Math. Program., Ser. A (2017) 163:1–23
DOI 10.1007/s10107-016-1053-z

FULL LENGTH PAPER

Min–max–min robust combinatorial optimization

Christoph Buchheim[1] · **Jannis Kurtz**[1]

Received: 27 January 2016 / Accepted: 2 July 2016 / Published online: 14 July 2016

Abstract The idea of k-adaptability in two-stage robust optimization is to calculate a fixed number k of second-stage policies here-and-now. After the actual scenario is revealed, the best of these policies is selected. This idea leads to a min–max–min problem. In this paper, we consider the case where no first stage variables exist and propose to use this approach to solve combinatorial optimization problems with uncertainty in the objective function. We investigate the complexity of this special case for convex uncertainty sets. We first show that the min–max–min problem is as easy as the underlying certain problem if k is greater than the number of variables and if we can optimize a linear function over the uncertainty set in polynomial time. We also provide an exact and practical oracle-based algorithm to solve the latter problem for any underlying combinatorial problem. On the other hand, we prove that the min–max–min problem is NP-hard for every fixed number k, even when the uncertainty set is a polyhedron, given by an inner description. For the case that k is smaller or equal to the number of variables, we finally propose a fast heuristic algorithm and evaluate its performance.

A preliminary version of this paper will appear in the Proceedings of the International Network Optimization Conference 2015 [9].

This work has partially been supported by the German Research Foundation (DFG) within the Research Training Group 1855.

✉ Jannis Kurtz
jannis.kurtz@math.tu-dortmund.de

Christoph Buchheim
christoph.buchheim@math.tu-dortmund.de

1 Technische Universität Dortmund, Vogelpothsweg 87, 44227 Dortmund, Germany

Keywords Robust optimization · k-Adaptability · Complexity

Mathematics Subject Classification 90C27 · 90C57

1 Introduction

The robust optimization approach, designed for tackling the uncertainty that is present in the parameters of many optimization problems, was introduced by Soyster [19] in 1973 and has received increasing attention since the seminal works of Ben-Tal and Nemirovski [3], El Ghaoui et al. [11], and Kouvelis and Yu [15] in the late 1990s. More recently, the focus of research has moved to the development of new approaches that try to avoid, or at least reduce, the so-called price of robustness [6]: since the original robust optimization approach asks for a worst-case optimal solution, this solution can be very conservative and hence far from optimal in the actual scenario.

In this paper, we are interested in robust counterparts of combinatorial optimization problems of the form

$$\min_{x\in X} \; c^\top x, \tag{M}$$

where $X \subseteq \{0,1\}^n$ contains the incidence vectors of all feasible solutions of the given problem and the objective function vector $c \in \mathbb{R}^n$ is assumed to be uncertain. As is common in robust optimization, we assume that a set U of objective function vectors is given, called the *uncertainty set* of the problem. In the strictly robust optimization approach [3], the aim is to find the worst-case optimal solution when taking into account all scenarios in U. This leads to the min–max problem

$$\min_{x\in X} \; \max_{(c,c_0)\in U} \; c^\top x + c_0 \tag{M^2}$$

with $U \subseteq \mathbb{R}^{n+1}$. In general, Problem (M^2) turns out to be NP-hard even for feasible sets X for which Problem (M) is tractable, both in the case of finite or polyhedral uncertainty sets [15] and in the case of ellipsoidal uncertainty sets [18]. This remains true even in the special case where the constant c_0 is certain, i.e., all vectors in U share that same last entry.

As mentioned above, the main drawback of the min–max robust approach is the price of robustness. A first method to address this problem was the so-called *gamma-uncertainty* presented by Bertsimas and Sim [5], allowing to define a parameter Γ to control the maximum number of parameters which may deviate from a given nominal value in a constraint or in the objective function. Liebchen et al. [16] proposed the concept of *recovery robustness*. Here a set of algorithms $\mathcal{A}$ is given and the objective is to find a solution x such that for every possible scenario ξ there exists an algorithm $A \in \mathcal{A}$ such that A applied to x and ξ constructs a solution which is feasible for scenario ξ. Büsing [10] used this idea to define a model for the shortest path problem which allows to change a fixed number k of edges of a given path in a second stage. Ben-Tal et al. [2] proposed the so-called *adjustable robustness*, where the set of variables is decomposed into *here and now* variables x and *wait and see* variables y. The objective is to find a solution x such that for all possible scenarios there exists a y such that (x, y) is feasible and minimizes the worst case.

Bertsimas and Caramanis [4] introduced the concept of k-adaptability. The idea is to compute k second-stage policies here-and-now; the best of these policies is chosen once the scenario is revealed. The authors analyze the gap between the static problem and the k-adaptability problem and give necessary conditions under which a certain level of improvement is achieved. Moreover, they prove that the problem is NP-hard in its general form, and devise a bilinear formulation for the 2-adaptability problem. The idea of k-adaptability was later used by Hanasusanto et al. [14] to approximate two-stage robust binary programs. This leads to a min–max–min problem for which the authors show that, if the uncertainty only occurs in the objective function, it suffices to calculate $n + 1$ second-stage policies to reach the optimal value of the two-stage problem. Still in the case of objective uncertainty and for polyhedral uncertainty sets, the authors provide a MILP formulation to solve the problem. When the uncertainty also occurs in the constraints, it is shown that the evaluation of the objective function can be performed in polynomial time if the number of second-stage policies is fixed but becomes strongly NP-hard otherwise. For this case, the authors devise a mixed-integer bilinear program depending on a parameter ε that approximates the problem arbitrarily well when ε tends to zero.

In this paper, we consider the k-adaptability approach in the case where no first stage variables exist. We propose to apply it for solving combinatorial optimization problems with uncertain objective functions, since such problems are naturally modeled by using only one stage. Instead of asking for a single worst-case optimal solution $x \in X$ as in (M^2), we thus aim at calculating k solutions $x^{(1)}, \dots, x^{(k)} \in X$, allowing to choose the best of them once the actual scenario is revealed. In fact, since in our case no first-stage variables exist, we can interpret the latter calculation as a robust preprocessing, concerning all possible scenarios, which is done before any solution is implemented. As a typical application for such an approach, consider a parcel service delivering to the same customers every day. Each morning, the company needs to determine a tour depending on the current traffic situation. However, computing an optimal tour from scratch may take too long in a real-time setting. Instead, in our approach a set of candidate tours is computed once and the company can choose the best one out of these solutions every morning. Apart from yielding better solutions in general compared to the min–max approach, this approach has the advantage that the solutions are more easily accepted by a human user if they do not change each time but are taken from a relatively small set of candidate solutions.

As mentioned before, the k-adaptability approach leads to a min–max–min problem. In our case, the latter is of the form

$$\min_{x^{(1)},\dots,x^{(k)} \in X} \; \max_{(c,c_0)\in U} \; \min_{i=1,\dots,k} c^\top x^{(i)} + c_0. \tag{M^3}$$

The main objective of this paper is to determine the computational complexity of Problem (M^3) for convex uncertainty sets U, which among others include polyhedral or ellipsoidal uncertainty sets. The complexity of course depends on the underlying set X or, more precisely, on the complexity of the corresponding certain problem (M). We will assume throughout that the set X is given implicitly by a linear optimization oracle for the certain problem (M).

Our main result is that Problem (M^3) is as easy as the underlying certain problem (M) if we can optimize a linear function over U in polynomial time and if $k \geq n+1$. This is in contrast to the NP-hardness of the more general problems studied in [14]. Note that for $k = n + 1$ the selection of the best solution out of the candidate set can be performed in $O(n^2)$ time once the scenario is revealed, independently of the feasible set X. To solve Problem (M^3) for $k \geq n + 1$, we provide an oracle-based algorithm that is applicable for any underlying combinatorial structure X, where the uncertainty set U can be specified by any linear optimization oracle. The second main result of this paper is that Problem (M^3) is NP-hard if the number of solutions k is fixed. As a corollary, it follows that the more general two-stage problem studied in [14] is NP-hard, which has not been proved yet. We also propose and evaluate a heuristic algorithm for Problem (M^3) for the case $k < n+1$ based on the algorithm for $k = n+1$, which turns out to yield good solutions in very short running times in our numerical experiments.

2 Preliminaries

In the next section, we will show that Problem (M^3) becomes as easy as the underlying certain optimization problem as soon as $k \geq n + 1$. The first step in the proof is the following reformulation, which follows from the proof of Theorem 1 in [14] by a straightforward generalization.

Lemma 1 *Let $U \subseteq \mathbb{R}^{n+1}$ be a non-empty convex set. Then*

$$\min_{x^{(1)},\ldots,x^{(k)}\in X} \max_{(c,c_0)\in U} \min_{i=1,\ldots,k} c^\top x^{(i)} + c_0 = \min_{x\in X(k)} \max_{(c,c_0)\in U} c^\top x + c_0$$

where

$$X(k) := \left\{ \sum_{i=1}^{k} \lambda_i x^{(i)} \mid \lambda_i \geq 0,\ \sum_{i=1}^{k} \lambda_i = 1, \quad x^{(i)} \in X \text{ for } i = 1,\ldots,k \right\}$$

is the set of all convex combinations of k elements of X.

In the following, we will thus consider the problem

$$\min_{x\in X(k)} \max_{(c,c_0)\in U} c^\top x + c_0 \tag{1}$$

in order to solve Problem (M^3). From Lemma 1 and Carathéodory's theorem, we immediately obtain

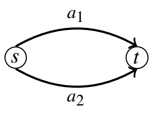
Corollary 1 *For $k \geq n + 1$ and for each non-empty convex set U we have*

$$\min_{x^{(1)},\ldots,x^{(k)}\in X} \max_{(c,c_0)\in U} \min_{i=1,\ldots,k} c^\top x^{(i)} + c_0 = \min_{x\in conv(X)} \max_{(c,c_0)\in U} c^\top x + c_0.$$

In the special case where no uncertain constant c_0 is considered, the objective function $\max_{c\in U} c^\top x$ is linear on any line through the origin. Therefore its optimum is obtained over the boundary of $\operatorname{conv}(X)$. Since the latter agrees with the boundary of $X(n)$, we obtain the latter result for all $k \geq n$ then.

Corollary 1 implies that considering more than $n+1$ solutions will not lead to any further improvement in the objective value of Problem (M^3), which was also shown in [14]. On the other hand, $k = n+1$ is a reasonable choice for the proposed application of our approach, namely to compute k alternative solutions in a preprocessing step and then to choose the best one out of these solutions every time a new scenario occurs. For $k = n+1$, the latter task can be performed in $O(n^2)$ time, as it reduces to computing the objective values of all $n+1$ solutions.

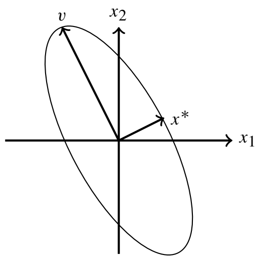
Example 1 Using Lemma 1, we give an example showing that the difference between the optimal values of (M^2) and (M^3) can be arbitrarily large. Consider the shortest path problem on the graph $G = (V, A)$ with $V = \{s, t\}$ and two edges $A = \{a_1, a_2\}$ both leading from s to t:



Let an ellipsoidal uncertainty set $U_\alpha = \left\{c \in \mathbb{R}^2 \mid c^\top \Sigma_\alpha c \leq 1\right\}$ be given, with positive definite matrix

$$\Sigma_\alpha = \begin{pmatrix} 4+\frac{1}{\alpha} & 2-\frac{2}{\alpha} \\ 2-\frac{2}{\alpha} & 1+\frac{4}{\alpha} \end{pmatrix} = 9(x^*)(x^*)^\top + \frac{1}{\alpha} v v^\top$$

for $\alpha \geq 1$, where $x^* = (\frac{2}{3}, \frac{1}{3})^\top$ and $v = (-1, 2)^\top$. The corresponding ellipsoid is given as follows:



Increasing α leads to a scaling of the ellipsoid in direction of v. The optimal solution of (M^2) is the path a_1 with value $\frac{1}{5}\sqrt{4+\alpha}$, which can be arbitrarily large if we increase α. On contrary, to solve Problem (M^3) for $k = 2$ we can use reformulation (1), then

$$x^* = \tfrac{2}{3}a_1 + \tfrac{1}{3}a_2 \in X(2)$$

is a feasible solution with an objective value of $\frac{1}{3}$, independently of α.

In the following, we investigate the solution of Problem (M^3) for $k = n + 1$. Based on Corollary 1, we propose the following two-stage approach: in the first step, we calculate an optimal solution x^* of the continuous problem

$$\min_{x \in \operatorname{conv}(X)} \max_{(c,c_0) \in U} c^\top x + c_0. \tag{2}$$

This step depends on the uncertainty set U and the underlying feasible set X. In the second step, we calculate a corresponding set of solutions $x^{(1)}, \ldots, x^{(n+1)}$ of Problem (M^3). The next result, following directly from Theorem 6.5.11 in [13], shows that the second step can be performed in polynomial time if the underlying certain problem can be solved in polynomial time.

Lemma 2 *Assume we are given an optimization oracle for the certain problem*

$$c \mapsto \min_{x \in X} c^\top x.$$

If $x^* \in \operatorname{conv}(X)$ is rational, then, in polynomial time, we can compute affinely independent vectors $x^{(1)}, \ldots, x^{(m)} \in X$ and rational coefficients $\lambda_1, \ldots, \lambda_m \geq 0$ with $\sum_{i=1}^m \lambda_i = 1$ such that $x^* = \sum_{i=1}^m \lambda_i x^{(i)}$ and $m \leq n + 1$.

Note that the algorithm given in [13], which is used in the latter lemma, computes a convex combination with the smallest possible m. The remaining task in the case $k = n + 1$ is thus to solve (2). In the following section, we investigate the complexity of the latter problem. For this, we use two further important results from [13], which we report here for convenience of the reader.

First note that for general convex sets the famous equivalence between optimization and separation does not hold in the strong sense. Instead, we have to take into account irrational values and therefore need a parameter $\varepsilon > 0$ which determines the accuracy of calculations. For an arbitrary convex set K we define

$$B_\varepsilon(K) = \{x \in \mathbb{R}^n \mid \|x - y\| \leq \varepsilon \text{ for some } y \in K\}$$

and

$$B_{-\varepsilon}(K) = \{x \in K \mid B_\varepsilon(x) \subseteq K\},$$

so that $B_{-\varepsilon}(K) \subset K \subset B_\varepsilon(K)$. Then a full-dimensional compact convex set K is called *centered convex body* if the following information is explicitly given: the integer n such that $K \subseteq \mathbb{R}^n$, a positive $R \in \mathbb{Q}$ such that $K \subseteq B_R(0)$, and some $r \in \mathbb{Q}$ and $a_0 \in \mathbb{Q}^n$ such that $B_r(a_0) \subseteq K$. We then write $K(n, R, r, a_0)$ and define the encoding length of K as the sum of the encoding lengths of R, r, and a_0.

Theorem 1 [13] *Let $\varepsilon > 0$ be a rational number, $K(n, R, r, a_0)$ a centered convex body given by a weak membership oracle and $f : \mathbb{R}^n \to \mathbb{R}$ a convex function given by an oracle which returns for every $x \in \mathbb{Q}^n$ and $\delta > 0$ a rational number t such that $|f(x) - t| \leq \delta$. Then there exists an oracle-polynomial time algorithm in the encoding length of K and* $\log \varepsilon$ *that returns a vector $y \in B_\varepsilon(K)$ such that $f(y) \leq f(x) + \varepsilon$ for all $x \in B_{-\varepsilon}(K)$.*

A weak membership oracle for K has to assert, for any given point $y \in \mathbb{Q}^n$ and any rational value $\varepsilon > 0$, either that $y \in B_\varepsilon(K)$ or that $y \notin B_{-\varepsilon}(K)$. Note that by definition at least one of the two assertions is true.

The problem solved in Theorem 1 can be seen as finding a vector almost in K which almost maximizes the objective function over all vectors which are deep in K. Since we ultimately want to find an almost optimal vector contained in K, we have to round the solution given by Theorem 1. This can be done in polynomial time in the cases we are interested in, as stated by the following result.

Lemma 3 [13] *Let $P \subseteq \mathbb{R}^n$ be a polyhedron such that each facet has encoding length at most $\varphi \in \mathbb{N}$ and let $v \in B_{2^{-6n\varphi}}(P)$. Then we can calculate $q \in \mathbb{Z}$ with $0 < q < 2^{4n\varphi}$ and a vector $w \in \mathbb{Z}^n$ in polynomial time such that*

$$\|qv - w\| < 2^{-3\varphi}$$

and such that $\frac{1}{q}w$ is contained in P.

3 Complexity for $k \geq n + 1$

The min–max problem (M^2) is well-known to be NP-hard for most classical combinatorial optimization problems when U is a general polytope, an ellipsoid, or a finite set, e.g., for the shortest path problem, the minimum spanning tree problem, or even in the unconstrained case with $X = \{0, 1\}^n$. Only few cases are known where (M^2) remains tractable (without uncertain constant), e.g., if U is an axis-parallel ellipsoid and $X = \{0, 1\}^n$ [1], if X corresponds to a matroid [17] or if U is a budgeted uncertainty set [5]. In particular, the min–max–min problem (M^3) is NP-hard in general for $k = 1$ for ellipsoidal, polyhedral and finite uncertainty.

In contrast to this, we show that Problem (M^3) is solvable in polynomial time for both polyhedral and ellipsoidal uncertainty sets whenever the underlying certain problem is solvable in polynomial time and $k \geq n + 1$. This result holds even if U is an arbitrary non-empty convex set provided that we can optimize any linear function over U in polynomial time. We will first consider the (easier) case of polyhedral sets U in Sect. 3.1 before dealing with the general case in Sect. 3.2.

3.1 Polyhedral uncertainty

In the case of a non-empty polyhedral uncertainty set U, we can show that Problem (M^3) is tractable as soon as the underlying certain problem is tractable and $k \geq n + 1$. More precisely, we have

Theorem 2 *Let $k \geq n+1$. Given an optimization oracle for the problem*

$$c \mapsto \min_{x \in X} c^\top x,$$

for any polyhedron $U = \{(c, c_0) \in \mathbb{R}^{n+1} \mid A(c, c_0)^\top \leq b\}$ with A and b rational we can solve (M^3) in polynomial time in the encoding length of (n, A, b).

Proof Let $A(c, c_0) = \bar{A}c + c_0 a$. Then, using Corollary 1, Problem (M^3) is equivalent to

$$\min_{x \in \mathrm{conv}(X)} \max \{c^\top x + c_0 \mid \bar{A}c + c_0 a \leq b,\ c \in \mathbb{R}^n,\ c_0 \in \mathbb{R}\}. \tag{3}$$

Replacing the inner linear program by its dual, we obtain

$$\min_{x \in P} b^\top y \tag{4}$$

with

$$P := \{(x, y) \in \mathbb{R}^n \times \mathbb{R}^m \mid x \in \mathrm{conv}\,(X)\,,\, y^\top \bar{A} = x, y^\top a = 1, y \geq 0\}.$$

Now using the famous Theorem 6.4.9 in [13], for solving Problem (4) in polynomial time it suffices to devise a polynomial-time algorithm for the strong separation problem for P. By the same theorem and since

$$\min_{x \in X} c^\top x = \min_{x \in \mathrm{conv}(X)} c^\top x,$$

the separation problem for conv (X) can be solved in polynomial time using the given oracle. On the other hand, the set

$$Q := \{(x, y) \in \mathbb{R}^n \times \mathbb{R}^m \mid y^\top \bar{A} = x,\ y^\top a = 1,\ y \geq 0\}$$

is a rational polyhedron and every point (x, y) can be separated by checking whether all equations are satisfied. The combination of both algorithms thus yields a separation algorithm for P and hence a polynomial-time algorithm for solving (3). As the computed optimal solution x^* is rational, the result follows from Lemma 2. □

3.2 General convex uncertainty

In the following, we assume that U is a non-empty convex set for which we have a weak optimization oracle, i.e., that for given $x \in \mathbb{Q}^n$ and rational $\varepsilon > 0$ we can compute in polynomial time a vector $(c, c_0) \in U \cap \mathbb{Q}^{n+1}$ with

$$c^\top x + c_0 \geq d^\top x + d_0 - \varepsilon \quad \text{for all } (d, d_0) \in U.$$

Moreover, we assume that U is bounded by a constant M, i.e., that

$$\|(c, c_0)\| \leq M \quad \text{for all } (c, c_0) \in U.$$

Note that our assumptions hold for polytopal uncertainty sets, but also for the important case of ellipsoidal uncertainty: if

$$U = \left\{(c, c_0) \in \mathbb{R}^{n+1} \mid \left(\binom{c}{c_0} - \bar{c}\right)^\top \Sigma^{-1}\left(\binom{c}{c_0} - \bar{c}\right) \leq 1\right\},$$

with $\bar{c} \in \mathbb{Q}^{n+1}$ denoting the center of the ellipsoid and $\Sigma \in \mathbb{Q}^{(n+1)\times(n+1)}$ being a positive definite symmetric matrix, we have

$$\max_{(c,c_0)\in U} c^\top x + c_0 = \bar{c}^\top \binom{x}{1} + \sqrt{\binom{x}{1}^\top \Sigma \binom{x}{1}}$$

for all $x \in \mathbb{Q}^n$, and M can be chosen as $\|\bar{c}\| + \lambda_{\min}(\Sigma)^{-1}$ with $\lambda_{\min}(\Sigma)$ being the minimal eigenvalue of Σ.

Theorem 3 *Let conv (X) be full-dimensional, $\varepsilon \in (0, 1) \cap \mathbb{Q}$, and U as above. Given an optimization oracle for the certain problem*

$$c \mapsto \min_{x\in X} c^\top x,$$

we can solve Problem (M^3) up to an accuracy of at most ε in time polynomial in $(n, \log M, \log \varepsilon)$ if $k \geq n + 1$.

Before we prove the latter theorem, we need to show two technical lemmas.

Lemma 4 *Let $f : \mathbb{R}^n \to \mathbb{R}$ be defined by $f(x) := \max_{(c,c_0)\in U} c^\top x + c_0$ where U is a convex set bounded by M. If $x, y \in \mathbb{R}^n$ with $\|x - y\| \leq \varepsilon$ then*

$$f(x) - f(y) \leq M\varepsilon.$$

Proof Let $x, y \in \mathbb{R}^n$ with $\|x - y\| \leq \varepsilon$. From $c^\top x + c_0 = c^\top(x - y) + c^\top y + c_0$ we obtain

$$\max_{(c,c_0)\in U} c^\top x + c_0 \leq \max_{(c,c_0)\in U} c^\top(x - y) + \max_{(c,c_0)\in U} c^\top y + c_0$$

and thus

$$f(x) - f(y) \leq \max_{(c,c_0)\in U} \|c\|\|x - y\| \leq M\varepsilon$$

by the Cauchy–Schwarz inequality. □

The following lemma states that if we can optimize over all elements deep in a convex set K with an arbitrary accuracy then we can optimize over all elements in K with an arbitrary accuracy.

Lemma 5 *Let f and U be as in Lemma 4. Let K be any convex set for which we know a radius $R > 0$ with $K \subseteq B_R(0)$ and for which we know that K contains a ball with radius r. Additionally, let $0 < \varepsilon < r$ and $x^* \in B_\varepsilon(K)$ such that for all $y \in B_{-\varepsilon}(K)$*

$$f(x^*) \leq f(y) + \varepsilon.$$

Then for all $y \in K$

$$f(x^*) \leq f(y) + \varepsilon\left(1 + \tfrac{2R}{r}M\right).$$

Proof Formula (0.1.14) in [13] states that

$$K \subseteq B_{\frac{2R\varepsilon}{r}}\left(B_{-\varepsilon}(K)\right)$$

for all $0 < \varepsilon < r$. Hence for all $y \in K$ there exists some $z_y \in B_{-\varepsilon}(K)$ such that $\|z_y - y\| \leq \frac{2R}{r}\varepsilon$. From Lemma 4 we obtain

$$f(z_y) - f(y) \leq M\tfrac{2R}{r}\varepsilon.$$

By our assumption on x^*, we derive

$$f(x^*) \leq f(z_y) + \varepsilon \leq f(y) + \varepsilon\left(1 + \tfrac{2R}{r}M\right).$$

which proves the result. □

Proof (of Theorem 3) By Corollary 1 we have

$$\min_{x^{(1)},\ldots,x^{(k)} \in X} \max_{(c,c_0)\in U} \min_{i=1,\ldots,k} c^\top x^{(i)} + c_0 = \min_{x \in \text{conv}(X)} \max_{(c,c_0)\in U} c^\top x + c_0.$$

Again define $f(x) := \max_{(c,c_0)\in U} c^\top x + c_0$ and note that f is a convex function, as it is defined as the maximum of affine–linear functions. The basic idea of the proof is to use Theorem 1 to calculate a near-optimal point $x^* \in B_{\varepsilon'}(\text{conv}(X))$, for an appropriately defined ε', and to use Lemma 3 to round x^* to a point in conv (X) which is optimal up to the given ε. As $X \subseteq \{0,1\}^n$ and conv (X) is full-dimensional, we have

$$B_r(x_0) \subseteq \text{conv}(X) \subseteq B_R(0)$$

for an appropriate rational center point $x_0 \in \text{conv}(X)$ and radii r and R that are polynomial in n. Hence conv (X) satisfies all assumptions of Theorem 1. All vertices of conv (X) have encoding length at most n and therefore all facets have encoding length at most $3n^3$ by Lemma 6.2.4 in [13]. Choose $\varphi \geq 3n^3$ such that $2^{-3\varphi} \leq \frac{\varepsilon}{2M}$, then also

$$\varepsilon' := \min\left\{\frac{\frac{\varepsilon}{2}}{1+\frac{2R}{r}M}, 2^{-6n\varphi}\right\}$$

has encoding length polynomial in $(n, \log M, \log \varepsilon)$ if φ is chosen polynomial in $(\log M, \log \varepsilon)$. Using the optimization oracle we can separate from conv (X) in polynomial time and in particular solve the weak membership problem in polynomial time. By Theorem 1 we can compute a rational $x^* \in B_{\varepsilon'}(\text{conv}\,(X))$ with $f(x^*) \leq f(y) + \varepsilon'$ for all $y \in B_{-\varepsilon'}(\text{conv}\,(X))$, in time polynomial in the binary encoding length of n and ε'. We may assume $M \geq 1$, then $\varepsilon' < r$ so that Lemma 5 can be applied. Hence

$$f(x^*) \leq f(y) + \varepsilon'(1 + \tfrac{2R}{r}M) \leq f(y) + \tfrac{\varepsilon}{2}$$

for all $y \in \text{conv}\,(X)$. Since the solution x^* is not necessarily contained in conv (X), we apply Lemma 3 to round x^*. For φ chosen as above and $v := x^*$, by Lemma 3 we can calculate $q \in \mathbb{Z}$ and $w \in \mathbb{Z}^n$ with $\|qx^* - w\| < 2^{-3\varphi}$, in polynomial time in the input. Since $x^* \in B_{\varepsilon'}(\text{conv}\,(X)) \subseteq B_{2^{-6n\varphi}}(\text{conv}\,(X))$, from the latter lemma it follows that $x' := \frac{1}{q}w$ is contained in conv (X). Moreover

$$\|x^* - x'\| = \tfrac{1}{q}\|qx^* - w\| < \tfrac{1}{q}2^{-3\varphi}.$$

By the choice of φ above, we obtain $\|x^* - x'\| < \frac{\varepsilon}{2M}$ and therefore by Lemma 4 we have $f(x') - f(x^*) \leq \frac{\varepsilon}{2}$. By the calculations above we obtain

$$f(x') \leq \frac{\varepsilon}{2} + f(x^*) \leq f(y) + \varepsilon$$

for all $y \in \text{conv}\,(X)$. Now by Lemma 2 we can calculate, in polynomial time, solutions $x^{(1)}, \ldots, x^{(n+1)} \in X$ with $x' = \sum_{i=1}^{n+1} \lambda_i x^{(i)}$ for appropriate $\lambda \geq 0$ with $\sum_{i=1}^{n+1} \lambda_i = 1$. Then $x^{(1)}, \ldots, x^{(n+1)}$ is the desired approximate solution of Problem (M^3). □

Note that the condition that conv (X) must be full-dimensional is only technical (see Section 6.1.2 in [13]). We can achieve a separation algorithm for lower dimensional sets by the following idea: if conv (X) is $(n-1)$-dimensional, then we can calculate the hyperplane in $\mathbb{R}^n$ which contains conv (X) in polynomial time by [12]. But then every point which is not contained in this hyperplane can be separated using the hyperplane itself. So the problem is reduced to the full-dimensional separation problem in dimension $n-1$ which is equivalent to the optimization problem in dimension $n-1$.

4 Practical algorithm for $k \geq n + 1$

The algorithm underlying Theorem 3 makes heavy use of the ellipsoid method and is thus not practical. In this section, we propose a variant of this algorithm that does not provably run in polynomial time, but works well in practice. It is based on the idea of column generation and uses two oracles: one for linear optimization over U

and one for solving the certain problem (M). Except for these oracles, the algorithm is independent of the considered type of uncertainty U and of the underlying combinatorial structure X. In particular, we can use any combinatorial algorithm for solving the certain problem (M); no polyhedral description of conv (X) is needed. This approach can be seen as an application of the algorithm presented in [8] to the dual problem of (2).

The algorithm is stated below. It does not only return the optimal solution set $\{x^{(1)}, \ldots, x^{(n)}\}$ of Problem (M^3), but also computes the optimal solution x^* of Problem (2) as well as the coefficients defining x^* as a convex combination of $x^{(1)}, \ldots, x^{(n)}$. The latter are used in Sect. 6 below when defining a heuristic algorithm for the case $k \leq n$.

Algorithm 1 Algorithm to solve Problem (M^3) for $k \geq n+1$

```
Input: $U$, $X$
Output: optimal solution of Problem (M$^3$) (and Problem 2)
1: $i := 0$
2: repeat
3:     calculate optimal solution $(z^*, (c^*, c_0^*))$ of
           $\max \{z \mid c^\top \bar{x}_j + c_0 \geq z \; \forall j = 1, \ldots, i, \; z \in \mathbb{R}, \; (c, c_0) \in U\}$
4:     calculate optimal solution $\bar{x}_{i+1}$ of
           $\min \{(c^*)^\top x + c_0^* \mid x \in X\}$
5:     $i := i + 1$
6: until $(c^*)^\top \bar{x}_i + c_0^* \geq z^*$
7: calculate a basic feasible solution of the linear system
       $z^* - c_0^* = \sum_{j=1}^i \lambda_j (c^*)^\top \bar{x}_j, \; \sum_{j=1}^i \lambda_j = 1, \; \lambda \geq 0$
8: $X^* := \{\bar{x}_j \mid \lambda_j > 0, \; j = 1, \ldots, i\}$
9: return $X^*$ and $x^* := \sum_{j=1}^i \lambda_j \bar{x}_j$
```

Theorem 4 *Algorithm 1 is correct and terminates in finite time.*

Proof First note that, for every subset $X' \subseteq X$, we have

$$\begin{aligned} \min_{x \in \mathrm{conv}(X')} \max_{(c,c_0) \in U} c^\top x + c_0 &= \max_{(c,c_0) \in U} \min_{x \in \mathrm{conv}(X')} c^\top x + c_0 \\ &= \max_{(c,c_0) \in U} \{z \mid z \leq c^\top x + c_0 \; \forall x \in \mathrm{conv}\left(X'\right)\} \\ &= \max_{(c,c_0) \in U} \{z \mid z \leq c^\top x + c_0 \; \forall x \in X'\} \end{aligned}$$

by the classical min–max theorem. To prove correctness of Algorithm 1, note that $(c^*)^\top x + c_0^* \geq z^*$ holds for all $x \in X$ after termination of the loop. Hence

$$\max_{(c,c_0) \in U} \{z \mid z \leq c^\top \bar{x}_j + c_0 \; \forall j = 1, \ldots, i\} = \max_{(c,c_0) \in U} \{z \mid z \leq c^\top x + c_0 \; \forall x \in X\}$$

and therefore, by the equivalence above, an appropriate convex combination of the calculated solutions $\bar{x}_j$ yields an optimal solution of Problem (2). Note that the convex combination of x^* calculated in Step 7 has the same objective value z^* and uses at most $n+1$ solutions $\bar{x}_j$. Finite running time follows directly from the finiteness of X. $\square$

Algorithm 1 does not run in polynomial time in general. However, in the following we give some evidence that it performs very well in practice. To solve problem (M^3), we implemented Algorithm 1 for the knapsack problem

$$X := \{x \in \{0,1\}^n \mid a^\top x \leq b\}$$

with ellipsoidal uncertainty for the profits and certain constant, i.e., for

$$U := \left\{(c,0) \in \mathbb{R}^{n+1} \mid (c-\bar{c})^\top \Sigma^{-1}(c-\bar{c}) \leq \Omega^2\right\},$$

where $\Sigma \in \mathbb{R}^{n\times n}$ is a symmetric positive semidefinite matrix and $\Omega \in \mathbb{N}$. Problem (M^3) for $k \geq n$ is then equivalent to

$$\min_{x\in \operatorname{conv}(X)} \bar{c}^\top x + \Omega\sqrt{x^\top \Sigma x}.$$

For our experiments, we created instances similar to those used in [7]: for any $n \in \{250, 500, 750\}$ we created 10 random knapsack instances together with 10 random ellipsoids. The weights a_i were chosen randomly from the set $\{100,\ldots,1500\}$ and b was set to $100n$. The ellipsoid center $\bar{c}$ was chosen randomly with $\bar{c}_i \in \{10{,}000,\ldots,15{,}000\}$. The extreme rays of the ellipsoid were calculated as random orthonormal bases where the length of the rays were chosen randomly as $\sqrt{\delta_j c_j}$, where δ_j is a random number in $[0,1]$. Note that the resulting ellipsoids are not axis-parallel in general. For any instance, we scaled the ellipsoid by varying the parameter Ω from 1 to 5. Additionally, to compare our algorithm to the MILP formulation given in [14], we implemented our algorithm for the knapsack problem with gamma-uncertainty sets

$$U^\Gamma := \left\{(c,0) \in \mathbb{R}^{n+1} \mid c_i = \bar{c}_i + \delta_i d_i,\ \sum_{i=1}^n \delta_i \leq \Gamma\right\},$$

where Γ is a given parameter. Again we created 10 random knapsack instances as above, each equipped with a gamma-uncertainty set. As in [18], the mean vector $\bar{c}$ was chosen randomly with $\bar{c}_i \in \{10{,}000,\ldots,15{,}000\}$, and d_i was set to $0.1\bar{c}_i$ for all $i = 1,\ldots,n$. Each instance has been solved for all values of Γ from the set $\{0.05n, 0.1n, 0.15n, 0.25n, 0.5n\}$, rounded down if fractional.

Concerning the oracles for Algorithm 1, we used CPLEX 12.5 to solve the second-order cone program in Step 3 for ellipsoidal uncertainty and the linear program for gamma-uncertainty. For solving the knapsack problem in Step 4 we used the classical dynamic programming algorithm.

Table 1 Results for the knapsack problem with ellipsoidal uncertainty

n	Ω	diff	$\|X^*\|$	iter	t_{dual}	t_{comb}	t_{tot}
250	1	6.3	7.2	9.8	1.1	0.7	2.5
	2	12.4	19.7	27.2	3.7	1.9	6.3
	3	18.3	44.4	54.8	8.3	3.9	13.0
	4	23.9	77.9	89.4	14.8	6.3	22.0
	5	29.2	135.5	154.5	29.8	11.0	41.7
500	1	4.5	10.5	13.9	9.5	3.9	18.5
	2	8.9	26.5	33.8	25.9	9.6	41.2
	3	13.2	72.4	79.1	67.1	22.4	94.5
	4	17.4	123.4	134.7	120.8	38.2	165.1
	5	21.5	147.3	194.9	182.5	55.3	243.1
750	1	3.6	14.9	19.1	42.6	12.4	69.8
	2	7.2	48.7	54.8	139.4	35.3	188.6
	3	10.7	142.1	146.3	383.6	93.1	493.2
	4	14.2	163.2	168.8	457.3	107.5	581.4
	5	17.5	243.0	252.0	808.8	160.7	986.8

Table 2 Results for the knapsack problem with gamma-uncertainty

n	Γ	diff	$\|X^*\|$	iter	t_{dual}	t_{comb}	t_{tot}
250	12	1.8	1.8	3.0	0.00	0.2	0.3
	25	3.6	3.0	4.6	0.00	0.3	0.4
	37	5.2	4.0	5.9	0.00	0.4	0.5
	62	8.2	8.9	14.7	0.00	1.0	1.1
	125	10.0	1.0	8.2	0.00	0.6	0.7
500	25	1.8	5.5	10.6	0.00	3.0	3.3
	50	3.6	9.3	21.1	0.00	5.9	6.3
	75	5.2	12.6	30.0	0.01	8.5	8.9
	125	8.2	18.5	50.7	0.02	14.3	14.7
	250	10.0	1.0	9.8	0.00	2.7	3.1
750	37	1.8	5.4	9.9	0.00	6.3	7.0
	75	3.6	10.1	28.5	0.01	18.0	18.9
	112	5.3	14.4	40.8	0.02	25.9	26.7
	187	8.3	25.2	82.3	0.07	52.5	53.5
	375	10.0	1.0	7.2	0.00	4.6	5.3

Results are listed in Tables 1 and 2. For each combination of n and Ω or n and Γ, respectively, we show the average over all 10 instances of the following numbers (from left to right): the difference (in percent) of the objective value of Problem (M^3) to the value of the certain problem with the ellipsoid center $\bar{c}$ or the mean vector $\bar{c}$, respectively, as cost function; the number of solutions in the computed set X^*; the number of major iterations; the running times used by the two oracles (t_{dual} for linear

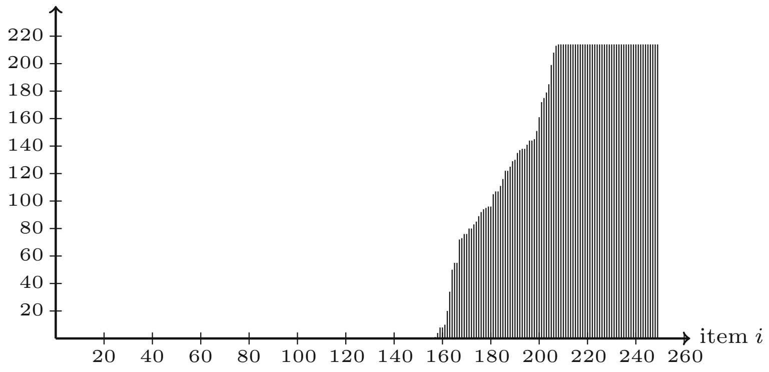


Fig. 1 One instance for the knapsack problem in dimension 250. The number of calculated solutions is 214; the y-axis shows the number of solutions in which an item i is selected

optimization over U and t_{comb} for solving the certain combinatorial problem (M)); and the total running time. All times are given in CPU seconds, on an Intel Xeon processor running at 2.5 GHz.

For ellipsoidal uncertainty the results show that running times increase with Ω and (of course) n. However, even the hardest instances with $n = 750$ and $\Omega = 5$ could be solved within 16.5 min on average. Interestingly, the number of solutions needed in order to solve Problem (M^3) to optimality usually remains well below n, in particular for small uncertainty sets. The oracle for linear optimization over U takes most of the running time.

For gamma-uncertainty, the results are even more positive, with much shorter running times, less iterations, and significantly smaller solution sets X^*. Contrarily to the computations with ellipsoidal uncertainty, here the combinatorial oracle takes most of the total running time, while the dual oracle runs less than a tenth of a second for all instance sizes. For comparison, we also performed experiments using the approach of [14]. It turned out however that CPLEX was not able to solve the corresponding MILP formulations within hours even for $n = 20$.

To obtain some insight into the typical structure of an optimal solution computed by Algorithm 1, we picked one instance with ellipsoidal uncertainty and counted in how many of the computed solutions a given object is used. The result is shown in Fig. 1, where objects are sorted (from left to right) by the number of appearances. It turns out that more than half of the objects are never used, about one fifth is used in every computed solution, while only the remaining objects are used in a non-empty proper subset of the solutions. Similar pictures are obtained when considering other instances.

5 NP-hardness for fixed *k*

For $k = 1$, Problem (M^3) agrees with Problem (M^2) and is thus NP-hard for many classes of uncertainty sets, even if the underlying certain problem is tractable. As a consequence, Problem (M^3) is NP-hard in the same cases as soon as k is considered part of the input. In the following, we show that Problem (M^3) remains NP-hard for any *fixed* number of solutions $k \in \mathbb{N}$, even in the special case that U is a polyhedron (given by an inner description) and $X = \{0,1\}^n$. To this end, for fixed $k \in \mathbb{N}$, we consider the problem

$$\min_{x \in \{0,1\}^n(k)} \max_{(c,c_0)\in U} c^\top x + c_0, \tag{M$^3[k]$}$$

where the input consists of vectors $v_1, \dots, v_r, w_1, \dots, w_s \in \mathbb{R}^{n+1}$ such that

$$U = \text{conv}\,(v_1, \dots, v_r) + \text{cone}\,(w_1, \dots, w_s)\,.$$

We use the following technical result.

Lemma 6 *Let $Y \subseteq \mathbb{R}^n$. Then the problem*

$$\min_{x\in Y,\, Ax\le b} \max_{(c,c_0)\in U} c^\top x + c_0 \tag{5}$$

with $U = conv\,(v_1, \dots, v_r) + cone\,(w_1, \dots, w_s)$ is equivalent to the problem

$$\min_{x\in Y} \max_{(c,c_0)\in V} c^\top x + c_0 \tag{6}$$

with $V = conv\,(v_1, \dots, v_r) + cone\left(w_1, \dots, w_s, (a_1, -b_1)^\top, \dots, (a_m, -b_m)^\top\right)$.

Proof Given an instance of Problem (5), let x be any solution of Problem (6). If $x \notin P := \{x \in \mathbb{R}^n \mid Ax \le b\}$, we derive $\max_{(c,c_0)\in V}\ c^\top x + c_0 = \infty$ by construction of V. On the other hand, for any $x \in P$ and for any scenario

$$(c, c_0) = v + \sum_{i=1}^{s} \mu_i w_i + \sum_{i=1}^{m} \lambda_i\,(a_i, -b_i) \in V$$

where $\lambda_i, \mu_i \ge 0$ and $v \in \text{conv}\,(v_1, \dots, v_r)$, the objective value is

$$c^\top x + c_0 = v^\top x + \sum_{i=1}^{s} \mu_i w_i^\top x + \sum_{i=1}^{m} \lambda_i \left(a_i^\top x - b_i\right) \le v^\top x + \sum_{i=1}^{s} \mu_i w_i^\top x.$$

Therefore the worst case scenario is obtained in U, which proves the result. □

In other words, we can add linear constraints to the feasible set Y without increasing the complexity of the problem, as the latter can be embedded into the uncertainty set U.

Corollary 2 *Problem ($\overline{M}^3[1]$) is NP-hard.*

Proof Using Lemma 6 applied to $Y = \{0,1\}^n$, we can easily model, e.g., the deterministic knapsack problem in the form of Problem ($\overline{\text{M}}^3[1]$) with a polyhedral uncertainty set. □

Theorem 5 *For any $k \in \mathbb{N}$, Problem ($M^3[k]$) can be polynomially reduced to Problem ($\overline{M}^3[k+1]$).*

Proof The idea of the proof is to reduce minimization over $\{0,1\}^n(k)$ to minimization over $\{0,1\}^{n+1}(k+1)$ using Lemma 6 again. To this end, first note that the encoding length of any convex hull of up to $k+1$ vectors in $\{0,1\}^{n+1}$ is at most $(k+1)(n+1)$. Using Lemma 6.2.4 and Lemma 6.2.7 in [13], it follows that every such convex hull either contains the center point $\bar{c} := \frac{1}{2}\mathbf{1} \in \mathbb{R}^{n+1}$, or the Euclidean distance between $\bar{c}$ and this convex hull is at least $2^{-6(k+1)(n+1)^3-1}$. In particular, switching to the max-norm and setting

$$d := \tfrac{1}{n+1} 2^{-6(k+1)(n+1)^3-1},$$

we derive that any convex combination of $k+1$ points in $\{0,1\}^{n+1}$ either contains $\bar{c}$ or does not intersect the box

$$B := \left[\bar{c} - \tfrac{1}{2}d\mathbf{1}, \bar{c} + \tfrac{1}{2}d\mathbf{1}\right] \cap \left\{x \in \mathbb{R}^{n+1} \mid x_{n+1} = \tfrac{1}{2} - \tfrac{1}{2}d\right\}.$$

Observe that the encoding length of d is polynomial in n. We now claim that the affine function $f(x) = (1-\frac{1}{d})\bar{c} + \frac{1}{d}x$ induces a bijection between the two sets $\{0,1\}^{n+1}(k+1) \cap B$ and $\{0,1\}^n(k) \times \{0\}$.

To prove our claim, let $x^* \in \{0,1\}^{n+1}(k+1) \cap B$ with $x^* = \sum_{i=1}^{k+1} \lambda_i x^{(i)}$. As $x^* \in B$, we obtain $(f(x^*))_{n+1} = 0$. Moreover, by the observations above, the center point $\bar{c}$ is contained in $\operatorname{conv}\left(x^{(1)}, \ldots, x^{(k+1)}\right)$, let $\bar{c} = \sum_{i=1}^{k+1} \mu_i x^{(i)}$. Then

$$f(x^*) = (1-\tfrac{1}{d})\bar{c} + \tfrac{1}{d}x^* = \sum_{i=1}^{k+1} \left((1-\tfrac{1}{d})\mu_i + \tfrac{1}{d}\lambda_i\right) x^{(i)}. \tag{7}$$

Since $x^*_{n+1} > 0$, there must exist some i with $(x^{(i)})_{n+1} = 1$. As $(f(x^*))_{n+1} = 0$, we derive $(1-\frac{1}{d})\mu_i + \frac{1}{d}\lambda_i = 0$ from (7). Now (7) implies $f(x^*) \in \{0,1\}^{n+1}(k)$ and therefore $f(x^*) \in \{0,1\}^n(k) \times \{0\}$.

To show the other direction, consider any $y^* \in \{0,1\}^{n+1}(k)$ with $y^*_{n+1} = 0$. Writing $y^* = \sum_{i=1}^k \nu_i x^{(i)}$, we have

$$\begin{aligned}
f^{-1}(y^*) &= -(d-1)\bar{c} + dy^* \\
&= \left(\tfrac{1}{2} - \tfrac{1}{2}d\right)\mathbf{1} + d\sum_{i=1}^{k} \nu_i x^{(i)} \\
&= \left(\tfrac{1}{2} - \tfrac{1}{2}d\right)\left(x^{(1)} + \bar{x}^{(1)}\right) + d\sum_{i=1}^{k} \nu_i x^{(i)}
\end{aligned}$$

where $\bar{x}^{(1)}$ denotes the complement vector of $x^{(1)}$, defined by $\bar{x}_i^{(1)} = 1 - x_i^{(1)}$ for all i. This is a convex combination of the binary vectors $x^{(1)}, \dots, x^{(k)}, \bar{x}^{(1)}$, hence $f^{-1}(y^*) \in \{0,1\}^{n+1}(k+1)$. It is easy to verify that $f^{-1}(y^*) \in B$.

To conclude the proof, we have

$$\begin{aligned}\min_{x \in \{0,1\}^n(k)} \max_{(c,c_0)\in U} c_0 + c^\top x &= \min_{x \in B \cap \{0,1\}^{n+1}(k+1)} \max_{(c,c_0)\in U} c_0 + c^\top f(x) \\ &= \min_{x \in B \cap \{0,1\}^{n+1}(k+1)} \max_{(c',c_0')\in U'} c_0' + (c')^\top x\end{aligned}$$

where U' is the image of U under the linear map

$$(c, c_0) \mapsto \left(\tfrac{1}{d}c, c_0 + \left(1 - \tfrac{1}{d}\right) c^\top \bar{c}\right).$$

Together with Lemma 6, modeling the box B by linear inequalities, this proves the result. □

By induction, the preceding two results imply

Corollary 3 *Problem (M^3) is NP-hard for any fixed $k \in \mathbb{N}$, even if U is a polyhedron given by an inner description and $X = \{0,1\}^n$.*

The special case of Problem (M^3) considered in Corollary 3 is probably the most elementary one possible: both the certain combinatorial optimization problem over $X = \{0,1\}^n$ and the linear optimization problem over any uncertainty set given as conv $(v_1, \dots, v_r)$ + cone $(w_1, \dots, w_s)$ can be trivially solved in polynomial time. Yet the min–max–min problem turns out to be NP-hard for any fixed k even in this case.

6 Heuristic algorithm for $k < n + 1$

As shown in the previous section, Problem (M^3) is NP-hard for fixed k even if the underlying certain problem can be solved in polynomial time. Nevertheless, the case of fixed (or small) k can be important for practical applications, since a set of $n+1$ solutions may be too large to be provided to—or accepted by—a human user. The idea of Theorem 3 can be used to design an efficient heuristic algorithm for the case of general k. The idea is to calculate an optimal solution $\{x^{(1)}, \dots, x^{(n+1)}\}$ for $k = n+1$ and then keep only those k solutions with the largest coefficients λ_i; see Algorithm 2. This heuristic is motivated by the observation that in our experiments we often find optimal solutions affinely spanned by significantly less than $n+1$ solutions, as was shown also in Tables 1 and 2.

Theorem 6 *Let $k \in \{1, \dots, n\}$ and $\varepsilon \in (0,1)$. Let conv (X) be full-dimensional and U as in Sect. 3.2. Given an optimization oracle for the certain problem*

$$c \mapsto \min_{x \in X} c^\top x,$$

Algorithm 2 Heuristic algorithm for Problem (M^3) for $k < n + 1$

```
Input: $U$, $X$, integer $k$ with $1 \leq k < n + 1$, $\varepsilon \in (0, 1)$
Output: feasible solution of Problem (M$^3$)
1: calculate an optimal solution $x^*$ of (1) for $k = n + 1$ up to accuracy $\varepsilon$, using Theorem 3
2: calculate $x^{(1)}, \dots, x^{(n+1)} \in X$, $\lambda_1, \dots, \lambda_{n+1} \geq 0$ with $\sum_{i=1}^{n+1} \lambda_i = 1$, $x^* = \sum_{i=1}^{n+1} \lambda_i x^{(i)}$, using Lemma 2
3: sort the $\lambda_i$ in decreasing order $\lambda_{i_1} \geq \dots \geq \lambda_{i_{n+1}}$
4: return $\{x^{(i_1)}, \dots, x^{(i_k)}\}$
```

Algorithm 2 calculates in polynomial time a solution of Problem (M^3) with an additive error of at most $M \frac{\sqrt{n}(n+1-k)}{k+1} + \varepsilon$.

Proof By Theorem 3, Algorithm 2 has polynomial running time under the given assumptions. Let $x_k \in X(k)$ be an optimal solution for Problem (1) and define

$$x_a := \sum_{j=1}^{k-1} \lambda_{i_j} x^{(i_j)} + \left(\sum_{j=k}^{n+1} \lambda_{i_j} \right) x^{(i_k)} \in X(k).$$

Since $X(k) \subseteq X(n + 1)$ and by the optimality of x^*, we have

$$\begin{aligned} \max_{(c,c_0)\in U} c^\top x_a + c_0 - \max_{c\in U} c^\top x_k + c_0 &\leq \max_{(c,c_0)\in U} c^\top x_a + c_0 - \max_{(c,c_0)\in U} c^\top x^* + c_0 \\ &\leq \max_{c\in U} \|c\| \|x_a - x^*\| \\ &\leq M \left\| \sum_{j=k+1}^{n+1} \lambda_{i_j} (x^{(i_k)} - x^{(i_j)}) \right\|. \end{aligned}$$

By the decreasing order of λ_{i_j}, we have $\lambda_{i_j} \leq \frac{1}{j}$. Additionally, as $X \subseteq \{0, 1\}^n$, we know $\|x^{(i_k)} - x^{(i_j)}\| \leq \sqrt{n}$. Therefore

$$\left\| \sum_{j=k+1}^{n+1} \lambda_{i_j} (x^{(i_k)} - x^{(i_j)}) \right\| \leq \sqrt{n} \sum_{j=k+1}^{n+1} \frac{1}{j} \leq \sqrt{n} \frac{n + 1 - k}{k + 1}.$$

This implies the result. □

For fixed U and n, the error bound given in Theorem 6 is strictly monotonously decreasing with growing $k \leq n + 1$. For $k = n + 1$, it coincides with ε.

To conclude this section, we provide some computational results showing that the above heuristic often calculates solutions that are close to optimal even for small k. To this end, we replace the theoretically fast algorithm underlying Theorem 3 by the practically fast Algorithm 1 of Sect. 4. We applied our heuristic to the instances of the shortest path problem used in [14]. The authors create graphs on 20, 25, ..., 50 nodes, corresponding to points in the Euclidean plane with random coordinates in [0, 10], and choose a budget uncertainty set of the form

$$\Xi = \left\{ \xi_{ij} \in [0,1] \mid \sum_{ij} \xi_{ij} \le \Gamma \right\}$$

where the costs on the edges are set to $c_{ij}(\xi) = (1 + \frac{\xi_{ij}}{2})d_{ij}$ and d_{ij} is the Euclidean distance of node i to node j. No uncertain constants are considered in the objective function. The parameter Γ is chosen from $\{3, 6\}$. For more details see Section 4.2 in [14].

In Table 3, the computational results for Algorithm 2 are shown. In columns indicated by #, we find the number of instances (out of 100) for which the problem was solved to optimality by the authors of [14] within a time limit of 2 h. For the latter instances, in columns marked Δ_{sol}, we state how far our heuristic solution value is above the exact minimum, on average (in percent). Similarly, for all 100 instances, we denote by Δ_{all} how far our heuristic solution value is above the best solution value found by the authors of [14] within the time limit, i.e., this number includes the instances which could not be solved to proven optimality in [14]. In both cases, we first state the mean and then the median value. For every type of instance, the running time for our heuristic was at most one CPU second on average, and is thus not reported in the table. Note that we are able to calculate heuristic solutions for all k up to $n + 1$ at one stroke.

Table 3 shows that, for every instance size considered, both the mean and the median of the differences Δ_{all} with respect to the best known solutions are within 8 %. In Fig. 2, we illustrate the mean differences in dependence of the problem size and k. Not surprisingly, the gap grows with the number of nodes, whereas a larger number k leads to better solutions.

Example 2 While Algorithm 2 performs very well on the random instances considered above, it is possible to construct instances where its solution value is arbitrarily far away from the optimum even if n and k are fixed. By Theorem 6, then M has to be unbounded. Consider the set $X := \{x \in \{0,1\}^n \mid \mathbf{1}^\top x \le 1\}$, so that conv (X) is a simplex. For $\alpha > 0$, we define an ellipsoid

$$U_\alpha = \{c \in \mathbb{R}^n \mid (c + \mathbf{1})^\top \Sigma_\alpha (c + \mathbf{1}) \le 1\},$$

where Σ_α is the inverse of the positive definite matrix $\alpha^2(I - (\frac{1}{n} - \frac{1}{2\alpha^2})\mathbf{1}\mathbf{1}^\top)$. Then the unique minimizer of

$$\max_{c \in U_\alpha} c^\top x = -\mathbf{1}^\top x + \sqrt{x^\top \Sigma_\alpha^{-1} x} = -\mathbf{1}^\top x + \alpha \sqrt{x^\top \left(I - \left(\frac{1}{n} - \frac{1}{2\alpha^2}\right)\mathbf{1}\mathbf{1}^\top\right) x}$$

over conv (X) is $x^* = \frac{1}{n}\mathbf{1}$. Its unique representation as convex combination of elements of X is $x^* = \sum_{i=1}^n \frac{1}{n} e_i$, where e_i denotes the ith unit vector. Hence, whenever $k \le n$, the heuristic will choose a set $X^* \subseteq \{e_1, \ldots, e_n\}$. Assuming without loss of generality that $X^* = \{e_1, \ldots, e_k\}$, the optimum of

$$\min_{x \in X^*(k)} \max_{c \in U_\alpha} c^\top x$$

Table 3 Computational results for Algorithm 2

Γ	k	20 nodes			25 nodes			30 nodes			35 nodes		
		#	Δ_{sol}	Δ_{all}	#	Δ_{sol}	Δ_{all}	#	Δ_{sol}	Δ_{all}	#	Δ_{sol}	Δ_{all}
3	1	100	4.1/3.3	4.1/3.3	100	4.6/3.7	4.6/3.7	100	5.4/5.2	5.4/5.2	100	6.5/6.1	6.5/6.1
	2	100	1.9/1.2	1.9/1.2	99	2.4/2.0	2.5/2.1	69	2.8/2.2	3.0/2.8	17	2.6/2.0	3.4/3.5
	3	97	0.8/0.2	0.8/0.2	31	0.5/0.3	1.2/0.6	6	0.1/0.0	1.8/1.2	0	−/−	2.2/2.1
	4	51	0.2/0.0	0.5/0.1	6	0.1/0.0	0.6/0.4	0	−/−	0.9/0.5	0	−/−	1.3/1.0
6	1	100	5.3/4.0	5.3/4.0	100	4.8/3.8	4.8/3.8	100	5.2/4.5	5.2/4.5	100	6.7/6.6	6.7/6.6
	2	100	3.7/3.3	3.7/3.3	99	4.6/4.3	4.7/4.5	67	4.8/4.7	5.2/5.4	16	7.1/6.9	5.8/5.3
	3	97	2.3/2.1	2.3/2.1	38	2.3/2.0	3.1/3.1	6	0.7/0.1	3.5/3.6	0	−/−	4.1/4.0
	4	55	1.0/0.5	1.4/0.9	7	0.5/0.1	2.0/1.8	0	−/−	2.3/2.2	0	−/−	3.2/3.1

Γ	k	40 nodes			45 nodes			50 nodes		
		#	Δ_{sol}	Δ_{all}	#	Δ_{sol}	Δ_{all}	#	Δ_{sol}	Δ_{all}
3	1	100	7.0/7.1	7.0/7.1	100	7.3/7.4	7.3/7.4	100	8.0/7.8	8.0/7.8
	2	6	3.1/3.0	3.9/3.5	0	−/−	3.9/3.9	0	−/−	4.3/4.1
	3	0	−/−	2.5/2.6	0	−/−	2.4/2.4	0	−/−	2.8/2.6
	4	0	−/−	1.6/1.5	0	−/−	1.6/1.4	0	−/−	1.9/1.8
6	1	100	7.2/6.0	7.2/6.0	100	7.8/7.2	7.8/7.2	100	7.8/7.8	7.8/7.8
	2	5	5.0/5.5	6.6/6.3	0	−/−	7.1/6.7	0	−/−	7.0/6.5
	3	0	−/−	4.5/4.6	0	−/−	4.9/4.6	0	−/−	4.9/4.6
	4	0	−/−	3.3/3.2	0	−/−	3.7/3.4	0	−/−	3.7/3.7

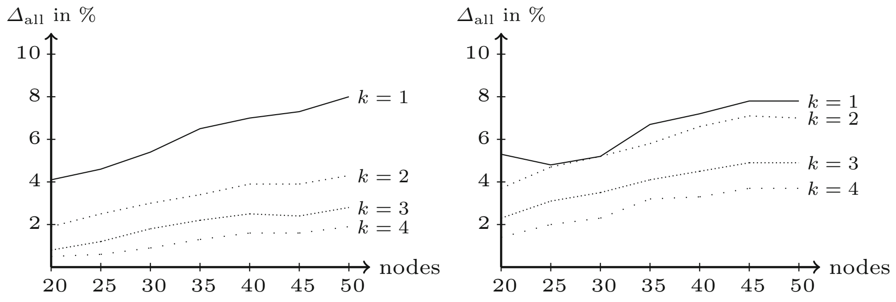


Fig. 2 Difference to the best known solution in % for $\Gamma = 3$ (*left*) and $\Gamma = 6$ (*right*)

is attained in $\frac{1}{k}\sum_{i=1}^{k} e_i$, the heuristic thus achieves the objective value

$$-1 + \alpha\sqrt{\frac{1}{k} - \frac{1}{n} + \frac{k^2}{2\alpha^2}}\,,$$

which becomes arbitrarily large with growing α if $k < n$. On contrary, for large enough α, every optimal solution contains the zero vector and has objective value zero. $\square$

Acknowledgements We would like to thank the authors of [14] for providing us their instances and computational results for the uncertain shortest path problem.

References

1. Baumann, F., Buchheim, C., Ilyina, A.: Lagrangean decomposition for mean-variance combinatorial optimization. In: Combinatorial Optimization—Third International Symposium, ISCO 2014, Lecture Notes in Computer Science, vol. 8596, pp. 62–74. Springer, Berlin (2014)
2. Ben-Tal, A., Goryashko, A., Guslitzer, E., Nemirovski, A.: Adjustable robust solutions of uncertain linear programs. Math. Program. **99**(2), 351–376 (2004)
3. Ben-Tal, A., Nemirovski, A.: Robust convex optimization. Math. Oper. Res. **23**(4), 769–805 (1998)
4. Bertsimas, D., Caramanis, C.: Finite adaptability in multistage linear optimization. IEEE Trans. Autom. Control **55**(12), 2751–2766 (2010)
5. Bertsimas, D., Sim, M.: Robust discrete optimization and network flows. Math. Program. **98**(1–3), 49–71 (2003)
6. Bertsimas, D., Sim, M.: The price of robustness. Oper. Res. **52**(1), 35–53 (2004)
7. Bertsimas, D., Sim, M.: Robust discrete optimization under ellipsoidal uncertainty sets (2004)
8. Blankenship, J.W., Falk, J.E.: Infinitely constrained optimization problems. J. Optim. Theory Appl. **19**(2), 261–281 (1976)
9. Buchheim, C., Kurtz, J.: Min-max-min robustness: a new approach to combinatorial optimization under uncertainty based on multiple solutions. In: International Network Optimization Conference—INOC 2015 (to appear)
10. Büsing, C.: Recoverable robust shortest path problems. Networks **59**(1), 181–189 (2012)
11. El Ghaoui, L., Lebret, H.: Robust solutions to least-squares problems with uncertain data. SIAM J. Matrix Anal. Appl. **18**(4), 1035–1064 (1997)
12. Grötschel, M., Lovász, L., Schrijver, A.: Geometric methods in combinatorial optimization. In: Proceedings of Silver Jubilee Conference on Combinatorics, pp. 167–183 (1984)
13. Grötschel, M., Lovász, L., Schrijver, A.: Geometric Algorithms and Combinatorial Optimization. Springer, Berlin (1993)

14. Hanasusanto, G.A., Kuhn, D., Wiesemann, W.: K-adaptability in two-stage robust binary programming. Optim. Online (2015)
15. Kouvelis, P., Yu, G.: Robust Discrete Optimization and Its Applications. Springer, Berlin (1996)
16. Liebchen, C., Lübbecke, M., Möhring, R., Stiller, S.: The concept of recoverable robustness, linear programming recovery, and railway applications. In: Robust and Online Large-scale Optimization, pp. 1–27. Springer, Berlin (2009)
17. Nikolova, E.: Approximation algorithms for reliable stochastic combinatorial optimization. In: Proceedings of APPROX '10, Barcelona, Spain (2010)
18. Sim, M.: Robust optimization. Ph.D. thesis, Massachusetts Institute of Technology (2004)
19. Soyster, A.L.: Convex programming with set-inclusive constraints and applications to inexact linear programming. Oper. Res. **21**(5), 1154–1157 (1973)