach, because item $i$ was included in the set $I_c$ even if $c_i/a_i = c_{m+1}/a_{m+1}$. In the optimistic approach, item $i$ may be packed only if strict inequality holds in the definition of $I_c$. Without changing the effect of the constraint $\sum_{i=1}^m (c_i - \hat{c}_i) \leq \Gamma$, this could be modelled by choosing $\hat{c}_{m+1}$ slightly smaller than $2M^2$ in the definition of the instance.

Again, also the adversary’s problem is NP-hard:

**Theorem 17.** Evaluating the objective function value of the robust bilevel continuous knapsack problem with simplicial uncertainty set $U_{\hat{c},\Gamma}$, where $\hat{c}$ and $\Gamma$ are part of the input, in some feasible point $b$, is NP-hard.

*Proof.* Looking at the proof of the previous theorem, it can be seen that the subset sum instance is a yes instance if and only if $f(W) = -W$, so the subset sum problem can also be solved by computing the objective function value at $b = W$.

The uncertainty set $U_{\hat{c},\Gamma}$ is a polytope defined explicitly by $n + 1$ linear inequalities. In particular, this shows NP-hardness for polytopal uncertainty sets given by an outer description:
Corollary 18. The robust bilevel continuous knapsack problem with an uncertainty set being a polytope, given by a set of linear inequalities, is NP-hard.

Theorem 15 also implies that the problem is NP-hard under Gamma-uncertainty, which is sometimes also called budgeted uncertainty [1]. However, this is only true in case the total amount of deviation is bounded. Since uncertainty sets cannot be replaced equivalently by their convex hulls in the bilevel case, the above result does not necessarily hold for the original definition of Gamma-uncertainty, where the number of deviating entries is bounded.

It is easy to check that $\hat{U}_{\hat{c}, \Gamma}$ agrees with $\text{conv}\{\hat{c}, \hat{c} + \Gamma e_1, \ldots, \hat{c} + \Gamma e_n\}$, where $e_i$ denotes the $i$-th unit vector. This proves

Corollary 19. The robust bilevel continuous knapsack problem with an uncertainty set being the convex hull of a finite set of vectors, which are explicitly given as part of the input, is NP-hard.

It follows from Theorem 17 that also the evaluation of the leader’s objective function is NP-hard in all cases mentioned in the last corollaries.

Recall that the problem is tractable if the uncertainty set $U$ is finite and given explicitly as part of the input; see Theorem 2. Together with Corollary 19, this again shows that replacing the uncertainty set by its convex hull may not only change the optimal solution, but even the complexity of the problem significantly.

Remark 20. In contrast to the result of Corollary 19, the problem can be solved in polynomial time if the uncertainty set is given as the convex hull of a constant number $k$ of vectors. In fact, the number of possible sortings in the follower’s solution can be bounded by $O(n^{2k})$ in this case, and they can be enumerated explicitly in polynomial time for constant $k$. The tractability then follows from Theorem 2.

7 Norm uncertainty

In Section 4, we have shown that the robust bilevel continuous knapsack problem can be solved efficiently if the uncertainty set is defined componentwise by intervals. This can be seen as a special case of an uncertainty set defined by a $p$-norm,

$$U_{\hat{c}, \Gamma}^p := \{c \in \mathbb{R}^n : ||c - \hat{c}||_p \leq \Gamma\},$$

where $\hat{c} \in \mathbb{R}^n_{>0}$, $\Gamma > 0$, and $p = \infty$. On the other hand, the case $p = 1$ is closely related to the simplicial case discussed in the previous section, which turned out to be NP-hard. In the following, we show NP-hardness for all $p \in [1, \infty)$.

Theorem 21. Let $p \in [1, \infty)$. Then the robust bilevel continuous knapsack problem with uncertainty set $U_{\hat{c}, \Gamma}^p$, where $\hat{c}$ and $\Gamma$ are part of the input, is NP-hard. This remains true under the assumption $U_{\hat{c}, \Gamma}^p \subseteq \mathbb{R}^n_{>0}$. 

20
Proof. This can be shown very similarly to Theorem 15. We use the same instance, except that
\[ \hat{c} := (2Mw_1 - w_{1/p}, \ldots, 2Mw_m - w_{m/p}, 2M^2) \]
and \( \Gamma := W^{1/p} \). By the same reasoning as before, we may assume that an adversary’s
optimum solution satisfies \( c_i \in \{ \hat{c}_i, \hat{c}_i + w_i^{1/p} \} \) for all \( i \in \{1, \ldots, m\} \) and \( c_{m+1} = \hat{c}_{m+1} \).
For this, note that, although it is now allowed to deviate from \( \hat{c} \) in any direction, the
adversary has no incentive to choose \( c_i < \hat{c}_i \) for any \( i \in \{1, \ldots, n\} \) in the given instance.
In particular, it is cheaper in terms of the constraint \( \sum_{i=1}^{n} |c_i - \hat{c}_i|^p \leq \Gamma^p \) to shift all
values \( c_1/a_1, \ldots, c_m/a_m \) to the right by some \( \varepsilon > 0 \) than moving \( c_{m+1}/a_{m+1} \) to the left
by \( \varepsilon \), i.e.,
\[ \sum_{i=1}^{m} (\varepsilon a_i)^p = \varepsilon^p \sum_{i=1}^{m} w_i^p \leq \varepsilon^p (\sum_{i=1}^{m} w_i)^p < \varepsilon^p M^p = (\varepsilon a_{m+1})^p. \]
The rest of the proof is analogous. \( \square \)

In the proof of Theorem 21, we implicitly assumed that we can compute \( p \)-th roots in
polynomial time when defining \( \hat{c} \) and \( \Gamma \). In general, these values cannot be computed or
even represented exactly in a polynomial time algorithm. However, they only influence
the set of possible sortings, not the actual leader’s solution and objective function val-
tues. For the former, a sufficiently good approximation of the \( p \)-th roots, which can be
computed in polynomial time, leads to the same result.

Remark 22. In the situation of Theorem 21, Remark 16 still applies: while the proof
is formulated for the pessimistic setting, the variation proposed in Remark 16 allows to
show NP-hardness also in the optimistic setting.

In the case \( p = 2 \), the uncertainty set \( U^2_{\hat{c}, \Gamma} \) is an axis-parallel ellipsoid. We thus derive
the following result:

Corollary 23. The robust bilevel continuous knapsack problem with (uncorrelated) el-
lipsoidal uncertainty is NP-hard.

Finally, we can again show NP-hardness even for the problem of evaluating the leader’s
objective function. The proof is analogous to the case of simplicial uncertainty discussed
in the previous section.

Theorem 24. Let \( p \in [1, \infty) \). Then evaluating the objective function of the robust bilevel
continuous knapsack problem with uncertainty set \( U^p_{\hat{c}, \Gamma} \), where \( \hat{c} \) and \( \Gamma \) are part of the
input, is NP-hard.

8 Stochastic approach

From now on, we investigate a stochastic instead of robust optimization approach: the
follower’s objective function is again uncertain for the leader, but we now assume a
probability distribution to be given from which the objective vector \( c \), now being a
random variable, is drawn. This happens after the leader’s, but before the follower’s decision is taken. Thus, also the follower’s optimal decision $x$ and the leader’s objective value $d^\top x$ can now be seen as random variables; to emphasize this, we denote the follower’s decision by $x_c$ in the following. The leader’s objective is now to optimize the expected value of her original objective $d^\top x_c$. The problem can be written as

$$\begin{align*}
\max & \quad E_c(d^\top x_c) \\
\text{s.t.} & \quad b^- \leq b \leq b^+ \\
& \quad x_c \in \arg\max c^\top x \\
& \quad \text{s.t.} \quad a^\top x \leq b \\
& \quad 0 \leq x \leq 1.
\end{align*}$$

(SP)

For the sake of simplicity, we do not distinguish between the optimistic and the pessimistic view in this formulation. However, the results presented in the following again hold in both settings.

In the following, we consider different types of probability distributions for $c$. In the easiest case, the distribution has finite support, which is similar to robust uncertainty with a finite uncertainty set; see Section 3. For this case, a simple efficient algorithm is presented in the following subsection. In the other two cases considered, we draw each component of $c$ independently from a uniform distribution, either a discrete or a continuous one. For both cases, we prove that optimizing or even evaluating the leader’s objective function is $\#P$-hard; see Section 8.2.

### 8.1 Distributions with finite support

In the robust case, we could understand the leader’s objective function as follows: for every possible follower’s objective $c$, an ordering of the items is determined in which the follower would choose them and which corresponds to a piecewise linear function $f_c$ for the leader. However, as there is not only one $c$, but an adversary choosing the $c \in U$ that is worst for the leader after her decision, the leader’s objective function can be seen as the pointwise minimum of the piecewise linear functions $f_c$. In the stochastic approach, the situation is similar, but instead of the pointwise minimum, the leader needs to maximize the pointwise expected value of the functions $f_c$.

Assuming that there is a finite set $U$ of possible follower’s objectives $c$ which occur with probabilities $0 < p_c \leq 1$, respectively, the leader’s objective is thus given as a finite sum of the piecewise linear functions $p_c f_c$. Then the following algorithm solves Problem (SP):

for every $c \in U$, use the algorithm described in Section 2 to compute the piecewise linear function $f_c$ in $O(n \log n)$ time, and multiply each function $f_c$ by the factor $p_c$. Then maximize the resulting weighted sum, which is a piecewise linear function again. Note that the sum can have $O(|U|n)$ segments and that it can be computed by sorting the vertices of all functions and traversing them from left to right while keeping track of the sum of the active linear pieces. This is possible in a running time of $O(|U|n \log(|U|n))$. Thus, we obtain the following result:
Theorem 25. The stochastic bilevel continuous knapsack problem with a distribution having a finite support $U$ can be solved in $O(|U|n \log(|U|/n))$ time.

Note that for any probability distribution, there are only finitely many ways to order the items and therefore only finitely many different piecewise linear functions $f_c$ to include in the weighted sum; the weight reflects the probability that some particular ordering is induced by the random vector $c$. However, the number of orderings having positive probability can be exponential, so that Theorem 25 does not yield an efficient algorithm in general. In fact, in the next section, we will show that such an efficient algorithm cannot exist for some probability distributions unless $P = NP$.

8.2 Componentwise uniform distributions

In this section, we consider the version of (SP) where the distribution of $c$ is uniform on a product of either finite sets or continuous intervals. Equivalently, each component of $c$ is drawn independently and according to some (discrete or continuous) uniform distribution. We will show that Problem (SP) is #P-hard in both cases.

The class #P contains all counting problems associated with decision problems belonging to NP, or, more formally, all problems that ask for computing the number of accepting paths in a polynomial time non-deterministic Turing machine. Using a natural concept of efficient reduction for counting problems, one can define a counting problem to be #P-hard if every problem in #P can be reduced to it. A polynomial time algorithm for a #P-hard counting problem can only exist if $P = NP$. In the following proofs, we will use the #P-hardness of the problem #Knapsack, which asks for the number of feasible solutions of a given binary knapsack instance [9].

In stochastic optimization with continuous distributions, problems often turn out to be #P-hard, and this is often even true for the evaluation of objective functions containing expected values. For an example, see [10], from where we also borrowed some ideas for the following proofs.

Theorem 26. The stochastic bilevel continuous knapsack problem with a discrete componentwise uniform distribution is #P-hard.

Proof. We show the result by a reduction from #Knapsack. More precisely, for some given $a^* \in \mathbb{Z}_{>0}^m$ and $b^* \in \{0, 1, \ldots, \sum_{i=1}^m a_i^*\}$, we will prove that one can compute

$$\#\{x \in \{0, 1\}^m : a^* \top x \leq b^*\}$$

in polynomial time if certain instances of (SP) can be solved in polynomial time. In case $b^* = \sum_{i=1}^m a_i^*$, this is clear, so from now on, assume $b^* < \sum_{i=1}^m a_i^*$.

We define a family of instances of (SP), parameterized by $\tau \in [-1, 1]$: each of the instances has $n := m + 1$ items, where

$$(a_1, \ldots, a_m, a_{m+1}) := (a_1^*, \ldots, a_m^*, \sum_{i=1}^m a_i^*)$$

$$(d_1, \ldots, d_m, d_{m+1}) := ((1 + \tau) \cdot a_1, \ldots, (1 + \tau) \cdot a_m, (-1 + \tau) \cdot a_{m+1})$$.
We set \( b^- := 0 \) and \( b^+ := a_{m+1} = \sum_{i=1}^{m} a_i \), fix \( c_{m+1} := 1 \), and assume
\[(c_1, \ldots, c_m) \sim U(\varepsilon, 1)^m \]
with
\[\varepsilon := \frac{\min_{i=1, \ldots, m} a_i}{2a_{m+1}} > 0.\]

In the following, we deal with slopes of piecewise linear functions \( f \) at certain points \( b \) of their range. In a non-differentiable point, this always refers to the slope of the linear piece directly right of \( b \). The same holds for the notation \( f'(b) \).

The proof consists of two main steps. First, we investigate the structure of the leader’s objective functions for the described instances and show that by determining the slope of any of them at \( b = b^* \), up to a certain precision, we can compute \( \# \{ x \in \{0, 1\}^m : a^* \top x \leq b^* \} \). In the second step, we show how to determine this slope up to the required precision by solving a polynomial number of these instances in a bisection algorithm.

As described in the previous section, the leader’s objective function can be thought of as a weighted sum of piecewise linear functions corresponding to the orderings induced by different choices of \( c \), with weights being the probabilities of the orderings, respectively. For fixed \( c \), consider the set

\[I_c := \{ i \in \{1, \ldots, m\} : \frac{c_i}{a_i} > \frac{c_{m+1}}{a_{m+1}} = \frac{1}{a_{m+1}} \}\]

of items the follower would choose before item \( m+1 \). The corresponding piecewise linear function \( f_c \) first has slope \( 1+\tau \) and then slope \( -1+\tau \), since \( \frac{d}{dx} a_i = 1+\tau \) for all \( i = 1, \ldots, m \), and \( \frac{d_{m+1}}{a_{m+1}} = -1+\tau \). The order of the items in \( I_c \) does not matter to the leader because they all result in the same slope in her objective. The slope changes from \( 1+\tau \) to \( -1+\tau \) at \( b = \sum_{i \in I_c} a_i \). The slope would change back to \( 1+\tau \) at \( b = \sum_{i \in I_c} a_i + a_{m+1} \), but this is outside of the range of the leader’s objective.

The actual leader’s objective is now a weighted sum of such functions. For obtaining the weights, we only need to know the probabilities for different sets \( I_c \), because all \( c \) resulting in the same \( I_c \) also result in the same piecewise linear function. The probability distribution is chosen such that all possible sets \( I_c \subseteq \{1, \ldots, m\} \) have equal probability \( \frac{1}{2^m} \), since each item is contained in \( I_c \) with probability exactly \( \frac{1}{2} \). Indeed, \( c_i = 1 \) means \( \frac{c_i}{a_i} = \frac{1}{a_i} > \frac{1}{a_{m+1}} \), hence \( i \in I_c \), whereas \( c_i = \varepsilon \) means \( \frac{c_i}{a_i} = \frac{\varepsilon}{a_i} < \frac{1}{2a_{m+1}} < \frac{1}{a_{m+1}} \), hence \( i \notin I_c \).

Thus, the leader’s objective function \( f_\tau \) is given as

\[f_\tau = \frac{1}{2^m} \sum_{M \subseteq \{1, \ldots, m\}} f_{M, \tau}\]

where \( f_{M, \tau} \) has slope \( 1+\tau \) for \( b \in [0, \sum_{i \in M} a_i) \) and slope \( -1+\tau \) afterwards. It follows
that, for any $b \in [b^-, b^+]$,

$$
\begin{align*}
    f'_x(b) &= \frac{1}{2m} ((-1 + \tau) \cdot \# \{ M \subseteq \{1, \ldots, m\} : \sum_{i \in M} a_i \leq b \} \\
    &\quad + (1 + \tau) \cdot \# \{ M \subseteq \{1, \ldots, m\} : \sum_{i \in M} a_i > b \} ) \\
    &= -\frac{1}{2m-1} \# \{ M \subseteq \{1, \ldots, m\} : \sum_{i \in M} a_i \leq b \} + 1 + \tau \\
    &= -\frac{1}{2m-1} \# \{ x \in \{0,1\}^m : a^T x \leq b \} + 1 + \tau .
\end{align*}
$$

(1)

This shows that by computing $f'_x(b^*)$ for any fixed $\tau$, we can determine the number $\# \{ x \in \{0,1\}^m : a^T x \leq b^* \} = 2^{m-1}(1 - f'_x(b^*))$. It is even enough to compute an interval of length less than $\frac{1}{2m-1}$ containing $f'_x(b^*)$, because the number of feasible knapsack solutions is an integer and this gives an interval of length less than 1 in which it must lie. This concludes the first step of our proof.

In the second step, we will describe a bisection algorithm to compute $f'_0(b^*)$ up to the required precision. We know that $f'_0(b^*) \in [-1,1]$ as it is a convex combination of values $-1$ and 1. Starting with $s_0^\tau := -1$ and $s_0^\tau := 1$, we iteratively halve the length of the interval $[s_k^-, s_k^+]$ by setting either $s_{k+1}^\tau := s_k^\tau$ and $s_{k+1}^\tau := \frac{1}{2}(s_k^- + s_k^+)$ or $s_{k+1}^- := \frac{1}{2}(s_k^- + s_k^+)$ and $s_{k+1}^+ := s_k^+$. After $m+1$ iterations, we have an interval of length $\frac{1}{2^m}$ containing $f'_0(b^*)$, which allows to compute $\# \{ x \in \{0,1\}^m : a^T x \leq b^* \}$.

It remains to show how to determine whether $f'_0(b^*) < \frac{1}{2}(s_k^- + s_k^+)$ or not, in order to choose the new interval. To this end, we first maximize $f_\tau$ for $\tau := -\frac{1}{2}(s_k^- + s_k^+)$ over $b \in [b^-, b^+]$. This can be done by solving \(\text{STP}\) for the corresponding instance, which by our assumption is possible in polynomial time. Suppose the maximum is attained at $b_{k+1}$. Since $f_\tau$ is concave, we know that $f'_x(b) \geq 0$ for all $b < b_{k+1}$, and $f'_x(b) \leq 0$ for all $b \geq b_{k+1}$. From (1), one can conclude that $f'_x(b) = f'_0(b) + \tau$ for all $\tau \in [-1,1]$ and all $b \in [b^-, b^+]$. We derive that $f'_0(b^*) \geq -\tau = \frac{1}{2}(s_k^- + s_k^+)$ if $b^* < b_{k+1}$, and $f'_0(b^*) \leq -\tau = \frac{1}{2}(s_k^- + s_k^+)$ otherwise.

**Theorem 27.** The stochastic bilevel continuous knapsack problem with a continuous componentwise uniform distribution is \#P-hard.

**Proof.** The result can be shown by a similar proof to the one of Theorem 20 instead of the discrete distribution used before, the continuous distribution

$$(c_1, \ldots, c_m) \sim \mathcal{U} \prod_{i=1}^m \left[ \frac{a_i}{2a_{m+1}}, \frac{3a_i}{2a_{m+1}} \right]$$

is considered, again fixing $c_{m+1} = 1$. The sets $I_c$ are defined as before and it can be shown again that each set has probability $\frac{1}{2^m}$: if $c_i \in \left( \frac{2a_i}{2a_{m+1}}, \frac{3a_i}{2a_{m+1}} \right]$, we have that $\frac{a_i}{a_i} > \frac{1}{a_{m+1}}$, hence $i \in I_c$, while $c_i \in [\frac{a_i}{2a_{m+1}}, \frac{2a_i}{2a_{m+1}}]$ implies $\frac{a_i}{a_i} < \frac{1}{a_{m+1}}$, so that $i \notin I_c$. Both events have probability $\frac{1}{2}$. The rest of the proof is analogous. \qed
Remark 28. In the constructions in both preceding proofs, the case \( \frac{\omega_i}{a_i} = \frac{1}{a_{m+1}} \) occurs with probability zero, for all \( i \in \{1, \ldots, m\} \). It thus follows that both results hold for the optimistic as well as the pessimistic approach.

To conclude this section, we can again state that even evaluating the leader’s objective function is hard:

**Theorem 29.** Evaluating the objective function of the stochastic bilevel continuous knapsack problem with a discrete or continuous componentwise uniform distribution is \#P-hard.

**Proof.** Using the same construction and notation as in the preceding proofs, we have shown that computing \( f_0'(b^*) \) is \#P-hard. Since \( f_0 \) is linear on \([b^*, b^*+1]\) by the integrality of \( a^* \), we obtain \( f_0'(b^*) = f_0(b^*+1) - f_0(b^*) \). Hence, it is as hard to evaluate \( f_0 \) as it is to compute \( f_0'(b^*) \). \( \qed \)

9 Conclusion

We have started the investigation of robust bilevel optimization by addressing the bilevel continuous knapsack problem with uncertain follower’s objective. It turned out that standard results from single-level robust optimization do not hold anymore: firstly, it is not possible in general to replace an uncertainty set by its convex hull without changing the problem. Secondly, the case of interval uncertainty is not trivial anymore. Even if we have shown that it is still tractable in the case of the bilevel continuous knapsack problem, we conjecture that in general the interval case cannot be reduced to the certain problem anymore. In the stochastic optimization approach, we showed that the problem is intractable already in the case where every objective coefficient is continuously uniformly distributed, which corresponds to interval uncertainty in robust optimization.

All hardness results presented in this paper are in the weak sense. This leaves open the question whether the corresponding problem variants are actually strongly NP-hard (or strongly \#P-hard in the stochastic case) or whether pseudopolynomial algorithms exist. This is left as future work.

References


