Scientific Research Publishing

American Journal of Operations Research, 2024, 14, 1-31
https://www.scirp.org/journal/ajor
ISSN Online: 2160-8849
ISSN Print: 2160-8830

A Dimensional Reduction Approach Based on Essential Constraints in Linear Programming

Eirini I. Nikolopoulou[1*], George S. Androulakis[2]

[1]Department of Mathematics, University of Patras, Patras, Greece
[2]Department of Business Administration, University of Patras, Patras, Greece
Email: *nikirene@upatras.gr, gandroul@upatras.gr

How to cite this paper: Nikolopoulou, E.I. and Androulakis, G.S. (2024) A Dimensional Reduction Approach Based on Essential Constraints in Linear Programming. *American Journal of Operations Research*, **14**, 1-31.
https://doi.org/10.4236/ajor.2024.141001

Received: November 7, 2023
Accepted: January 14, 2024
Published: January 17, 2024

Abstract

This paper presents a new dimension reduction strategy for medium and large-scale linear programming problems. The proposed method uses a subset of the original constraints and combines two algorithms: the weighted average and the cosine simplex algorithm. The first approach identifies binding constraints by using the weighted average of each constraint, whereas the second algorithm is based on the cosine similarity between the vector of the objective function and the constraints. These two approaches are complementary, and when used together, they locate the essential subset of initial constraints required for solving medium and large-scale linear programming problems. After reducing the dimension of the linear programming problem using the subset of the essential constraints, the solution method can be chosen from any suitable method for linear programming. The proposed approach was applied to a set of well-known benchmarks as well as more than 2000 random medium and large-scale linear programming problems. The results are promising, indicating that the new approach contributes to the reduction of both the size of the problems and the total number of iterations required. A tree-based classification model also confirmed the need for combining the two approaches. A detailed numerical example, the general numerical results, and the statistical analysis for the decision tree procedure are presented.

Keywords

Linear Programming, Binding Constraints, Dimension Reduction, Cosine Similarity, Decision Analysis, Decision Trees

1. Introduction

The popularity of linear programming can be attributed to many factors, in-

cluding its capacity to model large and complex problems and the ability to solve such problems in polynomial time [1] using effective algorithms. However, many large-scale real-life problems almost always contain a significant number of redundant constraints and variables, increasing their complexity and impacting post-optimal analysis [2]. Redundancy in linear programming is expected because of the complete lack of knowledge about the constraints and the desire to formulate the problem without omitting essential elements. Thus, researchers tend to include all the information given in the form of binding and non-binding constraints. Even including non-binding constraints does not alter the optimum solution; it may require additional iterations and increase the computational difficulties [3] [4]. Furthermore, the optimal strategy is to remove the types of redundancy, which reduces the total solution time [5]. Because they do not affect the solution structure, redundant or excessive constraints can be removed from the problem and not included in the formulation [2]. As a result, the constraints used should ideally be less than the original ones. This paper aims to present a method that uses a subset of the original constraints considered essential to the solution. It combines the cosine simplex algorithm [6] with a recently proposed method based on the weighted average of a linear programming problem [7] and eliminates redundant constraints while improving existing solving methods.

1.1. Literature Review

The contributions to algorithms and approaches for linear programming (LP) problems are reviewed in this section. The most popular techniques for solving LP problems are provided first. These techniques are used to characterize binding and redundant restrictions in addition to solving LP problems. The literature study also includes algorithms for eliminating redundant constraints, identifying binding constraints, or contributing to the dimension reduction of the problems. The first effective solution technique in linear programming was the Simplex method proposed by Dantzig [8] [9] [10], which is the most popular method that allows efficient post-optimality analysis; however, it is very computationally expensive for large-scale problems [11] [12]. Other computational methods for maximizing a linear function subject to linear inequalities were proposed [13] [14], and solving problems in polynomial time; nevertheless, they performed poorly in practice [15]. Corley *et al.* [6] presented the cosine simplex algorithm, an improvement of the Simplex algorithm that decreases the number of simplex iterations and calculations each iteration. The cosine criteria utilized detect active constraints faster than the usual Simplex method. A polynomial projection approach [16], the Exterior Point Simplex Algorithm (EPSA) [17] [18], and improved Simplex algorithms [19] [20] [21] [22] have also been studied.

In addition to methods for solving LP problems, other algorithms have been developed to identify and remove redundancies. According to Terlakey [23], the ultimate goal of any technique is to identify nonessential constraints, and works suggest that redundant constraints should be identified and trivial constraints should be deleted [24] [25] [26]. Several techniques dealing with redundancy in

linear programming were summarized in the work proposed by Karwan *et al.* [3]. This work also mentioned that identifying and removing redundant constraints may be as computationally demanding as finding the solution. Furthermore, many approaches have been devised using pivoting rules, variable selection rules, and Simplex-like matrices to remove nonbinding constraints from LP problems [4]. Redundancy was studied by Luenberger [27] in LP, transportation, and flow chart problems. Algorithms for determining irrelevant constraints in systems of linear inequalities and identifying redundant constraints in a system of linear constraints have also been developed [28]-[34] in primal and dual-form LP problems, while methods for classifying linear constraints as redundant or necessary were also developed [35] [36]. Presolving heuristic algorithms have been used for identifying redundancy [5] [37] [38], and routines have been developed for large and sparse LP problems prior to solving them with an interior point-based optimizer [39]. Stojković and Stanimirović [40] proposed two direct methods for identifying redundant constraints in linear programming based on cosine similarity and game theory, respectively. Bradley *et al.* [41] proposed several heuristic algorithms for detecting and exploiting structural redundancy in large-scale mathematical programming models, real-life linear programming, and mixed-integer models. New approaches have been developed to preprocess nonnegative large-scale problems to reduce their dimension by identifying and removing redundant constraints and variables [42] [43] [44]. Nikolopoulou *et al.* [7] proposed a method for identifying binding constraints in LP problems using the weighted average of each constraint. The method was accurate in problems without excessive or superfluous constraints, while in general LP problems, the accuracy of the proposed algorithm was tested using Type I and Type II errors, as described in a previous procedure [32].

Real-life problems are often formulated as LP problems, including all the operative information given as constraints to avoid missing essential elements. This constraint inclusion leads to augmented problems, which are often hard to solve. To address this issue, we propose a method that uses the subset of binding constraints, forming a reduced-size problem equivalent to the original one using the outcomes from identifying binding constraints of two existing methods [6] and [7]. Consequently, combining two methods leads to a significantly reduced size of LP problems when applied to medium and large-scale LP problems. The new method is based on a geometric approach to identifying the subset of binding constraints; it eliminates the need for artificial variables and is not computationally demanding. Moreover, it introduces new notions in LP programming: the sufficient and the minimal efficient problem.

1.2. Organization of the Paper

The rest of the paper is organized as follows: Section 2 discusses the new approach using the combination of the weighted average method and the cosine simplex method. The proposed algorithm is presented in Section 3. Section 4

presents the numerical results of the algorithm tested on a collection of well-known benchmarks and random LP problems. Further statistical analysis using decision trees is presented in Section 5. Finally, Section 6 discusses the results and remarks on the proposed approach and possible extensions.

2. Materials and Methods

2.1. Description of the Problem

Consider a linear programming problem with a nonempty convex feasible region:

$$\begin{aligned} &\max \boldsymbol{z}(\boldsymbol{x}) = \boldsymbol{c}^{\mathrm{T}}\boldsymbol{x} \\ &\text{subject to} \\ &\boldsymbol{A}\boldsymbol{x} \le \boldsymbol{b} \\ &\boldsymbol{x} \ge 0 \end{aligned} \tag{1}$$

where $\boldsymbol{x}$ is the n-dimensional vector of decision variables, $\boldsymbol{A} = \left[a_{ij}\right]_{m\times n}$ is the coefficient matrix of the problem, $\boldsymbol{b} = \left[b_i\right]_m$ is the righthanded vector, and $\boldsymbol{c} = \left[c_j\right]_n$ is the objective coefficient, $c_j \in \mathbb{R}$, $a_{ij} \in \mathbb{R}$ and $b_i > 0$, for $i = 1,2,\cdots,m$, and $j = 1,2,\cdots,n$. The coefficient function of the problem is $\boldsymbol{z}(\boldsymbol{x}) = \boldsymbol{c}^{\mathrm{T}}\boldsymbol{x}$. All types of constraints are included in problem formulation since it is not known a priori which constraints are binding, redundant, or excessive. Large-scale LP problems almost always contain a significant number of redundant and excessive constraints and variables. Unfortunately, no easy method exists to identify constraints except in two-dimensional problems [4].

2.2. Definitions and Assumptions

In this section, five definitions are given. The first two refer to the types of constraints in LP problems; the next introduces a new notion regarding the subset of constraints required to solve an LP problem; and the fourth refers to the LP problem that is produced by a subset of constraints of the original problem. Finally, the last definition refers to LP problems formulated only by the binding constraints.

Consider a linear programming problem (1) with a nonempty, convex feasible region.

Definition 2.1: A constraint is called "binding" or "active" if it is satisfied as an equality in the optimal solution. Otherwise, the constraint is called "redundant".

Definition 2.2: If the plane of a redundant constraint contains no feasible points (it lies beyond the feasible region), then the constraint is called "excessive" or "superfluous".

Definition 2.3: Let $S = \left\{s_1, s_2, \cdots, s_m\right\}$ be the set of constraints of problem (1). Let $S' = \left\{s_1, s_2, \cdots, s_u\right\}$, $u < m$ be the subset of constraints that includes all the binding and nonbinding constraints. Then, $S = \left\{s_1, s_2, \cdots, s_u, s_{u+1}, s_{u+2}, \cdots, s_m\right\}$. Furthermore:

- The S' subset is called "essential", and a constraint that belongs to the S'

subset is called "essential".

- In the special case where the number of constraints is less than the number of variables, then $u \le m$.

Definition 2.4: A linear programming problem consisting of several essential constraints is called a sufficient problem.

Definition 2.5: A linear programming problem consisting of binding constraints is called a minimal efficient problem.

Consequently, the following corollaries are formulated:

Corollary 2.1: The sufficient problem is equivalent to the original problem and has the same solution.

Corollary 2.2: There could be more than one sufficient problem equivalent to the original one having the same solution.

Therefore, the new problem with the nonempty convex feasible region can be formulated as

$$\begin{aligned} &\max \boldsymbol{z}(\boldsymbol{x}) = \boldsymbol{c}^{\mathrm{T}}\boldsymbol{x} \\ &\text{subject to} \\ &\boldsymbol{A}_u\boldsymbol{x} \le \boldsymbol{b}_u \\ &\boldsymbol{x} \ge 0 \end{aligned} \tag{2}$$

where $\boldsymbol{x}$ is the n-dimensional vector of decision variables, $\boldsymbol{A}_u = \left[a_{ij}\right]_{u\times n}$ is the coefficient matrix of the problem, u is the number of the essential constraints of the original problem (1), $\boldsymbol{b}_u = \left[b_i\right]_u$ is the right-handed vector, where $\boldsymbol{c} = \left[c_j\right]_n$ is the objective coefficient, $c_j \in \mathbb{R}$, $a_{ij} \in \mathbb{R}$ and $b_i > 0$, for $i = 1,2,\cdots,u$, and $j = 1,2,\cdots,n$. The coefficient function of the problem is $\boldsymbol{z}(\boldsymbol{x}) = \boldsymbol{c}^{\mathrm{T}}\boldsymbol{x}$. The constraints in the sufficient problem (2) are reduced compared to those in the original LP (1). Thus, the sufficient problem (2) is reduced in size compared to the original problem (1). The main purpose of this paper is to locate the essential constraints that form a sufficient problem, that is, the $S' = \left\{s_1, s_2, \cdots, s_u\right\}$ subset.

2.3. Weighted Average Method & Cosine Simplex Algorithm

In this section, the main idea of the two methods that are used to form the proposed algorithm is presented.

- Weighted average method [7].

Consider the linear programming problem (1). Assume that in this problem there are no excessive constraints and the feasible region is convex. Let $\boldsymbol{x}^* = \left(x_1^*, x_2^*, \cdots, x_n^*\right)$ be the optimal solution to the problem (1). Binding constraints hold equality in the system of inequalities in problem (1), thus $\boldsymbol{x}^*$ is a solution for each binding constraint of the LP problem (1). The goal of the method is to identify the constraints that hold equality in the optimal solution $\boldsymbol{x}^*$. The key idea behind the method is that binding constraints can be predicted if the solution $\boldsymbol{x}^*$ can be estimated. For this purpose,

$$\lambda_i = \frac{\sum_{j=1}^{n} a_{ij}x_{ij}}{\sum_{j=1}^{n} a_{ij}} = b_i \Big/ \sum_{j=1}^{n} a_{ij} \tag{3}$$

is considered the weighted average of the i-constraint, $i=1,2,\cdots,m$. Then, the n-dimensional vector $\boldsymbol{\lambda}_i^* = (\lambda_i, \lambda_i, \cdots, \lambda_i)$ is a solution to the i-constraint, $i=1,2,\cdots,m$. The $\boldsymbol{\lambda}_i^*$s, $i=1,2,\cdots,m$, that correspond to binding constraints are expected to have similar or quite similar values. The m-dimensional vector $\boldsymbol{\lambda} = (\lambda_1, \lambda_2, \cdots, \lambda_m)$ represents the weighted average for the set of constraints of the problem, and the aim was to use the $\boldsymbol{\lambda}_i^*$s, $i=1,2,\cdots,m$, to identify binding constraints. For this purpose, $\boldsymbol{\lambda}_i^*$s, $i=1,2,\cdots,m$, are used in ascending order. The n-dimensional point $M(\lambda_{\min}, \lambda_{\min}, \cdots, \lambda_{\min})$, where $\lambda_{\min} = \min\{\lambda_i : i=1,2,\cdots,m\}$, is the only point where the bisector of the orthogonal coordinate system's first angle intersects the feasible region's boundaries. Hence, the constraint that corresponds to $\lambda_{\min}$, is a constraint of the feasible area. The rest of $\boldsymbol{\lambda}_i^*$s, lie on the bisector but are outside the feasible area of the problem (for further details, see [7]). The n-dimensional function $\boldsymbol{T}(\boldsymbol{x}^*) = (\lambda_{\min}, \lambda_{\min}, \cdots, \lambda_{\min})$ is a solution to a constraint of the feasible area and can be considered an estimator of the optimal solution $\boldsymbol{x}^*$ to the problem (1); therefore, the constraint corresponding to point M could be considered essential to solving the LP problem [45].

- Cosine simplex algorithm [6].

  Consider the linear programming problem (1). Let

$$\cos\theta_i = \frac{a_i^{\mathrm{T}} c}{\|a_i\| \|c\|} \tag{4}$$

for $i=1,2,\cdots,m$, be the cosine of the angle θ_i between the vectors a_i and the vector c of the objective function. For a given problem (1), this method begins by solving a relaxed problem of the original objective function subject to a single constraint, yielding a nonempty, bounded feasible region. At each iteration of the algorithm, the constraint that is most parallel to the objective function among those violated by the solution to the current relaxed problem is appended to it. The algorithm adds active constraints until the Kuhn-Tucker conditions [11] and [1], both necessary and sufficient for (1), are satisfied. When no constraints are violated, the solution to the current relaxed problem is optimal for the original problem.

2.4. Related Work and Motivation

The main idea is based on a recently proposed notion: the segmentation of the area that is prescribed by the problem's constraints into two parts: the high-chance area of nonbinding constraints and the area of potential binding constraints. For this purpose, a heuristic procedure using cosine similarity and the weighted average is applied. These two methods were chosen since they are both based on methods for potential binding constraints. As a result, a segmentation rule is formed based on the outcomes from the graphical plot between the cosine and the weighted average method [46]. The outcome of the segmentation rule was that the essential constraints should be searched among those that are more parallel to the objective function and those that correspond to small weighted average values.

An example of a linear programming problem (1), where the number of constraints is much larger than the number of variables, was used to form the heuristic segmentation rule. In this problem, which constraints are binding, redundant, or excessive is unknown. The example was considered a random problem, created using a jitter function, and implemented in R [47]. At first, a vector that was considered a solution to the problem was chosen. Then, the coefficient matrix $\boldsymbol{A}=\left[a_{ij}\right]_{m\times n}$ was formed, and the coefficients a_{ij}, $i=1,2,\cdots,m$, $j=1,2,\cdots,n$, were generated independently and randomly from the uniform distribution. The vector $\boldsymbol{b}=\left[b_i\right]_m$ is formed by multiplying the above matrix $\boldsymbol{A}$ with the considered solution and adding random noise to this vector. To form the objective coefficient vector $\boldsymbol{c}=\left[c_j\right]_n$, the coefficients were generated independently and randomly, and $c_j\in\left[-1,1\right]$, $j=1,2,\cdots,n$.

Consider a random 750 × 50 LP problem (1). The binding constraints in this example are 18. The optimal value of the objective function is equal to 28.065, and the number of total iterations is 146. Consider each constraint's cosine (4) and weighted average (3). A graphical plot (**Figure 1**) is used between the cosine and the $1/\lambda_i^2$, $i=1,2,\cdots,750$. The black dotted vertical lines refer to the quartiles of the cosine and the $1/\lambda_i^2$, $i=1,2,\cdots,750$, on the plot. In **Figure 2**, the axis that corresponds to $1/\lambda_i^2$, $i=1,2,\cdots,750$, is represented by $1/l^2$, and the

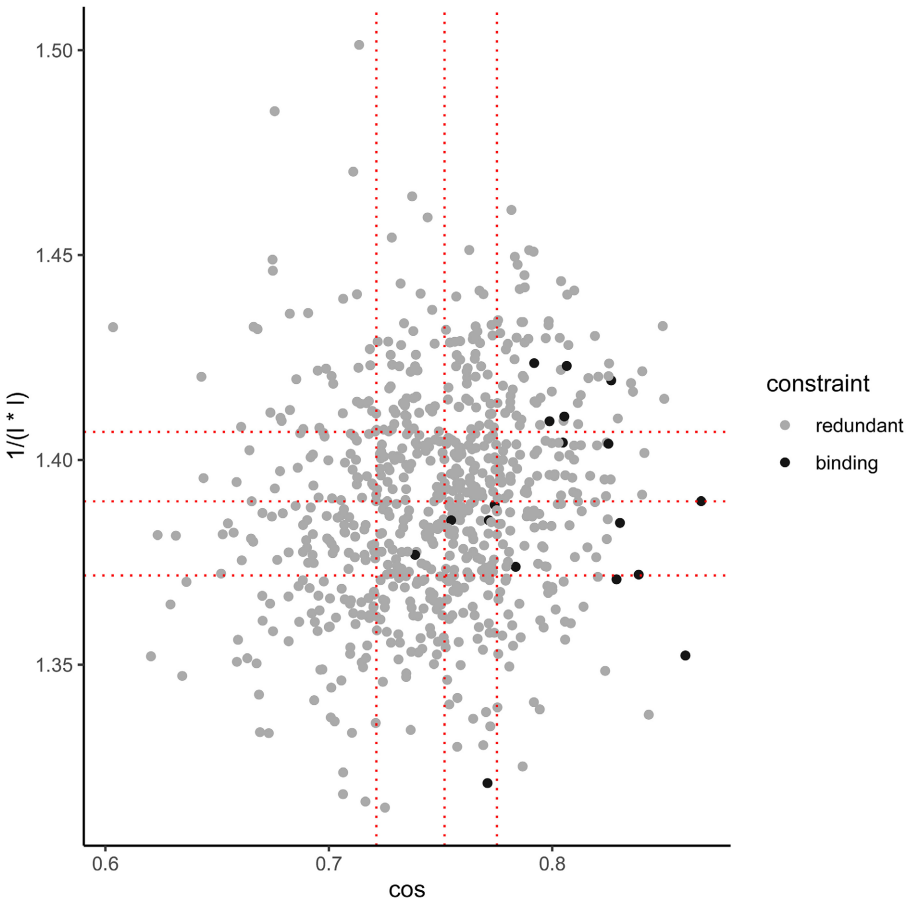

Figure 1. First segmentation of a random 750 × 50 LP problem.

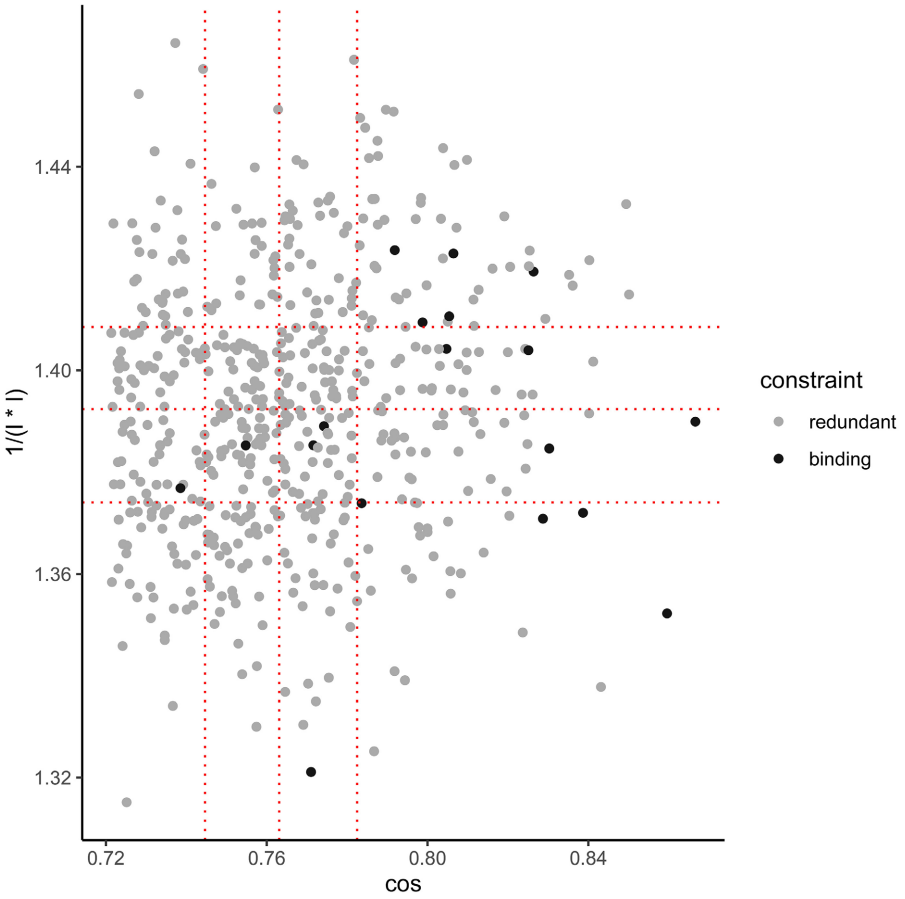

Figure 2. Second segmentation of the random LP problem.

axis that corresponds to $\cos\theta_i$, $i=1,2,\cdots,750$, is represented by cos. The binding constraints are represented as dark blue dots, while the redundant or excessive constraints are represented as light grey dots. According to **Figure 1** there are no binding constraints in the area that is defined below the value of the first quartile of cosine.

Therefore, the points in this area can be removed from the original problem and can be considered points referring to redundant or excessive constraints. This area can be considered a high-chance area of redundant or excessive constraints, while the complementary area can be considered an area where potential binding constraints can be found. The constraints considered essential to solving the problem are located in the complementary area.

After removing the points in the high-chance area of excessive constraints, the constraints of the new problem are 562, which is equal to a 25% reduction of the original number of constraints. The optimal value of the objective function of the new, reduced-sized problem is equal to 28.065, which is equal to the value of the objective function of the original problem. In the reduced problem, the binding constraints are the same as those of the original problem; however, the total iterations are 111, which is a 23.972% reduction of the original number of iterations. In **Figure 2**, where the same procedure is used, the axis that corres-

ponds to $1/\lambda_i^2$, $i=1,2,\cdots,562$, is represented by $1/l^2$, and the axis that corresponds to $\cos\theta_i$, $i=1,2,\cdots,562$, is represented by cos. The binding constraints are represented as dark blue dots, while the redundant or excessive constraints are represented as light grey dots. In the new problem, it is observed that three binding constraints are in the new high-chance area of redundant or excessive constraints. Furthermore, there are no binding constraints in the lower left corner area of the plot, which consists of dots that are below the line referring to the new first quartile of cosine and the line referring to the new first quartile of $1/l_i^2$, $i=1,2,\cdots,562$. In this area, 39 constraints can be removed from the problem.

The constraints of the new problem are now 523, which means that the constraint reduction of the original problem is 30.267%. The optimal value of the new, reduced-sized problem equals 28.065; the binding constraints and the non-zero variables are the same as those of the original problem, and the number of total iterations is 109. The final number of iterations is reduced by 25.343%.

The proposed two-step segmentation rule is summarized as follows:

Two-step segmentation rule
input: The number of decision variables (n), the number of constraints (m), the coefficient matrix of the problem (A), the vector of the right-hand side coefficients (b), the vector of the objective coefficients (c). Output:
(**Initialization**) Compute: $\cos\theta_i$, $i=1,2,\cdots,m$, using (4) Q_{1c}, the first quartile of the values of $\cos\theta_i$, $i=1,2,\cdots,m$ **Step 1:** Remove the constraints that: $\cos\theta_i \le Q_{1c}$, $i=1,2,\cdots,m$ **Step 2:** for the new problem, compute λ_i, $i=1,2,\cdots,k-m$, using (3) $1/\lambda_i^2$, for $i=1,2,\cdots,k-m$ $\cos\theta_i'$, $i=1,2,\cdots,m$, using (4) Q_{1c}', the first quartile of the values of $\cos\theta_i'$, $i=1,2,\cdots,k-m$ $Q_{1\lambda}'$, the first quartile of the values of $1/\lambda_i^2$, $i=1,2,\cdots,k-m$ Remove the constraints that: $\cos\theta_i' \le Q_{1c}'$ & $1/\lambda_i^2 \le Q_{1\lambda}'$, $i=1,2,\cdots,k-m$

The rule's effectiveness was proved by an application to 1000 random medium to large-scale LP problems, where the number of constraints was about 10 - 15 times larger than the number of variables. More specifically, the rule was applied in 500 random LP problems, where n = 50 and m = 500, and in 500 random LP problems, where n = 50 and m = 750. The problems were implemented according to the procedure described in this section regarding a random 750 × 50 problem using R. To check the effectiveness of the segmentation rule, the binding constraints of the original problems and the binding constraints of the problems after the first and second segmentations were identified using the Simplex method in R.

After the first segmentation, the binding constraints were identical in 98% of the 500 × 50 problems. After the second segmentation, the binding constraints were identical in 87.2% of the problems, while in 99.2% of the problems, the number of binding constraints differed by one at most. These results are reported in Table 1. In Table 2, the first and the second column refer to the statistics regarding the percentage of the difference between the original objective value and the objective value after the first segmentation (p.obj1) and the percentage of the difference between the original objective value and the objective value after the second segmentation (p.obj2); the third and the fourth column refer to the statistics of the percentage of the difference between the original variables and the variables after the first segmentation (p.sol1) and the percentage of the difference between the original variables and the variables after the second segmentation (p.sol2) and the last column refers to the statistics of the percentage of the total constraint reduction (p.reduced). As is observed from Table 3, the two segmentations did not affect the value of the objective function significantly.

In 94% of the 750 × 50 problems, the binding constraints were the same after the first segmentation, and in 99.8% of the problems, the number of binding constraints differed by one at most. After the second segmentation, the binding constraints were identical in 82% of the problems, while in 97.2% of the problems, the number of binding constraints differed by one at most. These results are reported in Table 3. Finally, Table 4 reports the statistics regarding the

Table 1. Binding constraints after the first and the second segmentation in 500 × 50 LPs.

Binding constraints	First segmentation			Second segmentation		
	Frequency	Percent	Cumulative percent	Frequency	Percent	Cumulative percent
0	490	98%	98%	437	87.4%	87.4%
1	10	2%	100%	59	11.8%	99.2%
2	0	0	0	3	0.6%	99.8%
3	0	0	0	1	0.2%	100%
Total	500	100%		500	100%	

Table 2. Statistics after the first and the second segmentation in 500 × 50 LPs.

Statistics		p.obj1	p.obj2	p.sol1	p.sol2	p.reduced
Mean		−0.0001	−0.0001	−0.0001	0.002	0.298
Std. deviation		0.0004	0.0006	0.012	0.018	0.009
Minimum		−0.01	−0.01	−0.1	−0.1	0.276
Maximum		$<10^{-4}$	$<10^{-4}$	0.1	0.14	0.334
Quartiles	**25%**	$<10^{-4}$	$<10^{-4}$	$<10^{-4}$	0.292	0.292
	50%	$<10^{-4}$	$<10^{-4}$	$<10^{-4}$	0.298	0.298
	75%	$<10^{-4}$	$<10^{-4}$	$<10^{-4}$	0.304	0.304

Table 3. Binding constraints after the first and the second segmentation in 750 × 50 LPs.

Binding constraints	First segmentation			Second segmentation		
	Frequency	Percent	Cumulative percent	Frequency	Percent	Cumulative percent
0	470	94%	94%	410	82%	82%
1	29	5.8%	99.8%	76	15.2%	97.2%
2	1	0.2%	100%	13	2.6%	99.8%
3	0	0	0	1	0.2%	100%
Total	500	100%		500	100%	

Table 4. Statistics after the first and the second segmentation in 750 × 50 LPs.

Statistics		p.obj1	p.obj2	p.sol1	p.sol2	p.reduced
Mean		$<10^{-5}$	−0.0001	$<10^{-4}$	0.001	0.298
Std. deviation		0.0002	0.0004	0.0045	0.018	0.008
Minimum		$<10^{-5}$	$<10^{-4}$	−0.08	−0.13	0.28
Maximum		$<10^{-5}$	$<10^{-4}$	0.07	0.21	0.32
Quartiles	**25%**	$<10^{-5}$	$<10^{-4}$	$<10^{-4}$	$<10^{-4}$	0.292
	50%	$<10^{-5}$	$<10^{-4}$	$<10^{-4}$	$<10^{-4}$	0.297
	75%	$<10^{-5}$	$<10^{-4}$	$<10^{-4}$	$<10^{-4}$	0.304

variables mentioned in Table 4; in this case, the two segmentations did not significantly affect the objective function's value. To conclude, the essential constraints should be searched among the constraints more parallel to the objective function and those corresponding to small weighted average values.

3. Discussion on the Proposed Method

3.1. Description of the Method

For a given LP problem (1), where $m > n$, a combination of the cosine simplex algorithm and the weighted average algorithm is used to reduce the dimension of the problem. The main objective of the reduction is to keep the essential constraints and subtract the nonessential ones to determine a sufficient problem. To assure the effectiveness of the proposed method regarding the essential constraints, we calculate the percentage of binding constraints after the reduction that is identical to the binding ones in the original problem. For the implementation of the proposed method, the i-constraints, $i = 1, 2, \cdots, m$, that

$$\sum_{j=1}^{n} a_{ij} x_j - b_i \le 10^{-8} \tag{5}$$

are considered binding. The original and reduced problem can be solved using any LP method; for the proposed procedure, the Simplex method was used.

The two algorithms, the cosine simplex and the weighted average algorithm, are combined and use the essential constraints proposed by the mentioned me-

thods individually. Applying this procedure leads to a relaxed problem consisting of the original objective function subject to a subset of the original set of constraints, forming a nonempty, bounded feasible region. Then, using a ranking rule, the proposed method adds essential constraints until the Karush-Kuhn-Tucker conditions [1] [11], which are both necessary and sufficient for the problem (1), are satisfied to form this subset. The ranking rule is related to the cosine simplex and the weighted average algorithm. It can be summarized as follows: The problem's constraints are sorted in ascending order by the weighted average criterion and in descending order by the cosine criterion. After locating the subset of the essential constraints, any LP method can solve the given LP problem.

Regarding the weighted average criterion, a random integer number r_1: $n \le r_2 \le m$ is chosen. This number is considered the threshold for the weighted average of the constraints. That is, the constraints whose rank (in ascending order) of $1/\lambda_i^2$ for $i = 1, 2, \cdots, m$, is above this threshold should be used to solve the original problem (1). Let m_1 be the subset of these constraints. This consideration is equivalent to selecting the constraints whose rank (in ascending order) of λ_i, for $i = 1, 2, \cdots, m$, is below the threshold. Thus, it is ensured that the constraint related to $\lambda_{\min}$ is used. Regarding the maximum cosine criterion, a random integer number r_2: $n \le r_2 \le m$ is also chosen, and it is considered a threshold for the cosines of the constraints. Let m_2 be the subset of these constraints. In this case, the constraints whose cosines (in ascending order) are above this threshold should be used to solve the original problem (1). This consideration ensures that the constraint with the maximum cosine closest to the objective function is used. Constraints that form a minimum angle with the objective function are considered active [40]. However, if the system of inequalities in problem (1) contains several excessive constraints, some of the minimal angles derived from the cosine similarity (4) may be determined by excessive constraints [40].

The two criteria could lead to the same or a different subset of the original constraints. Then, the new method uses k constraints, where k is less than or at most equal to the original constraints, $n \le k \le m$, to solve the problem (1). The k constraints are a combination of the constraints corresponding to both the ascending order of the weighted average values λ_is, $i = 1, 2, \cdots, m$, and the descending order of cosine values. More specifically, the constraints satisfying the two criteria are essential for the original problem. The set of constraints proposed by the two approaches—which is the sum of the constraints proposed by the weighted average and cosine method—is used to avoid a potentially non-feasible area. In the event that the constraints chosen do not lead to a feasible solution, the problem cannot be solved, and the procedure of setting new random numbers based on the weighted average and cosine criteria is repeated. More precisely, the new thresholds that are used to choose the two subsets of constraints m_1 and m_2, should have a lower value than the previous ones. The proposed algorithm terminates when a feasible solution with fewer constraints than the

original problem is obtained ($k < m$) or all the constraints of the original problem are used ($k = m$).

Combining the two methods leads to a subproblem equivalent to the original one since it consists of essential constraints. Using fewer of the original constraints may lead to a near-optimum because binding constraints may not be included in the subset. However, applying the proposed method to several known benchmarks has shown that this probability is extremely low (see Table 5). Consequently, the new method could use a subset of the original constraints to solve the problem. For large-scale problems where the number of constraints is much larger than the number of variables, e.g., 10 - 15 times larger, the number of essential constraints could be much smaller than the number of original ones. Therefore, the dimension reduction can be quite significant. Apart from the dimension reduction, one advantage of the proposed method is that only a subset of the original constraints is included in the overall calculations using constraint selection. Another advantage is that no artificial variables except those used to solve the problem are used. The number of artificial variables is also reduced when a reduced subset of constraints is used. Moreover, in the subset of constraints, there is at least one constraint that forms the feasible region and is proposed by the weighted average method, and at least one constraint that is considered active and is proposed by the weighted average method and/or the cosine method.

3.2. The Pseudocode of the Proposed Algorithm

A flow diagram of the weighted average and cosine algorithm (w.a.co) to provide an optimal solution for LP is shown in Figure 3.

Table 5. Computational results on a selection of well-known benchmark LPs.

Problem	n	m	used	n.bind	p.bind	m_1	m_2	Common
25fv47	1571	821	772	540	0.987	532	573	333
80bau3b	9799	2262	2262	1294	0.997	1427	2227	1392
addlittle	97	56	56	44	1	49	55	48
afiro	51	27	27	19	1	16	25	14
agg	163	488	488	67	1	339	465	316
agg2	302	516	506	84	1	342	488	324
agg3	302	516	516	89	1	371	512	367
bandm	472	305	305	245	1	129	281	105
beaconfd	262	173	122	106	1	12	110	0
blend	83	74	74	59	1	74	38	38
bnl1	1175	643	607	367	1	560	220	173
boeing1	384	350	336	239	1	237	335	236
boeing2	143	166	129	128	1	102	27	0
bore3d	315	233	232	101	1	109	227	104
brandy	249	220	220	166	1	117	197	94

Continued

capri	353	271	187	142	1	177	37	27
cycle	2871	1903	1903	1204	1	1468	1202	767
czprob	3523	929	901	875	0.996	598	815	512
d2q06c	5167	2171	2145	1388	0.987	2056	1721	1632
d6cube	6184	415	414	409	1	271	181	38
degen2	534	444	440	331	1	256	438	254
degen3	1818	1503	1503	1044	1	1441	554	492
dfl001	12,230	6071	5791	5188	1	5257	4174	3640
e226	282	223	203	123	1	172	112	81
etamacro	688	400	369	308	0.997	192	280	103
fffff800	854	524	524	330	1	448	426	350
finnis	614	497	406	130	0.992	365	120	79
fit1d	1026	24	24	24	1	23	6	5
fit1p	1677	627	499	364	0.997	412	382	295
fit2d	10,500	25	25	25	1	17	18	10
fit2p	13,525	3000	3000	1500	1	2250	2462	1712
forplan	421	161	148	142	1	125	81	58
ganges	1681	1309	1078	443	1	776	385	83
gfrd_pnc	1092	616	616	548	1	229	537	150
greenbea	5402	2392	2392	2199	1	1906	1132	646
greenbeb	5402	2392	2392	2199	1	1778	2211	1597
israel	142	174	174	70	1	126	173	125
kb2	41	43	43	9	1	17	34	8
lotfi	308	153	143	111	1	126	100	83
maros_r7	9408	3136	2863	984	1	2775	922	834
moodszk1	1620	687	687	673	1	539	381	233
nesm	2923	662	611	120	1	65	563	17
pilot	3652	1441	1159	1152	1	1056	451	348
recipe	204	91	91	40	1	79	58	46
sc105	103	105	105	98	1	78	74	47
sc205	203	205	205	198	1	124	102	21
sc50a	48	50	50	47	1	48	26	24
sc50b	48	50	50	50	1	26	33	9
scagr7	140	129	112	102	1	87	79	54
scfxm1	457	330	330	185	0.984	286	116	72
scfxm2	914	660	660	375	1	329	603	272
scfxm3	1371	990	990	565	1	896	612	518
scorpion	358	388	381	312	1	355	115	89
scsd1	760	77	77	69	1	75	39	37
scsd6	1350	147	147	140	1	119	35	7

Continued

scsd8	2750	397	397	391	1	178	279	60
sctap1	480	300	166	146	1	136	39	9
sctap2	1880	1090	930	569	1	371	916	357
sctap3	2480	1480	1020	798	1	714	888	582
seba	1028	515	514	505	0.998	75	514	75
share1b	225	117	117	117	1	116	80	79
share2b	79	96	96	59	1	47	86	37
shell	1777	536	536	534	1	393	372	229
ship04l	2118	402	397	139	1	374	256	233
ship04s	1458	402	376	139	1	229	273	126
ship08l	4283	778	778	365	1	447	707	376
ship08s	2387	778	764	365	1	736	515	487
ship12l	5427	1151	1096	498	1	752	859	515
ship12s	2763	1151	909	498	1	792	443	326
sierra	2036	1227	1221	566	0.993	587	1057	423
stair	467	356	338	89	1	136	277	75
standmps	1075	467	464	352	1	208	259	3
standata	1075	359	356	352	1	208	233	85
standgub	1184	361	361	354	1	210	251	100
stocfor1	111	117	110	74	1	88	82	60
tuff	587	333	333	300	1	259	97	23
vtp.base	203	198	198	139	1	87	142	31
wood1p	2594	244	244	242	1	201	159	116
woodw	8405	1098	1064	1064	1	673	528	137

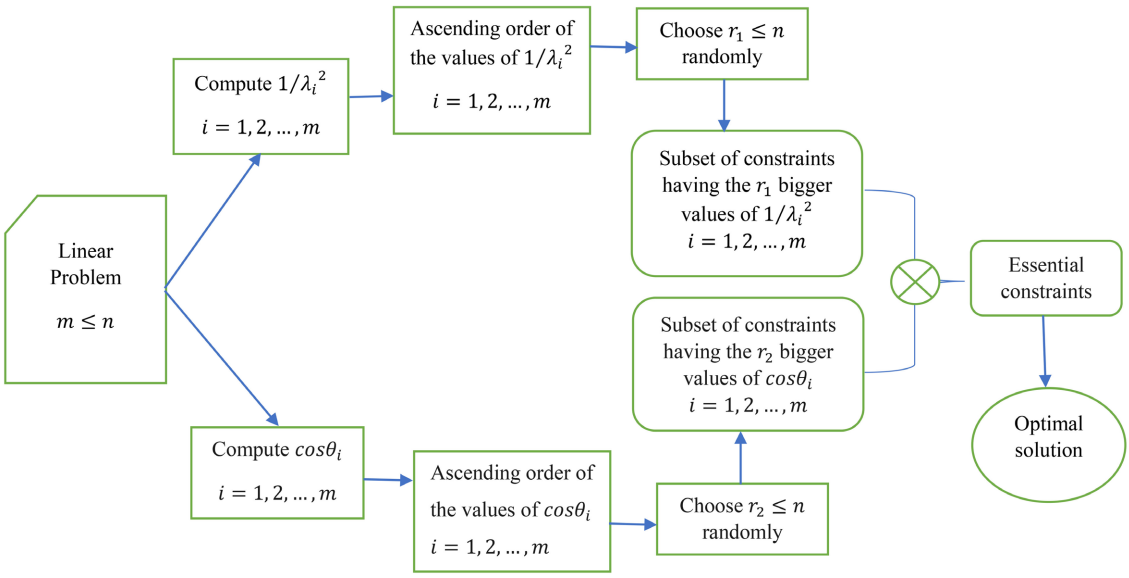

Figure 3. A flow diagram of the weighted average and cosine algorithm (w.a.co).

A formal description of the proposed algorithm is presented below:

Algorithm The weighted average and cosine algorithm (w.a.co)

input: The number of decision variables (n), the number of constraints (m), the coefficient matrix of the problem (A), the vector of the right-hand side coefficients (b), the vector of the objective coefficients (c), two random integer numbers $r_1 : n < r_1 \le m$, $r_2 : n < r_2 \le m$.

output: The percentage of binding constraints after the reduction that are identical to the binding ones in the original problem (p.bind), the total used constraints (used), the constraints of the weighted average method (m_1), the constraints of the cosine similarity method (m_2), the common constraints ($m_1 \cap m_2$).

(**Initialization**) Compute:

λ_i, $i = 1, 2, \cdots, m$, using (3)

$1/\lambda_i^2$, for $i = 1, 2, \cdots, m$

$\cos\theta_i$, $i = 1, 2, \cdots, m$, using (4)

Set $rnd1 = \left\{\text{increasing order}\left(1/\lambda_i^2\right)\right\} = \text{rank}\left(1/\lambda_i^2\right)$, $i = 1, 2, \cdots, m$

Set $rnd2 = \left\{\text{increasing order}\left(\cos\theta_i\right)\right\} = \text{rank}\left(\cos\theta_i\right)$, $i = 1, 2, \cdots, m$

(General loop)

While $r_1 : n < r_1 \le m$, $r_2 : n < r_2 \le m$, **do**

Select: m_1 = the constraints of $rnd1$ set that their rank is $\ge r_1$

Select: m_2 = the constraints of $rnd2$ set that their rank is $\ge r_2$

If $m_1 \cap m_2 = \varnothing$, **then**

use the subset $m_1 \cup m_2$

If the problem (1) is feasible, **then**

solve the problem (1) using an LP method - STOP

else

select $r_1\_new < r_1$ and $r_2\_new < r_2$

solve the problem (1) using an LP method

end second if

else

if $m_1 \cap m_2 = \varnothing$ & $m_1 + m_2 = m$, **then**

solve the problem (1) using an LP method - STOP

end first if

Print

p.bind, used, m_1, m_2, common

end while

3.3. An Example of a Linear Programming Problem

In this section, we solve an LP problem using the proposed method.

Consider the following LP problem (6):

$$\max \quad 4x_1 + 3x_2$$

$$\text{s.t.}$$

$$7x_1 + 2x_2 \le 14$$

$$3x_1 + 5x_2 \le 15$$

$$7x_1 + 6x_2 \le 42$$

$$x_1 + x_2 \le 7$$
$$3x_1 + 4x_2 \le 12$$
$$x_1, x_2 \ge 0 .$$

According to the definitions given in Section 2.2, the original set of constraints is $S = \{s_1, s_2, \cdots, s_5\}$. The main purpose is to find the S' subset of constraints using the proposed approach. LP problem (6) is first solved using Simplex in R to validate the effectiveness of the proposed procedure. The optimal value of the objective function is equal to 11.545. The first and fifth (the last) constraints are binding. Consequently, the essential constraints can be located according to the criteria proposed in Section 3. The values for $1/\lambda_i^2$, $i = 1,2,3,4,5$, are 0.413, 0.284, 0.096, 0.082, and 0.34, respectively. The biggest value among the values of $1/\lambda_i^2$, $i = 1,2,3,4,5$, is referring to the first constraint. The ascending order of the values of $1/\lambda_i^2$, $i = 1,2,3,4,5$, is 5, 3, 2, 1, 4. Using the weighted average method, r_1 is randomly chosen as 2. Then, the constraints having the two bigger values of $1/\lambda_i^2$, $i = 1,2,3,4,5$ are chosen. Therefore, the essential constraints are the first and the fifth. The values for $\cos\theta_i$, $i = 1,2,3,4,5$, are 0.934, 0.926, 0.998, 0.989, and 0.96. The biggest value of $\cos\theta_i$, $i = 1,2,3,4,5$, is referring to the third constraint. The ascending order of the values of $\cos\theta_i$, $i = 1,2,3,4,5$, is 2, 1, 5, 4, 3. Using the cosine similarity method, r_2 is randomly chosen as 4. Then, the constraints having the four bigger values of $\cos\theta_i$, $i = 1,2,3,4,5$ are chosen. In this case, the subset of essential constraints that are finally used are the first, the third, the fourth, and the fifth. The second constraint is not used since it does not satisfy either the weighted average or the cosine criterion. The second constraint is not binding, and its removal does not affect the optimal solution to the problem.

Therefore, the sufficient problem (7) consists of the four constraints is presented below:

$$\max \quad 4x_1 + 3x_2$$
$$\text{s.t.}$$
$$7x_1 + 2x_2 \le 14$$
$$7x_1 + 6x_2 \le 42$$
$$x_1 + x_2 \le 7$$
$$3x_1 + 4x_2 \le 12$$
$$x_1, x_2 \ge 0 .$$

LP problem (7) is solved again using Simplex in R language. The optimal value of the objective function of the reduced problem is equal to 11.545, which is the same as the optimal value of the objective function of the original problem (6). In the final problem (7), the first and fourth constraints are binding; the binding constraints of LP problem (6) are the same as the binding constraints of LP problem (7). According to the definitions given in Section 2.2, in LP problem (7), the subset of the essential constraints is $S' = \{s_1, s_3, s_4, s_5\}$ while in LP problem (6), the original set of constraints is $S = \{s_1, s_3, s_4, s_5, s_2\}$. The constraint re-

duction for the problem is 20%.

4. Numerical Results

In this section, the numerical results of the proposed algorithm are presented. At first, we apply the proposed method to several well-known benchmarks and random LP problems. Then, the proposed procedure was implemented in R for more than 2000 LP problems. Finally, after applying the procedure, the whole set of the reduced LPs, consisting of several well-known benchmarks and random LP problems, was solved using Simplex in R.

4.1. Application of the w.a.co. Algorithm in Netlib Problems

The proposed algorithm was applied to well-known benchmarks in the literature as netlib problems [48]. The netlib library includes medium and large-scale LP problems where some of them refer to real-life scenario problems; for example, blend is a variant of the oil refinery problem; boeing1 and boeing2 refer to flap settings on aircraft for economical operations; dfl001 is a real-world airline schedule planning (fleet assignment) problem; finnis is a model for the selection of alternative fuel types; lotfi involves audit staff scheduling; and pilot is one of the economic models developed by George Dantzig's group. In these problems, the original constraints are the given constraints. The main goal was to determine the essential constraints that form the sufficient problem according to the definitions given in Section 2.2, that is, the $S' = \{s_1, s_2, \cdots, s_u\}$ subset. However, only 16.45% of the set of problems (13 out of 79 problems) have constraints greater than the variables. Consequently, to deal with this particularity, some steps of the proposed algorithm were partially adapted:

Adapted w.a.co. algorithm

(General loop)
While $r_1 : 2 < r_1 \le m$, $r_2 : 2 < r_2 \le m$, **do**
Select: m_1 = the constraints of *rnd*1 set that their rank is $\ge r_1$
Select: m_2 = the constraints of *rnd*2 set that their rank is $\ge r_2$
If $m_1 \bigcap m_2 = \varnothing$, **then**
use the subset $m_1 \bigcup m_2$
If the problem (1) is feasible, **then**
solve the problem (1) using an LP method - STOP
else
select $r_1\_new < r_1$ and $r_2\_new < r_2$
solve the problem (1)
end second if
else
if $m_1 \bigcap m_2 = \varnothing$ & $m_1 + m_2 = m$, **then**
solve the problem (1) using an LP method - STOP
end first if
Print
p.bind, used, m_1, m_2, common
end while

Moreover, we should mention that LP problems where the number of variables is greater than the number of constraints can be solved in dual form [27] apart from using the adapted algorithm. Table 5 reports the numerical results of applying the algorithm to a set of netlib problems. Since the number of constraints is less than the number of variables in 83.3% of netlib problems, in some problems there is no significant constraint reduction. More specifically, the number of essential constraints was less than the original constraints in 50% of the problems and equaled that in 50% of the problems. We also observe that the two procedures chose different sets of constraints (m_1 and m_2). In some problems, the average weighted method proposed more constraints than the cosine similarity method; in others, the cosine similarity method proposed more constraints than the weighted average method; however, it is unclear which procedure is preferred. The weighted method selects fewer constraints than the cosine similarity method in 48.7% of the netlib problems. The detailed numerical results are summarized in Table 5.

Regarding the names of the columns in Table 5, n refers to the number of variables, m to the number of constraints, used refers to the number of essential constraints proposed by the proposed method, n.bind is the number of binding constraints according to (5), p.bind is the percentage of binding constraints located by the proposed method to the binding constraints of the original LP problem, m_1 is the set of constraints proposed by the weighted average method, m_2 is the set of constraints proposed by the cosine similarity method, and common refers to the common constraints between m_1 and m_2. According to the definitions given in Section 2.2, the numbers in column "used" refer to the subset $S' = \{s_1, s_2, \cdots, s_u\}$ of the essential constraints, while the numbers in column "m" refer to the set $S = \{s_1, s_2, \cdots, s_m\}$ of the original constraints. Regarding the number of iterations, the required number of iterations using the Simplex method is, on average, $(m+n)/2$ [12] [49]. Therefore, it is obvious that even when the constraint reduction is small using the proposed method for some netlib problems, the number of iterations and operations is significantly reduced.

Descriptive statistics regarding the application of the algorithm are reported in Table 6. The first column refers to the percentage of the binding constraints

Table 6. Statistics on a selection of well-known benchmark LPs.

Statistics		p.bind	used_m	m1_m	m2_m	m1_used	m2_used	m1_m2	c_used
Mean		0.999	0.95	0.668	0.652	0.704	0.678	1.333	0.369
Std. deviation		0.003	0.089	0.204	0.237	0.211	0.235	0.96	0.208
Minimum		0.984	0.553	0.069	0.13	0.098	0.198	0.109	0
Maximum		1	1	1	0.998	1	1	4.784	0.857
	25%	1	0.94	0.532	0.502	0.579	0.504	0.674	0.211
Quartiles	**50%**	1	1	0.677	0.679	0.738	0.703	1.055	0.385
	75%	1	1	0.823	0.861	0.872	0.897	1.578	0.518

(p.bind) after the reduction that are identical to the binding ones in the original problem; the second is the percentage of the used constraints (used_m), while the third and fourth columns are the percentages of m_1 constraints (m1_m) and m_2 constraints (m2_m) to the original ones, respectively. The following three columns are: the percentage of the m_1 constraints regarding the essential ones (m1_used), the percentage of the m_2 constraints regarding the essential ones (m2_used) and finally, the ratio of the m_1 constraints to the m_2 ones (m1_m2). The last column (c.used) is the percentage of the common constraints between the two subsets m_1 and m_2 and the essential ones.

The method used 95% on average of the original constraints, while the minimum percentage of used constraints is 55.3%, and the maximum percentage is 100%. The median percentage of correct binding constraints for the problems equals 100% (99.9% on average) (Table 6). According to Table 5, the correctly binding constraints are 100% in 87.3% of the netlib problems, and in 12.7% of the problems, the percentage of correctly binding constraints is between 98.4% and 99.8%. Therefore, there is a slight chance, 0.09%, of not including a binding constraint using the proposed method. In terms of average, the percentage of the constraints proposed from the average weighted method and the cosine similarity method was 66.8% and 65.2% of the original constraints of the problem, respectively, and the constraints proposed from the average weighted procedure and the cosine procedure were 70.4% and 67.8% of the essential constraints of the problem, respectively (Table 6). The constraint reduction can be calculated up to 64% (initial constraints minus used ones). Since the mean percentage of the common constraints of the two algorithms to the essential ones is 36.9%, there is a need to exploit the proposed union of constraints between the two algorithms (Table 6).

4.2. Application of the w.a.co. Algorithm in Random LP Problems

Apart from netlib problems, the proposed method was also applied to random LPs. We considered applying it to random LPs to exploit the proposed algorithm and prove its efficiency. The problems were created using the same R procedure presented in Section 2.4. The method was applied to 1000 random medium-scale LP problems, where $n \in [3,50]$ and $m \in [30,747]$ and to 1000 random large-scale LP problems, where $n \in [50,100]$ and $m \in [511,1332]$. The number of constraints was set to 10 - 15 times larger than the number of variables. After the application, the random LP problems were solved using Simplex. The proposed algorithm contributed to reducing the number of original constraints and performed fewer iterations when solving the reduced LP problems. The results are summarized in Table 7, where the column names are the same as those described in Table 5. Furthermore, in Table 7, the last column, iter_red, refers to the percentage of the iteration reduction.

The method used 85.9% on average of the original constraints in medium-scale problems, while the minimum percentage of used constraints equals 36% and

Table 7. Statistics on medium-scale random LPs.

Statistics		p.bind	used_m	m1_m	m2_m	m1_used	m2_used	m1_m2	c_used	iter_red
Mean		0.933	0.859	0.605	0.602	0.694	0.694	1.482	0.423	0.196
Std. deviation		0.065	0.121	0.249	0.253	0.247	0.252	1.545	0.198	0.173
Minimum		0.778	0.36	0.075	0.082	0.084	0.097	0.084	0.039	0
Maximum		1	1	1	1	1	1	10.176	0.983	0.737
Quartiles	25%	0.884	0.781	0.408	0.392	0.509	0.516	0.596	0.271	0.054
	50%	0.95	0.891	0.637	0.627	0.762	0.767	0.999	0.404	0.143
	75%	1	0.859	0.821	0.82	0.914	0.912	1.685	0.557	0.307

the maximum percentage is 100%. The percentage of binding constraints in the problems is equal to 93.3% on average. In 25% of the random problems, the percentage is 100%, and in 75%, the percentage is between 77.78% and 100%. In terms of the average percentage of constraints, the constraints proposed from the weighted average method and the cosine similarity method were 60.5% and 60.2% of the original ones, respectively, and the constraints proposed from the weighted average method and the cosine similarity method were 69.4% and 69.4% of the essential ones, respectively. The constraint reduction is 14.1% on average and can be calculated up to 64% (initial constraints minus used ones). The ratio of the common constraints of the two methods to the essential ones is 42.3% on average, which means there is a need to exploit the proposed union of constraints between the two methods.

Our method performed fewer iterations on the reduced problems than on the original ones. The number of iterations demanded was 19.6% less on average than the original. The iteration reduction was calculated at 73.7%. As was mentioned in the netlib problems, the two procedures chose a different set of constraints (m_1 and m_2). According to Table 8, the weighted method selects more constraints than the cosine similarity method in 51.1% of the random medium-scale problems, while in large-scale problems, it selects more constraints than the cosine similarity method in 48.6% of the random large-scale problems.

Moreover, in large-scale problems, the method used 86.9% on average of the original constraints. The minimum percentage of essential constraints is equal to 55%, and the maximum percentage is equal to 100%. The percentage of binding constraints is 92.5% on average; in 25% of random problems, the percentage is 100%, and in the remaining 87%, the percentage is between 76.5% and 98.3%. In terms of the average percentage of constraints, the constraints proposed from the weighted average method and the cosine similarity method were 62.5% and 60.7% of the original constraints of the problem, respectively, and the constraints proposed from the weighted method and the cosine similarity method were 70.9% and 69% of the essential constraints, respectively. The constraint reduction is 13.1% on average and can be calculated up to 44.5% (initial constraints minus used ones). The percentage of the common constraints of the two

Table 8. Constraints m_1 and m_2 in medium and large-scale random LPs.

Constraints of the two methods	Medium scale problems			Large scale problems		
	Frequency	Percent	Cumulative percent	Frequency	Percent	Cumulative percent
m_1	511	51.1%	51.1%	486	48.6%	48.6%
m_2	489	48.9%	100%	514	51.4%	100%
Total	1000	100%		1000	100%	

methods compared to the essential ones is 40%, which means there is a need to exploit the union of the proposed constraints between the two methods. These results are summarized in Table 9.

The names of the columns in Table 9 are the same as those in Table 7. In terms of the average number of iterations, the number is 18.9% less than those in the original problem. The iteration reduction was calculated at 77.3%, and in this case, the two methods chose a different set of constraints (m_1 and m_2).

4.3. Statistical Analysis

In this section, the results of a statistical analysis that was conducted using the characteristics of the problems related to the two subsets of constraints, m_1 and m_2 are presented. At first, statistical analysis was used to confirm the need to use a combination of constraints m_1 and m_2, as proposed by the two procedures, the weighted average and the cosine similarity procedure. Furthermore, statistical analysis was used to identify the most critical factors that may affect the accuracy of the solution that was obtained using the proposed method and describe relationships between these factors regarding the accuracy of finding the optimal solution. These factors arise from the characteristics of the problems that are related to the new approach and its sub-methods. The percentage of correct binding constraints in LP problems using the proposed method was determined as a factor to specify the accuracy of the approach.

4.3.1. Decision Trees for Medium and Large-Scale Problems

The further goal of the statistical analysis was to predict the characteristics of the problem with respect to the two methods and confirm that it is necessary to use both methods to find an optimal solution. According to the numerical results in Section 4.2, the mean percentage of the common constraints of the two algorithms to the essential ones is 42.3% on average in random LPs. Therefore, we conducted further analysis to explain and determine the proposed union of constraints between the two methods. More specifically, our main goal was to study which method should be used in different types of LPs to achieve better accuracy in finding the subset of binding constraints. For this purpose, a tree-based classification model was performed. Decision trees are techniques for determining which subsets of explanatory variables are most relevant to predicting a response variable. A tree structure is generated by recursively dividing the sample into a

Table 9. Statistics on large-scale random LPs.

Statistics		p.bind	used_m	m1_m	m2_m	m1_used	m2_used	m1_m2	c_used	iter_red
Mean		0.925	0.869	0.625	0.607	0.709	0.69	1.517	0.4	0.189
Std. deviation		0.939	0.8994	0.667	0.627	0.782	0.759	1.023	0.199	0.175
Minimum		0.765	0.555	0.079	0.073	0.088	0.082	0.088	0.069	0
Maximum		1	1	1	1	1	1	11.933	0.964	0.773
Quartiles	25%	0.881	0.792	0.431	0.414	0.526	0.516	0.611	0.242	0.047
	50%	0.939	0.899	0.667	0.627	0.782	0.759	1.0231	0.366	0.135
	75%	0.983	0.964	0.839	0.817	0.919	0.908	1.709	0.536	0.292

series of groups. A problem in tree construction is determining the splitting of the initial information into smaller and smaller pieces. The fundamental idea is to select each subset split so that the data in each descendant subset is "purer" than the data in the parent subset. Each subdivision is made such that the difference in the response variable in the resulting two groups is maximized [50] [51]. In addition, this technique does not require distributional assumptions; it is more resistant to the effects of outliers, and no advanced statistical knowledge is required to apply or interpret decision trees [50].

Moreover, decision trees are data-driven and not model-driven. Therefore, this technique has been preferred in statistical analysis over other classification methods or models, such as discriminant analysis and logistic regression, which have their common origin in the general linear model [50] [51]. Two tree-based classification models were constructed from the two datasets of 1000 random medium and 1000 random large-scale problems.

To perform a tree-based classification analysis, we considered the following variables according to the proposed method:

- The variable p. bind_95 refers to the percentage of correctly binding constraints. The result is measured in two categories: the percentage that is less than 95% accurate and the percentage that is more than 95% accurate. It was considered a nominal bivariate variable.
- P.bind_95 is a variable that we wanted to predict, and it was considered a measure of the accuracy of the proposed method.
- Variables m1_m_cat and m2_m_cat refer to the percentage of binding constraints used by the weighted average method and the cosine similarity method, respectively, in the total binding constraints of the problems.

These percentages are considered nominal bivariate variables with two options:

- the number of binding constraints (m_1) that is less than 50% (median) of the original number of constraints and the number of binding constraints (m_1) that is more than 50% (median) of the original number of constraints,
- or the number of binding constraints (m_2) that is less than 50% (median) of the original number of constraints and the number of binding constraints (m_2) that is more than 50% (median) of the original number of constraints.

- The variable m1_m2 is the ratio between the constraints from the weighted mean method and those from the cosine similarity method.

The ratio was considered a nominal binary variable with two options: the first is when more constraints from the weighted average method are used, and the second is when more constraints from the cosine similarity method are used.

We use the tree-based classification model in medium and large problems, where p.bind_95 is assumed to be the dependent variable and the other variables are independent. According to the performance measures used, the methods QUEST [52] [53], CHAID/exhausted CHAID [54] [55], and CART [56] [57] constructed the same trees. Moreover, the risk of the method and the percentage of correct classification were the same for the three methods [58] [59]. Therefore, without loss of generality, the CHAID growing method is used. This method builds a predictive model or tree to help determine how variables best merge to explain the outcome in the given dependent variable p.bind_95. More specifically, we use the CHAID method to predict the correct binding constraints using the constraints proposed by the weighted average method and/or the cosine Simplex algorithm. Furthermore, we want to predict which method should be used more to form the subset of essential constraints. For statistical analysis, we used IBM SPSS Statistics 28.

4.3.2. Statistical Analysis and Numerical Results

The classification trees for medium and large-scale problems are illustrated in **Figure 4** and **Figure 5** respectively. For each node, the number of problems and the percent of binding constraints, which is less or more than 95%, are given. The splits occur in order of importance. For example, m2_m_cat was the most significant factor regarding the percentage of binding constraints. The "parent" node is the accuracy percent of binding constraints and contains splits into two "child" nodes, one containing the percentage of the constraints suggested by the cosine similarity method, which is less than 50% (median), and the other containing the percentage, which is more than 50% (median). In medium-scale problems, there are eight termination nodes; in large-scale problems, there are six termination nodes. The classification in **Figure 4** and **Figure 5** is highlighted in gray. For example, in the medium-scale LP tree-based model that is described in **Figure 4**, the binding constraints in node 5 are classified in the first category (the percentage is less than 95%).

For both medium and large LP classification models, a suitable proposal for the two subsets of constraints, m_1 and m_2 can lead to more than 95% accuracy in constraint binding. More precisely, the constraints from the weighted average method should be less than half of the original ones to achieve more than 95% accuracy in binding constraints. However, the constraints proposed by the cosine method should be less than or more than half of the original constraints. If the constraints proposed by the weighted average method are more than half of the original constraints, the accuracy of the binding constraints is less than 95%. This consideration is described in **Figure 4** and **Figure 5**.

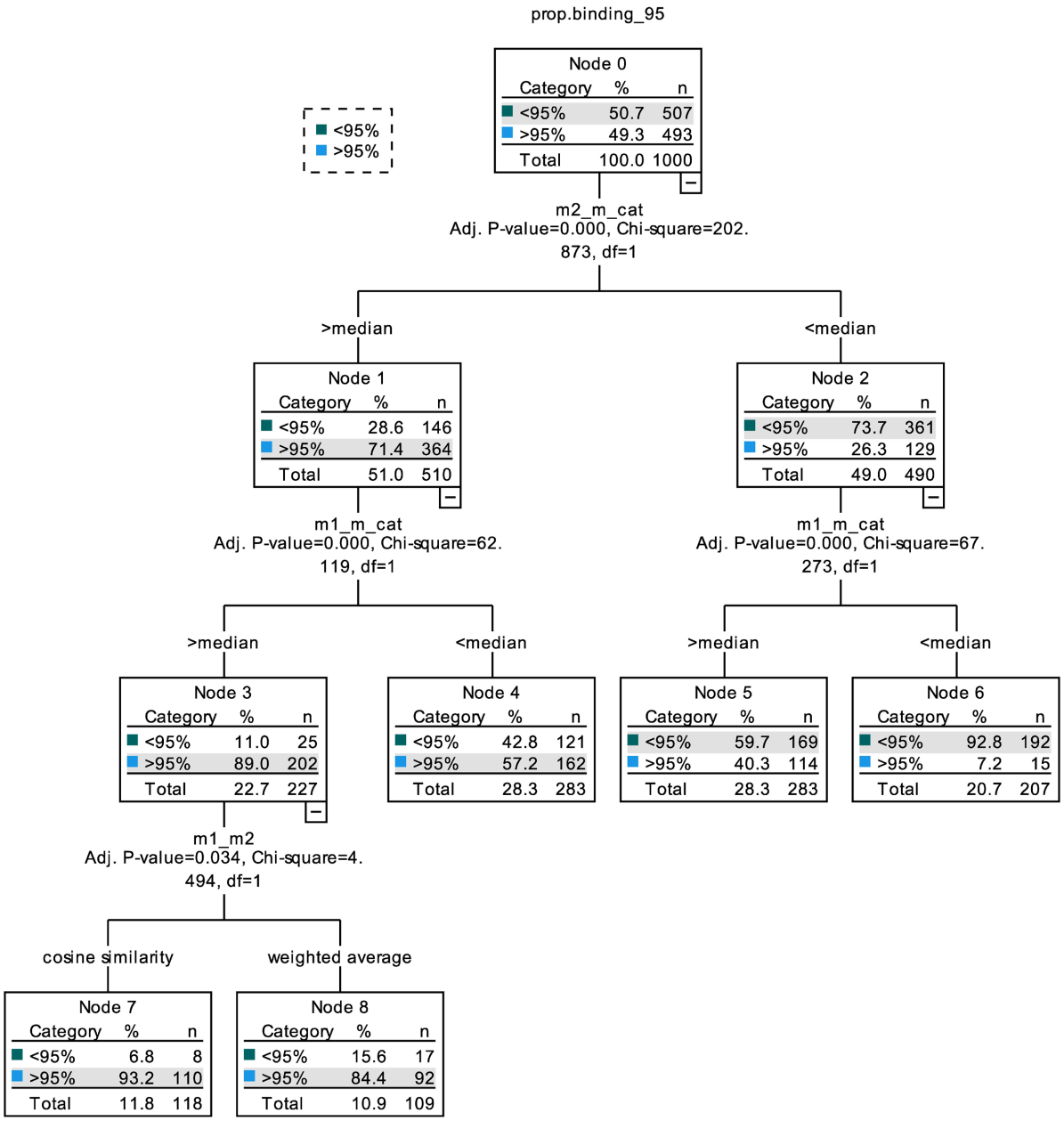

Figure 4. Tree-based classification model in medium-scale random LPs.

The tree-based classification model calculates a measure of prediction accuracy (risk) to indicate the proportion of cases misclassified by the proposed classification. For medium-sized problems, the risk for resubstitution and cross-validation is 0.275 (standard error = 0.014), whereas for large problems, the risk for substitution is 0.266, and for cross-validation is 0.275 (standard error = 0.014). Table 10 reports these results. For medium-scale problems, the model correctly classified 71.2% of the first class of binding constraints (percentage of correct identification < 95%) and 73.8% of the second class of binding constraints (percentage of correct identification > 95%), while the overall percentage of correct classification was 72.5% (Table 11). For large-scale problems, the model correctly classified

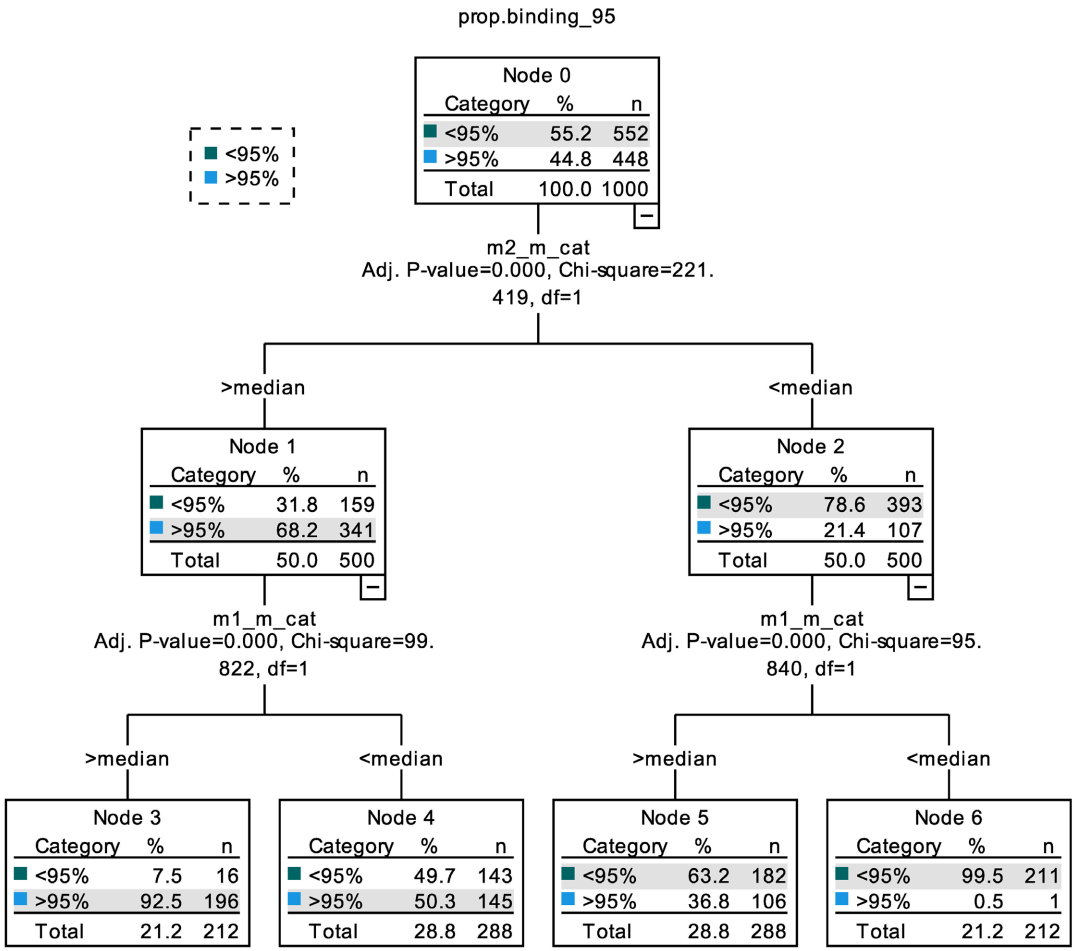

Figure 5. Tree-based classification model in large-scale LPs.

Table 10. Classification risk in medium and large-scale random LPs.

Method	Medium-scale problems		Large-scale problems	
	Estimate	Std. error	Estimate	Std. error
Resubstitution	0.275	0.014	0.266	0.014
Cross-validation	0.275	0.014	0.274	0.014

Table 11. Classification in medium and large-scale random LPs.

Observed	Predicted medium-scale			Predicted large-scale		
	<95%	>95%	Correct percent	<95%	>95%	Correct percent
<95%	361	146	71.2%	393	159	71.2%
>95%	129	364	73.8%	107	341	76.1%
Overall percentage	49%	51%	72.5%	50%	50%	73.4%

71.2% of the first class of binding constraints (percentage of correct identification < 95%) and 73.8% of the second class of binding constraints (percentage of

correct identification > 95%), while the overall percentage of correct classification was 73.4%. These results are summarized in Table 11. Consequently, for medium and large problems, there is a 72.5% and 73.4% chance of achieving accuracy greater than 95% for binding constraints using the characteristics of the problems concerning the two subsets of constraints, respectively.

The tree-based classification model supported the combination of the two methods. According to the classification model, the constraints proposed by the weighted average method should be less than 50% of the original ones, while the constraints proposed by the cosine method should be more than 50% of the original ones to achieve accuracy in solving LP problems.

5. Conclusions

Researchers tend to include constraints that are not binding to the optimal solution when formulating linear programming problems for fear of excluding necessary constraints. This inclusion does not affect the optimal solutions; nonetheless, it may necessitate more iterations and raise the computing difficulty. Thus, algorithms inevitably need to be developed to eliminate redundancy using necessary constraints and reduce the dimension of the problem under study.

For this purpose, this paper proposes a method for dimension reduction of medium and large-scale linear programming problems using a subset of the original constraints considered essential. This subset includes the binding constraints and forms a sufficient problem equivalent to the original one. It is chosen from two complementary procedures: the weighted average procedure and the cosine procedure. The weighted average procedure ensures that the proposed method uses at least one constraint that forms the feasible region and is potentially binding, while the cosine similarity procedure uses at least one constraint with a high probability of binding. According to the numerical results, the combination of the two algorithms mentioned seems to be promising. The accuracy of identifying significant constraints using this method was tested on a collection of known benchmarks and 2000 random medium and large-scale LP problems. Statistical analysis showed that neither method should be preferred but should be used in a complementary manner. More specifically, the constraints proposed by the weighted average method should be less than 50% of the initial ones, while the constraints suggested by the cosine method should be more than 50% to achieve accuracy in solving LP problems. The reduction of constraints was calculated up to 44.7% in netlib problems, 64% in medium-scale problems, and 44.5% in large-scale problems, and the iteration reduction was calculated up to 73.7% and 77.3% in medium and large-scale problems, respectively.

Future work includes extending the computational studies to a larger number of tested problems from available benchmark collections and in integer or mixed-integer linear programming and improving the reduction and accuracy level by implementing the algorithms to use an appropriate subset of the original constraints. In addition, the quality of the linear model can be improved by

checking if the proposed subsets contain constraints that are not necessary by using specific properties as used in constraint satisfaction problems. The proposed method can be applied to large-scale real-life problems like logistics and warehouse LP-formulated problems, transportation and transshipment problems, scheduling problems, product mix problems, airline operations problems, and problems related to personnel allocation and the public sector. Using the proposed method, we can also have information about the subset of constraints that can be used to solve the problem and consider alternative scenarios in case the optimal solution cannot be obtained for real-life problems for unpredictable reasons.

Conflicts of Interest

The authors declare no conflicts of interest regarding the publication of this paper.

References

[1] Sergeant, C.R., Bazaraa, M.S. and Jarvis, J.J. (1978) Linear Programming and Network Flows. *Journal of the Operational Research Society*, **29**, 510. https://doi.org/10.2307/3009778

[2] Gal, T. (1992) Weakly Redundant Constraints and Their Impact on Post Optimal Analysis in LP. *European Journal of Operational Research*, **60**, 315-326. https://doi.org/10.1016/0377-2217(92)90083-L

[3] Karwan, M.H., Lotfi, V., Telgen, J. and Zionts, S. (1983) Redundancy in Mathematical Programming: A State-of-the-Art Survey. Springer-Verlag, Berlin. https://doi.org/10.1007/978-3-642-45535-3

[4] Thompson, G.L., Tonge, F.M. and Zionts, S. (1966) Techniques for Removing Nonbinding Constraints and Extraneous Variables from Linear Programming Problems. *Management Science*, **12**, 588-608. https://doi.org/10.1287/mnsc.12.7.588

[5] Andersen, E.D. and Andersen, K.D. (1995) Presolving in Linear Programming. *Mathematical Programming*, **71**, 221-245. https://doi.org/10.1007/BF01586000

[6] Corley, H.W., Rosenberger, J., Yeh, W.C. and Sung, T.K. (2006) The Cosine Simplex Algorithm. *International Journal of Advanced Manufacturing Technology*, **27**, 1047-1050. https://doi.org/10.1007/s00170-004-2278-1

[7] Nikolopoulou, E.I., Manoussakis, G.E. and Androulakis, G.S. (2019) Locating Binding Constraints in LP Problems. *American Journal of Operations Research*, **9**, 59-78. https://doi.org/10.4236/ajor.2019.92004

[8] Dantzig, G.B. (1951) Maximization of a Linear Function of Variables Subject to Linear Inequalities. In: Koopmans, T.C., Ed., *Activity Analysis of Production and Allocation*, Wiley & Chapman-Hall, New York, 339-347.

[9] Dantzig, G.B. (1955) Linear Programming under Uncertainty. *Management Science*, **1**, 197-206. https://doi.org/10.1287/mnsc.1.3-4.197

[10] Dantzig, G.B. (2016) Application of the Simplex Method to a Transportation Problem. In: Koopmans, T.C., Ed., *Activity Analysis of Production and Allocation*, John Wiley and Sons, New York, 359-373.

[11] Bazaraa, M.S., Jarvis, J.J. and Sherali, H.D. (2011) Linear Programming and Network Flows. 4th Edition, Wiley, Hoboken.

[12] Vanderbei, R.J. (2020) Linear Programming: Foundations and Extensions. International Series in Operations Research and Management Science, Vol. 285, Springer, Berlin. https://doi.org/10.1007/978-3-030-39415-8

[13] Brown, G.W. and Koopmans, T.C. (1952) Computational Suggestions for Maximizing a Linear Function Subject to Linear Inequalities. In: Koopmans, T.C., Ed., *Activity Analysis of Production and Allocation*, Wiley, New York, 625-628. https://doi.org/10.2307/2226909

[14] Khachiyan, L.G. (1980) Polynomial Algorithms in Linear Programming. *USSR Computational Mathematics and Mathematical Physics*, **20**, 53-72. https://doi.org/10.1016/0041-5553(80)90061-0

[15] Bland, R.G., Goldfarb, D. and Todd, M.J. (1981) Ellipsoid Method: A Survey. *Operations Research*, **29**, 1039-1091. https://doi.org/10.1287/opre.29.6.1039

[16] Karmarkar, N. (1984) A New Polynomial-Time Algorithm for Linear Programming. *Combinatorica*, **4**, 373-395. https://doi.org/10.1007/BF02579150

[17] Paparrizos, K. (1991) An Infeasible (Exterior Point) Simplex Algorithm for Assignment Problems. *Mathematical Programming*, **51**, 45-54. https://doi.org/10.1007/BF01586925

[18] Paparrizos, K. (1993) An Exterior Point Simplex Algorithm for (General) Linear Programming Problems. *Annals of Operations Research*, **46-47**, 497-508. https://doi.org/10.1007/BF02023111

[19] Basu, A., De Loera, J.A. and Junod, M. (2014) On Chubanov's Method for Linear Programming. *INFORMS Journal on Computing*, **26**, 336-350. https://doi.org/10.1287/ijoc.2013.0569

[20] Gleixner, A.M., Steffy, D.E. and Wolter, K. (2016) Iterative Refinement for Linear Programming. *INFORMS Journal on Computing*, **28**, 449-464. https://doi.org/10.1287/ijoc.2016.0692

[21] Omer, J., Rosat, S., Raymond, V. and Soumis, F. (2015) Improved Primal Simplex: A More General Theoretical Framework and an Extended Experimental Analysis. *INFORMS Journal on Computing*, **27**, 773-787. https://doi.org/10.1287/ijoc.2015.0656

[22] Triantafyllidis, C.P. and Samaras, N. (2020) A New Non-Monotonic Infeasible Simplex Type Algorithm for Linear Programming. *PeerJ Computer Science*, **6**, e265. https://doi.org/10.7717/peerj-cs.265

[23] Terlaky, T. (1996) Interior Point Methods of Mathematical Programming. Springer, Berlin. https://doi.org/10.1007/978-1-4613-3449-1

[24] Boot, J.C.G. (1962) On Trivial and Binding Constraints in Programming Problems. *Management Science*, **8**, 419-441. https://doi.org/10.1287/mnsc.8.4.419

[25] Zionts, S. (1965) Size of Reduction Techniques of Linear Programming and Their Applications. Carnegie Institute of Technology, Pittsburgh.

[26] Lisy, J. (1971) Metody pro nalezini redundant omezini v ulohach linearniho programovani. *Economicko Mathematicky Obzor*, **7**, 198-285.

[27] Luenberger, D.G. (1973) Introduction to Linear and Nonlinear Programming. Addison Wesley Publishing Company, San Francisco.

[28] Mattheiss, T.H. (1973) An Algorithm for Determining Irrelevant Constraints and All Vertices in Systems of Linear Inequalities. *Operations Research*, **21**, 247-260. https://doi.org/10.1287/opre.21.1.247

[29] Brearley, A.L., Mitra, G. and Williams, H.P. (1975) Analysis of Mathematical Programming Problems Prior to Applying the Simplex Algorithm. *Mathematical Pro-*

gramming, **8**, 54-83. https://doi.org/10.1007/BF01580428

[30] Gal, T. (1979) Weakly Redundant Constraints and Their Impact on Post Optimal Analysis. *European Journal of Operational Research*, **60**, 315-326. https://doi.org/10.1016/0377-2217(92)90083-L

[31] Telgen, J. (1983) Identifying Redundant Constraints and Implicit Equalities in Systems of Linear Constraints. *Management Science*, **39**, 1209-1222. https://doi.org/10.1287/mnsc.29.10.1209

[32] Hebert, T. and Tsai, S.Y.W. (1981) The Identification of Binding Constraints: A Directional Derivative Heuristic Approach. *Computers, Environment and Urban Systems*, **6**, 115-125. https://doi.org/10.1016/0198-9715(81)90007-7

[33] Gal, T. (1984) Linear Parameteric Programming—A Brief Survey. In: Fiacco, A.V., Ed., *Sensitivity, Stability and Parametric Analysis*, Springer, Berlin, 43-68. https://doi.org/10.1007/BFb0121210

[34] Megiddo, N. (1984) Linear Programming in Linear Time When the Dimension Is Fixed. *Journal of the ACM* (*JACM*), **31**, 114-127. https://doi.org/10.1145/2422.322418

[35] Berbee, H.C.P., Boender, C.G.E., Rinnooy Ran, A.H.G., Scheffer, C.L., Smith, R.L. and Telgen, J. (1987) Hit-and-Run Algorithms for the Identification of Nonredundant Linear Inequalities. *Mathematical Programming*, **37**, 184-207. https://doi.org/10.1007/BF02591694

[36] Caron, R.J., McDonald, J.F. and Ponic, C.M. (1989) A Degenerate Extreme Point Strategy for the Classification of Linear Constraints as Redundant or Necessary. *Journal of Optimization Theory and Applications*, **62**, 225-237. https://doi.org/10.1007/BF00941055

[37] Ioslovich, I. (2002) Robust Reduction of a Class of Large-Scale Linear Programs. *SIAM Journal on Optimization*, **12**, 262-282. https://doi.org/10.1137/S1052623497325454

[38] Paulraj, S., Chellappan, C. and Natesan, T.R. (2006) A Heuristic Approach for Identification of Redundant Constraints in Linear Programming Models. *International Journal of Computer Mathematics*, **83**, 675-683. https://doi.org/10.1080/00207160601014148

[39] Gondzio, J. (1997) Presolve Analysis of Linear Programs Prior to Applying an Interior Point Method. *INFORMS Journal on Computing*, **9**, 73-91. https://doi.org/10.1287/ijoc.9.1.73

[40] Stojković, N.V. and Stanimirović, P.S. (2001) Two Direct Methods in Linear Programming. *European Journal of Operational Research*, **131**, 417-439. https://doi.org/10.1016/S0377-2217(00)00083-7

[41] Bradley, G.H., Brown, G.G. and Graves, G.W. (1983) Structural Redundancy in Large-Scale Optimization Models. In: Karwan, M.H., Ed., *Redundancy in Mathematical Programming: A State-of-the-Art Survey*, Springer-Verlag, Berlin, 145-169. https://doi.org/10.1007/978-3-642-45535-3_12

[42] Gutman, P.O. and Isolovich, I. (2007) On the Generalized Wolf Problem: Preprocessing of Nonnegative Large-Scale Linear Programming Problems with Group Constraints. *Automation and Remote Control*, **68**, 1401-1409. https://doi.org/10.1134/S0005117907080115

[43] Sumathi, P. and Paulraj, S. (2013) Identification of Redundant Constraints in Large Scale Linear Programming Problems with Minimal Computational Effort. *Applied Mathematical Sciences*, **7**, 3963-3974. https://doi.org/10.12988/ams.2013.36325

[44] Sumathi, P. and Gangadharan, A. (2014) Selection of Constraints with a New Approach in Linear Programming Problems. *Applied Mathematical Sciences*, **8**, 1311-1321. https://doi.org/10.12988/ams.2014.42121

[45] Nikolopoulou, E.I. (2020) A Survey on Linear Constrained Models and Applications in Operational Research. Ph.D. Thesis, University of Patras, Patras.

[46] Nikolopoulou, E. and Androulakis, G.S. (2021) A Segmentation Rule to Determine Areas of Potential Binding and Non-Binding Constraints in LP Problems. 31*st European Conference on Operational Research*, Athens, 11-14 July 2021, 261.

[47] R Core Team (2019) R: A Language and Environment for Statistical Computing. R Foundation for Statistical Computing, Vienna. https://www.R-project.org/

[48] Browne, S., Dongarra, J., Grosse, E. and Rowan, T. (1995) The Netlib Mathematical Software Repository. *D-Lib Magazine*, **1**, 96. https://doi.org/10.1045/september95-browne

[49] Maurer, S.B., Nering, E.D. and Tucker, A.W. (1994) Linear Programs and Related Problems. *The American Mathematical Monthly*, **101**, 1022-1026. https://doi.org/10.2307/2975180

[50] Breiman, L., Friedman, J.H., Olshen, R.A. and Stone, C.J. (2017) Classification and Regression Trees. Routledge, New York. https://doi.org/10.1201/9781315139470

[51] Everitt, B.S. and Skrondal, A. (2010) The Cambridge Dictionary of Statistics. https://www.stewartschultz.com/statistics/books/Cambridge%20Dictionary%20Statistics%204th.pdf

[52] Fielding, A.H. (2006) Cluster and Classification Techniques for the Biosciences. Cambridge University Press, Cambridge. https://doi.org/10.1017/CBO9780511607493

[53] Lim, T.S., Loh, W.Y. and Shih, Y.S. (2000) Comparison of Prediction Accuracy, Complexity, and Training Time of Thirty-Three Old and New Classification Algorithms. *Machine Learning*, **40**, 203-228. https://doi.org/10.1023/A:1007608224229

[54] Kass, G.V. (1980) An Exploratory Technique for Investigating Large Quantities of Categorical Data. *Applied Statistics*, **29**, 119-127. https://doi.org/10.2307/2986296

[55] Lewicki, P. and Hill, T. (2006) Statistics: Methods and Applications—A Comprehensive Reference for Science, Industry and Data Mining. Vol. 1, StatSoft Inc., St Tulsa.

[56] Phelps, M.C. and Merkle, E.C. (2008) Classification and Regression Trees as Alternatives to Regression. 4*th Annual GRASP Symposium*, Wichita State University, 25 April 2008, 77-78.

[57] Taylor, P.C. and Silverman, B.W. (1993) Block Diagrams and Splitting Criteria for Classification Trees. *Statistics and Computing*, **3**, 147-161. https://doi.org/10.1007/BF00141771

[58] Han, J., Kamber, M. and Pei, J. (2012) Data Mining: Concepts and Techniques. Morgan Kaufmann Publishers, Waltham.

[59] Loh, W.Y. and Shin, Y.S. (1997) Split Selection Methods for Classification Trees. *Statistica Sinica*, **7**, 815-840.