Column-Randomized Linear Programs: Performance Guarantees and Applications

Yi-Chun Chen
UCLA Anderson School of Management, University of California, Los Angeles, California 90095, United States,
yi-chun.chen.phd@anderson.ucla.edu

Velibor V. Mišić
UCLA Anderson School of Management, University of California, Los Angeles, California 90095, United States,
velibor.misic@anderson.ucla.edu

We propose a randomized method for solving linear programs with a large number of columns but a relatively small number of constraints. Since enumerating all the columns is usually unrealistic, such linear programs are commonly solved by column generation, which is often still computationally challenging due to the intractability of the subproblem in many applications. Instead of iteratively introducing one column at a time as in column generation, our proposed method involves sampling a collection of columns according to a user-specified randomization scheme and solving the linear program consisting of the sampled columns. While similar methods for solving large-scale linear programs by sampling columns (or, equivalently, sampling constraints in the dual) have been proposed in the literature, in this paper we derive an upper bound on the optimality gap that holds with high probability and converges with rate $1/\sqrt{K}$, where $K$ is the number of sampled columns, to the value of a linear program related to the sampling distribution. To the best of our knowledge, this is the first paper addressing the convergence of the optimality gap for sampling columns/constraints in generic linear programs without additional assumptions on the problem structure and sampling distribution. We further apply the proposed method to various applications, such as linear programs with totally unimodular constraints, Markov decision processes, covering problems and packing problems, and derive problem-specific performance guarantees. We also generalize the method to the case that the sampled columns may not be statistically independent. Finally, we numerically demonstrate the effectiveness of the proposed method in the cutting-stock problem and in nonparametric choice model estimation.

1. Introduction

We consider solving a linear program (LP) in standard form:

$$\begin{align*}
\text{minimize} & \quad c^T x \\
\text{such that} & \quad Ax = b, \\
& \quad x \geq 0,
\end{align*}$$

where $x \in \mathbb{R}^n$, $c \in \mathbb{R}^n$, $A \in \mathbb{R}^{m \times n}$, and $b \in \mathbb{R}^m$. In various applications of linear programming, such as the cutting-stock problem (Gilmore and Gomory 1961) and the vehicle routing problem (Dumas et al. 1991), it is often the case that the number of variables $n$ is much larger than the number of constraints $m$. Given that there are many more columns than constraints and enumerating all of the columns is impossible in most cases, a standard solution method is column generation (CG), which works as follows: (i) start with an initial set of columns from $A$; (ii) solve the corresponding restricted linear program to optimality; (iii) solve a subproblem to find the column with the lowest reduced cost; (iv) add the new column to the current set of columns; (v) go back to step (i) until problem (1) is solved to optimality (i.e., the minimum reduced cost in step (iii) is nonnegative). The subproblem that ones solves to introduce a new column is often computationally challenging. For example, in the cutting-stock problem, the subproblem is a knapsack problem, which is NP-hard.
Chen and Mišić: Column-Randomized Linear Programs

(Garey and Johnson 1979). In practice, the subproblem is often formulated as an integer program, and can be difficult to solve at a large scale. In addition, CG is a sequential method, that is, the subproblem that one solves to introduce the ith column depends on the computational results of the previous i − 1 iterations. Such a structure prohibits one from applying parallel computing techniques to implement the column generation method.

Instead of searching for columns by a subproblem that is potentially NP-hard, we propose a randomized method, called column randomization. In this method, one first samples a collection of columns according to a user-specified randomization scheme, and then solves the corresponding restricted linear program. We refer to this restricted linear program that consists of sampled columns as the column-randomized linear program. This approach is attractive because computationally, it is often significantly easier to randomly sample columns than it is to optimize over columns (as is the case in CG). In addition, while CG operates sequentially, the sampling step in column randomization is well-suited to parallelization.

We note that similar sampling-based methods for large-scale LPs have been previously considered in the operations research literature. In particular, there is a significant literature on solving problems with large numbers of constraints by randomly sampling constraints (De Farias and Van Roy 2004, Calafiore and Campi 2005). By strong duality of linear programs, sampling the columns of problem (1) is equivalent to sampling the constraints of its dual problem. However, the behavior of the sampled LP in terms of its optimality gap – the difference in objective value between the sampled problem and the complete problem – has received scarce attention in the literature. In this paper, our main goal is to answer the following question: Given a user-specified randomization scheme for sampling columns from a linear program, is it possible to probabilistically bound the optimality gap of the column-randomized linear program?

We provide theoretical results to answer this question and demonstrate how these results can be applied to common applications of large-scale linear programming. We make the following specific contributions:

1. **Theoretical Guarantees**. We show that with high probability over the sample of columns, the optimality gap of the column-randomized linear program is bounded by the sum of two terms: the optimality gap of a linear program related to the sampling distribution and a term that is of order 1/√K, where K is the number of sampled columns. To best of our knowledge, this is the first theoretical result that addresses the behavior of the optimality gap of the column/constraint sampling technique for general linear programs.

2. **Problem-Specific Bounds**. We apply the proposed method to several applications of large-scale linear programming and derive problem-specific upper bounds for the optimality gap. The problems include LPs with totally unimodular constraints, Markov decision processes (MDP), covering problems and packing problems. We also extend our approach to the portfolio optimization problem, in which the objective function is only assumed to be Lipschitz continuous (and is not necessarily linear or convex).

3. **Generalization to Non-I.I.D. Samples**. While the literature has mainly focus on independent and identically distributed (i.i.d.) samples, we generalize the randomization scheme to the case where the sampled columns may be statistically dependent, and develop a theoretical guarantee for this case. We apply our guarantee to a simple non-independent randomization scheme, where one samples n_c columns from each of n_G groups of columns, which applies to many LPs with columns that have a natural group structure (such as MDPs).

4. **Numerical Results**. We numerically demonstrate the effectiveness of the proposed method on two optimization problems that are commonly solved by CG: the cutting-stock problem, which is a classical application of linear programming; and the nonparametric choice model estimation problem, which is a modern application of linear programming. We compare the performance of the column randomization method to that of the CG method and show that for a fixed optimality gap, the column randomization method can attain that optimality gap within a fraction of the time required by CG. Thus, for some problems, the column
randomization method can be a viable alternative to CG or can otherwise be used to provide a good warm start solution for CG.

We organize the paper as follows. In Section 2, we review the related literature and highlight our contribution. In Section 3, we state our theoretical results and discuss their implications. We provide proofs of our main results in Section 4. In Section 5, we apply our method to several applications of large-scale LP and derive problem-specific guarantees. In Section 6, we generalize our approach to sampling non-i.i.d. columns. In Section 7, we present our numerical results, and we conclude in Section 8. Omitted proofs are provided in the electronic companion.

2. Literature Review

In this section, we review four streams of literature. First, we discuss the CG method and large-scale LPs. Second, we discuss existing papers on column/constraint sampling, and highlight our contributions. Third, we briefly review work in randomized methods, stochastic optimization and online linear programming. Lastly, we also describe several other papers from the machine learning literature, particularly on random feature methods, that relate closely to the proof technique used in our results.

Column Generation. CG has been widely used to solve optimization problems that have a huge number of columns compared to the number of constraints (Ford Jr and Fulkerson 1958, Dantzig and Wolfe 1960, du Merle et al. 1999). Applications include vehicle routing (Dumas et al. 1991, Feillet 2010), facility location problems (Klose and Drexl 2005), and choice model estimation (van Ryzin and Vulcano 2015, Mišić 2016); we refer readers to Desrosiers and Lübbecke (2005) for a comprehensive review. By strong duality of linear programs, CG is equivalent to constraint generation that solves linear programs with a large number of constraints (Bertsimas and Tsitsiklis 1997). A key component of both methods is the subproblem that one solves to iteratively introduce columns or constraints. Usually, this subproblem is computationally challenging and is often solved by integer programming. For example, in the cutting-stock problem, the CG subproblem is a knapsack problem, which is NP-hard (Gilmore and Gomory 1961, Garey and Johnson 1979).

Sampling Columns/Constraints. Another approach to solving LPs with huge numbers of columns (or equivalently, with huge numbers of constraints), is by sampling (De Farias and Van Roy 2004, Calafiore and Campi 2005, 2006, Campi and Garatti 2008, 2018). Specifically, one first samples a set of columns (or constraints) according to a given distribution then solves a linear program that consists of the sampled columns (or constraints). The seminal paper of De Farias and Van Roy (2004) proposed the constraint sampling method for linear programs that arise in approximate dynamic programming (ADP). Given a distribution for sampling the constraints, the paper showed that with high probability over the sampled set of constraints, any feasible solution of the sampled problem is nearly feasible for the complete problem (that is, there is a high probability of satisfying a new random constraint, sampled according to the same distribution). Under the additional assumption that the constraint sampling distribution is a Lyapunov function, the paper also develops a specific guarantee on the error between the optimal value function and the approximate value function that is obtained by solving the sampled problem, but does not relate the objective value of the sampled and complete problems. In contrast, the results of our paper pertain specifically to the objective value of the sampled problem, are free from any assumptions on the sampling distribution and are applicable to general linear programs beyond those arising in ADP. Around the same period, Calafiore and Campi (2005, 2006) pioneered the sampling approach to robust convex optimization. With a different perspective from De Farias and Van Roy (2004), Calafiore and Campi (2005, 2006) also characterized the sample complexity needed for the optimal solution (as opposed to an arbitrary feasible solution) of the sampled problem to be nearly feasible for the original problem. However, the performance of the sampled problem in terms of the objective value, and its dependence on the number of samples, was not addressed.

Since the works of Calafiore and Campi (2005) and De Farias and Van Roy (2004), there has been some work that has quantified the dependence of the objective value on the number of sampled
constraints. In particular, the paper of Mohajerin Esfahani et al. (2014) considers a convex program where the decision variable $x$ satisfies a family of convex constraints, which are later sampled, and is also constrained to lie in an ambient set $X$. The paper develops a probabilistic bound on the difference in objective value between the complete problem and its sampled counterpart in terms of a uniform level-set bound (ULB), which is a quantile function of the worst-case probability over all feasible solutions in set $X$. Our work differs significantly from Mohajerin Esfahani et al. (2014) in two aspects. First, in terms of the problem setting, Mohajerin Esfahani et al. (2014) assumes that even before any constraints are sampled, the decision variable is already constrained in the convex compact (and thus bounded) set $X$, and the associated performance guarantees also rely on properties of $X$. In our setting, this corresponds to the dual solutions of problem (1) being bounded, which need not be the case in general. Moreover, we do not assume that the linear program is initialized with a specific set of variables (or equivalently, a set of constraints in the dual) before we sample columns. Consequently, the result of Mohajerin Esfahani et al. (2014) is not directly applicable to the research question discussed in this paper. Second, as noted earlier, the performance bound in Mohajerin Esfahani et al. (2014) relies on the ULB function of the sampling distribution. While sufficient conditions for the existence of a ULB are provided in the paper, in general a ULB cannot be represented explicitly and thus the resulting performance guarantee is less interpretable. In contrast, our theoretical results do not require a ULB or other related functions, and have a more interpretable dependence on the sampling distribution (via the distributional counterpart; see problem (7) in Theorem 1). In addition, we also believe our results are more straightforward technically: one only needs McDiarmid’s inequality and standard linear programming results to prove them. As we will show in Section 5, our theoretical results and proof technique can be applied to many common types of LPs to derive application-specific guarantees.

Randomized Methods, Stochastic Optimization and Online Linear Programming.

Besides column/constraint sampling, many other randomized methods have been proposed to solve large-scale optimization problems, including methods based on random walks (Bertsimas and Vempala 2004) and random projection (Pilanci and Wainwright 2015, Vu et al. 2018). In addition to these randomized methods, there is also a separate literature on optimization problems where stochasticity is part of the problem definition; some examples include stochastic programming (Birge and Louveaux 2011, Shapiro et al. 2014), contextual optimization (Elmachtoub and Grigas 2017), and online optimization (Shalev-Shwartz 2012). Within this literature, the problem setting of online linear programming, where columns of a linear program are revealed sequentially to a decision maker, bears a resemblance to ours; some examples of papers in this area include Agrawal et al. (2014), Eghbali et al. (2018), Li and Ye (2019). Despite this similarity, this problem setting differs significantly from ours in that a decision maker is making irrevocable decisions in an online fashion: the decision maker must decide how much to use of a variable/column at the time that it is revealed, and cannot revise this decision in the future.

Other Related Literature. Our proof technique is inspired by the literature on random feature selection in machine learning (Moosmann et al. 2007, Rahimi and Recht 2008, 2009). In particular, our paper generalizes the result of Rahimi and Recht (2009), which considers the problem of learning a predictive model that is a weighted sum of random feature functions, to the problem of solving linear programs that consist of random columns. The major difference between our setup and that of Rahimi and Recht (2009) is that the decision variables in a linear program must satisfy constraints (i.e., constraints (1b) and (1c)), while the weights of random feature functions in the setup of Rahimi and Recht (2009) are not constrained in any way. Because of this difference, the results of Rahimi and Recht (2009) cannot directly be applied to our problem setting. To overcome this feasibility issue, we utilize classical LP sensitivity analysis and relate a possibly infeasible solution constructed using the random sample of columns to a feasible solution of the sampled LP (see Section 4.2).
3. Theoretical Results

In this section, we first describe the basic notations and definitions that will be used throughout the paper (Section 3.1). Then we formally define the column randomization method and investigate its theoretical properties (Section 3.2). We end this section by discussing implications and interpretations of the theoretical results (Section 3.3). Proofs of the results are relegated to Section 4.

3.1. Notation and Definitions

For any positive integer \( n \), let \( [n] \equiv \{1, 2, \ldots, n\} \). Let \( e_i \) be the \( i \)th standard basis vector for \( \mathbb{R}^n \); that is, \( e_i = (e_{ij}) \) where \( e_{ij} = 1 \) if \( j = i \) and \( e_{ij} = 0 \) if \( j \neq i \). Thus, for any \( x \in \mathbb{R}^n \), we can represent it as \( x = \sum_{i \in [n]} x_i e_i \).

We consider a linear program in standard form:

\[
P: \quad \min \{ c^T x \mid Ax = b, \ x \geq 0 \},
\]

where \( A \) is an \( m \times n \) matrix and \( c \in \mathbb{R}^n \). We will refer to the problem \( P \) as the complete problem throughout the paper, as it contains all of the columns of \( A \).

We define the dual problem of problem (2) as

\[
D: \quad \max \{ p^T b \mid p^T A \leq c^T \}.
\]

For any optimization problem \( P' \), we denote its optimal objective value by \( v(P') \) and its feasible region by \( \mathcal{F}(P') \). By LP strong duality, we have \( v(P) = v(D) \). Furthermore, for any optimization problem \( P'' \) that shares same objective function as the complete problem \( P \) and satisfies \( \mathcal{F}(P'') \subseteq \mathcal{F}(P) \), we define \( \Delta v(P'') \equiv v(P'') - v(P) \), which is nonnegative and can be interpreted as the optimality gap of solving \( P'' \) instead of \( P \).

We make two assumptions on problem \( P \). First, we assume that problem \( P \) is feasible and bounded; this assumption is not too restrictive, since the cases where the complete problem \( P \) is either unbounded or infeasible are not interesting to consider. The second assumption we make is that \( \text{rank}(A) = m \), i.e., the rows of \( A \) are linearly independent. This is also not too restrictive, as one can remove any rows of \( A \) that are linear combinations of the other rows without changing the problem.

For each \( i \in [m] \) and \( j \in [n] \), we use \( A^i \) and \( A_j \) to denote the \( i \)th row and \( j \)th column of matrix \( A \), respectively. For any collection of indices \( J \subseteq [n] \), we let \( A_J \) represent the submatrix of \( A \) that consists of columns whose indices belong to \( J \). In this paper, instead of solving either the complete problem \( P \) or its dual \( D \), we consider solving a linear program whose columns are randomly selected. We call such a linear program a column-randomized linear program, which we formally define below.

**Definition 1.** (Column-Randomized Linear Program) Let \( J \) be a finite collection of random indices, i.e., \( J \equiv \{j_1, j_2, \ldots, j_K\} \) for an integer \( K \), where \( j_k \in [n] \) is a random variable for \( k = 1, 2, \ldots, K \). Then the problem

\[
P_J: \quad \min \{ c_j^T \bar{x} \mid A_J \bar{x} = b, \ \bar{x} \geq 0 \}
\]

is called a column-randomized linear program.

Clearly, \( P_J \) is equivalent to \( \min \{ c^T x \mid Ax = b, \ x \geq 0, \ x_j = 0 \ \forall j \notin J \} \). With this reformulation, any feasible solution of \( P_J \) can be represented as an element in \( \mathcal{F} \). We can thus define \( \Delta v(P_J) \) for the column-randomized LP \( P_J \). We sample random indices in \( J \) by a randomization scheme \( \rho \), which is a computational procedure that randomly selects indices from \([n]\), or equivalently, randomly generates columns from \( A \). Let \( \xi \) be the probability distribution over \([n]\) that corresponds to \( \rho \); that is, the \( j \)th component of \( \xi \), denoted by \( \xi_j \), is the probability that index \( j \) is selected by \( \rho \). Throughout this section, we assume \( \rho \) samples each index independently and identically according to \( \xi \). We will relax this assumption in Section 6.
We use $D_J$ to denote the dual of $P_J$, which is defined as

$$D_J : \max \{ p^T b \mid p^T A_J \leq c_J^T \}. \tag{5}$$

We will also require the notions of a basis, basic solutions and reduced costs in our theoretical results. A collection of indices $B \subseteq [n]$ of size $m$ is called a basis if the matrix $A_B$ is nonsingular, i.e., the collection of $m$ columns $\{ A_j \}_{j \in B}$ is linearly independent. A basic solution $x$ of the primal problem $P$ corresponding to the basis $B$ is the solution $x$ obtained by setting $x_B = A_B^{-1} b$, where $x_B$ is the subvector corresponding to the columns in $B$, and $x_N = 0$, where $x_N$ is the subvector corresponding to the columns in $[n] \setminus B$. A solution $x$ is called a basic feasible solution of $P$ if it is a basic solution for some basis $B$ and satisfies $x \geq 0$. For the dual problem, a basic solution $p$ corresponding to the basis $B$ is the solution $p$ defined by setting $p^T = c_B^T A_B^{-1}$; if it additionally satisfies $p^T A \leq c^T$, then it is also a basic feasible solution. Given a basis $B$, we define the reduced cost vector $\bar{c}$ for that basis as $\bar{c} \equiv c^T - c_B^T A_B^{-1} A$.

Finally, we use $\| \cdot \|$ to denote norms. For a vector $v \in \mathbb{R}^n$, we let $\| v \|_1 = \sum_{j=1}^n | v_j |$ be its $\ell_1$ norm, $\| v \|_2 = \sqrt{\sum_{j=1}^n v_j^2}$ be its Euclidean or $\ell_2$ norm, and $\| v \|_\infty = \max_{j=1, \ldots, n} | v_j |$ be its $\ell_\infty$ norm. For a matrix $A$, we let $\| A \|_{\max} = \max_{i,j} | A_{i,j} |$. Without loss of generality, we assume that the cost vector $c$ has unit Euclidean norm, i.e., $\| c \|_2 = 1$. This is not a restrictive assumption, because by normalizing the cost vector $c$ to have unit Euclidean norm, the objectives of the complete problem $P$ and the column-randomized problem $P_J$ are both scaled by $1/\| c \|_2$. Thus, the relative performance of problem $P_J$ to the complete problem $P$, which is the main focus of our paper, remains the same.

### 3.2. Main Theoretical Results

We propose the column randomization method in Algorithm 1. We first sample $K$ indices, $j_1, j_2, \ldots, j_K$, by a randomization scheme $\rho$ and let $J = \{ j_1, \ldots, j_K \}$. We then collect the corresponding columns of $A$ as matrix $A_J$ and the corresponding components of $c$ as vector $c_J$. After forming $A_J$ and $c_J$, we solve the LP (6) and return its optimal value $v(P_J)$ and optimal solution.

**Algorithm 1** The Column Randomization Method

1. Sample $K$ indices as $J \equiv \{ j_1, \ldots, j_K \}$ by a randomization scheme $\rho$.
2. Define $A_J = [A_{j_1}, \ldots, A_{j_K}]$ and $c_J = [c_{j_1}, \ldots, c_{j_K}]$.
3. Solve the column-randomized linear program, which only has $K$ columns:

$$P_J : \min \{ c_J^T \bar{x} \mid A_J \bar{x} = b, \ \bar{x} \geq 0 \}. \tag{6}$$

4. **return** optimal objective value $v(P_J)$ and an optimal solution $\bar{x}^*$.

Notice that an optimal solution $\bar{x}^*$ of problem $P_J$ can be immediately converted to a feasible solution for the complete problem $P$ by enlarging $\bar{x}^*$ to length $n$ and setting $\bar{x}^*_j = 0$ for $j \in [n] \setminus J$.

We now present two theorems that bound the duality gap $\Delta v(P_J) \equiv v(P_J) - v(P)$ of problem $P_J$; we defer our discussion of these two theorems to Section 3.3. Since several preliminary results are needed before we prove the theorems, we also relegate the proofs of the theorems to Section 4.

**Theorem 1.** Let $C$ be a positive constant and define the linear program $P_{\text{distr}}$ as

$$P_{\text{distr}} \equiv \min_{x \in \mathbb{R}^n} c^T x \quad \text{subject to} \quad Ax = b, \quad 0 \leq x \leq C \cdot \xi. \tag{7a}$$

Notice that an optimal solution $\bar{x}^*$ of problem $P_J$ can be immediately converted to a feasible solution for the complete problem $P$ by enlarging $\bar{x}^*$ to length $n$ and setting $\bar{x}^*_j = 0$ for $j \in [n] \setminus J$.
Let $P_J$ be the column-randomized LP solved by Algorithm 1, and $A_J$ be the corresponding constraint matrix. For any $\delta \in (0, 1)$, with probability at least $1 - \delta$ over the sample $J$, the following holds: if $P_J$ is feasible and $\text{rank}(A_J) = m$, then

$$\Delta v(P_J) \leq \Delta v(P_{\text{distr}}) + \frac{C(1 + m\gamma\|A\|_{\max})}{\sqrt{K}} \left(1 + \sqrt{2\log\frac{2}{\delta}}\right), \quad (8)$$

where $\gamma$ is an upper bound on $\|p\|_\infty$ for every basic solution $p$ of the dual problem $D$ and $\|A\|_{\max} = \max_{ij} |A_{ij}|$.

Theorem 1 shows that, with probability at least $1 - \delta$, the optimality gap $\Delta v(P_J)$ of the column-randomized LP $P_J$ is upper bounded by the sum of two terms. The first term is the optimality gap $\Delta v(P_{\text{distr}})$ of the problem $P_{\text{distr}}$, which we refer to as the distributional counterpart. The second term involves $\|A\|_{\max}$, the largest absolute value of elements in the constraint matrix; $\gamma$, the upper bound of the $\ell_\infty$ norm of any basic solution of the dual problem; $\delta$, the confidence parameter; and $K$, the number of sampled columns. Most importantly, the second term converges to zero with a rate $1/\sqrt{K}$. In Section 5, we will see how $\gamma$ and $\|A\|_{\max}$ can be further bounded for certain special cases.

We now present our second theorem, which relates the optimality gap to the reduced costs of the complete problem.

**Theorem 2.** Define $C_J$, $P_{\text{distr}}, P_J$ and $A_J$ as in Theorem 1. For any $\delta \in (0, 1)$, with probability at least $1 - \delta$ over the sample $J$, the following holds: if $P_J$ is feasible and $\text{rank}(A_J) = m$, then

$$\Delta v(P_J) \leq \Delta v(P_{\text{distr}}) + \frac{C}{\sqrt{K}} \cdot \chi \cdot \left(1 + \sqrt{2\log\frac{1}{\delta}}\right), \quad (9)$$

where $\chi$ is an upper bound on $\|c\|_2$ for every basic solution of the complete problem $P$.

Theorem 2 has a similar structure to Theorem 1. Compared to Theorem 1, the upper bound in Theorem 2 does not involve $\gamma$ and $\|A\|_{\max}$, but instead requires a bound on the norm of the reduced cost vector for all the bases of $P$.

### 3.3. Discussion on main theorems

Both Theorem 1 and 2 provide bounds on the optimality gap $\Delta v(P_J)$ of the following form:

$$\Delta v(P_J) \leq \Delta v(P_{\text{distr}}) + \frac{C_P \cdot C_\delta}{\sqrt{K}}, \quad (10)$$

where $C_P$ only depends on properties of the complete problem $P$ and $C_\delta$ only depends on the confidence parameter $\delta$. In Theorem 1, $C_P = 1 + m\gamma\|A\|_{\max}$ and $C_\delta = 1 + \sqrt{2\log(2/\delta)}$; in Theorem 2, $C_P = \chi$ and $C_\delta = 1 + \sqrt{2\log(1/\delta)}$. In the following discussion, we first focus on the general structure of the upper bounds given in (10), and subsequently we address the differences between Theorem 1 and Theorem 2.

**Role of Problem $P_{\text{distr}}$:** The distributional counterpart $P_{\text{distr}}$ is the restricted version of the complete problem $P$, which includes the additional constraint $x \leq C_\xi$. Thus, $\Delta v(P_{\text{distr}}) \geq 0$. If there exists an optimal solution $x^*$ of the complete problem $P$ such that $0 \leq x^* \leq C_\xi$, then $\Delta v(P_{\text{distr}}) = 0$. Notice that neither Theorem 1 nor 2 implies that the optimality gap $\Delta v(P_J)$ of the column-randomized linear program $P_J$ can be arbitrarily small with large $K$. Indeed, if $\xi$ is not “comprehensive” enough – that is, its support is small, and does not include the complete set of columns of any optimal basis for $P$ – then one would not expect the column-randomized program $P_J$ to perform closely to the complete problem $P$, even if $K$ is sufficiently large. In other words, problem $P_{\text{distr}}$ reflects the “coverage” ability of the distribution $\xi$, or equivalently, its randomization scheme $\rho$. 
Role of Constant $C$: Given a randomization scheme $\rho$ and its corresponding distribution $\xi$, as the constant $C$ increases, the optimality gap $\Delta v(P_{\text{distr}})$ of problem $P_{\text{distr}}$ decreases since its feasible set $F(P_{\text{distr}})$ is enlarged. On the other hand, the second term on the RHS of bound (10) increases since it is proportional to $C$. To interpret this phenomenon, we can view bound (10) as a type of bias-complexity/bias-variance tradeoff, which is common in statistical learning theory (Shalev-Shwartz and Ben-David 2014):

$$\Delta v(P_J) \leq \Delta v(P_{\text{distr}}) + \frac{C \cdot C_P \cdot C_\xi}{\sqrt{K}}.$$  \hspace{1cm} (11)

When the constant $C$ increases, the feasible set $F(P_{\text{distr}})$ gradually becomes a better approximation of the feasible set $F(P)$, as more feasible solutions in $F(P)$ are included in $F(P_{\text{distr}})$. The optimality gap $\Delta v(P_{\text{distr}})$, which can be viewed as the approximation error, is thus narrowed. On the other hand, as the set $F(P_{\text{distr}})$ expands, one needs more samples to ensure that the sampled feasible set $F(P_J)$ can approximate $F(P_{\text{distr}})$. In that sense, as we increase $C$, the second term of the right-hand side of (11) also increases.

Feasibility of $P_J$: We make several important remarks regarding the feasibility of $P_J$ and how feasibility is incorporated in our guarantee. First, note that in general, the sampled problem $P_J$ need not be feasible. As a simple example, consider the following complete problem:

$$P = P_1 \equiv \min \{1^T x \mid I x = 1, x \geq 0\},$$

where $I$ is the $n$-by-$n$ identity matrix and $m = n$. In this problem, the only way that the sampled problem $P_J$ can be feasible is if the collection $j_1, \ldots, j_K$ includes every index in $[n]$; if any column $j \in [n]$ is not part of the sample $J$, then the sampled problem $P_J$ is automatically infeasible. Thus, when $K < n$, $P_J$ is infeasible almost surely. When $K \geq n$, it is still possible that $j_1, \ldots, j_K$ does not include all indices in $[n]$, and thus $P_J$ is infeasible with positive probability.

For this reason, our guarantee on the optimality gap is stated as a conditional guarantee: with high probability over the sample $j_1, \ldots, j_K$, the optimality gap of $P_J$ obeys a particular bound if the column-randomized LP is feasible. We note that this is distinct from probabilistically conditioning on $j_1, \ldots, j_K$, i.e., our guarantee is not the same as

$$\Pr \left[ \Delta v(P_J) \leq \Delta v(P_{\text{distr}}) + \frac{C \cdot C_P \cdot C_\xi}{\sqrt{K}} \mid P_J \text{ is feasible} \right] \geq 1 - \delta,$$

because upon conditioning on the feasibility of $P_J$, the random variables $j_1, \ldots, j_K$ are in general no longer an i.i.d. sample. As an example of this, consider again problem $P_1$ above, with $K = n$ and a randomization scheme $\rho$ corresponding to the uniform distribution $\xi = (1/n, \ldots, 1/n)$ over $[n]$. By conditioning on the event that $P_J$ is feasible, the sample $J = \{j_1, \ldots, j_K\}$ must then be exactly equal to $[n]$, and we obtain that $\Pr[j_k = t, j_{k'} = t] = 0 \neq \Pr[j_k = t] \cdot \Pr[j_{k'} = t]$ for any $k, k' \in [K]$ with $k \neq k'$ and $t \in [n]$. In this example, the indices $j_1, \ldots, j_K$ are thus not independent.

With regard to the feasibility of column-randomized LPs, it appears to be difficult to guarantee feasibility in general. However, one can use similar techniques as in the proofs of our main results to characterize the near-feasibility of a column-randomized LP. Consider the following complete problem, and its sampled and distributional counterparts:

$$P_{\text{feas}} = \min \{\|Ax - b\|_1 \mid x \geq 0\},$$

$$P_{j\text{feas}} = \min \{\|A_j \hat{x} - b\|_1 \mid \hat{x} \geq 0\},$$

$$P_{\text{distr feas}} = \min \{\|Ax - b\|_1 \mid 0 \leq x \leq C \xi\}.$$
The objective function in each problem measures how close $Ax$ is to $b$ for a given nonnegative solution $x$, and the optimal value measures the minimum total infeasibility, as measured by the lowest attainable $\ell_1$ distance between $Ax$ and $b$. Note that an optimal value of zero for a given problem implies that the feasible region contains a solution $x$ that satisfies $Ax = b$. With a slight abuse of notation, let us use $v(P_{\text{feas}}^J)$, $v(P_{\text{feas}}^J)$ and $v(P_{\text{distr}}^J)$ to denote the optimal objective value of each problem. We then have the following result.

**Proposition 1.** Let $C$ be a nonnegative constant. For any $\delta \in (0,1)$, with probability at least $1 - \delta$ over the sample $J$,

$$v(P_{\text{feas}}^J) \leq v(P_{\text{distr}}^J) + \frac{C}{\sqrt{K}} \cdot m \cdot \|A\|_{\max} \cdot \left(1 + \sqrt{2 \log \frac{1}{\delta}}\right).$$

The proof of Proposition 1 (see Section EC.1.1 of the ecompanion) follows using a similar but simpler procedure than those used in the proofs of Theorems 1 and 2. The guarantee in Proposition 1 has a similar interpretation to Theorems 1 and 2: the magnitude of the total infeasibility of the columns $J$ is bounded with high probability by the minimum infeasibility of the distributional counterpart $P_{\text{distr}}^J$ plus a $O(1/\sqrt{K})$ term.

**Interpretation of $\gamma$ and $\chi$:** We first note that the technique of bounding the objective value of a linear program using the $\ell_\infty$ norm of basic feasible solutions has been applied previously in the literature (Ye 2011, Kitahara and Mizuno 2013). The presence of $\gamma$ and $\chi$ in Theorem 1 and 2, respectively, arises due to the use of sensitivity analysis results from linear programming with respect to the right-hand side vector $b$. As we will see in Section 4, we show that any optimal solution $x^{\ast\ast}$ of problem $P_{\text{distr}}$ has a sparse counterpart $x'$ in the space $S_J \equiv \{x \mid x_j = 0 \ \forall j \notin J\}$ such that it is in the vicinity of $x^{\ast\ast}$ in terms of Euclidean distance. However, $x'$ does not necessarily belong to the feasible set $F(P_J)$ of the column-randomized linear program $P_J$, since $F(P_J)$ is a subset of $S_J$. To relate the optimal objective value $v(P_J)$ of problem $P_J$ to $c^T x'$, which is close to $c^T x^{\ast\ast}$, we use sensitivity analysis arguments which involve either $\gamma$ or $\chi$.

**Comparison of Theorems 1 and 2:** While both Theorem 1 and 2 provide valid bounds for the optimality gap $\Delta v(P_J)$, Theorem 1 is in general easier to apply; indeed, in Section 5 we discuss two notable examples where $\gamma$ can be easily computed (specifically, LPs with totally unimodular constraint matrices $A$ and infinite horizon discounted Markov decision processes). For problems that are not standard form LPs, neither guarantee directly applies, but we can obtain specialized guarantees by carefully modifying a result (Proposition 2 in Section 4.2) that leads to Theorem 1 and designing bounds for the $\ell_\infty$ norm of feasible or optimal solutions of $D_J$ (as opposed to basic solutions of $D$). We will later showcase two examples of such guarantees, for covering LPs (Section 5.3) and packing LPs (Section 5.4).

With regard to Theorem 2, we expect for most problems that Theorem 2 will be difficult to apply, as it requires a universal bound for the norm of the reduced cost vector for every basis, feasible or not, of problem $P$. Nevertheless, Theorem 2 is interesting because it involves reduced costs, which are also of importance in column generation. For a basic feasible solution, the reduced cost of a non-basic variable $j$ can be thought of as the rate at which the objective changes as one increases $x_j$ to move from the current basic feasible solution to an adjacent/neighboring basic feasible solution in which $j$ is part of the basis. With this perspective of reduced costs, one can informally interpret the result in the following way: if $\chi$ is small, then the rate at which the objective changes between adjacent basic feasible solutions is small. In such a setting, it is reasonable to expect that there will be many basic feasible solutions that are close to being optimal and that solving the sampled problem $P_J$ should return a solution that performs well. On the other hand, if there exist non-optimal basic feasible solutions where the reduced cost vector has a very large magnitude (which would imply a large $\chi$), then this would suggest that the objective changes by a large amount between certain adjacent basic feasible solutions, and that there are certain “good”
columns that are more important than others for achieving a low objective value. In this setting, we would expect the sampled problem objective \( v(P_J) \) to only be close to \( v(P) \) if \( J \) includes the “good” columns, which would be unlikely to happen in general.

**Design of Randomization Scheme** \( \rho \): The quantity \( \xi_j \), which is the probability that the \( j \)th column is drawn by the randomization scheme \( \rho \), can be interpreted as the relative importance of \( x_j \) compared to other components of \( x \in \mathbb{R}^n \) in the complete problem \( P \); indeed, when the corresponding column is randomly chosen, \( x_j \) is allowed to be nonzero, and can thus be utilized to solve the optimization problem. For example, in a network flow optimization problem, \( x_j \) represents the amount of flow over edge \( j \); a nonzero \( \xi_j \) can thus be interpreted as the belief that edge \( j \) should be used for flow. As another example, consider the LP formulation of an MDP, where each component of \( x \) corresponds to a state-action pair \((s,a)\) (i.e., \( x_{(s,a)} \) is the expected discounted frequency of the system being in state \( s \) and action \( a \) being taken). In this setting, a nonzero \( \xi_{(s,a)} \) can be interpreted as the relative importance of \((s,a)\) to other state-action pairs.

One can design the randomization scheme based on prior knowledge of the problem. For example, one could use a heuristic solution to a network flow problem to design a randomization scheme \( \rho \) resulting in a distribution \( \xi \) that is biased towards this heuristic solution. Similarly, if one has access to a good heuristic policy for an MDP, one can design a distribution \( \xi \) that is biased towards state-action pairs \((s,a)\) that occur frequently for this policy. If such prior knowledge is not available, a uniform or nearly-uniform distribution over \([n]\) is adequate. We provide two concrete examples on how to design randomization schemes in our numerical experiments in Section 7. Finally, we note that the indices in \( J \) have been assumed to be i.i.d. It turns out that this assumption can actually be relaxed: in Section 6, we derive an upper bound on the optimality gap \( \Delta v(P_J) \) for the case when the indices are sampled non-independently.

**Minor Remarks on the Upper Bound:** We remark on two other interesting properties of the bound (10). First, the second term in (10) is independent of the distribution \( \xi \); no matter how \( \xi \) is designed, the optimality gap \( \Delta v(P_J) \) is guaranteed to converge with rate \( 1/\sqrt{K} \). Second, the dependence of the bound on the confidence parameter \( \delta \) is via \( \sqrt{2\log(2/\delta)} \) in Theorem 1 or \( \sqrt{2\log(1/\delta)} \) in Theorem 2. This implies that very small values of \( \delta \) will not significantly increase the upper bound on \( \Delta v(P_J) \).

**Computational Strengths and Weaknesses** We compare the column randomization method to the CG method from a computational viewpoint. An obvious characteristic of the CG method is that it is a serial algorithm: to introduce a new column, one needs the dual solution of the restricted problem that consists of columns generated in previous iterations. This sequential nature unfortunately prevents the CG method from being parallelized. In contrast, the column randomization method is amenable to parallelization. Given a collection of processors, each processor can be used to sample a column and compute the constraint and objective coefficients in parallel, until \( K \) columns in total are sampled across all processors. This can be especially advantageous in cases where the objective or constraint coefficients require significant effort compute, such as solving a dynamic program or integer program. For example, Bertsimas et al. (2019) considers a set partitioning model of a pickup and delivery problem arising in airlift operations, where each decision variable \( x_{v,S} \) corresponds to an aircraft \( v \) being assigned to a collection of shipments \( S \) and the cost coefficient \( c_{v,S} \) is the optimal value of a scheduling problem that determines the sequence of pickups and dropoffs of the shipments in \( S \).

An obvious disadvantage of the column randomization method is that it does not guarantee optimality. Even if there exists an optimal solution of the complete problem \( P \) that belongs to the feasible set \( F(P_{\text{dist}}) \) of problem \( P_{\text{dist}} \), the optimality gap still converges with rate \( 1/\sqrt{K} \), which implies that the “last-mile” shrinkage of the optimality gap requires an increasing number of additional sampled columns. If optimality is a concern, instead of solely using the column randomization method, one could use it as a warm-start for the CG method. Specifically, let \( J_{\text{ns}} = \{ j \mid x_j^* > 0 \} \), where \( x^* \) is the solution returned by Algorithm 1. Then, the set of variables \( (x_j)_{j \in J_{\text{ns}}} \) and the columns \( A_{J_{\text{ns}}} \) can be used as the initial solution for the CG method.
4. Proofs

In this section, we prove Theorem 1 and 2. We start with some preliminary results (Section 4.1) then prove the main theorems (Section 4.2).

4.1. Preliminary Results and Lemmas

Lemma 1 and 2 bound the distance between the sample mean and the expected value of a collection of i.i.d. vectors, in terms of $\ell_2$ norm and $\ell_1$ norm, respectively. Lemma 1 is Lemma 4 from Rahimi and Recht (2009), which utilizes McDiarmid's inequality to show that the scalar function $\|\bar{w} - \mathbb{E} [\bar{w}] \|_2$, where $\bar{w}$ is the mean of $K$ i.i.d. vectors $w_1, \ldots, w_K$, concentrates to zero with rate $O \left( \frac{1}{\sqrt{K}} \right)$.

**Lemma 1.** (Rahimi and Recht 2009) Let $w_1, w_2, \ldots, w_K$ be i.i.d. random vectors such that $\|w_k\|_2 \leq C$ for $k = 1, \ldots, K$. Let $\bar{w} = (1/K) \cdot \sum_{k=1}^{K} w_k$. Then for any $\delta \in (0, 1)$, we have, with probability at least $1 - \delta$,

$$\|\bar{w} - \mathbb{E} [\bar{w}]\|_2 \leq \frac{C}{\sqrt{K}} \cdot \left( 1 + \sqrt{\frac{2 \log \frac{1}{\delta}}{}} \right).$$

**Lemma 2.** Let $w_1, w_2, \ldots, w_K$ be i.i.d. random vectors of size $m$ such that $\|w_k\|_\infty \leq C$ for $k = 1, \ldots, K$. Let $\bar{w} = (1/K) \cdot \sum_{k=1}^{K} w_k$. Then for any $\delta \in (0, 1)$, we have, with probability at least $1 - \delta$,

$$\|\bar{w} - \mathbb{E} [\bar{w}]\|_1 \leq \frac{mC}{\sqrt{K}} \cdot \left( 1 + \sqrt{\frac{2 \log \frac{1}{\delta}}{}} \right).$$

**Proof:** Since $\|w_k\|_2 \leq \sqrt{m}\|w_k\|_\infty \leq \sqrt{m}C$, we apply Lemma 1 and obtain that with probability $1 - \delta$, $\|\bar{w} - \mathbb{E} [\bar{w}]\|_2 \leq \sqrt{m} \cdot C/\sqrt{K} \cdot \left( 1 + \sqrt{2 \log \frac{1}{\delta}} \right)$. Combining this with the fact that $\|\bar{w} - \mathbb{E} [\bar{w}]\|_1 \leq \sqrt{m} \cdot \|\bar{w} - \mathbb{E} [\bar{w}]\|_2$, we obtain the desired result. \(\square\)

Lemma 3 is a standard result of sensitivity analysis of linear programming; see Chapter 5 of Bertsimas and Tsitsiklis (1997). In fact, one can view the optimal objective value of problem $P$ as a convex function in $b$ and show that the dual solution $p$ is a subgradient at $b$.

**Lemma 3.** Let $z(b) = \min \{ c^T y \mid A_0 y = b, y \geq 0 \}$ and $z(b') = \min \{ c_0^T y \mid A_0 y = b', y \geq 0 \}$. Then $z(b) - z(b') \leq p^T (b - b')$, where $p$ is an optimal dual solution of the former problem.

4.2. Proofs of Main Theorems

We first establish a useful result.

**Proposition 2.** Let $C$ be a nonnegative constant and define the linear program $P_{\text{distr}}$ as in Theorem 1, i.e., $P_{\text{distr}} : \min \{ c^T x \mid Ax = b, 0 \leq x \leq C\xi \}$. Let $P_J$ be the column-randomized LP solved by Algorithm 1. For any $\delta \in (0, 1)$, with probability at least $1 - \delta$ over the sample $J$, the following holds: if $P_J$ is feasible, then

$$\Delta v(P_J) \leq \Delta v(P_{\text{distr}}) + \frac{C}{\sqrt{K}} \cdot \left( 1 + \|p\|_\infty \cdot m \cdot \|A\|_{\text{max}} \right) \cdot \left( 1 + \sqrt{\frac{2 \log \frac{2}{\delta}}{}} \right)$$

for any optimal solution $p$ of problem $D_J$ (the dual of problem $P_J$).

**Proof:** Let $j_1, \ldots, j_K$ be the set of indices sampled according to the distribution $\xi$ by the randomization scheme $\rho$. Let $x^{*0}$ be an optimal solution of the distributional counterpart problem $P_{\text{distr}}$. Consider the solution $x'$ that is defined as

$$x' = \frac{1}{K} \sum_{k=1}^{K} \frac{x^{*0}_{j_k}}{\xi_{j_k}} \cdot e_{j_k},$$
where we use $e_j$ to denote the $j$th standard basis vector for $\mathbb{R}^n$. In addition, define the vector $b'$ as

$$b' \equiv Ax'.$$

To prove our result, we proceed in three steps. In the first step, we show how we can probabilistically bound $\|x' - x^{*0}\|_2$. In the second step, we show how we can probabilistically bound $\|b' - b\|_1$. In the last step, we use the results of our first two steps, together with sensitivity results for linear programs, to derive the required bound. In what follows, we use $I_+$ to denote the support of $\xi$, that is, $I_+ = \{j \in [n] \mid \xi_j > 0\}$.

**Step 1: Bounding $\|x' - x^{*0}\|_2$.** To show that $x'$ will be close to $x^{*0}$, let us first define the vector $w_k$ as

$$w_k = \frac{x^{*0}}{\xi_{jk}} \cdot e_{jk}$$

for each $k \in [K]$. The vectors $w_1, \ldots, w_K$ constitute an i.i.d. collection of vectors, and possess three special properties. First, observe that $x'$ is just the sample mean of $w_1, \ldots, w_K$. Second, observe that the expected value of each $w_k$ can be calculated as

$$\mathbb{E}[w] = \sum_{j \in I_+} \frac{\xi_j}{\xi_{jk}} \cdot x_{jk} e_j = \sum_{j \in I_+} \xi_j e_j = \sum_{j \in [n]} x_{jk} e_j = x^{*0}$$

where we use $w$ to denote a random vector following the same distribution as each $w_k$. In the above, we note that the third step follows because the distributional counterpart $P_{\text{distr}}$ includes the constraint $x \leq C\xi$, so $j \notin I_+$ automatically implies that $x_{jk} = 0$.

Finally, observe that the $\ell_2$ norm of each $w_k$ can be bounded as

$$\|w_k\|_2 = \left| \frac{x_{jk}}{\xi_{jk}} \right| \cdot \|e_j\|_2 \leq C \cdot 1 = C,$$

where the inequality follows because $x^{*0}$ satisfies the constraint $0 \leq x \leq C\xi$. With these three properties in hand, and recognizing that $\|x' - x^{*0}\|_2 = \|(1/K) \sum_{k=1}^K w_k - \mathbb{E}[w]\|_2$, we can invoke Lemma 1 to assert that, with probability at least $1 - \delta/2$,

$$\|x' - x^{*0}\|_2 \leq \frac{C}{\sqrt{K}} \cdot \left( 1 + \sqrt{2\log \frac{2}{\delta}} \right). \quad (12)$$

**Step 2: Bounding $\|b' - b\|_1$.** To show that $b'$ will be close $b$, we proceed similarly to Step 1. In particular, we define $b_k$ for each $k \in [K]$ as

$$b_k \equiv Ax_k = \frac{x_{jk}}{\xi_{jk}} \cdot Ae_{jk} = \frac{x_{jk}}{\xi_{jk}} A\xi.$$

Observe that by definition of $b_k$, we have that the sample mean of $b_1, \ldots, b_K$ is equal to $b'$:

$$\frac{1}{K} \sum_{k=1}^K b_k = \frac{1}{K} \sum_{k=1}^K Aw_k = A \left( \frac{1}{K} \sum_{k=1}^K w_k \right) = Ax' \equiv b'. \quad (13)$$

In addition, the expected value of each $b_k$ can be calculated; letting $\tilde{b}$ denote a random variable with the same distribution as each $b_k$, we have

$$\mathbb{E}[\tilde{b}] = A\mathbb{E}[w_k] = Ax^{*0} = b.$$
Lastly, we can bound the $\ell_{\infty}$ norm of each vector $b_k$ as
\[
\|b_k\|_{\infty} = \left\| \frac{x_k^0}{\xi_k} A_{jk} \right\|_{\infty} = \frac{x_k^0}{\xi_k} \cdot \|A_{jk}\|_{\infty} \leq C \|A\|_{\max},
\]
where the inequality follows by the definition of $\|A\|_{\max}$ and the fact that $x^0$ satisfies $0 \leq x \leq C \xi$.
With these observations in hand, we now recognize that $\|b' - b\|_1 = \left\| (1/K) \sum_{k=1}^{K} b_k - E[b]\right\|_1$, i.e., $\|b' - b\|_1$ is just the $\ell_1$ norm of the deviation of a sample mean from its true expectation; we can therefore invoke Lemma 2 to assert that, with probability at least $1 - \delta/2$,
\[
\|b' - b\|_1 \leq \frac{m \cdot C \cdot \|A\|_{\max}}{\sqrt{K}} \cdot \left(1 + \sqrt{2 \log \frac{2}{\delta}}\right). \tag{14}
\]

**Step 3: Completing the proof.** With Steps 1 and 2 complete, we are now ready to bound the optimality gap. For any vector $b'' \in \mathbb{R}^m$, we define the linear program $P_f(b'')$ as
\[
P_f(b'') : \min \left\{ c^T x \mid Ax = b'' , \; x \geq 0, \; x_j = 0 \; \forall j \notin J \right\}. \tag{15}
\]
Then $v(P_f(b'')) \leq c^T x'$; this follows because $Ax' = b'$ and $x' \geq 0$, which means that $x'$ is a feasible solution to problem $P_f(b')$. In addition, since $c^T x^0 = v(P_{\text{distr}})$, we have
\[
v(P_f(b')) \leq c^T x' = c^T (x^0 + (x' - x^0)) = v(P_{\text{distr}}) + c^T (x' - x^0). \tag{16}
\]
If the column-randomized problem $P_f$ is feasible, then by letting $p$ be any optimal solution of the dual of $P_f$ and applying Lemma 3, we have
\[
v(P_f) = v(P_f(b)) \leq v(P_f(b')) + p^T (b - b') \tag{17}
\leq v(P_{\text{distr}}) + c^T (x' - x^0) + p^T (b - b') \tag{18}
\leq v(P_{\text{distr}}) + \|c\|_2 \cdot \|x' - x^0\| + \|p\|_{\infty} \cdot \|b' - b\|_1 \tag{19}
= v(P_{\text{distr}}) + \|x' - x^0\|_2 + \|p\|_{\infty} \cdot \|b' - b\|_1, \tag{20}
\]
where the first inequality comes from Lemma 3, the second inequality comes from (16), the third inequality comes from the Cauchy-Schwarz inequality and Hölder’s inequality, and the last equality comes from the assumption that $\|c\|_2 = 1$.

We now bound expression (20) by applying the inequalities (12) and (14), each of which hold with probability at least $1 - \delta/2$, and combining them using the union bound. We thus obtain that, with probability at least $1 - \delta$,
\[
v(P_f) \leq v(P_{\text{distr}}) + \frac{C}{\sqrt{K}} \left(1 + \|p\|_{\infty} \cdot m \cdot A_{\max}\right) \cdot \left(1 + \sqrt{2 \log \frac{2}{\delta}}\right). \tag{21}
\]
Subtracting $v(P)$ from both sides gives us the required inequality. \hfill \square

With Proposition 2, we can smoothly prove Theorem 1 as follows.

**Proof of Theorem 1:** By invoking Proposition 2, we obtain that with probability at least $1 - \delta$, if $P_f$ is feasible, then
\[
\Delta v(P_f) \leq \Delta v(P_{\text{distr}}) + \frac{C}{K} \cdot \left(1 + \|p\|_{\infty} \cdot m \cdot A_{\max}\right) \cdot \left(1 + \sqrt{2 \log \frac{2}{\delta}}\right),
\]
for any dual optimal solution $p$ of $D_f$. To prove the theorem, let us set $p$ to an optimal basic feasible solution of the problem $D_f$. Note that such a dual optimal solution is guaranteed to exist by the assumption that $\text{rank}(A_j) = m$. Since $p$ is a basic feasible solution of $D_f$, it is automatically a basic (but not necessarily feasible) solution of the complete dual problem $D$. By the definition of $\gamma$ in the theorem, we have that $\|p\|_{\infty} \leq \gamma$, and the theorem follows. \hfill \square
To prove Theorem 2, we prove a complementary result to Proposition 2.

**Proposition 3.** Let \( C, P_J \) and \( P_{\text{distr}} \) be defined as in the statement of Proposition 2. For any \( \delta \in (0,1) \), with probability at least \( 1 - \delta \) over the sample \( J \), the following holds: if \( P_J \) is feasible, then
\[
\Delta v(P_J) \leq \Delta v(P_{\text{distr}}) + \frac{C}{\sqrt{K}} \cdot \|c^T - p^T A\|_2 \cdot \left(1 + \sqrt{2 \log \frac{1}{\delta}}\right)
\]
for any optimal solution \( p \) of problem \( D_J \) (the dual of problem \( P_J \)).

**Proof:** We follow the proof of Proposition 2 until inequality (18) and continue as follows:
\[
v(P_J) = v(P_J(b)) \leq v(P_{\text{distr}}) + p^T(b - b') \\
\leq v(P_{\text{distr}}) + c^T(x' - x^0) + p^T(b - b') \\
= v(P_{\text{distr}}) + c^T(x' - x^0) + p^T A(x^0 - x') \\
= v(P_{\text{distr}}) + (c^T - p^T A) (x' - x^0) \\
\leq v(P_{\text{distr}}) + \|c^T - p^T A\|_2 \cdot \|x' - x^0\|_2,
\]
where the bound holds for any optimal solution \( p \) of the sampled dual problem \( D_J \). By invoking Lemma 1 with \( \delta \) to bound \( \|x' - x^0\|_2 \), and subtracting \( v(P) \) from both sides, we obtain the desired result. \( \square \)

Using Proposition 3, we now prove Theorem 2.

**Proof of Theorem 2:** We invoke Proposition 3 and set \( p \) to be an optimal basic feasible solution of the sampled dual problem \( D_J \); then \( p^T = c^T B^{-1} A_B^{-1} \) for some set of basic variables \( B \subset [n] \). In this case, we observe that the dual slack vector \( c^T - p^T A \) becomes \( c^T - c_B B^{-1} A_B^{-1} \), which is exactly the reduced cost vector \( \bar{c} \) associated with the basis \( B \) within the full problem \( P \). By using the hypothesis that any such reduced cost vector satisfies \( \|\bar{c}\|_2 \leq \chi \), we obtain the desired result. \( \square \)

## 5. Special Structures and Extensions

In this section, we demonstrate how the results of Sections 3 and 4 can be applied to LPs with specific problem structures, including LPs with totally unimodular constraints (Section 5.1), Markov decision processes (Section 5.2), covering problems (Section 5.3) and packing problems (Section 5.4). In Section 5.5, we consider the portfolio optimization problem, which is in general not an LP, but is amenable to the same type of analysis.

### 5.1. LPs with Totally Unimodular Constraints

Consider a linear program with a totally unimodular constraint matrix, i.e., every square submatrix of \( A \) has determinant 0, 1, or \(-1\). Such LPs appear in various applications, such as minimum cost network flow problems and assignment problems (Bertsekas 1998). In such problems, it is not uncommon to encounter the situation where the number of variables is much larger than the number of constraints. For example, in a minimum cost network flow problem, each constraint corresponds to a flow-balance constraint at a given node, while each variable corresponds to the flow over an edge; in a graph of \( n \) nodes, one will therefore have \( n \) constraints and as many as \( \binom{n}{2} \) decision variables. We can thus consider solving the problem using the column randomization method. We obtain the following guarantee on the objective value of the column randomization method when applied to linear programs with totally unimodular constraints.

**Proposition 4.** Assume the constraint matrix of \( A \) of the complete problem \( P \) is totally unimodular. Define \( C, P_{\text{distr}}, P_J \) and \( A_J \) as in Theorem 1. For any \( \delta \in (0,1) \), with probability at least \( 1 - \delta \) over the set \( J \), the following holds: if \( P_J \) is feasible and \( \text{rank}(A_J) = m \), then
\[
\Delta v(P_J) \leq \Delta v(P_{\text{distr}}) + \frac{C(1 + m^2 \|c\|_\infty)}{\sqrt{K}} \left(1 + \sqrt{2 \log \frac{2}{\delta}}\right).
\]
Proof: Any basic solution \( \mathbf{p} \) to the dual problem \( D \) can be written as \( \mathbf{p}^T = \mathbf{c}_B^T \mathbf{A}_B^{-1} \), where \( B \) is a basis. In addition, since \( \mathbf{A} \) is totally unimodular, any element of \( \mathbf{A}_B^{-1} \) is either 1, -1, or 0. Therefore, the \( i \)th component of \( \mathbf{p} \) satisfies \( p_i = \sum_{j=1}^{\max n} [\mathbf{A}_B^{-1}]_{ji} (\mathbf{c}_B)_j \leq m \cdot \|\mathbf{c}\|_\infty \) for all \( i \in [m] \). Thus, we set \( \gamma = m \|\mathbf{c}\|_\infty \). Along with the fact \( \|\mathbf{A}\|_{\max} = 1 \) for any totally unimodular matrix \( \mathbf{A} \), we finish the proof by invoking Theorem 1.

5.2. Markov Decision Processes

Consider a discounted infinite horizon MDP, with \( n_s \) states and \( n_a \) actions. The cost function \( c(s, a) \) represents the immediate cost of taking action \( a \) in state \( s \). The transition probability \( P_s(s', a) \) represents the probability of being in state \( s' \) after taking action \( a \) in state \( s \). Let \( \theta \in (0, 1) \) be the discount factor. One can solve the MDP by formulating a linear program (Manne 1960):

\[
\begin{align*}
\text{minimize} & \quad \mathbf{c}_s^T \mathbf{x}_1 + \ldots + \mathbf{c}_s^T \mathbf{x}_s + \ldots + \mathbf{c}_s^T \mathbf{x}_n,
\text{subject to} & \quad (\mathbf{E}_s - \theta \mathbf{P}_s) \mathbf{x}_1 + \ldots + (\mathbf{E}_s - \theta \mathbf{P}_s) \mathbf{x}_s + \ldots + (\mathbf{E}_s - \theta \mathbf{P}_s) \mathbf{x}_n = 1,
& \quad \mathbf{x}_1, \ldots, \mathbf{x}_s, \ldots, \mathbf{x}_n \geq 0,
\end{align*}
\]

where \( \mathbf{E}_s \) is an \( n_s \times n_s \) matrix such that the \( j \)th row is all ones and every other entry is zero. The vector \( \mathbf{c}_s \) is of size \( n_s \) such that its \( j \)th component is equal to \( c(s, a) \). The matrix \( \mathbf{P}_s \) is of size \( n_s \times n_s \) such that its \( (s', a) \)-th component represents \( P_s(s', a) \). Notice that matrix \( \mathbf{P}_s \) is a column stochastic matrix, i.e., \( \mathbf{1}^T \mathbf{P}_s = 1^T \) and \( \mathbf{P}_s \geq 0 \) for all \( s \in [n_s] \). The decision variable vector \( \mathbf{x}_s \) is of size \( n_a \), where the \( j \)th entry represents the expected discounted long-run frequency of the system being in state \( s \) and action \( a \) being taken. If one sorts the decision variables by actions (Ye 2005), then the linear program can be re-written as:

\[
\begin{align*}
\text{minimize} & \quad \tilde{\mathbf{c}}_s^T \tilde{\mathbf{x}}_1 + \ldots + \tilde{\mathbf{c}}_s^T \tilde{\mathbf{x}}_s + \ldots + \tilde{\mathbf{c}}_s^T \tilde{\mathbf{x}}_n,
\text{subject to} & \quad (\mathbf{I} - \theta \tilde{\mathbf{P}}_s) \tilde{\mathbf{x}}_1 + \ldots + (\mathbf{I} - \theta \tilde{\mathbf{P}}_s) \tilde{\mathbf{x}}_s + \ldots + (\mathbf{I} - \theta \tilde{\mathbf{P}}_s) \tilde{\mathbf{x}}_n = 1,
& \quad \tilde{\mathbf{x}}_1, \ldots, \tilde{\mathbf{x}}_s, \ldots, \tilde{\mathbf{x}}_n \geq 0,
\end{align*}
\]

where \( \tilde{\mathbf{c}}_s = [c(1, a); \ldots; c(s, a); \ldots; c(n_s, a)] \) for \( a \in [n_a] \) and \( \tilde{\mathbf{P}}_s \) is a \( n_s \times n_s \) matrix such that its \( (s', a) \)-th element is equal to \( P_s(s', a) \). Note that problem (24) is a standard form LP and has more columns than rows. We can therefore apply the column randomization method to solve problem (24), leading to the following proposition.

Theorem 3. Consider solving a discounted infinite horizon MDP with \( n_s \) states and \( n_a \) actions by the column randomization method. Define \( C, P_{\text{distr}}, P_J \) and \( \mathbf{A}_J \) as in Theorem 1. For any \( \delta \in (0, 1) \), with probability at least \( 1 - \delta \), the following holds: if \( P_J \) is feasible and rank(\( \mathbf{A}_J \)) = \( n_s \), then

\[
\Delta v(P_J) \leq \Delta v(P_{\text{distr}}) + \frac{C}{\sqrt{K}} \cdot \left( 1 + \frac{n_s \|\mathbf{c}\|_{\infty}}{1 - \theta} \right) \cdot \left( 1 + 2 \log_2 \frac{2}{\delta} \right).
\]

Proof: Similarly to Proposition 4, we prove Theorem 3 by bounding \( \|\mathbf{A}\|_{\max} \) and \( \gamma \). Obviously, \( \|\mathbf{A}\|_{\max} \leq 1 \). Again, any basic solution \( \mathbf{p} \) of the dual has the form \( \mathbf{p}^T = \mathbf{c}_B^T \mathbf{A}_B^{-1} \), where \( B \) is a basis of the linear program (24). Note that \( \mathbf{A}_B \) has the form \( \mathbf{A}_B = \mathbf{I} - \theta \mathbf{P} \), where \( \mathbf{P} \) is an \( n_s \times n_s \) matrix such that each of its columns is selected from the columns of \( [\tilde{\mathbf{P}}_1, \ldots, \tilde{\mathbf{P}}_{n_a}] \) (see Ye 2005). In addition, a standard property of \( \mathbf{A}_B^{-1} \) is that it can be written as the following infinite series:

\[
\mathbf{A}_B^{-1} = \mathbf{I} + \theta \mathbf{P} + \theta^2 \mathbf{P}^2 + \cdots = \mathbf{I} + \sum_{n=1}^{\infty} \theta^n \cdot \mathbf{P}^n.
\]
Thus, we can bound \( \|p\|_\infty \) as \( \|p^T\|_\infty \leq \|c_B^T\|_\infty + \sum_{n=1}^{\infty} \theta^n \cdot \|c_B^T P^n\|_\infty \). Note that for any \( n \in \mathbb{N} \) and vector \( v \in \mathbb{R}^{n_s} \), we have

\[
\|v^T P^n\|_\infty = \max_{s \in [n_s]} \left| \sum_{s' \in [n_s]} v_{s'} P^n_{(s',s)} \right|
\]

where \( P^n_{(s',s)} \) is the \((s',s)\)th entry of \( P^n \). Therefore, we obtain that

\[
\|p^T\|_\infty = \|c_B^T A^{-1}\|_\infty 
\leq \|c_B\|_\infty + \sum_{n=1}^{\infty} \theta^n \cdot \|c_B^T P^n\|_\infty 
\leq \|c_B\|_\infty / (1 - \theta) 
\leq \|c\|_\infty / (1 - \theta).
\]

Since \( p \) was an arbitrary basic solution of the complete dual of problem (24), we can therefore set \( \gamma = \|c\|_\infty / (1 - \theta) \). The rest of the proof follows by an application of Theorem 1.

\[\Box\]

### 5.3. Covering Problems

A covering linear program can be formulated as

\[
P^{\text{covering}} : \begin{align*}
\text{minimize} & \quad c^T x \\
\text{subject to} & \quad Ax \geq b, \\
& \quad x \geq 0,
\end{align*}
\]

where \( A, b \) and \( c \) are all nonnegative, and we additionally assume that for every \( i \in [m] \), there exists a \( j \in [n] \) such that \( A_{i,j} > 0 \). This type of problem arises in numerous applications such as facility location (Owen and Daskin 1998). The column-randomized counterpart of this problem and its dual can be written as

\[
P^{\text{covering}}_J : \min \{ c^T \tilde{x} \mid A_J \tilde{x} \geq b, \tilde{x} \geq 0 \},
\]

\[
D^{\text{covering}}_J : \max \{ p^T b \mid p^T A_J \leq c^T, p \geq 0 \}.
\]

Although \( P^{\text{covering}} \) is not a standard form LP, it is straightforward to extend Proposition 2 to this problem, leading to the following result. We omit the proof for brevity.

**Proposition 5.** Let \( C \) be a nonnegative constant and define \( P^{\text{covering}}_J \) as

\[
P^{\text{covering}} \equiv \min \{ c^T x \mid Ax \geq b, 0 \leq x \leq C \xi \}.
\]

For any \( \delta \in (0, 1) \), with probability at least \( 1 - \delta \) over the sample \( J \), the following holds: if \( P^{\text{covering}}_J \) is feasible, then

\[
\Delta v(P^{\text{covering}}_J) \leq \Delta v(P^{\text{covering}}) + \frac{C}{\sqrt{R}} : (1 + \|p\|_\infty \cdot m \cdot \|A\|_\infty) \cdot \left( 1 + \sqrt{2 \log \frac{2}{\delta}} \right)
\]

for any optimal solution \( p \) of \( D^{\text{covering}}_J \).
To now use this result, we need to be able to bound $\|p\|_\infty$ for any solution $p$ of any dual $D_j^{\text{covering}}$ of the column-randomized problem. Let us define the quantity $U^{\text{covering}}$ as

$$U^{\text{covering}} = \max_{i,j} \left\{ \frac{c_j}{A_{i,j}} \mid A_{i,j} > 0 \right\}.$$  

We then have the following result.

**Theorem 4.** Let $C$ and $P_{\text{distr}}^{\text{covering}}$ be defined as in Proposition 5. For any $\delta \in (0,1)$, with probability at least $1 - \delta$ over the sample $J$, the following holds: if $P_{\text{covering}}^J$ is feasible, then

$$\Delta v(P_{\text{covering}}^J) \leq \Delta v(P_{\text{distr}}^{\text{covering}}) + \frac{C}{\sqrt{K}} \cdot (1 + U^{\text{covering}} \cdot m \cdot \|A\|_{\text{max}}) \cdot \left(1 + \sqrt{2 \log \frac{2}{\delta}}\right).$$

The proof (see Section EC.1.2) follows by showing that $U^{\text{covering}}$ is a bound on $\|p\|_\infty$ for any feasible solution $p$ of the dual $D_j^{\text{covering}}$ for any $J$ such that $P_{\text{covering}}^J$ is feasible. (Note that the bound applies to any feasible solution of $D_j^{\text{covering}}$, not just the optimal solutions of $D_j^{\text{covering}}$.)

### 5.4. Packing Problems

A packing linear program is defined as

$$P_{\text{packing}} : \text{maximize } \quad c^T x$$

subject to $\quad Ax \leq b,$

$$x \geq 0,$$

where we assume that $c \geq 0$, $b > 0$, and that $b$ is such that for every column $j \in [n]$, there exists an $i \in [m]$ such that $A_{i,j} > 0$. Packing problems have numerous applications, such as network revenue management (Talluri and van Ryzin 2006).

The column-randomized counterpart of this problem and its dual can be written as

$$P_{\text{packing}}^J : \text{max } \{c_j^T \bar{x} \mid A_j \bar{x} \leq b, \bar{x} \geq 0\},$$

$$D_{\text{packing}}^J : \text{min } \{p^T b \mid p^T A_j \geq c_j, p \geq 0\}.$$  

As with covering problems, the packing problem $P_{\text{packing}}$ is not a standard form LP, but we can derive a counterpart of Proposition 2 for $P_{\text{distr}}^{\text{packing}}$. Note that in this guarantee, for a problem $P'$ with the same feasible region as $P_{\text{packing}}$, the optimality gap $\Delta v(P')$ is defined as $\Delta v(P') = v(P_{\text{packing}}) - v(P')$, since the complete problem $P_{\text{packing}}$ is a maximization problem. As with Proposition 5, the proof is straightforward, and thus omitted.

**Proposition 6.** Let $C$ be a nonnegative constant and define $P_{\text{distr}}^{\text{covering}}$ as

$$P_{\text{distr}}^{\text{covering}} \equiv \max\{c^T x \mid Ax \leq b, 0 \leq x \leq C\xi\}.$$  

For any $\delta \in (0,1)$, with probability at least $1 - \delta$ over the sample $J$, the following holds: if $P_{\text{packing}}^J$ is feasible, then

$$\Delta v(P_{\text{packing}}^J) \leq \Delta v(P_{\text{distr}}^{\text{covering}}) + \frac{C}{\sqrt{K}} \cdot (1 + \|p\|_\infty \cdot m \cdot \|A\|_{\text{max}}) \cdot \left(1 + \sqrt{2 \log \frac{2}{\delta}}\right)$$

for any optimal solution $p$ of $D_{\text{packing}}^J$. 
To obtain a more specific guarantee, define for each $i$ the following quantities:

$$r_i = \max \left\{ \frac{c_j}{A_{i,j}} \mid A_{i,j} > 0 \right\},$$

$$j_i^* = \arg \max_j \left\{ \frac{c_j}{A_{i,j}} \mid A_{i,j} > 0 \right\}.$$ 

These two quantities can be understood by interpreting each $i$ as a resource constraint, and $b_i$ as the available amount of resource $i$. The column $j_i^*$ is the column that has the best rate of objective value garnered per unit of resource $i$ consumed, and the quantity $r_i$ is that corresponding rate.

Define now $W$ as

$$W = \sum_{i' = 1}^m r_{i'} b_{i'},$$

and $U_{\text{packing}}$ as the maximum over $i$ of $W/b_i$, i.e.,

$$U_{\text{packing}} = \max_{i \in [m]} \frac{W}{b_i} = \frac{\sum_{i' = 1}^m r_{i'} b_{i'}}{\min_{i \in [m]} b_i}.$$ 

We then have the following specific guarantee for packing LPs.

**Theorem 5.** Let $C$ and $P_{\text{packing}}$ be defined as in Proposition 6. For any $\delta \in (0, 1)$, with probability at least $1 - \delta$ over the sample $J$, the following holds: if $P_{\text{packing}}^J$ is feasible, then

$$\Delta v(P_{\text{packing}}^J) \leq \Delta v(P_{\text{distr}}^J) + \frac{C}{\sqrt{K}} \cdot (1 + U_{\text{packing}} \cdot m \cdot \|A\|_{\text{max}}) \cdot \left(1 + \sqrt{2 \log 2 / \delta}\right).$$

The proof of this result (see Section EC.1.3) follows by establishing that $W$ is an upper bound on $v(P_{\text{packing}}^J)$, and then bounding each $|p_i|$ by solving a modified version of $D_{\text{packing}}^J$ which is defined using $W$. We remark that our choice of $W$ is special only in that it bounds $v(P_{\text{packing}}^J)$. For particular packing problems, if one has access to a problem-specific bound $W'$ on $v(P_{\text{packing}}^J)$, one could define $U_{\text{packing}}$ with $W'$ instead to obtain a more refined bound.

### 5.5. Portfolio Optimization

In this last section, we deviate slightly from our previous examples by showing how our approach can be applied to problems that are not linear programs. The specific problem that we consider is the portfolio optimization problem, which is defined as

$$P_{\text{portfolio}}: \begin{array}{ll}
\text{minimize} & f(r_1, \ldots, r_m) \\
\text{such that} & \sum_{j=1}^n \alpha_{ij} x_j = r_i, \quad \forall i \in [m] \\
& \sum_{j=1}^n x_j = 1, \\
& x \geq 0,
\end{array}$$

where both $x$ and $r$ are decision variables. Problem (28) can be interpreted as follows: a decision maker seeks an optimal portfolio, which is a distribution over instruments, according to some objectives. The decision variable $x_j$ represents the fraction of allocation committed to instrument $j$, the constraint parameter $\alpha_{ij}$ represents the return of instrument $j$ in scenario $i$, and $r_i$ is the total return in $i$th scenario. The objective function $f$ is a function measuring the risk of the returns.
Unlike the optimization problems we discussed so far, we assume that $f$ is any Lipschitz continuous function with Lipschitz constant $L$, and is not necessarily a linear function of $r$.

Although problem $P_{\text{portfolio}}$ is not in general a linear program, we can still apply the column randomization method to solve the problem. We describe the procedure in Algorithm 2. Notice that, unlike Algorithm 1 which samples columns associated with all variables, here we only sample columns associated with $x$.

**Algorithm 2** The Column Randomization Method - Portfolio Optimization

1: Sample $K$ i.i.d. indices in $[n]$ as $J \equiv \{J_1, \ldots, J_K\}$ according to a randomization scheme $\rho$.
2: Solve the sampled optimization problem:

$$P_{\text{portfolio}}^J: \min \left\{ f(r) \left| \sum_{j \in J} \alpha_{ij} \tilde{x}_j = r_i, \forall i \in [m], \sum_{j \in J} \tilde{x}_j = 1, \tilde{x} \geq 0 \right. \right\}$$

3: return optimal solution $(\tilde{x}^*, r^*)$ and optimal objective value $f(r^*)$.

**Proposition 7.** Assume vectors $\alpha_j = (\alpha_{ij})_{i \in [m]}$ in problem $P_{\text{portfolio}}$ satisfying $\|\alpha_j\|_2 \leq H$ for all $j \in [n]$. Let $C \geq 1$ be an arbitrary constant and define the optimization problem

$$P_{\text{distr}}: \min_{x, r} \left\{ f(r) \left| \sum_{j \in [n]} \alpha_j x_j = r, 1^T x = 1, 0 \leq x \leq C \right. \right\}.$$

Denote $F$, $F_{\text{distr}}$, and $F_J$ as optimal objective values of problems $P_{\text{portfolio}}$, $P_{\text{distr}}$, and $P_{\text{portfolio}}^J$, respectively. Define $\Delta F_J = F_J - F$ and $\Delta F_{\text{distr}} = F_{\text{distr}} - F$. For any $\delta \in (0, 1)$, with probability at least $1 - \delta$, the following statement holds:

$$\Delta F_J \leq \Delta F_{\text{distr}} + \frac{C L H}{\sqrt{K}} \left(1 + 3 \sqrt{\frac{1}{2} \log \frac{4}{\delta}}\right).$$

For brevity, the proof is relegated to the companion (see Section EC.1.4). While the proof is similar to that of Proposition 2 in the construction of a random solution that is close to the solution of the distributional counterpart problem $P_{\text{distr}}$, the main difference is that it relies on Lipschitz continuity, rather than LP duality.

It is worthwhile to point out several aspects about this result and the portfolio optimization problem. First, the portfolio optimization problem (28) is not required to be a convex optimization problem; the objective function $f$ can be non-convex, so long as it is Lipschitz continuous. Second, this result is related to a more specific result from our prior work (Chen and Mišić 2019). In that paper, we consider the problem of estimating the decision forest choice model, which is a probability distribution over a collection of decision trees, and show that by solving an optimization problem over a random sample of trees, one can obtain a gap on the $\ell_1$ training error of the model that decays with rate $1/\sqrt{K}$ (Theorem 5 of Chen and Mišić 2019). Proposition 7 is a generalization of that result to more general optimization problems outside of choice model estimation, and allows for objective functions more general than those based on $\ell_1$ distance.

### 6. Statistically-Dependent Columns

So far we have assumed that each column in the column-randomized linear program is sampled independently. In this section, we show how this assumption can be relaxed. We state our main performance guarantee in Section 6.1. In Section 6.2, we consider a specific non-i.i.d. column sampling scheme – *groupwise sampling* – which has natural applications in problems such as Markov decision processes, and apply our guarantee from Section 6.1 to this sampling scheme.
6.1. Performance Guarantees via Dependency Graph and Forest Complexity

We begin by assuming that the randomization scheme $\rho$ is such that $j_1, \ldots, j_K$ still follow the distribution $\xi$, i.e., $\Pr[j_k = t] = \xi_t$ for $k \in [K]$ and $t \in [n]$, but they are no longer independent. Thus, the indices $j_1, \ldots, j_K$ are no longer an i.i.d. sample from $\xi$, and we require a different set of tools to analyze Algorithm 1 and $\Delta v(P_J)$ in this setting.

To analyze the column randomization method, we will make use of a specific concentration inequality from Liu et al. (2019), which requires specifying the dependence structure of a collection of random variables through a specific type of graph. We thus begin by briefly defining the relevant graph-theoretic concepts.

Given an undirected graph $G$, we use $V(G)$ to denote the vertices of $G$, and $E(G)$ to denote the edges of $G$. Given two vertices $u, v \in V(G)$, the edge between $u$ and $v$ is denoted by $\langle u, v \rangle$. We say that $u$ and $v$ are adjacent if $\langle u, v \rangle \in E(G)$. We say that $u$ and $v$ are non-adjacent if they are not adjacent. For two sets of nodes $U, V \subseteq V(G)$, we say that $U$ and $V$ are non-adjacent if $u$ and $v$ are non-adjacent for every $u \in U$ and $v \in V$.

Lastly, a graph $G$ is a forest if it does not contain any cycles, and is a tree if it does not contain any cycles and consists of a single connected component.

With these definitions, we now define the dependency graph, which is a representation of the dependency structure within a collection of random variables.

**Definition 2.** (Dependency graph) An undirected graph $G$ is called a dependency graph of a set of random variables $X_1, X_2, \ldots, X_K$ if it satisfies the following two properties:

1. $V(G) = [K]$.
2. For every $I, J \subseteq [K]$, $I \cap J = \emptyset$ such that $I$ and $J$ are non-adjacent, $\{X_i\}_{i \in I}$ and $\{X_j\}_{j \in J}$ are independent.

We now introduce the concept of a forest approximation from Liu et al. (2019).

**Definition 3.** (Forest approximation, Liu et al. (2019)) Given a graph $G$, a forest $F$, and a mapping $\phi : V(G) \rightarrow V(F)$, we say that $(\phi, F)$ is a forest approximation of $G$ if, for any $u, v \in V(G)$ such that $\langle u, v \rangle \in E(G)$, either $\phi(u) = \phi(v)$ or $\langle \phi(u), \phi(v) \rangle \in E(F)$.

In words, a forest approximation is a mapping of a general graph $G$ to a smaller forest $F$ that is obtained by merging nodes in $G$. For a given node $v \in V(F)$, the set $\phi^{-1}(v)$ corresponds to the set of nodes in $V(G)$ that were merged to obtain the node $v$. Using the notion of a forest approximation, we can now define the forest complexity of a graph $G$.

**Definition 4.** (Forest complexity, Liu et al. (2019)) Let $\Phi(G)$ denote the set of all forest approximations of $G$. Given a forest approximation $(\phi, F)$, define $\lambda(\phi, F)$ as

$$
\lambda(\phi, F) = \sum_{(u, v) \in E(F)} \left( |\phi^{-1}(u)| + |\phi^{-1}(v)| \right)^2 + \sum_{i=1}^k \min_{u \in V(T_i)} |\phi^{-1}(u)|^2
$$

where $T_1, \ldots, T_k$ is the collection of trees that comprise $F$. We call $\Lambda(G) = \min_{(\phi, F) \in \Phi(G)} \lambda(\phi, F)$ the forest complexity of $G$.

The forest complexity $\Lambda(G)$ quantifies how much the graph $G$ looks like a forest. Notice that $\Lambda(G) \geq |V(G)|$ for any graph $G$. In practice, we only need an upper bound on $\Lambda(G)$, rather than its exact value; we refer readers to Liu et al. (2019) for several examples on how $\Lambda(G)$ can be bounded.

Given a dependency graph $G$ for the random indices in the set $J$, we now bound the optimality gap of the column-randomized linear program.

**Theorem 6.** Let $C$ be a nonnegative constant, define $P_{\text{distr}}$ as in Theorem 1 and assume the random indices in $J$ follow the dependency graph $G$ with forest complexity $\Lambda(G)$. For any $\delta \in (0, 1)$, with probability at least $1 - \delta$ over the sample $J$, the following holds: if $P_J$ is feasible and $\text{rank}(A_J) = m$, then

$$
\Delta v(P_J) \leq \Delta v(P_{\text{distr}}) + C \cdot (1 + m \gamma \|A\|_{\text{max}}) \cdot \left( \sqrt{\frac{K + 2|E(G)|}{K^2}} + \sqrt{\frac{2\Lambda(G) \log(2/\delta)}{K^2}} \right),
$$

(32)
where \( \gamma \) and \( \|A\|_{\max} \) are defined as in Theorem 1.

Under the same conditions, with probability at least \( 1 - \delta \) over the sample \( J \), the following holds: if \( P_J \) is feasible and \( \text{rank}(A_J) = m \), then

\[
\Delta v(P_J) \leq \Delta v(P_{\text{distr}}) + C \cdot \chi \cdot \left( \sqrt{\frac{K + 2|E(G)|}{K^2}} + \sqrt{\frac{2\Lambda(G)\log(1/\delta)}{K^2}} \right),
\]

(33)

where \( \chi \) is defined as in Theorem 2.

The proof (see Section EC.1.5) follows by utilizing the McDiarmid inequality for dependent random variables from Liu et al. (2019). We note that Theorem 6 is a generalization of Theorems 1 and 2. If \( j_1, j_2, \ldots, j_K \) are independent, then the dependency graph \( G \) has no edges, and thus \( |E(G)| = 0 \) and \( \Lambda(G) = K \). Therefore, when each column is generated independently, the upper bounds in Theorem 6 are equivalent to the bounds in Theorem 1 and 2.

6.2. Groupwise Column Sampling

In many linear programs, we can naturally rearrange and group related columns together. For example, in the LP formulation of an MDP, one can collect columns associated with state \( s \). In many linear programs, we can naturally rearrange and group related columns together. For example, in the LP formulation of an MDP, one can collect columns associated with state \( s \). The reason for this is that we may sample the columns in such a way that we do not sample any columns corresponding to a particular state \( \tilde{s} \); in such a scenario, the sampled problem \( P_J \) will automatically be infeasible.

In the presence of a natural group structure of the columns, rather than sampling columns in total across all \( n \) columns, one could consider sampling \( n_g \) columns from each group. In the MDP example, this would correspond to sampling \( n_g \) columns (which correspond to state-action pairs) for each state \( s \). The resulting column-randomized linear program \( P_J \) corresponds to an MDP where there is a random set of \( n_g \) actions out of the complete set of \( n_a \) actions available in each state \( s \). Most importantly, \( P_J \) is guaranteed to be feasible.

It turns out that our results for dependent columns can be used to study column-randomized LPs where columns are sampled by groups. We refer to such a mechanism as a groupwise randomization scheme and define it formally below.

Definition 5. (Groupwise Randomization Scheme) Assume the set of indices \( [n] \) can be organized into \( n_G \) groups, i.e., \( [n] \) is the disjoint union of sets \( G_g \) for \( g = 1, 2, \ldots, n_G \). Consider a randomization scheme \( \rho \) such that (i) it samples indices in \( n_g \) rounds of sampling; (ii) in each round, it samples \( n_G \) indices as follows: for \( i = 1, \ldots, n_G \), it first uniformly at random chooses an index \( g_i \) from \( [n_G] \setminus \{g_j \mid j \in [i - 1]\} \) then samples an index from group \( G_{g_i} \) according to a distribution \( \xi^{g_i} \). We refer to such a randomization scheme \( \rho \) as a groupwise randomization scheme.

Note that the randomization scheme \( \rho \) samples \( K = n_r n_G \) indices in total, and samples \( n_r \) columns in each group. By design, each random index \( j \) follows the distribution \( \xi \), whose probabilities are given by

\[
\xi_t \equiv \Pr [j = t] = \frac{1}{n_G} \sum_{g \in [n_G]} \mathbb{1} \{ t \in G_g \} \cdot \xi^{g}_t = \frac{1}{n_G} \cdot \xi^{g}_{\tilde{t}},
\]

where \( G(t) \) is the group to which column \( t \in [n] \) belongs to.

By using our general result for dependent columns (Theorem 6), we obtain a specific guarantee for column-randomized LPs obtained by groupwise randomization schemes.

Theorem 7. Let \( J \) be a sample of \( K = n_r n_G \) indices sampled according to a groupwise randomization scheme \( \rho \). Let \( C \) be a nonnegative constant and define \( P_{\text{distr}} \) as in Theorem 1. For any
\[ \delta \in (0,1), \text{ with probability at least } 1 - \delta, \text{ the following holds: if } P_J \text{ is feasible and } \text{rank}(A_J) = m, \text{ then} \]

\[ \Delta v(P_J) \leq \Delta v(P_{distr}) + \frac{C(1 + m\gamma \|A\|_{\text{max}})}{\sqrt{n_r}} \left(1 + \sqrt{2\log \frac{2}{\delta}}\right), \]

where \( \gamma \) and \( \|A\|_{\text{max}} \) are defined as in Theorem 1. Under the same assumption, with probability at least \( 1 - \delta \), the following holds: if \( P_J \) is feasible and \( \text{rank}(A_J) = m \), then

\[ \Delta v(P_J) \leq \Delta v(P_{distr}) + \frac{C \cdot \chi}{\sqrt{n_r}} \left(1 + \sqrt{2\log \frac{1}{\delta}}\right), \]

where \( \chi \) is defined as in Theorem 2.

Proof: The dependency graph \( G \) of \( K = n_r n_G \) random indices that are sampled by \( \rho \) consists of \( n_r \) cliques of size \( n_G \); Figure 1 provides an example of the dependency graph for \( n_r = 3 \) and \( n_G = 4 \). Therefore, \( |E(G)| = n_r n_G (n_G - 1)/2 \) and \( \Lambda(G) \leq \lambda(\phi, F) = n_r n_G^2 \) for a forest approximation \((\phi, F)\) that maps each clique in \( G \) as a node in \( F \). By upper bounding \( \Lambda(G) \) by \( n_r n_G^2 \) in Theorem 6, and using the fact that \( K = n_r n_G \), we complete the proof. \( \square \)

Theorem 7 can be interpreted as a guarantee on the optimality gap as a function of the number of columns sampled per group: for a groupwise randomization scheme, the gap decreases at a rate of \( 1/\sqrt{n_r} \), where \( n_r \) is the number of columns sampled per group. Compared to Theorem 1 and 2, the rate of convergence in Theorem 7 in terms of the total number of columns sampled, which is \( K = n_r n_G \), is slower; Theorem 1 and 2 both have a rate of \( 1/\sqrt{K} \), while Theorem 7 has a rate of \( 1/\sqrt{n_r} \equiv \sqrt{n_G/K} \).

7. Numerical Experiments

In this section, we apply the column randomization method to two applications of large-scale linear programs that are commonly solved by CG. We demonstrate the effectiveness of the column randomization method by comparing its performance to that of the CG method. We also use these two applications to show that how one can design a randomization scheme based on the problem structure. All linear and mixed-integer programs in this section are formulated in the Julia programming language (Bezanson et al. 2017) with the JuMP package (Dunning et al. 2017) and solved by Gurobi (Optimization 2020).

7.1. Cutting-Stock Problem

The first application we consider is the classic cutting-stock problem. We follow the notation in Bertsimas and Tsitsiklis (1997) and for completeness, briefly review the problem. A paper company needs to satisfy a demand of \( b_i \) rolls of paper of width \( w_i \) for each \( i \in [m] \). The company has supply of large rolls of paper of width \( W \) such that \( W \geq w_i \) for \( i \in [m] \). To meet the demand, the company slices the large rolls into smaller rolls according to patterns. A pattern is a vector of nonnegative integers \((a_1, a_2, \ldots, a_m)\) that satisfies \( \sum_{i=1}^{m} a_i w_i \leq W \), where each \( a_i \) is the number of rolls of width \( w_i \) to cut from the large roll. Let \( n \) be the number of all feasible patterns and let \((a_{ij}, a_{2j}, \ldots, a_{mj})\) be the \( j \)th pattern for \( j \in [n] \). Let \( A \) be the matrix such that \( A_{ij} = a_{ij} \) for \( i \in [m] \) and \( j \in [n] \). The
cutting-stock problem is to minimize the number of large rolls of papers used while satisfying the demand, which can be formulated as the following covering LP:

\[
P_{\text{CS}}: \begin{align*}
\text{minimize} \quad & \sum_{j=1}^{n} x_j \\
\text{subject to} \quad & \sum_{j=1}^{n} a_{ij} x_j \geq b_i, \quad \forall i \in [m], \\
& x_j \geq 0, \quad \forall j \in [n].
\end{align*}
\] (34a)

Explicitly representing the constraint matrix \( A \) in full is usually impossible: the number of feasible patterns \( n \) can be huge even if the number of demanded widths \( m \) is small. A typical solution method is column generation, in which each iteration proceeds as follows. Given a set of patterns \( J = \{j_1, j_2, \ldots, j_K\} \), solve the restricted problem \( P_{\text{CS}}(J) : \min_{x \in \mathbb{R}^n} \left\{ \sum_{k=1}^{K} \bar{x}_k \mid \sum_{k=1}^{K} A_{j_k} \bar{x}_k \geq b, \bar{x} \geq 0 \right\} \) and let \( p \) be the optimal dual solution. Then find a new pattern \( j_{K+1} \) such that the corresponding new column has the most negative reduced cost \( 1 - p^T A_{j_{K+1}} \). If the reduced cost is nonnegative, the current solution is optimal and the procedure terminates; otherwise, we add \( j_{K+1} \) to the collection \( J \) and repeat the procedure. The problem of finding the column with the most negative reduced cost is equivalent to solving the following subproblem:

\[
P_{\text{CS-sub}}: \begin{align*}
\text{maximize} \quad & \sum_{i=1}^{m} p_i a_i \\
\text{subject to} \quad & \sum_{i=1}^{m} w_i a_i \leq W, \\
& a_i \in \mathbb{N}^+, \quad \forall i \in [m].
\end{align*}
\] (35a)

where \( \mathbb{N}^+ \) is the set of nonnegative integers; if the optimal value \( v(P_{\text{CS-sub}}) \) is smaller than 1, then we terminate the column generation procedure; otherwise, we let pattern \( j_{K+1} \) correspond to the optimal solution of \( P_{\text{CS-sub}} \) and add it to \( J \).

Instead of column generation, we can consider solving the cutting-stock problem by the column randomization method. In our implementation of the column randomization method, we consider the randomization scheme described in Algorithm 3. The randomization scheme essentially starts with an empty pattern, i.e., \( (a_1, \ldots, a_m) = (0, \ldots, 0) \) and at each iteration, it increments \( a_i \) for a randomly chosen \( i \), while ensuring that it does not run out of unused width. We note that Algorithm 3 is not the only way to sample columns of \( A \), and one can consider other randomization schemes that would lead to potentially better performance of the column randomization method. Our intention here is to provide a simple example of how one can design a randomization scheme based on problem structure.

In Figure 2, we illustrate the performance of column-randomized linear programs for the cutting-stock problem with respect to number of columns \( K \in \{2 \times 10^4, 4 \times 10^4, 6 \times 10^4, 8 \times 10^4\} \) and number of required widths \( m \in \{1000, 2000, 4000\} \). We note that the value of \( m \) significantly affects size and complexity of the problem: as \( m \) increases, there are more possible patterns and thus \( n \) increases as well. For the CG approach, \( m \) defines the number of integer variables in the subproblem (35); as it increases, the subproblem becomes more challenging. We set \( W = 10^5 \); we draw each \( w_i \) uniformly at random from \( \{W/10, W/10 + 1, \ldots, W/4 - 1, W/4\} \) without replacement; and we draw each \( b_i \) independently uniformly at random from \( \{1, \ldots, 100\} \). We measure the performance of column-randomized linear programs \( P_{J}^{\text{CS}} \), where each column is obtained by Algorithm 3, by its relative optimality gap \( \Delta v(P_{J}^{\text{CS}}) / v(P_{\text{CS}}) \). For each value of \( m \) and \( K \), we run the column-randomized
Algorithm 3 Sampling a Column for the Cutting-Stock Problem

1: Column $a$ is a zero vector of length $m$ and $\zeta \leftarrow W$.
2: while $\zeta > 0$ do
3:     $I \leftarrow \{i \mid w_i \leq \zeta\}$.
4:     if $|I| \geq 1$ then
5:         Sample an index $i$ uniformly at random from $I$.
6:         Update $a_i \leftarrow a_i + 1$ and $\zeta \leftarrow \zeta - w_i$.
7:     else
8:         Break the while loop
9:     return Column $a$.

method 20 times and compute the average optimality gap, which is plotted in Figure 2. Before continuing, we note here that there are many ways to randomly generate cutting-stock instances. Our goal is not to exhaustively evaluate the numerical performance of the column randomization method on every possible family of instances, but rather to understand its performance on a reasonably general set of instances.

We first observe that the curves in Figure 2 approximately match the convergence rate of $1/\sqrt{K}$ in Theorems 1 and 2. In addition, the speed of convergence significantly slows down after the optimality is smaller than 2%; see the curve for $m = 1000$. Second, as the problem size increases, we need more samples to return comparable performance in terms of optimality gap. This is reflected by the fact that for a fixed number of columns $K$, the optimality gap is larger for larger value of $m$.

![Figure 2](image-url)  

**Figure 2** Performance of the column randomization method on the cutting-stock problem with respect to number of columns $K$ and number of required widths $m$.

We further compare the runtime of the column randomization method to that of the CG method in Table 1. The first column of the table indicates the value of $m$, which quantifies the problem size and subproblem complexity. The second column indicates the number of sampled columns $K$ in the column-randomized linear program. The third and fourth columns indicate relative optimality gap $\Delta v(P^{CS}(J))/v(P^{CS})$ and runtime of the column randomization method, respectively; for both of these metrics, we report the average over 20 runs of the column-randomized method. The fifth
Table 1 Performance of the column randomization method on the cutting stock problem for different problem sizes and numbers of sampled columns.

<table>
<thead>
<tr>
<th>Demand Types ((m))</th>
<th>Columns ((K))</th>
<th>Optimality Gap (%)</th>
<th>Runtime (s)</th>
<th>CG Runtime (s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1000</td>
<td>(2 \times 10^4)</td>
<td>0.78</td>
<td>28.4</td>
<td>365.5</td>
</tr>
<tr>
<td></td>
<td>(4 \times 10^4)</td>
<td>0.36</td>
<td>56.4</td>
<td>411.7</td>
</tr>
<tr>
<td></td>
<td>(6 \times 10^4)</td>
<td>0.20</td>
<td>89.3</td>
<td>456.4</td>
</tr>
<tr>
<td></td>
<td>(8 \times 10^4)</td>
<td>0.16</td>
<td>122.5</td>
<td>475.1</td>
</tr>
<tr>
<td>(total)</td>
<td></td>
<td></td>
<td></td>
<td>775.4</td>
</tr>
<tr>
<td>2000</td>
<td>(2 \times 10^4)</td>
<td>1.65</td>
<td>58.9</td>
<td>1330.6</td>
</tr>
<tr>
<td></td>
<td>(4 \times 10^4)</td>
<td>0.65</td>
<td>120.1</td>
<td>1622.8</td>
</tr>
<tr>
<td></td>
<td>(6 \times 10^4)</td>
<td>0.43</td>
<td>197.9</td>
<td>1732.2</td>
</tr>
<tr>
<td></td>
<td>(8 \times 10^4)</td>
<td>0.31</td>
<td>287.6</td>
<td>1805.0</td>
</tr>
<tr>
<td>(total)</td>
<td></td>
<td></td>
<td></td>
<td>2932.92</td>
</tr>
<tr>
<td>4000</td>
<td>(2 \times 10^4)</td>
<td>5.10</td>
<td>139.4</td>
<td>4979.8</td>
</tr>
<tr>
<td></td>
<td>(4 \times 10^4)</td>
<td>1.59</td>
<td>314.2</td>
<td>7175.2</td>
</tr>
<tr>
<td></td>
<td>(6 \times 10^4)</td>
<td>0.95</td>
<td>527.1</td>
<td>7670.1</td>
</tr>
<tr>
<td></td>
<td>(8 \times 10^4)</td>
<td>0.68</td>
<td>768.6</td>
<td>7940.0</td>
</tr>
<tr>
<td>(total)</td>
<td></td>
<td></td>
<td></td>
<td>13336.1</td>
</tr>
</tbody>
</table>

column shows the time required by the CG method to reach the same (average) relative optimality gap. We also list the total duration for CG (i.e., the time required for CG to reach a 0% optimality gap) in the fifth column, and denote it by “(total)”.

Table 1 shows that, when the problem is small \((m = 1000)\), the column randomization method returns a high-quality solution with an optimality gap below 1%, within 30 seconds and with \(2 \times 10^4\) sampled columns. Doubling or tripling the number of sampled columns does not significantly improve the performance, as the optimality gap is already small. Meanwhile, CG also works well when \(m = 1000\), obtaining the optimal solution in a reasonable time (within fifteen minutes). On the other hand, when the problem is large \((m = 4000)\), the runtime of CG dramatically increases, as it needs almost 5000 seconds (just under 1.5 hours) to reach a 5% optimality gap. The computational limiting factor comes from solving the subproblem, which becomes more difficult as \(m\) increases. On the other hand, the column randomization method only needs ten minutes to reach a 1% optimality gap. This demonstrates the value of solving linear programs by the column randomization method in lieu of CG when the subproblem is intractable. If one requires perfectly optimal solutions (gap of 0%), one can use the result of the column randomization method as an initial warm-start solution for the column generation approach. In the case of \(m = 4000\), if one uses the result of column-randomization method with \(K = 4 \times 10^4\) as a warm start, the runtime of the column generation method could potentially be reduced by more than 50%.

### 7.2. Nonparametric Choice Model Estimation

The second problem we consider is nonparametric choice model estimation, which is a modern application of large-scale linear programming and CG. In particular, we consider estimating the ranking-based choice model from data (Farias et al. 2013, van Ryzin and Vulcano 2015, Mišić 2016). In this model, we assume that a retailer offers \(N\) different products, indexed from 1 to \(N\). We use the index 0 to represent the no-purchase alternative, which is always available to customer. Together, we refer to the set \([N]^+ \equiv \{0, 1, 2, \ldots, N\}\) as the set of purchase options. A ranking-based choice model \((\Sigma, \lambda)\) consists of two components. The first component \(\Sigma\) is a collection of rankings over options \([N]^+\), in which each ranking represents a customer type. We use \(\sigma(i)\) to indicate the rank of option \(i\), where \(\sigma(i) < \sigma(j)\) implies that \(i\) is more preferred to \(j\) under the ranking \(\sigma\). When a set of products \(S \subseteq [N]\) is offered, a customer of type \(\sigma\) selects option \(i\) from the set \(S \cup \{0\}\) with the lowest rank, i.e., the option \(\arg\min_{i \in S \cup \{0\}} \sigma(i)\). The second component \(\lambda\) is a probability
distribution over rankings in the set \( \mathbf{\Sigma} \); the element \( \lambda_\sigma \) can be interpreted as the probability that a random customer would make decisions according to ranking \( \sigma \).

To estimate a ranking-based model, we utilize data in the form of past sales rate information. Here we consider the type of data described in Farias et al. (2013); we refer readers to that paper for more details. Assume that the retailer has provided \( M \) assortments \( \mathcal{S} = \{S_1, S_2, \ldots, S_M\} \) in the past, where each \( S_m \subseteq [N] \). For each assortment \( S_m \), the retailer observes the choice probability \( \nu_{i,m} \) for assortment \( S_m \) and option \( i \), which is the fraction of past transactions in which a customer chose \( i \), given that assortment \( S_m \) was offered. We let \( \nu_{i,m} \equiv 0 \) if \( i \notin S \cup \{0\} \).

The estimation of a ranking-based choice model \( (\mathbf{\Sigma}, \mathbf{\lambda}) \) can be formulated in the form of problem \( \hat{D} \) (Section 5.5). We first notice that there are in total \( (N+1)! \) rankings over \( [N]^+ \), which we enumerate as \( \sigma_1, \sigma_2, \ldots, \sigma_{(N+1)!} \). We let the \( k \)th column of the problem correspond to ranking \( \sigma_k \), for \( k \in [(N+1)!] \). We use \( \alpha_{i,m,k} \) to indicate whether a customer following ranking \( \sigma_k \) would choose option \( k \) when offered assortment \( S_m \). The estimation problem can then be written as

\[
P^{\text{EST}}: \quad \min_{\mathbf{\lambda}, \mathbf{v}} \quad \mathcal{D}(\hat{\mathbf{v}}, \mathbf{v})
\]

\[
\text{such that} \quad \sum_{k=1}^{(N+1)!} \alpha_{i,m,k} \cdot \lambda_k = \hat{\nu}_{i,m}, \quad \forall m \in [M], i \in [N]^+,
\]

\[
\sum_{k=1}^{(N+1)!} \lambda_k = 1,
\]

\[
\mathbf{\lambda} \geq \mathbf{0},
\]

where \( \hat{\mathbf{v}} \) and \( \mathbf{v} \) are vectors of \( \hat{\nu}_{i,m} \) and \( \nu_{i,m} \) values, respectively, for \( i \in [N]^+ \) and \( m \in [M] \). The function \( \mathcal{D} \) measures the error between the predicted choice probabilities \( \hat{\mathbf{v}} \) and the actual choice probabilities \( \mathbf{v} \). We follow Mišić (2016) and set \( \mathcal{D} = \|\hat{\mathbf{v}} - \mathbf{v}\|_1 \), which has Lipschitz constant \( \sqrt{M(N+1)} \).

We notice that even if \( N \) is merely 10, problem \( P^{\text{EST}} \) has nearly \( 4 \times 10^7 \) columns. Given that problem \( P^{\text{EST}} \) may have an intractable number of columns, van Ryzin and Vulcano (2015) and Mišić (2016) applied CG to solve the problem. Alternatively, we can apply the column randomization method. We consider the randomization scheme described in Algorithm 4, where we first randomly generate a ranking (line 2) and then map its decision under each assortment to form a column (lines 3-5). Before continuing, we pause to make three important remarks. First, we note that sampling a ranking uniformly at random (line 2) requires minimal computational effort, and can be done by a single function call in most programming languages. Second, we also note that while in Algorithm 3 we directly sample the coefficients of a column, in Algorithm 4 we instead first sample the underlying “structure” of the column (a ranking) then obtain the corresponding coefficients; this illustrates the problem-specific nature of the randomization scheme. Lastly, we note that the paper of Farias et al. (2013) considered a linear program for computing the worst-case revenue of an assortment, which is effectively the minimization of a linear function of \( \mathbf{\lambda} \) subject to constraints (36b)–(36d). The paper considered a solution method for this problem based on sampling constraints in the dual (which is equivalent to sampling columns in the primal), but did not compare this approach to column generation, which will do shortly.

We compare the performance of the column randomization method to that of CG with the following experiment setup. We assume that customers follow multinomial logit (MNL) model to make decision, that is, the choice probability \( \nu_{i,m} \) follows \( \nu_{i,m} = \exp(u_i) / \left( 1 + \sum_{j \in S_m} \exp(u_j) \right) \) for a given assortment \( S_m \), where each parameter \( u_i \) represents the expected utility of product \( i \). We choose each \( u_i \sim U[0,1] \), i.e., uniformly at random from interval \([0,1]\). We also choose the set of historical assortments \( \mathcal{S} = \{S_1, \ldots, S_M\} \) uniformly at randomly from all possible \( 2^N \) assortments of \( N \) products. We examine the performance of the column randomization method under various problem sizes, using different values of \( N \) and \( M \). For the CG method, we use the method in Mišić
Algorithm 4 Sampling a Column for the Ranking Estimation Problem

1: Initialize $\alpha_{(i,m)} \leftarrow 0$ for $i \in [N]^+$ and $m \in [M]$.
2: Sample a ranking/permutation $\sigma: [N]^+ \rightarrow [N]^+$ uniformly at random.
3: for $m \in [M]$ do
4: $i^* \leftarrow \arg \min_{i \in S_m \cup \{0\}} \sigma(i)$.
5: $\alpha_{(i^*,m)} \leftarrow 1$
6: return Column $\alpha = (\alpha_{(i,m)})_{i \in [N]^+, m \in [M]}$.

(2016), and solve the subproblem as an IP using the formulation from van Ryzin and Vulcano (2015).

Table 2 shows the performance of the column randomization method. The first two columns of the table indicate the problem size. The third column shows the number of sampled columns. The fourth and fifth columns display the optimality gap and the runtime, respectively; for both of these metrics, we report the average value of the metric over 20 runs of the column randomization method. The sixth column denotes the duration of the CG method to reach the same (average) optimality gap as the column randomization method. We remark that the optimal objective value $v(P_{\text{EST}})$ is always zero, since random utility maximization models such as the MNL model can be represented as ranking-based models (Block and Marschak 1959). Thus, instead of showing relative optimality gap as in Table 1, we directly show the objective value of the column-randomized linear program in Table 2.

In all cases listed in Table 2, the column randomization method outperforms the CG method by a large margin. It only requires a fraction of the runtime of the CG method to reach the same optimality level. In particular, when $(N, M) = (10, 150)$, the column randomization method only needs three seconds to reach the optimal objective value, which is zero, while the CG method needs over ten thousand seconds (almost three hours). In real-world applications, the number of products $N$ is usually significantly larger than 10. In those cases, the advantage of column randomization will be even more pronounced. We note that in the IP formulation of the CG subproblem, the number of binary variables scales as $O(N^2 + NM)$. Thus, as $N$ increases, the subproblem quickly becomes intractable. (We additionally note that van Ryzin and Vulcano 2015 showed this subproblem to be NP-hard.)

8. Conclusion

In this paper, we analyzed the column-randomization method for solving large-scale linear programs with an intractably large number of columns, which involves simply randomly sampling a collection of $K$ columns from the constraint matrix, and solving the corresponding problem. We developed two performance guarantees for the solution one obtains from this approach, one involving a bound on dual solution and one involving a bound on reduced costs, and showed how these guarantees and the overall approach can be applied to specific problems, such as LPs with totally unimodular constraints, Markov decision processes and covering problems. In numerical experiments with the cutting stock problem and the nonparametric choice model estimation problem, we showed that the proposed approach can obtain near-optimal solutions in a fraction of the computational time required by column generation. Given the computational simplicity of randomly sampling columns in many problems, we hope that this paper will spur further research into large-scale optimization that leverages the synergy of randomization and optimization.

References


### Table 2 Performance of the column randomization method on the estimation problem \( \tilde{P}^{\text{EST}} \) under varying problem sizes and numbers of sampled columns.

<table>
<thead>
<tr>
<th>( N )</th>
<th>( M )</th>
<th>Columns (K)</th>
<th>Objective</th>
<th>Runtime (s)</th>
<th>CG Runtime (s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>6</td>
<td>50</td>
<td>500</td>
<td>0.05</td>
<td>0.03</td>
<td>20.58</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1000</td>
<td>0.00</td>
<td>0.07</td>
<td>30.44</td>
</tr>
<tr>
<td>8</td>
<td>50</td>
<td>500</td>
<td>0.13</td>
<td>0.10</td>
<td>52.32</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1000</td>
<td>0.00</td>
<td>0.12</td>
<td>88.25</td>
</tr>
<tr>
<td>8</td>
<td>100</td>
<td>500</td>
<td>0.92</td>
<td>0.21</td>
<td>120.14</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1000</td>
<td>0.07</td>
<td>0.45</td>
<td>414.43</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1500</td>
<td>0.00</td>
<td>0.66</td>
<td>632.23</td>
</tr>
<tr>
<td>10</td>
<td>50</td>
<td>500</td>
<td>0.27</td>
<td>0.17</td>
<td>11.93</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1000</td>
<td>0.00</td>
<td>0.22</td>
<td>282.78</td>
</tr>
<tr>
<td>10</td>
<td>100</td>
<td>500</td>
<td>1.60</td>
<td>0.28</td>
<td>240.23</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1000</td>
<td>0.40</td>
<td>0.53</td>
<td>774.66</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1500</td>
<td>0.06</td>
<td>0.71</td>
<td>1423.71</td>
</tr>
<tr>
<td></td>
<td></td>
<td>2000</td>
<td>1.57</td>
<td></td>
<td>2234.52</td>
</tr>
<tr>
<td>10</td>
<td>150</td>
<td>500</td>
<td>2.91</td>
<td>0.69</td>
<td>507.63</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1000</td>
<td>0.98</td>
<td>1.07</td>
<td>1399.22</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1500</td>
<td>0.43</td>
<td>1.33</td>
<td>2635.36</td>
</tr>
<tr>
<td></td>
<td></td>
<td>2000</td>
<td>0.18</td>
<td>2.01</td>
<td>4524.72</td>
</tr>
<tr>
<td></td>
<td></td>
<td>2500</td>
<td>0.00</td>
<td>3.14</td>
<td>10143.93</td>
</tr>
</tbody>
</table>


This page is intentionally blank. Proper e-companion title page, with INFORMS branding and exact metadata of the main paper, will be produced by the INFORMS office when the issue is being assembled.
EC.1. Omitted Proofs

EC.1.1. Proof of Proposition 1

Let $x^*$ be an optimal solution of $P_{\text{distr}}$. Define the solution $x'$ as

$$x' = \frac{1}{K} \sum_{k=1}^{K} \frac{x^*}{\xi_{jk}} \cdot e_{jk}.$$

With $x'$, we can bound the objective value of $P_{\text{feas}}$ as follows:

$$v(P_{\text{feas}}) \leq \|Ax' - b\|_1$$

$$= \|Ax' - Ax^* + Ax^* - b\|_1$$

$$\leq \|Ax' - Ax^*\|_1 + \|Ax^* - b\|_1$$

$$= \|Ax' - Ax^*\|_1 + v(P_{\text{distr}}) \quad \text{(EC.1)}$$

where the first step follows by the fact that $x'$, when restricted to the indices in $J$, is a feasible solution of $P_{\text{feas}}$; the third step follows by the triangle inequality; and the fourth follows by the definition of $x^*$ as an optimal solution of $P_{\text{distr}}$.

The only remaining step is to bound $\|Ax' - Ax^*\|_1$. To do this, let us define the vector $v_k$ as

$$v_k = \frac{x^*}{\xi_{jk}} A_{jk}$$

for each $k \in [K]$. The vectors $v_1, \ldots, v_K$ are special for three reasons. First, their sample mean is exactly

$$\frac{1}{K} \sum_{k=1}^{K} v_k = \frac{1}{K} \sum_{k=1}^{K} \frac{x^*}{\xi_{jk}} A_{jk}$$

$$= \frac{1}{K} \sum_{k=1}^{K} \frac{x^*}{\xi_{jk}} A e_{jk}$$

$$= Ax'.$$

Second, letting $v$ denote a random variable following the same distribution as each $v_k$, the expected value of each $v_k$ is

$$\mathbb{E}[v] = \sum_{j \in I_+} \frac{x^*}{\xi_j} A_j$$

$$= \sum_{j \in I_+} x^* A_j$$

$$= \sum_{j \in [n]} x^* A_j$$

$$= Ax^*$$

where $I_+$ is the subset of indices in $[n]$ such that $\xi_j > 0$. Note that the third step is justified by observing that $\xi_j^* = 0$ whenever $j \notin I_+$ (this is because of the constraint $0 \leq x \leq C \xi$ in the definition of $P_{\text{distr}}$).

Lastly, observe that each $v_k$ is bounded as

$$\|v_k\|_\infty = \frac{x^*}{\xi_{jk}} \cdot \|A_{jk}\|_\infty \leq C \cdot H,$$
where we use the hypothesis that \(|A_j\|_\infty \leq |A|_{\max}\) and the fact that \(x^0\) satisfies \(0 \leq x^0 \leq C\xi\).

With all of these properties, the quantity \(|Ax' - Ax^0|_1\) can be re-written as \(|(1/K)\sum_{k=1}^{K} v_k - \mathbb{E}[v]|_1\), which we can bound using Lemma 2 (see Section 4.1). Invoking Lemma 2, we get that

\[
|Ax' - Ax^0|_1 = \frac{1}{K} \sum_{k=1}^{K} v_k - \mathbb{E}[v]|_1 \\
\leq \frac{mC|A|_{\max}}{\sqrt{K}} \left(1 + \sqrt{2\log \frac{1}{\delta}}\right),
\]

with probability at least \(1 - \delta\). Using this within the bound (EC.1), we obtain that

\[
v(P_{\text{feas}}^{\text{distr}}) \leq v(P_{\text{feas}}^{\text{distr}}) + |Ax' - Ax^0|_1 \\
\leq v(P_{\text{feas}}^{\text{distr}}) + \frac{C}{\sqrt{K}} \cdot m \cdot |A|_{\max} \cdot \left(1 + \sqrt{2\log \frac{1}{\delta}}\right)
\]

holds with probability at least \(1 - \delta\), which completes the proof. \(\square\)

**EC.1.2. Proof of Theorem 4**

We prove the result by showing that the bound \(U_{\text{covering}}\) is a valid bound on \(|p|_\infty\) for any feasible solution of the dual \(D^\text{covering}_J\), no matter what the sample of columns \(J\) is, and then invoking Proposition 5. Fix an \(i \in [m]\), and consider the LP

\[
D^\text{B-covering}_J : \max\{p_i \mid p^T A_j \leq c_j^T, \ p \geq 0\}. \tag{EC.2}
\]

The optimal objective value of this problem, \(v(D^\text{B-covering}_J)\), is an upper bound on \(p_i\) for any feasible solution \(p\) of \(D^\text{covering}_J\) (and thus, it is an upper bound on \(p_i\) for any optimal solution \(p\) of \(D^\text{covering}_J\)).

Consider the dual of this problem:

\[
P^\text{B-covering}_J : \min\{c_j^T \tilde{x} \mid A_j \tilde{x} \geq e_i, \ \tilde{x} \geq 0\}, \tag{EC.3}
\]

where \(e_i\) is the \(i\)th standard basis vector for \(\mathbb{R}^m\). By weak duality, the objective value of any feasible solution of \(P^\text{B-covering}_J\) is an upper bound on \(v(D^\text{B-covering}_J)\).

We now construct a particular feasible solution. Let \(j'\) be any column in \(J\) such that \(A_{i,j'} > 0\); such a column is guaranteed to exist by our assumption on the matrix \(A\). Define a solution \(\tilde{x}\) as

\[
\tilde{x}_j = \begin{cases} 
1/A_{i,j'} & \text{if } j = j', \\
0 & \text{otherwise}.
\end{cases}
\]

It is easy to see that \(\tilde{x}\) is a feasible solution of \(P^\text{B-covering}_J\), and that its objective value is \(c_j^T \tilde{x} = c_{j'}/A_{i,j'}\). Since this objective value is bounded by \(U_{\text{covering}}\), it follows that \(U_{\text{covering}} \geq \max\{p_i \mid p^T A_j \leq c_j^T, \ p \geq 0\}\).

Since our choice of \(i\) was arbitrary, it follows that \(|p|_\infty \leq U_{\text{covering}}\) for any feasible solution of \(D^\text{covering}_J\). The result then follows by invoking Proposition 5. \(\square\)

**EC.1.3. Proof of Theorem 5**

As with Theorem 4, we will prove the result by showing that \(U_{\text{packing}}\) is a valid upper bound on \(|p|_\infty\) for any optimal solution of the dual problem \(D^\text{packing}_J\), no matter what \(J\) is, and then invoking Proposition 6.

We first establish a useful property of \(W\): the quantity \(W\) is actually an upper bound on \(v(P)\).

To see this, define the solution \(\tilde{x}^{(i)}\) for each \(i\) as

\[
\tilde{x}^{(i)} = \frac{b_i}{A_{i,j^*_i}} \cdot e_{j^*_i},
\]
and define $\tilde{x} = \sum_{i=1}^{m} \tilde{x}^{(i)}$. Let $x$ be any feasible solution of the complete problem $P_{\text{packing}}$. Note that for each $\tilde{x}^{(i)}$, we have:

$$
c^T \tilde{x}^{(i)} = \frac{c_i^* b_i}{A_{i,j}^*} \\
\geq \frac{c_i^*}{A_{i,j}^*} \left[ \sum_{j=1}^{n} A_{i,j} x_j \right] \\
= \frac{c_i^*}{A_{i,j}^*} \left[ \sum_{j:A_{i,j}>0} A_{i,j} x_j \right] \\
\geq \sum_{j:A_{i,j}>0} A_{i,j} \cdot \frac{c_j}{A_{i,j}} \cdot x_j \\
= \sum_{j:A_{i,j}>0} c_j x_j.
$$

where the first inequality follows because $x$ satisfies $Ax \leq b$, and the second follows by the definition of $j_i^*$. Using this bound, we have

$$
c^T \tilde{x} = \sum_{i=1}^{m} c^T \tilde{x}^{(i)} \\
\geq \sum_{i=1}^{m} \left[ \sum_{j:A_{i,j}>0} c_j x_j \right] \\
\geq \sum_{j=1}^{n} c_j x_j \\
= c^T x,
$$

where the second inequality follows by our assumption that for each $j$, there exists an $i$ such that $A_{i,j} > 0$.

Now, let us fix an $i \in [m]$. We wish to bound $|p_i|$ for an optimal solution $p$ of $D_{j}^{\text{packing}}$. We can compute a bound on $|p_i|$ by solving the following LP:

$$
D_{j}^{\text{B-packing}} : \max \{ p_i | p^T b \leq v(P_{j}^{\text{packing}}), p^T A_J \geq c_J^T, p \geq 0 \}.
$$

Note that by weak duality, the feasible region of $D_{j}^{\text{B-covering}}$ is exactly the set of all optimal solutions to the sampled dual problem, $D_{j}^{\text{packing}}$. Observe that for any $J$, $v(P_{j}^{\text{packing}}) \leq v(P_{j}^{\text{packing}}) \leq W$. Thus, a valid upper bound on $v(D_{j}^{\text{B-packing}})$ can be obtained by solving the following relaxation of $D_{j}^{\text{B-packing}}$:

$$
D_{j}^{\text{B-packing-rlx}} : \max \{ p_i | p^T b \leq W, p \geq 0 \}.
$$

This problem is a valid relaxation, because we have simply removed the constraint $p^T A_J \geq c_J^T$, and we have replaced the value $v(P_{j}^{\text{packing}})$ with the larger value of $W$. The optimal objective value of this relaxation is simply $W/b_i$. Therefore, we obtain that for any dual optimal solution $p$ of $D_{j}^{\text{packing}}$, $|p_i| \leq W/b_i$. It follows that $\|p\|_{\infty} \leq \max_{i \in [m]} (W/b_i) \equiv U^{\text{packing}}$, for any optimal solution $p$ of $D_{j}^{\text{packing}}$. Invoking Proposition 6 with this bound gives the desired result. $\square$
EC.1.4. Proof of Proposition 7
Let \((x^*, r^0)\) be an optimal solution of \(P_{\text{distr}}\). Consider the solution \((x', r')\) defined relative to the sample \(J\):

\[
x' = \frac{1}{K} \sum_{k=1}^{K} \frac{x_k^0}{\xi_{jk}} \mathbf{e}_{jk}, \tag{EC.4}
\]

\[
r' = \sum_{j \in [n]} \alpha_j x'_j = \frac{1}{K} \sum_{k=1}^{K} (x_k^0/\xi_{jk}) \alpha_j. \tag{EC.5}
\]

The significance of \((x', r')\) is that we will be able to show that \(r'\) will be close to \(r^0\), and that \(f(r')\) will be close to \(f(r^0)\) = \(F_{\text{distr}}\). However, \((x', r')\) is not necessarily a feasible solution to problem \(P_{\text{portfolio}}\), because \(x'\) will in general not satisfy the unit sum constraint. To turn it into a feasible solution for problem \(P_{\text{portfolio}}\), we consider the solution \((x'', r'')\) obtained by normalizing \(x'\) by its sum:

\[
x'' = \frac{x'}{1^T x'}, \tag{EC.6}
\]

\[
r'' = \frac{r'}{1^T x'}. \tag{EC.7}
\]

Note that \((x'', r'')\) is a feasible solution of \(P_{\text{portfolio}}\).

To understand why we consider \((x', r')\) and \((x'', r'')\), we show how these two solutions can be used to bound the difference between \(F_J\) and \(F_{\text{distr}}\). Let \((x, r)\) be an optimal solution of \(P_J\). We now bound \(F_J - F_{\text{distr}}\) as follows:

\[
F_J - F_{\text{distr}} = f(r) - f(r^0)
\]

\[
\leq f(r'') - f(r^0)
\]

\[
= f(r'') - f(r') + f(r') - f(r^0)
\]

\[
\leq |f(r'') - f(r')| + |f(r') - f(r^0)|
\]

\[
\leq L \|r'' - r'\|_2 + L \|r' - r^0\|_2 \tag{EC.8}
\]

where the first step follows by the definitions of \((x, r)\) and \((x^0, r^0)\); the second step follows because \((x'', r'')\) is a feasible solution of \(P_J\); the third and fourth step follow by algebra and basic properties of absolute values; and the last step follows by the fact that \(f\) is Lipschitz continuous with constant \(L\).

We now proceed to show that \(\|r' - r^0\|_2\) and \(\|r'' - r'\|_2\) can be bounded with high probability.

**Bounding** \(\|r' - r^0\|_2\): To bound this term, let us define for each \(k \in [K]\) the random vector \(r_{jk}\) as

\[
r_{jk} = \frac{x_{jk}^0}{\xi_{jk}} \alpha_{jk}.
\]

We make three important observations about \(r_{j1}, \ldots, r_{jk}\). First, for each \(k\), the norm of \(r_{jk}\) is bounded as

\[
\|r_{jk}\|_2 = \left\| \frac{x_{jk}^0}{\xi_{jk}} \alpha_{jk} \right\|_2 \leq \frac{x_{jk}^0}{\xi_{jk}} \cdot \|\alpha_{jk}\|_2 \leq \frac{C \xi_{jk}}{\xi_{jk}} \cdot H = CH.
\]
Second, observe that \( \mathbf{r}' \) is just the sample mean of \( \mathbf{r}_{j1}, \ldots, \mathbf{r}_{jK} \), i.e., \( \mathbf{r}' = (1/K) \sum_{k=1}^{K} \mathbf{r}_{jk} \). Lastly, we observe that the expected value of each \( \mathbf{r}_{jk} \) is

\[
\mathbb{E}[\mathbf{r}_{jk}] = \sum_{j \in [n]: \xi_j > 0} \frac{x_{j0}^*}{\xi_j} \alpha_j
\]

\[
= \sum_{j \in [n]: \xi_j > 0} x_{j0}^* \alpha_j
\]

\[
= \sum_{j \in [n]} x_{j0}^* \alpha_j
\]

\[
= \mathbf{r}^*,
\]

where the third step uses the fact that \( x_{j0}^* = 0 \) when \( \xi_j = 0 \) (by virtue of the constraint \( 0 \leq \mathbf{x} \leq C \xi \)). Therefore, the term \( \| \mathbf{r}' - \mathbf{r}^* \|_2 \) is just the distance between the sample mean of an i.i.d. collection of random vectors from its expected value, where the \( \ell_2 \) norm of each random vector is bounded. We can therefore invoke Lemma 1 to assert that

\[
\| \mathbf{r}' - \mathbf{r}^* \|_2 \leq \frac{CH}{\sqrt{K}} \left( 1 + \sqrt{2 \log \frac{2}{\delta}} \right)
\]

(EC.9)

with probability at least \( 1 - \delta/2 \).

Bounding \( \| \mathbf{r}'' - \mathbf{r}' \|_2 \): For this term, observe first that since \( \mathbf{r}'' = \mathbf{r}' / (1^T \mathbf{x}') \), we can re-arrange this to obtain that \( \mathbf{r}' = (1^T \mathbf{x}') \mathbf{r}'' \). Let us use \( s \) to denote the normalization constant, i.e., \( s = 1^T \mathbf{x}' \). We can now bound \( \| \mathbf{r}'' - \mathbf{r}' \|_2 \) in the following way:

\[
\| \mathbf{r}'' - \mathbf{r}' \|_2 = \| \mathbf{r}'' - s \mathbf{r}' \|_2
\]

\[
= |s - 1| \cdot \| \mathbf{r}' \|_2.
\]

We now bound \( |s - 1| \). Note that \( s \) can be written as

\[
s = 1^T \mathbf{x}' = \frac{1}{K} \sum_{k=1}^{K} x_{j0}^* \xi_{jk} 1^T \mathbf{e}_j = \frac{1}{K} \sum_{k=1}^{K} x_{j0}^* \xi_{jk}.
\]

Letting \( w_k = (x_{j0}^*/\xi_{jk}) \), we obtain \( s = (1/K) \sum_{k=1}^{K} w_k \); in other words, \( s \) is the average of \( K \) i.i.d. random variables, \( w_1, \ldots, w_K \). Note that each \( w_k \) has expected value \( \mathbb{E}[w_k] = \sum_{j \in [n]: \xi_j > 0} (x_{j0}^*/\xi_j) \cdot \xi_j = \sum_{j \in [n]: x_{j0}^* = 1} \) \( \theta \); therefore, the term \( |s - 1| \) represents how much the sample mean \( s \) deviates from its expected value of \( 1 \). We also observe that each \( w_k \) is contained in the interval \( [0, C] \). Therefore, using Hoeffding’s inequality, we obtain that

\[
\mathbb{P}r[|s - 1| > \epsilon] = \mathbb{P}r[|s - \mathbb{E}[s]| > \epsilon] \leq 2 \cdot \exp \left( -\frac{2K \epsilon^2}{C^2} \right),
\]

(EC.10)

for any \( \epsilon > 0 \); by setting \( \epsilon = C \sqrt{\log(4/\delta)}/(2K) \), we obtain that

\[
|s - 1| \leq C \sqrt{\frac{1}{2K} \log \frac{4}{\delta}},
\]

(EC.11)

with probability at least \( 1 - \delta/2 \).
With this bound in hand, let us now bound \( \|r''\|_2 \). Observe that
\[
\|r''\|_2 \leq \frac{1}{K} \sum_{k=1}^{K} \left( \frac{x_{0j}}{\xi_{jk}} \right) \cdot \|\alpha_j\|_2 \leq \frac{1}{K} \sum_{k=1}^{K} \left( \frac{x_{0j}}{\xi_{jk}} \right) \cdot H = s \cdot H,
\]
so it follows that \( \|r''\|_2 = (1/s)\|r''\|_2 \leq H \). We therefore have that \( \|r'' - r'\|_2 \) satisfies
\[
\|r'' - r'\|_2 \leq \frac{CH}{\sqrt{K}} \sqrt{\frac{1}{2} \log \frac{4}{\delta}},
\]
with probability at least \( 1 - \delta/2 \).

**Completing the proof:** We now put these two bounds together to complete the bound in (EC.8). Combining inequalities (EC.1.4) and (EC.9) together using the union bound, we have that with probability at least \( 1 - \delta \),
\[
F_j - F_{distr} \leq L\|r'' - r'\|_2 + L\|r' - r^*\|_2 \\
\leq L \cdot \frac{CH}{\sqrt{K}} \sqrt{\frac{1}{2} \log \frac{4}{\delta}} + L \cdot \frac{CH}{\sqrt{K}} \left( 1 + \sqrt{2 \log \frac{2}{\delta}} \right) \\
\leq \frac{CHL}{\sqrt{K}} \left( 1 + 3 \sqrt{\log \frac{4}{\delta}} \right).
\]

By moving \( F_{distr} \) to the right hand side, and subtracting \( F \) from both sides, we obtain the desired inequality.

**EC.1.5. Proof of Theorem 6**

Before we can prove Theorem 6, we need to establish two auxiliary results. The first result is the analog of Lemma 1 for a collection of possibly dependent random variables, formulated in terms of forest complexity.

**Lemma EC.1.** Let \( w_1, w_2, \ldots, w_K \) be \( K \) random vectors with same distribution. Let \( G \) be the dependency graph of \( w_1, w_2, \ldots, w_K \). In addition, assume \( \|w_k\|_2 \leq C \) for \( k = 1, \ldots, K \). Let \( \bar{w} = (1/K) \cdot \sum_{k=1}^{K} w_k \). Then for any \( \delta \in (0,1) \), we have, with probability at least \( 1 - \delta \),
\[
\|\bar{w} - \mathbb{E}\bar{w}\|_2 \leq C \cdot \left( \sqrt{\frac{K + 2 \cdot |E(G)|}{K^2}} + \sqrt{\frac{2 \cdot \Lambda(G)}{K^2} \cdot \log \frac{1}{\delta}} \right).
\]

**Proof of Lemma EC.1:** Define a space \( W = \{ z | \|z\|_2 \leq C \} \). Consider a scalar function \( f : W^K \to \mathbb{R} \) defined as
\[
f(z_1, z_2, \ldots, z_K) = \left\| \frac{1}{K} (z_1 + z_2 + \ldots + z_K) - \mathbb{E}\bar{w} \right\|_2
\]
For any \( k \in [K] \) and any \( z_1, \ldots, z_k, \ldots, z_K, z'_k \in W \), we have
\[
|f(z_1, \ldots, z_k, \ldots, z_K) - f(z_1, \ldots, z'_k, \ldots, z_K)| \leq \frac{\|z_k - z'_k\|_2}{K} \leq \frac{2C}{K}.
\]
Therefore, \( f \) has the bounded differences property (note that in Liu et al. 2019, this is referred to as the \( c \)-Lipschitz property; see Definition 2.1 of that paper). By Theorem 3.6 of Liu et al. (2019), for any \( \epsilon > 0 \), we have
\[
\Pr[f(w_1, \ldots, w_K) - \mathbb{E}f(w_1, \ldots, w_K) \geq \epsilon] \leq \exp \left( -\frac{K^2\epsilon^2}{2C^2 \cdot \Lambda(G)} \right)
\]
On the other hand, define \( u_i = w_i - Ew_i \). Then

\[
E[u_i^T u_j] = \begin{cases} 
E[w_i^T w_j] - \|Ew_i\|^2 \leq E[\|w_i\|_2^2] \leq C^2, & \text{if } i = j \text{ or } \langle i, j \rangle \in E(G), \\
0, & \text{otherwise.}
\end{cases}
\]

Therefore,

\[
E[f(w_1, \ldots, w_K)^2] = \left\| \frac{1}{K} (w_1 + \ldots + w_K) - E\bar{w} \right\|^2_2 \\
= \frac{1}{K^2} \left( \sum_{i,j \in [K]} E[u_i^T u_j] \right) \\
= \frac{1}{K^2} \left( \sum_{i \in [K]} E[u_i^T u_i] + \sum_{(i,j) \in E(G)} E[u_i^T u_j] \right) \\
\leq C^2 \cdot \frac{K + 2|E(G)|}{K^2}.
\]

As a result,

\[
Ef(w_1, \ldots, w_K) \leq \sqrt{Ef(w_1, \ldots, w_K)^2} \leq C \cdot \sqrt{\frac{K + 2|E(G)|}{K^2}},
\]

where the first inequality comes from the concavity of square root function. With all the results above, we have

\[
P \left[ f(w_1, \ldots, w_K) - C \cdot \sqrt{\frac{K + 2|E(G)|}{K^2}} \geq \epsilon \right] \leq P \left[ f(w_1, \ldots, w_K) - Ef(w_1, \ldots, w_K) \geq \epsilon \right] \\
\leq \exp \left( -\frac{K^2 \epsilon^2}{2C^2 \cdot \Lambda(G)} \right).
\]

Let \( \epsilon = \sqrt{2C^2 \Lambda(G) \log(1/\delta)} / K^2 \). Then with probability at least \( 1 - \delta \), we have

\[
f(w_1, \ldots, w_K) \leq C \cdot \sqrt{\frac{K + 2|E(G)|}{K^2}} + C \sqrt{\frac{2 \cdot \Lambda(G)}{K^2} \cdot \log \left( \frac{1}{\delta} \right)}.
\]

We thus prove the statement. \( \square \)

From Lemma EC.1, we can also straightforwardly prove the following result, which is the analog of Lemma 2 for possibly dependent random variables.

**Corollary EC.1.** Let \( w_1, w_2, \ldots, w_K \) be \( K \) random vectors of size \( m \) and with same distribution. Let \( G \) be the dependency graph of \( w_1, w_2, \ldots, w_K \). In addition, assume \( \|w_k\|_\infty \leq C \) for \( k = 1, \ldots, K \). Let \( \bar{w} = (1/K) \cdot \sum_{k=1}^{K} w_k \). Then for any \( \delta \in (0, 1) \), we have, with probability at least \( 1 - \delta \),

\[
\|\bar{w} - E\bar{w}\|_1 \leq \sqrt{m} \cdot C \cdot \left( \sqrt{\frac{K + 2 \cdot |E(G)|}{K^2}} + \sqrt{\frac{2 \cdot \Lambda(G)}{K^2} \cdot \log \left( \frac{1}{\delta} \right)} \right).
\]

With these two results, we can now proceed with proving Theorem 6. We define \( x^0 \) and construct random vectors \( w_{j_1}, \ldots, w_{j_K}, b_{j_1}, \ldots, b_{j_K} \) as in the proof of Proposition 2; we note that this construction is valid even if there exists dependency between the indices \( j_1, \ldots, j_K \). We further
define \( x' \) as the sample mean of \( w_{j_1}, \ldots, w_{j_K} \) and \( b' \) as the sample mean of \( b_{j_1}, \ldots, b_{j_K} \). By Proposition 2 and Expression (20), we have

\[
\Delta v(P_J) \leq \Delta v(P_{\text{distr}}) + \|x' - x^0\|_2 + \|p^*_J\|_\infty \cdot \|b' - b\|_1.
\]

(EC.12)

By invoking Lemma EC.1, with probability at least \( 1 - \delta \),

\[
\|x' - x^0\|_2 \leq C \cdot \left( \sqrt{\frac{K + 2 \cdot |E(G)|}{K^2}} + \sqrt{\frac{2 \cdot \Lambda(G)}{K^2} \cdot \log \frac{1}{\delta}} \right).
\]

(EC.13)

Similarly, by Corollary EC.1, with probability at least \( 1 - \delta \),

\[
\|b' - b\|_1 \leq \sqrt{m} \cdot C \cdot \|A\|_{\max} \cdot \left( \sqrt{\frac{K + 2 \cdot |E(G)|}{K^2}} + \sqrt{\frac{2 \cdot \Lambda(G)}{K^2} \cdot \log \frac{1}{\delta}} \right).
\]

(EC.14)

Combining inequalities (EC.12), (EC.13), and (EC.14) and applying the union bound, we conclude that, with probability at least \( 1 - \delta \), the following holds: if \( P_J \) is feasible and \( \text{rank}(A_J) = m \),

\[
\Delta v(P_J) \leq \Delta v(P_{\text{distr}}) + C \cdot (1 + m \gamma \|A\|_{\max}) \cdot \left( \sqrt{\frac{K + 2 |E(G)|}{K^2}} + \sqrt{\frac{2 \Lambda(G) \log(2/\delta)}{K^2}} \right).
\]

(EC.15)

Similarly, by Proposition 2 and inequality (22), we have

\[
\Delta v(P_J) \leq \Delta v(P_{\text{distr}}) + \chi \cdot \|x' - x^0\|_2.
\]

(EC.16)

Combining with inequality (EC.13), we conclude that, with probability \( 1 - \delta \), the following holds: if \( P_J \) is feasible and \( \text{rank}(A_J) = m \),

\[
\Delta v(P_J) \leq \Delta v(P_{\text{distr}}) + C \cdot \chi \cdot \left( \sqrt{\frac{K + 2 |E(G)|}{K^2}} + \sqrt{\frac{2 \Lambda(G) \log(1/\delta)}{K^2}} \right),
\]

(EC.17)

which completes the proof. \( \square \)