GRÖBNER AND GRAVER BASES FOR CALCULATING OPPORTUNITY COST MATRICES

YUCHEN GE, JANOSCH ORTMANN, AND WALTER REI

Abstract. Opportunity cost matrices, first introduced in [15], are interesting in the context of scenario reduction. We provide new algorithms, based on ideas from algebraic geometry, to efficiently compute the opportunity cost matrix using computational algebraic geometry. We demonstrate the efficacy of our algorithms by computing opportunity cost matrices for two stochastic integer programs.

1. Introduction

In many real-world applications, decision-making under uncertainty is a challenging problem that requires the consideration of multiple possible scenarios. However, the number of scenarios can grow exponentially with the number of uncertain parameters, making the decision problem intractable. Scenario reduction methods aim to address this issue by reducing the number of scenarios while preserving the key characteristics of the original problem.

Recently [1, 15, 16] there has been progress in developing problem-driven scenario reduction methods. All of these methods require the computation of a variant of the *opportunity cost matrix* introduced in [15]: assign a decision x_j^* to each scenario s_j. In [15, 16] this is done by choosing an optimal solution to the single-scenario problem. The opportunity cost matrix is computed by evaluating the solution associated to scenario j against scenario i. Intuitively, suppose that we hae an oracle that predicts that scenario j will occur. The (i, j) entry of the opportunity cost matrix gives the cost incurred by trusting the oracle and taking a decision under the viewpoint of scenario j if it then turns out that in fact scenario i occurs.

The opportunity cost matrix naturally leads to a decision-based distance on the scenario set: two scenarios i and j can be considered to be close if the (i, j) entry of the opportunity cost matrix is small (and hence the cost of predicting scenario j when actually scenario i occurs leads to a small cost) and far away if the (i, j) entry is large. [15, 16] propose clustering algorithms based on this distance, which has shown to lead to good approximative solutions and tight bounds on the true objective value.

In this paper, we present a novel approach to efficiently compute opportunity cost matrices using Gröbner and Graver bases. Gröbner [5] and Graver [11] bases are algebraic objects

2000 *Mathematics Subject Classification.* 90C15 (primary) 90C10, 13P25 (secondary).

This work was conceived while YG was a research intern in Montreal with financial support from the MITACS Globalink programme. The authors are grateful to MITACS for this support.

JO would like to thank Rüdiger Schultz for valuable discussions on the subject of the article. He gratefully acknowledges the support provided by an NSERC discovery grant.

While working on this paper, the third author was the Canada Research Chair (CRC) in Stochastic Optimization of Transport and Logistics Systems. In addition, he was supported by the National Sciences and Engineering Research Council of Canada (NSERC) through the discovery grants program. He gratefully acknowledges all the support that is provided by the CRC and NSERC programs.

that can be used to solve systems of polynomial equations and inequalities. These have been successfully applied to optimisation [19, 20] and have seen a wide range of applications

Our approach leverages the structure of opportunity cost matrices to formulate them as polynomial systems, which can then be solved using Gröbner and Graver bases. The resulting method is fast, accurate, and scales with the number of scenarios and variables.

We demonstrate the effectiveness of our approach through numerical experiments on a well-known integer program from [13]. Our results show that the Gröbner basis approach is particularly efficient for a large number of scenarios, whereas the Graver basis approach stays stable with a large number of decision variables. Our paper contributes to the literature by providing two new approaches to computing opportunity cost matrices. We also prove theoretical results about Graver bases for deterministic and stochastic integer programs.

The rest of the paper is organized as follows. In Section 2, we provide background on opportunity cost matrices and their usefulness. In Section 3, we introduce the Gröbner and Graver bases and present our mathematical results. Section 4 describes the algorithms that we then apply to generate our numerical results in Section 5. Finally, Section 6 concludes.

Notation. Throughout this paper, $\mathbb{Z}$ denotes the set of integers and $\mathbb{Z}_{\geqslant 0}$ the subset of non-negative integers.

2. Opportunity cost matrices

Consider the following two-stage stochastic optimisation problem

$$\min_{x\in\mathcal{X}} \mathbb{E}F(x,\xi) = \min_{x\in\mathcal{X}} \gamma^\top x + \mathbb{E}Q(x,\xi) \tag{2.1}$$

where $\gamma \in \mathbb{R}^d$ and $\mathcal{X}$ is the intersection of the integer lattice with a convex polyhedron. The random part Q of the objective function is given by

$$Q(x,\xi) = \min\left\{c_\xi^\top y \colon Tx + Ay = h_\xi \colon y \in \mathbb{Z}^d\right\} \tag{2.2}$$

where $c_\xi \in \mathbb{Z}^d$ and $b_\xi \in \mathbb{Z}^k$ for all ξ and T and A are integer matrices of the required dimension. Observe that we assume T and A to be deterministic matrices, i.e. they are not random.

In general, it is not obvious how to obtain an explicit solution to (2.1). The scenario approach in stochastic programming [4] consists of approximating the probability measure $\mathbb{P}$ on ξ that gives rise to the expectation operator $\mathbb{E}$ by a discrete measure assigning probabilities[1] $p_1, ..., p_N$ on a finite set of outcomes $\xi_1, ..., \xi_N$(often called *scenarios*) for ξ. The set $\mathcal{S} = \{\xi_1, \dots, \xi_N\}$ is then referred to as the *scenario set.*

For example, this can be achieved by the *sample average approximation* [17], where the ξ_i are independent samples from $\mathbb{P}$ and are each assigned the same probability $1/N$ of occurring. In any case, the scenario approach yields optimisation problems of the form

$$\min\left\{\gamma^\top x + \sum_{i=1}^{N} p_i c_i^\top y_i : x \in \mathcal{X}, Ay_i = h_i - Tx, y_i \in \mathbb{Z}_+^n \ \forall\, i\right\}. \tag{2.3}$$

In order to properly model the incertainty represented by $\mathbb{P}$, often a large number of scenarios must be generated. On the other hand solving problems of the form 2.3 when N is large can be very computationally challenging. This motivates the idea of *scenario reduction*: can one

[1]i.e. $p_n \geqslant 0$ for all n and $\sum_{n=1}^{N} p_n = 1$

find a small subset $\widehat{\mathcal{S}}$ of the scenario set $\mathcal{S} = \{\xi_1, \dots, \xi_N\}$ such that replacing $\mathcal{S}$ by $\widehat{\mathcal{S}}$ in (2.3) only incurs a small approximation error.

Generally speaking, scenario reduction methods can be separated into *distribution-based* and *problem-based* methods. The former seek to minimise the difference (defined in some suitable sense) between the empirical measure of the subset and the original scenario set. See for example [8, 14, 18] for examples of this approach. On the other hand, problem-based methods [1, 15, 16] look to minimise the difference between the solutions associated to the larger and smaller problems, in other words scenarios are considered to be similar if they lead to similar decisions according to (2.3). In [15], the opportunity cost matrix was introduced.

Definition 2.1. Consider a set of feasible decisions $x_1, ..., x_N \in \mathcal{X}$ such that decision x_j is associated to scenario s_j. The *opportunity cost matrix* is the $N \times N$ matrix $\mathcal{A}$ whose (i, j) entry is $\mathcal{A}_{ij} = F(x_i, \xi_j)$.

Clearly there are many interesting choices for the solutions x_j.

Example 2.2. In [15] and [16], the decisions x_n were chosen to be the single-scenario solutions of (2.1), that is

$$x_j \in \operatorname{argmin} F(x, \xi_j), \tag{2.4}$$

In this paper, we are looking for efficient methods of computing the matrix $\mathcal{A}$ in the context of the stochastic linear two-stage problem defined in (2.1). Since the functions Q and F only differ by the linear term $\gamma \cdot x$, we turn our attention to efficiently computing

$$Q\left(x_i, \xi_j\right) = \min\left\{c_j^\top y \colon Ay = b_{ij}\right\}, \tag{2.5}$$

where $c_j = c(\xi_j)$, $b_{ij} = b(x_i, \xi_j)$. Here, we write $b(x, \xi) = h_\xi - T_\xi x$. Since we had assumed A to be deterministic deterministic, the feasible region on the right-hand side of (2.5) changes only through b_{ij} as i and j vary.

3. Gröbner and Graver bases

In this section we develop the theory of Gröbner and Graver bases that we require in order to justify and motivate the algorithms that will be described in Section 4.

In the previous section we saw that in order to compute the opportunity cost matrix, we are looking to solve integer optimisation problems of the form

$$(IP)_{c,b}: \quad \min\left\{c^\top z : Az = b, z \in \mathbb{N}^d\right\} \tag{3.1}$$

for a given integer matrix A but with varying cost vectors $c \in \mathcal{C}$ and constraint vectors $b \in \mathcal{B}$. Here, $\mathcal{C}$ and $\mathcal{B}$ are finite sets corresponding to the values that the $c(\xi_m)$ and $b(x, \xi_m)$ of (2.5) can take. Since we consider the matrix A to be fixed throughout, we do not include it in the notation.

Note that the optimal solution to (3.1) may not be unique, which is sometimes inconvenient. Whenever required, we can therefore replace (3.1) by its refinement $(IP)_{>_c,b}$ that we will define now. We begin my defining a total order[2] on $\mathbb{Z}^d_{\geqslant 0}$.

Definition 3.1. Let $>$ be any total order on $\mathbb{Z}^n_{\geqslant 0}$ and fix $c \in \mathbb{Z}^d$. We define another order relation on $\mathbb{N}^n$ by saying that $x >_c$ y and only if

(1) $c \cdot x > c \cdot y$ or

[2]A *total order* on a set S is a binary relation $\prec$ on S such that $a \prec b$ or $b \prec a$ for all $a, b \in S$ with $a \neq b$.

(2) $c \cdot x = c \cdot y$ and $x > y$.

Unless otherwise specified, the total order $\prec$ in Definition 3.1 will be the lexicographical order, under which $x > y$ if and only if the leftmost non-zero entry of $x - y$ is positive.

Definition 3.2. A set $\mathcal{T}_c \subseteq \mathbb{Z}^d$ is called a *test set* for the family of problems $\left\{(IP)_{c,b} \colon b \in \mathbb{R}^l\right\}$ if

(1) $c \cdot t > 0$ for all $t \in \mathcal{T}_c$
(2) for every $b \in \mathbb{R}^l$ and for every non-optimal feasible solution $z_0 \in \mathbb{Z}_+^d$ to $Az = b$, there exists a vector $t \in \mathcal{T}_c$ such that $z_0 - t$ is feasible. Such a vector is called an *augmentation vector*.

Given a test set $\mathcal{T}_c$, the *augmentation algorithm* (Algorithm 1) provides a way to compute the optimum of $(IP)_{>c,b}$ for any $b \in \mathcal{B}$. Intuitively, while there is $t \in \mathcal{T}_c$ with $c^\top t > 0$ such that $z_0 - t$ is feasible, we repetitively iterate z_0 by assigning $z_0 := z_0 - t$. Thus, we are looking to efficiently construct a test set $\mathcal{T}_c$ for each $c \in \mathcal{C}$. Observe that $\mathcal{T}_c \subseteq \ker(A)$ since both z_0 and $z_0 - t$ are feasible for any $t \in \mathcal{T}_c$ which implies that

$$At = A(z_0 - (z_0 - t)) = Az_0 - A(z_0 - t) = b - b = 0. \tag{3.2}$$

In the following we will exhibit two approaches to constructing such a test set. The first uses Gröbner bases and is described in Section 3.1. The second, based on Graver bases, goes one step further and constructs a *universal test set*, that is, a set $\mathcal{T}$ containing a test set $\mathcal{T}_c$ for each cost vector $c \in \mathcal{C}$.

3.1. **The Gröbner Basis Approach.**

It is now well known [2, 19, 20, 23] that Gröbner bases [5, 7] yield test sets. The *Buchberger algorithm* [5, 6] takes as input a set of polynomials $f_1, \ldots, f_M$ and returns the reduced Gröbner basis of the ideal generated by the f_j. A geometric version of the Buchberger algorithm was given in [24]. This algorithm is at its most efficient if the generating set is small. We achieve this by using the toric ideal and in particular the methodology of [3]. The advantage is that this can be done once for all cost vectors, which is efficient in the context of computing opportunity cost matrices.

We will require the following result, which is a direct consequence of Dickson's Lemma [7] and can be found as Lemma 2.1.4 of [24].

Lemma 3.3. *Let $A \in \mathbb{Z}^{l \times d}$ and let $\mathcal{N}_{c,b}$ denote the set of non-optimal feasible solutions of $(IP)_{c,b}$. Then there exist $\alpha(t), \ldots, \alpha(t) \in \mathbb{N}^d$ such that*

$$\bigcup_{b \in \mathbb{R}^l} \mathcal{N}_{c,b} = \bigcup_{i=1}^{t} (\alpha(i) + N^n) \tag{3.3}$$

The vectors $\alpha(1), \ldots, \alpha(t)$ allow us to compute a test set for the family of integer programs $\{(IP)_{c,b} \colon b \in \mathcal{B}\}$, via the the following result, which follows from the Gordon-Dickson Lemma and is stated as Corollary 2.1.10 of [24]. See also Lemma 2.1 in [21].

Theorem 3.4. *Fix $c \in \mathbb{Z}^d$, let $\alpha(1), \ldots, \alpha(t)$ as in Lemma 3.3 and let $\beta(i)$ be the (unique) optimum of the program $(IP)_{>c,A\alpha(i)}$. Then the set*

$$\mathcal{G}_{A,c} = \{(\alpha(i) - \beta(i)), i = 1, 2, ..., t\} \tag{3.4}$$

is a minimal test set for the family of problems $\left\{(IP)_{c,b} \colon b \in \mathbb{R}^l\right\}$.

Definition 3.5. The set $\mathcal{G}_{A,c}$ is called the *Gröbner basis* for the matrix A.

While this result gives a formula for the desired test set, it requires computation of the $\alpha(i)$ from Lemma 3.3 and, crucially, of the $\beta(i)$, which are given in terms of solutions of integer problems itself. A geometric construction of the test set was given in [24].

As explained above, we need to compute the solutions to $(IP)_{>_c,b}$ as c varies. If we would like to take the Gröbner basis approach, we would need to recompute the Gröbner basis for every cost vector $c \in \mathcal{C}$. Therefore, we proceed in a different manner, based on ideas from the theory of polynomial rings.

Let $R = \mathbb{R}[x_1, \ldots, x_n]$ denote the ring of polynomials in n variables. We will make frequent use of a close relation between vectors of integers and differences of monomials in R. Any n-vector of non-negative integers $\alpha = (\alpha_1, \ldots, \alpha_n) \in \mathbb{Z}^n_{\geqslant 0}$ can be identified with a monomial $x^\alpha \in R$ defined by

$$x^\alpha = x_1^{\alpha_1} \cdots x_n^{\alpha_n} = \prod_{j=1}^n x_j^{\alpha_j}, \tag{3.5}$$

and this mapping can be extended to integer-valued vectors as follows. For any $\alpha = (\alpha_1, \ldots, \alpha_n)\mathbb{Z}^n$, define $\alpha^+ = (\alpha_1^+, \ldots, \alpha_n^+)$ and $\alpha^- = (\alpha_1 -, \ldots, \alpha_n^-)$, recalling that the positive part of a real number x is defined to be $x^+ = \max(x, 0)$ and the negative part of x is $x^- = (-x)^+$. Then both $\alpha^+, \alpha^- \in \mathbb{Z}^n_{\geqslant 0}$ and $\alpha = \alpha^+ - \alpha^-$. We now define a mapping

$$\phi\colon \mathbb{Z}^n \longrightarrow R \quad \text{by} \quad \phi(\alpha) = x^{\alpha^+} - x^{\alpha^-}. \tag{3.6}$$

The notion of a Gröbner basis associated to a matrix from Definition 3.5 is actually a special case of a more general concept, namely that of a Gröbner basis of a polynomial ideal. In the following, we give a very brief account of the definition and basic properties of Gröbner bases. For more details see for example Chapter 1 of [22].

Definition 3.6. A subset $I \subseteq R$ is said to be an *ideal* of R if it satisfies the following two properties:

a) For any $x, y \in I$ we have $x - y \in I$ (that is, I is an additive subgroup of R),
b) For any $x \in I$ and $r \in R$ we have $rx \in I$.

Example 3.7. Given polynomials $f_1, \ldots, f_n$, the *ideal generated by* $f_1, \ldots, f_n$ is the set of all R-linear combinations of the f_j:

$$\langle f_1, \ldots, f_n \rangle = \left\{ \sum_{j=1}^n r_j f_j \colon r_1, \ldots, r_n \in R \right\}. \tag{3.7}$$

Example 3.8. The *toric ideal* I_A of a matrix $A \in \mathbb{Z}^{d \times n}$ is generated by differences of monomials $x^u - x^v$ such that $u - v$ lies in the kernel of A:

$$I_A = \langle x^u - x^v \colon Au = Av \rangle. \tag{3.8}$$

See Section 4 of [22] for the theory of toric ideals. A polynomial $f \in R$ consists of monomials x^α with *degree* $\alpha \in \mathbb{Z}^n_{\geqslant 0}$. Throughout, we fix a total order (recall Definition 3.1 and the discussion surrounding it) $\prec$ on $\mathbb{N}^n$. In principle, any total order will do, but for us, $\prec$ will always be the total order $<_c$ defined in Definition 3.1. Given $\prec$, we can compare the different monomials by comparing their degrees. This allows us to define the *initial monomial* of f to be that with the greatest degree (with respect to $\prec$). The initial monomial of f is denoted

$\mathrm{in}_{<}(f)$. Given an ideal I of R, the *initial ideal* of I is the ideal generated by the initial monomials of the elements of I:

$$\mathrm{in}_{<}(f) = \langle \mathrm{in}_{<}(f) : f \in I \rangle . \tag{3.9}$$

We are now in a position to define Gröbner bases for polynomial ideals.

Definition 3.9. Let I be an ideal of R. A subset $\mathcal{G}$ of I is said to be a *Gröbner basis* of I if the inital ideal of I is generated by the inital monomials of the elements of $\mathcal{G}$:

$$\mathrm{in}_{<}(I) = \langle \mathrm{in}_{<}(g) : g \in \mathcal{G} \rangle . \tag{3.10}$$

A Gröbner basis $\mathcal{G}$ of I is said to be *minimal* if (3.10) no longer holds whenever any one element is removed from $\mathcal{G}$. Also, $\mathcal{G}$ is said to be *reduced* if for any distinct $g_1, g_2 \in \mathcal{G}$ it holds that no term of g_1 is a multiple of $\mathrm{in}_{<}(g_2)$.

Given an ideal I of R and a total order on $\mathbb{N}^n$, there is only one reduced Gröbner basis $\mathcal{G}$ such that the coefficients of $\mathrm{in}_{<}(g)$ are equal to 1 for each $g \in \mathcal{G}$. In other words, reduced Gröbner bases are unique up to multiplying the elements by constants.

Proposition 3.10. *Recall the vectors $\alpha(i)$ and $\beta(i)$ from 3.4. The set of polynomials*

$$\mathcal{G}_{I_A} = \left\{ x^{\alpha(j)} - x^{\beta(j)} : j \in \{1, \ldots, t\} \right\} \tag{3.11}$$

is the reduced Gröbner basis of the toric ideal I_A with respect to the ordering $<_c$.

We conclude that the two notions of Gröbner basis (of a matrix on the one hand and of a polynomial ideal on the other) are actually the same: they are related via the map ϕ defined in (3.6). In fact, the original Buchberger algorithm [5] is formulated in this setting.

In [3], an efficient algorithm to compute a small generating set of the toric ideal I_A, see Algorithm 2. Observe that the toric ideal only depends on the matrix A, and not on the cost vector c. This motivates our *kernel algorithm*, described in detail in Algorithm 3:

(1) Given the matrix A, compute a small generating set of the toric ideal I_A following [3].
(2) For each $j \in \{1, \ldots, N\}$, apply Buchberger's algorithm to compute the Gröbner base with respect to A and $<_{c_j}$.
(3) Use the correspondence $(x^u - x^v) \longleftrightarrow u - v$ from (3.6) to obtain the test set with respect to (A, c_j).
(4) For each $i, j \in \{1, \ldots, N\}$, apply the augmentation algorithm (Algorithm 1) to compute $Q(x_i, \xi_j)$, and hence the (i, j)-entry of the opportunity cost matrix.

3.2. **The Graver basis approach.** In the previous section, computing the $N \times N$ opportunity cost matrix still required computing N test sets via the Gröbner bases. This motivates the question whether it is possible to construct a single test set that not only covers all right-hand side vectors b but also all cost vectors c.

Definition 3.11. A set $\mathcal{T} \subseteq$ is an *universal test set* for the matrix A if it contains a test set $\mathcal{T}_c$ for every cost vector c.

Clearly, the Gröbner basis approach alone is insufficient for computing a universal test set, as this would require computing an infinite number of Gröbner bases. Instead, the Graver basis approach will be used. We begin by stating the definition of a Graver basis.

Definition 3.12. Assume $a = (a_1, a_2, ..., a_n), b = (b_1, b_2, ..., b_n) \in \mathbb{Z}^n$. Then $a \sqsubseteq b$ if $\forall i$: $a_i b_i \geqslant 0$ and $|a_i| \leqslant |b_i|$.

Definition 3.13. Fix a integer matrix A. The Graver Basis of A is the set of $\sqsubseteq$-minimal elements in $\mathrm{Ker}_{\mathbb{Z}}(A) := \{u \in \mathbb{Z}^n : A \cdot u = 0, u \neq 0\}$, denoted by Γ_A.

It is not immediately obvious that Γ_A is finite. However, finiteness of Γ_A follows from the following result gathered from [12].

Proposition 3.14. *Let $\mathcal{G}_{A,c}$ be the reduced Gröbner basis of A with respect to $> c$. Then*

$$\bigcup_{c \in \mathbb{Z}^n} \mathcal{G}_{A,c} \subseteq \mathcal{GR}_A.$$

Next, we state and prove a result about the relationship between the Graver Basis of the stochastic integer program 2.3 and that of the deterministic integer program obtained by choosing a single scenario and assigning probability 1 to it:

$$\min\left\{c^\top x + pq^\top y : Ax = b, x \in \mathbb{Z}_+^m, Tx + Wy = h, y \in \mathbb{Z}_+^n\right\} \tag{3.12}$$

Then we assert a new result, which connects the Graver Basis of the SIP problem and the Graver Basis of the IP problem:

Theorem 3.15. *Denote the Graver Basis of the SIP problem (5) by Γ_N, and that of the IP problem (3.12) by Γ_1 for $i = 1, 2, ..., N$. If $\mathrm{Ker}_{\mathbb{R}}(A) = 0$, we have the following relationship*

$$\Gamma_N = \{(0, 0, ..., v_i, ..., 0) : (0, v_i) \in \Gamma_1, i = 1, 2, ..., n\}$$

Proof. The corresponding coefficient matrix of (2.3) is of the form

$$A_N := \begin{pmatrix} A & 0 & 0 & \cdots & 0 \\ T & W & 0 & \cdots & 0 \\ T & 0 & W & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ T & 0 & 0 & \cdots & W \end{pmatrix}$$

and the corresponding coefficient matrix of (6) is of the form

$$A_1 := \begin{pmatrix} A & 0 \\ T & W \end{pmatrix}.$$

Since $\mathrm{Ker}_{\mathbb{R}}(A) = 0$, we have $\mathrm{Ker}_{\mathbb{Z}}(A_N) = (0, v_1, v_2, ..., v_n) : (0, v_i) \in \mathrm{Ker}_{\mathbb{Z}}(A_1)\}$. From the definition of the Graver Basis, we can arrive at the conclusion. $\square$

4. Algorithms

In this section, we describe numerical algorithms that follow from the mathematical development of the previous section. We first introduce the augmentation algorithm that computes an optimal solutions to $(IP)_{c,b}$ from a test set $\mathcal{T}_c$ and a feasible solution z_0.

For details about Algorithm 2 see [3].

The kernel algorithm takes as input the matrix A, a sequence of cost vectors c and an array of right-hand side vectors b and computes the corresponding opportunity cost matrix, by first applying Algorithm 2 and then repeatedly the Buchberger algorithm. In this way, a Gröbner basis for A is computed with respect to each $>_c$. The augmentation algorithm 1 then gives the entries of the opportunity cost matrix for every c and every b.

For the Graver basis approach, we apply the modified augmentation algorithm 4 for a family of integer problems of the form $\min\left\{c^\top x \colon Ax = b\right\}$ as both b and c vary.

Algorithm 1 The augmentation algorithm

```
Require: Cost vector $c$, right-hand side vector $b$
Require: Matrix $A$
Require: Test set $\mathcal{T}_c$
Require: $z_0$ such that $Az_0 = b$.
  while $\exists\, t \in \mathcal{T}_c$ such that $z_0 - t$ is feasible do
      $z_0 \leftarrow z_0 - t$
  end while
      return Optimal solution $z_0 = \operatorname{argmin}\left\{c^T x \colon Ax = b\right\}$
```

Algorithm 2 Algorithm to compute a small generating set of the Toric ideal [3]

```
Require: Integer matrix $A$
  Calculate a $\mathbb{Z}-$basis $K$ for $\ker(\pi_*)$
  Find an equivalent basis $K'$ such that all rows of $K'$ lie in the same orthant.
  $J \leftarrow$ index set of all columns with negative entries and let $K'_J$ be the matrix obtained from
  $K'$ by reversing the signs of the columns indexed by $J$.
  $G_J \leftarrow \varphi\,(K'_J)$.
  while $J \neq \varnothing$ do
      Take $j \in J$ and let $G_{J\setminus\{j\}}$ be the result of $T_j$ operating on the reduced Gröbner basis
  for $\langle G_J \rangle$ with respect to a term order that eliminates $x_j$
      $J \leftarrow J \setminus \{j\}$
  end while
      return The generating set $G_\varnothing$ of $I_A$.
```

Algorithm 3 The Kernel Algorithm

```
Require: Cost vectors $\{c_1, \ldots, c_N\}$
Require: Right-hand side vectors $\{b_{i,j}\}_{i,j=1}^N$
Require: Matrix $A$
  Compute a generating set $\{f_1, \ldots, f_r\}$ of the Toric ideal using Algorithm 2
  $i \leftarrow 1$
  while $i \leqslant N$ do
      Compute the Gröbner basis $\mathcal{G}_i$ of $(A, >_{c_i})$ via the Buchberger algorithm
      $j \leftarrow 1$
      while $j \leqslant N$ do
          Compute $\mathcal{A}_{i,j} = (IP)_{c_i, b_{i,j}}$ by inputting $\mathcal{G}_i$ into the Augmentation Algorithm 1
          $j \leftarrow j + 1$
      end while
      $i \leftarrow i + 1$
  end while
      return The opportunity cost matrix $(\mathcal{A}_{i,j})_{i,j=1}^N$
```

Algorithm 4 Graver Basis method

```
Require: Cost vectors $\{c_1, \dots, c_N\}$
Require: Right-hand side vectors $\{b_{i,j}\}_{i,j=1}^N$
  Compute the Graver Basis $\Gamma_A$ of $A$
Require:
  $i \leftarrow 1$
  while $i \leqslant N$ do
      $j \leftarrow 1$
      while $j \leqslant N$ do
          Compute a feasible solution $z_{i,j}$ to $Ax = b_{i,j}$
          while $\exists\, t \in \Gamma_A : c_i^\top t > 0$ and s.t. that $z_0 - t$ is feasible do
              $z_{i,j} := z_{i,j} - t$
          end while
          Compute $\mathcal{A}_{i,j} = c_j^\top z_{i,j}$
      end while
  end while
      return The opportunity cost matrix $\mathcal{A}$
```

5. Preliminary results

In this section we test our approach on some numerical examples. The first is a well-known purely combinatorial stochastic program introduced in [13]. The second is a stochastic network design problem.

5.1. **The example of [13].** In [13], the following stochastic integer program was introduced and studied.

$$
\begin{aligned}
&\min \left\{ 35x_1 + 40x_2 + \frac{1}{N}\sum_{\nu=1}^{N} 16y_1^\nu + 19y_2^\nu + 47y_3^\nu + 54y_4^\nu \right\} \\
&\text{s.t. } x_1 + y_1^\nu + y_3^\nu \geqslant \xi_1^\nu \\
&\quad\;\; x_2 + y_2^\nu + y_4^\nu \geqslant \xi_2^\nu \\
&\quad\;\; 2y_1^\nu + y_2^\nu \leqslant \xi_3^\nu \\
&\quad\;\; y_1^\nu + 2y_2^\nu \leqslant \xi_4^\nu, \\
&\quad\;\; x_1, x_2, y_1^\nu, y_2^\nu, y_3^\nu, y_4^\nu \in \mathbb{Z}_+
\end{aligned} \tag{5.1}
$$

As in [13] the scenarios ξ^ν are chosen uniformly from the four-dimensional squares $[300, 12000] \times [300, 12000] \times [200, 12000] \times [200, 12000]$. In order to bring the problem in the form (2.1), we introduce slack variables u_j^ν for $j \in \{1, 2, 3, 4\}$ and obtain

$$
\begin{aligned}
&\min \left\{ \frac{1}{N}\sum_{\nu=1}^{N} 35x_1 + 40x_2 + 16y_1^\nu + 19y_2^\nu + 47y_3^\nu + 54y_4^\nu \right\} \\
&\text{s.t. } x_1 + y_1^\nu + y_3^\nu - u_1^\nu = \xi_1^\nu \\
&\quad\;\; x_2 + y_2^\nu + y_4^\nu - u_2^\nu = \xi_2^\nu \\
&\quad\;\; 2y_1^\nu + y_2^\nu + u_3^\nu = \xi_3^\nu
\end{aligned} \tag{5.2}
$$

$$y_1^\nu + 2y_2^\nu + u_4^\nu = \xi_4^\nu,$$
$$x_1, x_2, y_1^\nu, y_2^\nu, y_3^\nu, y_4^\nu, u_1^\nu, u_2^\nu, u_3^\nu, u_4^\nu \in \mathbb{Z}_+.$$

In order to test our approach, we calculated the seconds of CPU time used by the programs on the Apple Silicon M1 chip. We observe that it is easy to read off a feasible solution to the problem (5.2)

$$x_1 = x_2 = y_1^\nu = y_2^\nu = 0, y_3^\nu = \xi_1^\nu, y_4^\nu = \xi_2^\nu \quad (\nu = 1, \dots N) \tag{5.3}$$

The task is to compare the computational costs of the opportunity-cost matrix in the SIP problem in the last section. In other words, we compute a family of integer programming problems with three algorithms. We test the program's running time in CPU seconds on an Apple Silicon M1 chip with Python's `time` package. Also, we introduce the modern MINLP solver, specially designed for mixed-integer programming problems, implemented by Python's `gekko` package. We shall compare the computational costs of our algorithms 3 and 4 with the MINLP solver in two cases. One is when we fix the sub-problem, and the other is when we fix the number of scenarios.

We shall compute the computational time when the quantity of scenarios equals 1, 20, 40, 60, 80, 100, 120, 140, 160, 180, and 200. We gather the numerical results in the table below.

TABLE 1. Comparison between different solvers (Timing units: seconds)

Scenario Number	Kernel Method	MINLP solver	Graver Method
1	0.20098	11.33629	131.06186
20	1.21699	219.18796	131.66931
40	2.33655	439.71981	132.02384
60	3.53545	643.3165	132.35069
80	4.68832	994.41339	132.72208
100	5.93403	1250.31321	133.14721
120	7.24906	1508.7674	133.61075
140	8.56163	1803.57992	134.06991
160	10.08377	2075.8631	134.61703
180	11.25169	2333.33849	135.21214
200	12.88583	2584.11191	135.8798

It is remarkable that it only took about 0.1 seconds to compute the generating set of the toric ideal of A, essential for the Gröbner Bases computation in all scenarios. By contrast, it took approximately 130 seconds to compute the Graver Basis of A. Also we visualize the table in a line chart below. The horizontal axis represents the increasing quantity of scenarios, and the vertical axis represents the running time. The three curves with different colors in the picture represent three different methods.

Figure 5.1 shows that Algorithm 4 took almost 130 seconds to initialize and then the curve rises slowly. The computational cost of the MINLP solver grows extremely fast and starts to overpower the Algorithm 4 when $N = 20$. Generally, the curve of Algorithm 3 shows superior performance when compared with the other two algorithms.

Next, we compare when the complexity of the sub-problems increases. Below is the graph showing the trends of computational costs when the size of scenarios is fixed to 200 but the number of scenarios is increased. In Figure 5.2, with a fixed 200 scenarios, var*x represents x variables in the particular programming problem. As can be seen, by comparing the curves,

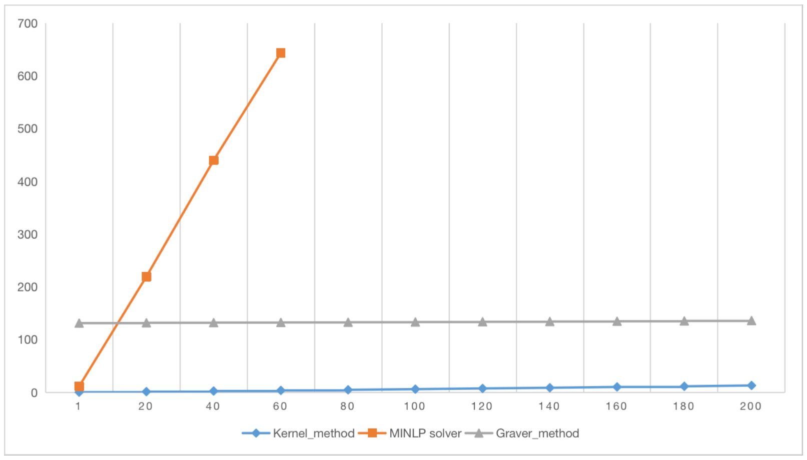

FIGURE 5.1. Trends of Computational Costs when the Sub-problem is Fixed

the Gröbner Basis approach (the blue curve) is overall superior to the Graver Basis approach (the grey curve) and the traditional MINLP solver (the yellow curve). Also, regarding details, the Graver Basis method begins to suffer from the curse of dimensionality, starting from the problem's complexity being 20 variables, and so is the Gröbner Basis method but more dramatically. However, the MINLP solver does not reflect the apparent complexity when it suffers from the curse. Generally, the MINLP solver applies to situations where the sub-problem is large, and the number of scenarios is small. In contrast, the Gröbner Basis applies to situations where the sub-problem has little complexity, and the number of scenarios is significant.

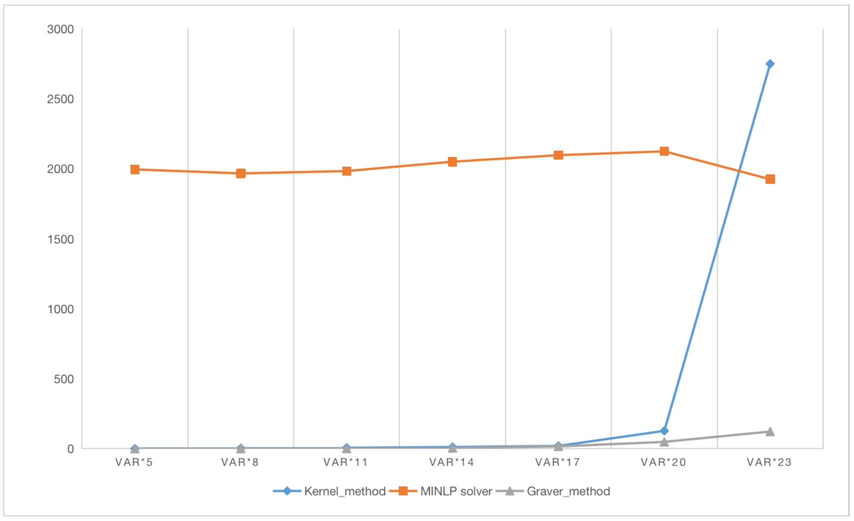

FIGURE 5.2. Trends of Computational Costs when the Quantity of Scenarios is Fixed

We have yet to reflect one thing in the experiments. By testing our algorithm hundreds of times, we have come to the empirical conclusion that the time increase of our proposed algorithm in computing Grobner Basis will be dramatic for $N > 15$. And there will be geometric growth for the computational time regarding Graver Basis when the complexity is between 10 and 15 variables. Also, surprisingly, we can no longer compute the Gröbner Basis after 15 variables' complexity. We refer to this phenomenon as the curse of dimensionality in computational algebraic geometry regarding SIP problems. We conjecture that the crux is that the number of ideal minimal generating elements will increase dramatically as the dimension increases, thus making Bunchberger's algorithm computationally costly.

Finally, the crux of the problem, that two algebraic methods don't function when the variables' number exceeds 15, is the computation of the Gröbner Basis of some specific ideals, specifically the toric ideal of the coefficient matrix A, unavoidable in each of the two algorithms. We have tried many applications specialized in algebraic calculations, for example, SageMath, Macaulay2, etc. Still, when the variables' number exceeds 15, usually the exponential of the generating elements of the polynomial ideal, which is required in the algorithm, is too great to be computable. For example, the computation of the Gröbner Basis of the matrix $A = (B, I)$ where I denotes the identity matrix and

$$B = \begin{pmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 & 10 \\ 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 & 10 & 11 \\ 3 & 4 & 5 & 6 & 7 & 8 & 9 & 10 & 11 & 12 \\ 4 & 5 & 6 & 7 & 8 & 9 & 10 & 11 & 12 & 13 \\ 5 & 6 & 7 & 8 & 9 & 10 & 11 & 12 & 13 & 14 \\ 6 & 7 & 8 & 9 & 10 & 11 & 12 & 13 & 14 & 15 \end{pmatrix}$$

involves the computation of the Gröbner Basis of the toric ideal below

$$\text{ideal}(x_8^2 - x_7 * x_9,\ x_7 * x_8 - x_6 * x_9, x_6 * x_8 - x_5 * x_9, x_5 * x_8 - x_4 * x_9, x_4 * x_8 - x_3 * x_9,\ x_3 * x_8 - x_2 * x_9,\ x_2 * x_8 - x_1 * x_9,\ x_1 * x_8 - x_0 * x_9,\ x_7^2 - x_5 * x_9,\ x_6 * x_7 - x_4 * x_9, x_5 * x_7 - x_3 * x_9, x_4 * x_7 - x_2 * x_9, x_3 * x_7 - x_1 * x_9,\ x_2 * x_7 - x_0 * x_9,\ x_1 * x_7 - x_0 * x_8, x_6^2 - x_3 * x_9,\ x_5 * x_6 - x_2 * x_9, x_4 * x_6 - x_1 * x_9, x_3 * x_6 - x_0 * x_9, x_2 * x_6 - x_0 * x_8,\ x_1 * x_6 - x_0 * x_7,\ x_5^2 - x_1 * x_9,\ x_4 * x_5 - x_0 * x_9, x_3 * x_5 - x_0 * x_8, x_2 * x_5 - x_0 * x_7, x_1 * x_5 - x_0 * x_6, x_4^2 - x_0 * x_8,\ x_3 * x_4 - x_0 * x_7, x_2 * x_4 - x_0 * x_6, x_1 * x_4 - x_0 * x_5, x_3^2 - x_0 * x_6,\ x_2 * x_3 - x_0 * x_5, x_1 * x_3 - x_0 * x_4, x_2^2 - x_0 * x_4,\ x_1 * x_2 - x_0 * x_3, x_1^2 - x_0 * x_2,\ x_8 * x_{11} * x_{12} * x_{13} * x_{14} * x_{15} * x_{16} - x_9, x_7 * x_{11} * x_{12} * x_{13} * x_{14} * x_{15} * x_{16} - x_8, x_6 * x_{11} * x_{12} * x_{13} * x_{14} * x_{15} * x_{16} - x_7, x_5 * x_{11} * x_{12} * x_{13} * x_{14} * x_{15} * x_{16} - x_6, x_4 * x_{11} * x_{12} * x_{13} * x_{14} * x_{15} * x_{16} - x_5, x_3 * x_{11} * x_{12} * x_{13} * x_{14} * x_{15} * x_{16} - x_4, x_2 * x_{11} * x_{12} * x_{13} * x_{14} * x_{15} * x_{16} - x_3, x_1 * x_{11} * x_{12} * x_{13} * x_{14} * x_{15} * x_{16} - x_2, x_0 * x_{11} * x_{12} * x_{13} * x_{14} * x_{15} * x_{16} - x_1, x_9 * x_{10} * x_{14} * x_{15}^2 * x_{16}^3 - x_0 * x_6 * x_{11}^2 *\ x_{12}, x_8 * x_{10} * x_{14} * x_{15}^2 * x_{16}^3 - x_0 * x_5 * x_{11}^2 *\ x_{12}, x_7 * x_{10} * x_{14} * x_{15}^2 * x_{16}^3 - x_0 * x_4 * x_{11}^2 *\ x_{12}, x_6 * x_{10} * x_{14} * x_{15}^2 * x_{16}^3 - x_0 * x_3 * x_{11}^2 *\ x_{12}, x_5 * x_{10} * x_{14} * x_{15}^2 * x_{16}^3 - x_0 * x_2 * x_{11}^2 *\ x_{12}, x_4 * x_{10} * x_{14} * x_{15}^2 * x_{16}^3 - x_0 * x_1 * x_{11}^2 *\ x_{12}, x_3 * x_{10} * x_{14} * x_{15}^2 * x_{16}^3 - x_0^2 * x_{11}^2 *\ x_{12}, x_0 * x_5 * x_{11}^3 * x_{12}^2 * x_{13} - x_9 * x_{10} * x_{15} * x_{16}^2, x_0 * x_4 * x_{11}^3 * x_{12}^2 * x_{13} - x_8 * x_{10} * x_{15} * x_{16}^2, x_0 * x_3 * x_{11}^3 * x_{12}^2 * x_{13} - x_7 * x_{10} * x_{15} * x_{16}^2, x_0 * x_2 * x_{11}^3 * x_{12}^2 * x_{13} - x_6 * x_{10} * x_{15} * x_{16}^2, x_0 * x_1 * x_{11}^3 * x_{12}^2 * x_{13} - x_5 * x_{10} * x_{15} * x_{16}^2, x_0^2 * x_{11}^3 * x_{12}^2 * x_{13} - x_4 * x_{10} * x_{15} * x_{16}^2, x_2 * x_{10} * x_{13} * x_{14}^2 * x_{15}^3 * x_{16}^4 - x_0^2 * x_{11}, x_1 * x_{10} * x_{12} * x_{13}^2 * x_{14}^3 * x_{15}^4 * x_{16}^5 - x_0^2, x_{10} * x_{11} * x_{12}^2 * x_{13}^3 * x_{14}^4 * x_{15}^5 * x_{16}^6 - x_0),$$

In future work, we will study how the situation can be improved by optimising the algorithm for computation of Gröbner bases, for example, the Faugère F_4 [9] and F_5 [10] algorithms.

5.2. **Stochastic network design.** We have also tested our approach on a small stochastic network design problem, formulated as follows:

$$\min \sum_{a \in A} c_a x_a + \frac{1}{N} \sum_{i=1}^{N} \sum_{c \in C} \sum_{a \in A} q_{ac} y^i_{ac} \tag{5.4}$$

$$\text{s.t.} \quad \sum_{\substack{a \in A \\ a(0)=v}} y^i_{ac} - \sum_{\substack{a \in A \\ a(1)=v}} y^i_{ac} = d^i_{v,c} \quad \forall (v,c,i) \in V \times C \times \{1,\dots,N\} \tag{5.5}$$

$$\sum_{c \in C} y^i_{ac} \leqslant u_a x_a \quad \forall (a,i) \in A \times \{1,\dots,N\} \tag{5.6}$$

$$x_a \in \{0,1\}, y^i_{ac} \in \mathbb{Z}_{\geqslant 0} \tag{5.7}$$

Following the same procedure as in Section 5.1, we compare our two algorithms with a commercial solver. Our instances concern a small problem, with three vertices and three arcs. Figure 5.3 shows the time taken to compute the opportunity cost matrix for each approach. We once more observe the superior efficiency of the kernel approach when the

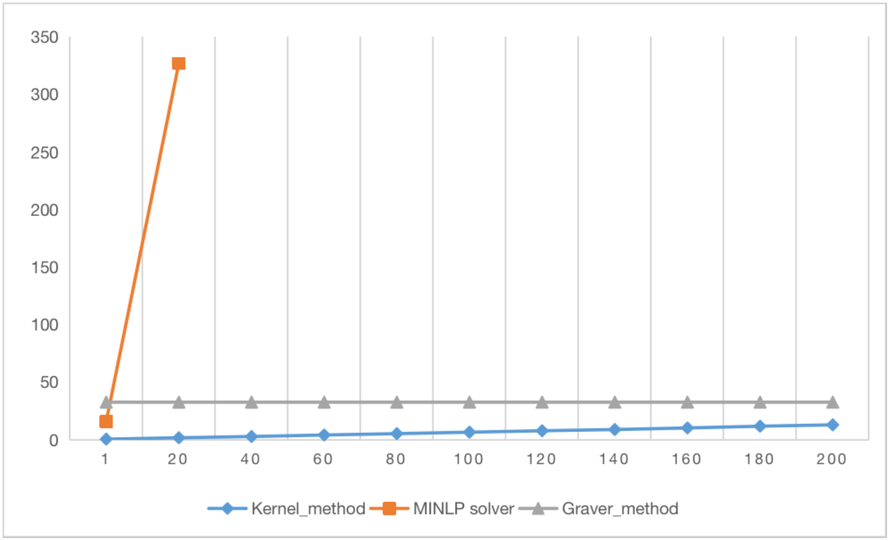

FIGURE 5.3. Trends of Computational Costs when the Sub-problem is Fixed (Stochastic Network Design Problem)

number of decision variables is small.

6. Conclusion

In this paper, we have presented new algorithms for computing the opportunity cost matrix of [15] using algebraic methods using Gröbner and Graver bases. We have also provided mathematical results showing the relationship of Graver bases for deterministic and stochastic integer programs. Our numerical results show that the Gröbner basis is very efficient as the number of scenarios increases if the number of variables of the underlying integer program does not grow too fast. On the other hand, the Graver basis approach is interesting when there are a large number of variables.

In future work, we propose to include more recent methods of computing Gröbner basis, such as the Faugère algorithms. We will also study how to combine the algebraic approach with approximations such as Lagrangian cuts in order to increase further the number of decision variables that our approach can handle.

References

[1] Dimitris Bertsimas and Nishanth Mundru. "Optimization-Based Scenario Reduction for Data-Driven Two-Stage Stochastic Optimization". *Operations Research* (2022). ISSN: 0030-364X. DOI: 10.1287/opre.2022.2265.

[2] Dimitris Bertsimas, Georgia Perakis, and Sridhar Tayur. "A New Algebraic Geometry Algorithm for Integer Programming". *Management Science* 46 (7 2000), pp. 999–1008. ISSN: 0025-1909. DOI: 10.1287/mnsc.46.7.999.12033.

[3] Fausto Di Biase and Rüdiger Urbanke. "An Algorithm to Calculate the Kernel of Certain Polynomial Ring Homomorphisms". *Experimental Mathematics* 4 (3 1995), pp. 227–234. ISSN: 1058-6458. DOI: 10.1080/10586458.1995.10504323.

[4] John R. Birge and François Louveaux. *Introduction to Stochastic Programming.* Springer New York, 2011. ISBN: 978-1-4614-0236-7. DOI: 10.1007/978-1-4614-0237-4.

[5] Bruno Buchberger. *Gröbner Bases: An Algorithmic Method in Polynomial Ideal Theory.* Ed. by N K Bose. 1985. DOI: 10.1007/978-94-017-0275-1_4.

[6] Pasqualina Conti and Carlo Traverso. *Buchberger algorithm and integer programming.* Ed. by H. F. Mattson, T. Mora, and T. R. N Rao. 1991. DOI: 10.1007/3-540-54522-0_102.

[7] David A. Cox, John Little, and Donal O'Shea. *Ideals, Varieties, and Algorithms.* Springer International Publishing, 2015. ISBN: 978-3-319-16720-6. DOI: 10.1007/978-3-319-16721-3.

[8] Jitka Dupacová, Nicole Gröwe-Kuska, and Werner Römisch. "Scenario reduction in stochastic programming". *Mathematical Programming* 95 (3 2003), pp. 493–511. ISSN: 0025-5610. DOI: 10.1007/s10107-002-0331-0.

[9] Jean-Charles Faugère. "A new efficient algorithm for computing Gröbner bases (F4)". *Journal of Pure and Applied Algebra* 139 (1-3 1999), pp. 61–88. ISSN: 00224049. DOI: 10.1016/S0022-4049(99)00005-5.

[10] Jean-Charles Faugère. "A new efficient algorithm for computing Gröbner bases without reduction to zero (F5)". ACM, 2002, pp. 75–83. ISBN: 1581134843. DOI: 10.1145/780506.780516.

[11] Jack E. Graver. "On the foundations of linear and integer linear programming I". *Mathematical Programming* 9 (1 1975), pp. 207–226. ISSN: 0025-5610. DOI: 10.1007/BF01681344.

[12] Raymond Hemmecke. "On the positive sum property and the computation of Graver test sets". *Mathematical Programming* 96 (2 2003), pp. 247–269. ISSN: 0025-5610. DOI: 10.1007/s10107-003-0385-7.

[13] Raymond Hemmecke and Rüdiger Schultz. "Decomposition of test sets in stochastic integer programming". *Mathematical Programming* 94 (2-3 2003), pp. 323–341. ISSN: 0025-5610. DOI: 10.1007/s10107-002-0322-1.

[14] René Henrion, Christian Küchler, and Werner Römisch. "Scenario reduction in stochastic programming with respect to discrepancy distances". *Computational Optimization and Applications* 43 (1 2009), pp. 67–93. ISSN: 0926-6003. DOI: 10.1007/s10589-007-9123-z.

[15] Mike Hewitt, Janosch Ortmann, and Walter Rei. "Decision-based scenario clustering for decision-making under uncertainty". *Annals of Operations Research* 315 (2 2022), pp. 747–771. ISSN: 15729338. DOI: 10.1007/s10479-020-03843-x.

[16] Julien Keutchayan, Janosch Ortmann, and Walter Rei. "Problem-Driven Scenario Clustering in Stochastic Optimization" (2021).

[17] Anton J. Kleywegt, Alexander Shapiro, and Tito Homem de Mello. "The Sample Average Approximation Method for Stochastic Discrete Optimization". *SIAM Journal on Optimization* 12 (2 2002), pp. 479–502. ISSN: 1052-6234. DOI: 10.1137/S1052623499363220.

[18] Werner Römisch. *Scenario Reduction Techniques in Stochastic Programming*. Ed. by Osamu Watanabe and T Zeugmann. 2009. DOI: 10.1007/978-3-642-04944-6_1.

[19] Rüdiger Schultz. "On structure and stability in stochastic programs with random technology matrix and complete integer recourse". *Mathematical Programming* 70 (1995), pp. 73–89.

[20] Rüdiger Schultz, Leen Stougie, and Maarten H. van der Vlerk. "Solving stochastic programs with integer recourse by enumeration: A framework using Gröbner basis". *Mathematical Programming* 83 (1-3 1998), pp. 229–252. ISSN: 0025-5610. DOI: 10.1007/BF02680560.

[21] Bernd Sturmfels. "Algebraic Recipes for Integer Programming". 2004, pp. 99–113.

[22] Bernd Sturmfels. *Grobner bases and convex polytopes*. American Mathematical Society, 1996.

[23] Rekha R. Thomas. *Gröbner Bases in Integer Programming*. 1998. DOI: 10.1007/978-1-4613-0303-9_8.

[24] R.R. Thomas and R. Weismantel. "Truncated Gröbner bases for integer programming". *Applicable Algebra in Engineering, Communications and Computing* 8 (4 1997), pp. 241–256. DOI: 10.1007/s002000050062.

Yuchen Ge, Shandong University, Jinan, China
Email address: gycdwwd@163.com
URL: https://gycdwwd.github.io/

Janosch Ortmann, CRM, GERAD and Université du Québec à Montréal, Case postale 8888, succ. Centre-ville, Montréal (QC) H3C 3P8, Canada
Email address: ortmann.janosch@uqam.ca
URL: http://crm.umontreal.ca/~ortmann/

Walter Rei, CIRRELT and Université du Québec à Montréal, Case postale 8888, succ. Centre-ville, Montréal (QC) H3C 3P8, Canada
Email address: rei.walter@uqam.ca