UNIVERSIDAD DE CHILE
FACULTAD DE CIENCIAS FÍSICAS Y MATEMÁTICAS
DEPARTAMENTO DE INGENIERÍA INDUSTRIAL

COLUMN GENERATION-BASED DECOMPOSITION FOR LARGE-SCALE FEATURE SELECTION PROBLEMS

TESIS PARA OPTAR AL GRADO DE MAGÍSTER EN GESTIÓN DE OPERACIONES

MEMORIA PARA OPTAR AL TÍTULO DE INGENIERO CIVIL INDUSTRIAL

NICOLÁS ACEVEDO VILLENA

PROFESOR GUÍA:
Fernando Ordóñez Pizarro

MIEMBROS DE LA COMISIÓN:
Renaud Chicoisne
Charles Thraves Cortés-Monroy

Este trabajo ha sido parcialmente financiado por: ANID–Subdirección de Capital Humano/Magíster Nacional 2022–Folio 22220861 y CONICYT-FONDECYT Proyecto N°1201844

SANTIAGO DE CHILE
2023

MÉTODO DE GENERACIÓN DE COLUMNAS PARA PROBLEMAS DE SELECCIÓN DE ATRIBUTOS DE GRAN ESCALA

El actual crecimiento en los datos disponibles y su utilización en campos como el aprendizaje automático ha aumentado la demanda de algoritmos capaces de abordar problemas de gran escala. En consecuencia, el problema de selección de atributos es fundamental para construir modelos precisos y coherentes a partir de grandes conjuntos de datos. A pesar de que existe una gran cantidad de métodos para este problema, sólo un número limitado de estos es capaz de producir soluciones en un tiempo razonable y sin comprometer la calidad de las soluciones en problemas de este tipo.

Esta investigación propone un algoritmo escalable, usando un método de descomposición para problemas de optimización cónica de gran complejidad. Se trata de un método de generación de columnas que descompone una versión cónica de segundo orden del problema de selección de atributos, regularizando coeficientes asignando parámetros de penalización. Se demuestra que el método es equivalente a LASSO, y es capaz de resolver instancias de escala "mediana a grande" (miles de atributos y observaciones) en cuestión de segundos, en instancias donde otras alternativas demoran minutos o incluso horas.

El método es similar a un enfoque Dantzig-Wolfe, dado que resuelve dos subproblemas en cada iteración: (1) el problema cónico de segundo orden original, pero en un subconjunto de su región factible, y (2) un problema relajado obtenido mediante la relajación Lagrangiana de las restricciones cónicas de alta complejidad. El problema relajado genera nuevas soluciones para expandir la región factible del problema maestro. En este trabajo se modifica el método de generación de columnas para considerar casos de problemas relajados no acotados y construir soluciones artificiales en ellos. Estas soluciones artificiales se construyen a partir de información de los problemas no acotados.

El método propuesto puede resolver eficientemente instancias de escala "mediana a grande", con una penalización considerable, en cuestión de minutos, mientras que el problema original y LASSO requieren más de 20 minutos para producir esa solución. Sin embargo, el método propuesto muestra un rendimiento más lento en algunos casos. Específicamente, el método de descomposición es más eficiente que el problema original cuando selecciona 40% o menos atributos, y es más eficiente que LASSO cuando selecciona 4% o más atributos.

COLUMN GENERATION-BASED DECOMPOSITION FOR LARGE-SCALE FEATURE SELECTION PROBLEMS

The current growth in the size of available datasets and its uses in machine learning has increased the demand for fast algorithms capable of addressing large-scale problems. Consequently, the feature selection problem is central to construct meaningful and accurate models from large-scale data. While numerous methods for this problem exist, only a limited number of them can effectively handle large-scale datasets in a reasonable time without compromising solution quality.

This research proposes a scalable algorithm based on a decomposition method for hard-to-solve instances of conic optimization problems. The solution involves a column generation method that deconstructs a Second-Order Cone formulation of the feature selection problem. It regularizes the coefficients using penalization parameters and has been proven to be equivalent to LASSO. The method can solve medium-to-large scale instances (thousands of features and observations) in a matter of seconds, whereas other exact optimization alternatives take minutes or even hours.

The method is similar to a Dantzig-Wolfe approach. It solves two subproblems in each iteration: (1) the original Second-Order Cone Problem on a subset of its feasible region, and (2) a relaxed problem obtained via Lagrange relaxation of the challenging conic constraints. The relaxed problem provides new solutions to expand the master problem's feasible region. In this work, the column generation method is adjusted by considering cases of unbounded relaxed problems, and constructing artificial solutions when that is the case. These artificial solutions take into account the characteristics of the unbounded problems.

The proposed method can efficiently solve medium-to-large instances with substantial penalization within a couple of minutes, while the original problem and LASSO require over 20 minutes to produce a solution. However, the proposed method exhibits slower performance in some cases. Specifically, the decomposition method outperforms the execution time of the original problem when selecting 40% or fewer features, while it outperforms LASSO when selecting 4% or more features.

Para mi Nana y mis hermanas

Agradecimientos

En primer lugar, quiero agradecer a mis padres, con quienes tengo muchas diferencias en múltiples aspectos de la vida, pero aún así siempre dan más de lo necesario por mí. A mis dos hermanas, quienes han sido mi soporte y refugio en más de una ocasión. A mi abuela (la Nana), que en paz descanse, a mi madrastra y a mis tíos, quienes sin tener ninguna obligación me recibieron en su casa por años, en distintas épocas, y sigo recibiendo su cariño hasta el día de hoy. A mis dos hermanos, quienes formaron su familia antes de lo que puedo recordar, por lo que no hemos compartido demasiado, pero aún así se han preocupado por mí. El Cristian incluso me recibió medio año en su casa cuando era un niño.

Quiero agradecer también a Camila y Maite, dos personas que fueron muy importantes en distintos momentos de mi vida. Me apoyaron, creyeron en mí, me incentivaron y me regalaron su cariño. También agradecer a Alejandro, Carlos, Cristóbal y Vicente, quienes son mis amigos del INBA, los CHP, con quienes he compartido de todo, me he reído más de lo que puedo dimensionar y con quienes puedo ser yo mismo. A mis amigos de industrias, Androli, Caro, Gonzalo, Mariano, Pavlo, Tomás, Yordano, quienes fueron un pilar fundamental durante la pandemia y lo siguen siendo hasta el día de hoy. Especialmente Pavlito, con quién he compartido música, carrera, magíster, investigación, escalada y mucho más. Al Branco, Perroni y Sopapo también, aunque estemos en caminos muy distintos. Al Cucho y al Juanma, por la motivación y las risas. A NotGroup, Ninijo y a todos mis amigos de la U.

Al profe Fernando, mi profe guía, quien me ha enseñado, aconsejado, recomendado y hasta me ha conseguido trabajo. También al profe Charles, que me abrió las puertas al mundo de la investigación, me mostró lo difícil que son los problemas reales y me motivó a meterme al magíster. A Renaud y al profe Ricardo por su apoyo en la tesis y en investigación, respectivamente. A Linda Valdés por su tremenda ayuda con el proceso de titulación.

Table of Content

List of Tables

List of Figures

Chapter 1

Introduction

Feature selection is a critical element in the field of machine learning and data analysis. Its goal is to identify the most informative or relevant features within a given set when attempting to explain the value of certain feature of interest. Feature selection has gained significant importance in recent decades due to the exponential increase in data usage in various domains, which has also led to more diverse and larger datasets. The ability to effectively select relevant features not only enhances the performance and interpretability of machine learning models but also addresses challenges such as high-dimensional data, computational complexity, and overfitting.

Given the importance of this technique, over time, there have been many proposals on how to carry out this task. Well-known classic examples of this technique in the field of data analysis include the use of genetic algorithms employed by Holland (1975), the information gain criterion proposed by Quinlan (1986), the LASSO (Least Absolute Shrinkage and Selection Operator) method by Tibshirani (1996), and the sequential feature selection by Kohavi & John (1997).

The following chapters focus on a specific subcategory of these methods: those that perform feature selection with exact optimization while simultaneously minimizing the sum of squared errors. Among the methods mentioned earlier, the only one that falls into this category is LASSO, which minimizes the sum of squared errors subject to the constraint that the ℓ^1 norm of the weights assigned to each feature is less than a certain threshold. Other exact optimization models have been proposed for this problem, with mechanisms relatively similar to LASSO. An example of this is the model described by Bertsimas et al. (2016), where instead of constraining the magnitudes of the coefficients associated with each feature, penalties are imposed on the nonzero coefficients. In other words, it solves the *best-subset selection problem* while minimizing the sum of squared errors. The problem with this modeling approach is that it involves integer variables, so to solve the problem in competitive times, the feature set should be of small to medium size. It can handle problems with thousands of observations and hundreds of features, solving them in a matter of minutes.

This work aims to address instances of the feature selection problem with at least thousands of features and tens of thousands of observations (medium-to-large size instances) in a matter of minutes. To develop an efficient algorithm to solve problems of this size, this text proposes

the following steps.

1. Formulate a simple relaxation form of the Mixed-Integer Quadratic Problem addressed by Bertsimas et al. (2016), which can minimize squared errors and select relevant features, but in considerably less time than the initial problem.
2. Propose a conic formulation of the relaxed problem, to make use of the nonlinear decomposition method proposed by Chicoisne (2023). The objective of the decomposition method is to solve the problem in less than 10% of the original time for instances of medium-to-large size.
3. Conduct various experiments to validate the efficacy of the proposed method in generating meaningful solutions, comparing it against the conic formulation of the relaxed problem as well as other classic exact optimization methods for feature selection.

Different approaches for the second step, and further analysis of each of these approaches, are discussed in the methodology, available in Chapter 3. The results for the different alternatives proposed, with different parameter configurations, are presented in Chapter 4. The selected decomposition method excels at efficiently solving medium-to-large problems, outperforming the original conic relaxation and the LASSO method in several instances.

Chapter 2

Theoretical Framework

2.1 Feature Selection Problem

The feature selection problem is a process that aims to reduce the quantity of features of a given dataset. In other words, this method selects a subset of relevant features (according to a certain metric) from an original set of them, which may include features with no relevance at all. Features that are not part of the aforementioned relevant subset are removed from the problem or the working database. This technique often leads to improved results in various aspects, including a decrease in computational costs, enhanced accuracy of machine learning models, and greater interpretability when analyzing results (Miao & Niu, 2016).

This process has become a necessity today in various fields due to the exponential growth in the size (both in quantity and dimensionality) of data collected and stored each year. This is primarily attributable to the ever-increasing volume of information gathered from various sources, including various types of sensors, surveillance images, videos, social networks, and many others. The challenges associated with this rapid data growth have been studied for decades across various fields, including astronomy, statistics, optimization, and other disciplines.

In his book, "Dynamic Programming" (Bellman, 1957), Richard Bellman coined the term "the curse of dimensionality" to describe one of the challenges that arise in problems with high-dimensional databases. With this term, he explains that as the dimensionality of the data increases, the number of observations required to achieve good coverage (representation) of the feature space increases exponentially. One consequence of insufficient coverage of the feature space today is that many machine learning techniques may struggle to learn and effectively generalize to new datasets (see Figure 2.1). In this context, reducing data dimensionality decreases the number of features without diminishing the number of available observations, resulting in a more effective representation of the final feature space. Therefore, methods that reduce the dimension of a problem emerge as a viable alternative to address this "curse".

Additional challenges are linked to a high number of features within a dataset, which can be further explored in this section. Nonetheless, the phenomena detailed previously suffice

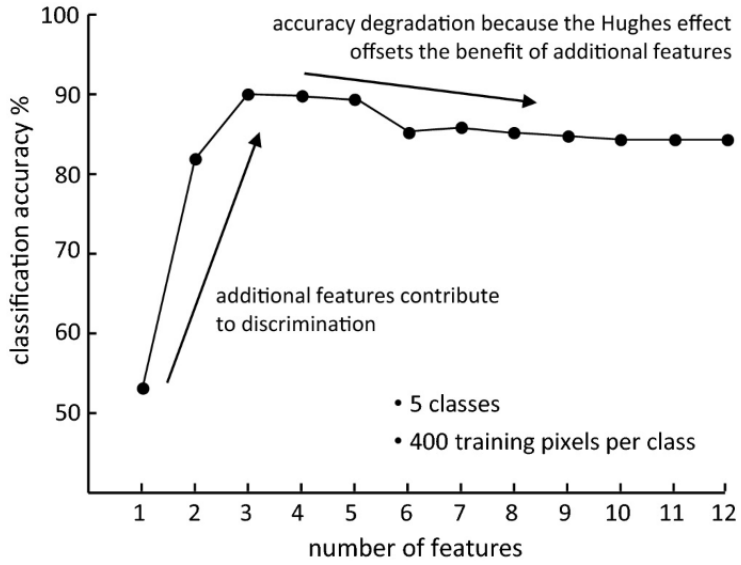

Figure 2.1: Detailed graph of the "Hughes' Phenomenon" from the book "Supervised Classification Techniques" by Richards (2022). The accuracy of a model improves with more features up to a point, but beyond that, an increase in dimensionality results in a decrease in model accuracy.

to underscore the relevance of feature selection methods nowadays.

Regarding existing methods, as can be inferred from the previous paragraphs, there is a wide variety of them. According to Miao & Niu (2016), feature selection methods can be classified in two ways: in terms of availability of label information data and also by the different strategies of searching the most relevant features.

2.1.1 Feature Selection Methods in Terms of Availability of Label Information

Miao & Niu (2016) classified feature selection methods into three families: supervised methods, unsupervised methods, and semi-supervised methods. The following provides a description of these three categories and an overview of some of the most relevant methods within each group.

1. **Supervised methods:** These methods rely on the availability of labeled data and use the target variable to guide the selection of relevant features. They assess the relationship between features and the target variable, considering the predictive power of each feature with respect to the outcome of interest.

   The work presented in this document focuses on supervised methods providing an exact optimization solution for regression problems. Examples of this subcategory include LASSO (Tibshirani, 1996) and Elastic Net (Zou & Hastie, 2005). Both methods minimize the sum of squared errors between the objective variable and a linear combination of all available features, while imposing a penalty on the number of selected features or their magnitude.

Other common examples of supervised methods include those constructing a Feature Ranking (Akman et al., 2023). These methods rely on a pre-defined metric to rank features, from which the selection is performed. The scoring process may involve measuring the correlation of each feature with the independent variable (Cui et al., 2010), setting a certain threshold as in Minimum Variance Thresholding (Hou et al., 2006), performing a Chi-squared test (Pearson, 1900), using Recursive Feature Elimination (Guyon et al., 2002), and more.

2. **Unsupervised methods:** These techniques perform feature selection by relying solely on the structure of the data, without considering any specific outcome or target variable.

   Examples include the widely known Principal Component Analysis (PCA) (Rahmat et al., 2023), clustering methods such as k-means (Khaleel, 2011), and autoencoders (Xu et al., 2019).

3. **Semi-supervised methods:** These approaches make use of a combination of labeled and unlabeled data to select features, identifying patterns and structures within the dataset to make decisions about feature relevance. Semi-supervised methods are especially useful when labeled data is limited or expensive to obtain.

   A wide variety of semi-supervised feature selection methods exists, with their classification within this family as described by Sheikhpour et al. (2017). These methods go beyond the scope of this work.

2.1.2 Feature Selection Methods According to the different Strategies of Searching the Most Relevant Features

Miao & Niu (2016) also classify feature selection methods into three search strategies to look for the most relevant features in the dataset. The following provides a description of these three categories and an overview of some of the most relevant methods within each group.

1. **Filter Methods:** These methods utilize statistical metrics to assign a score to each of the available features, independently of the method or model used to solve the original problem. They are useful for large-scale problems, as they are often efficient compared to other types of feature selection methods.

   Common filter methods include Correlation-based methods (Cui et al., 2010), Chi-squared test (Pearson, 1900) , and Variance thresholding (Hou et al., 2006).

2. **Wrapper Methods:** These methods use the algorithm or model applied to address the original problem, typically a prediction task, to obtain information and evaluate different features based on their results. In other words, these methods test models with different subsets, measuring their predictive performance in each case to determine the ultimate feature space.

   Common wrapper methods include Forward selection (Aha & Bankert, 1996), that starts with an empty subset of features and incorporates the one that performs best regarding the prediction of the target variable. It iteratively adds features that improve performance until the model's performance decreases (stopping criterion). On the

opposite, there is also Backward elimination (Aha & Bankert, 1996), that begins with the subset of features containing them all, then removes less relevant features (whose removal improves model performance) until the elimination of a feature does not yield any performance gain.

3. **Embedded Methods:** These methods include feature selection within the same model that solves the original problem. They are not independent stages or concepts in the whole process.

   Common embedded methods are LASSO (Tibshirani, 1996), Elastic Net (Zou & Hastie, 2005), and feature selection methods based on decision trees (Chen et al., 2020). In all these methods, feature selection is performed simultaneously with problem solving. They are part of the model itself.

As can be inferred from the previous subsection, this work focuses on embedded methods, using exact optimization as a prediction and feature selection model at once. Subsection 2.4.1 introduces notation and other details associated with the models of interest.

2.2 Exact Optimization

To understand what feature selection methods with exact optimization are, one must first define the concept of exact optimization. According to Rothlauf (2011), optimization methods can be broadly categorized into two groups: exact optimization methods, which guarantee the discovery of an optimal solution (either the minimum or maximum solution achievable for a given problem), and heuristic optimization methods, which cannot ensure an optimal solution, even though they may occasionally found one. The demand for optimization heuristics arises from the complexities involved in solving problems using exact optimization methods. There exist problems of such complexity in their feasible space, that the execution times required to identify an optimal solution render exact solution methods impractical. Heuristics take advantage of the structure of a problem to locate solutions that closely approximate the optimal solution, and this can be accomplished in significantly less time than through exact optimization. Nevertheless, certain problem contexts need precision, leaving no room for errors, and in such cases, exact optimization methods become indispensable. Furthermore, exact optimization methods offer a baseline for assessing and determining the efficiency of heuristics in discovering solutions that closely approximate the optimal outcome. Hence, it is almost always imperative to model and solve the exact optimization method.

2.2.1 Exact Optimization Problems

In general, an exact optimization problem can be defined in several ways. Using the notation of Boyd & Vandenberghe (2004), an optimization problem has the form

$$\begin{aligned} \min_{x \in \mathbb{R}^n} \quad & f_0(x) \\ \text{subject to} \quad & f_i(x) \leq b_i, \quad i = 1, \ldots, m. \end{aligned} \tag{2.1}$$

In this formulation, $\min_x \ f_0(x)$ refers to the minimization problem of the function $f_0 : \mathbb{R}^n \to \mathbb{R}$ with respect to the variable vector $x = (x_1, \ldots, x_n) \in \mathbb{R}^n$. The function f_0 is the objective function of the problem, the functions $f_i : \mathbb{R}^n \to \mathbb{R}$, $i = 1, \ldots, m$, are the constraint functions that the problem is subject to, and $b_1, \ldots, b_m$ are the bounds of these constraints. Note that any optimization problem with a scalar objective function can be written in this form. In the case of problems aiming to maximize a function, it suffices to take f_0 as the negative of that function. Regarding inequality constraints, a similar principle can be applied to express all constraints as in (2.1). In the case of equality constraints, they can be bounded from both above and below by the same value.

To further generalize the concept of an optimization problem and simplify the notation, it can be expressed as

$$\begin{aligned} \min_{x} \quad & f_0(x) \\ \text{s. t.} \quad & x \in \mathcal{X}, \end{aligned} \tag{2.2}$$

where $\mathcal{X}$ represents the intersection of all constraints applied to x, including equalities, inequalities, and, in general, any set in which x must reside. This set is commonly referred to as the feasible region.

A vector $x^* \in \mathbb{R}^n$ that satisfies $x^* = \text{argmin}_{x \in \mathcal{X}} f_0(x)$ is referred to as the optimal solution of the problem or simply the optimum. Algorithms designed to solve these optimization problems (i.e., search for x^*) exploit the specific structure of each problem. There are various methods and criterion to verify the optimality of a particular point. One of the most extensively studied problem types, featuring numerous criteria and solution algorithms, is convex optimization. All exact optimization problems addressed in this work belong to this category.

2.2.2 Convex Optimization Problems

Based on the definition provided by Boyd & Vandenberghe (2004), in Chapter 4, pages 136-146, a convex optimization problem can be expressed as

$$\begin{aligned} \min_{x \in \mathbb{R}^n} \quad & f_0(x) \\ \text{s. t.} \quad & f_i(x) \leq 0, \quad i = 1, \ldots, m \\ & h_i(x) = 0, \quad i = 1, \ldots, p. \end{aligned} \tag{2.3}$$

Here, $f_0, \ldots, f_m : \mathbb{R}^n \to \mathbb{R}$ are convex functions, and $h_1, \ldots, h_p : \mathbb{R}^n \to \mathbb{R}$ are affine functions. This definition requires convexity in the objective function and in functions that should take negative values. Furthermore, it imposes that equality constraints, depending on x, should be affine. However, this definition does not require that all these constraints necessarily exist. In other words, problems like unconstrained convex objective functions, linear problems, and problems with only inequality constraints less than zero that meet convexity are also convex optimization problems.

Numerous advantages exist when solving convex optimization problems, including:

1. The feasible region $\mathcal{X}$ of a convex problem is a convex set, meaning that any local minimum of the problem is necessarily a global optimum. Therefore, there's no possibility of algorithms getting stuck in some local minima.
2. As a consequence of the previous point, these problems, especially when $f_0, \ldots, f_m$, and $h_1, \ldots, h_p$ are differentiable functions, can be addressed using algorithms such as gradient descent (Curry, 1944), the Newton method (Fletcher, 2013), and various derivatives of these techniques.
3. In certain instances, optimality conditions can be utilized to determine an optimal solution of a convex problem. A notable example of this is the sufficiency of the Karush–Kuhn–Tucker conditions to characterize an optimal solution for a differentiable convex problem (Karush, 1939).

2.2.3 Lagrange Relaxation of an Optimization Problem

Lagrange relaxation is an important concept to understand the different models that are detailed in the next chapter. It is detailed in Lemaréchal (2001), where the optimization problem involves the maximization of an objective function subject to equality constraints. In this work, however, the focus is on problems with constraints of all types. Boyd & Vandenberghe (2004), Chapter 2, pages 215–223, define the Lagrange function of a problem as the sum of the objective function and weighted sums of the problem's constraints in the form

$$L(x, \lambda, \nu) = f_0(x) + \sum_{i=1}^{m} \lambda_i f_i(x) + \sum_{i=1}^{p} \nu_i h_i(x), \tag{2.4}$$

where λ_i is the Lagrange multiplier associated with the constraints of the form $f_i(x) \leq 0$, and similarly, ν_i is the Lagrange multiplier for the constraints of the form $h_i(x) = 0$.

The common aim when using this function is to optimize the objective function of a problem, either minimizing or maximizing it, while also penalizing any violation of the problem's constraints using the parameters λ and ν. This approach often simplifies the problem-solving process, making it more tractable compared to directly dealing with the original problem. The introduction of the Lagrange multipliers λ and ν allows the incorporation of the constraints as penalties in the objective function, enabling the optimization algorithms to handle the problem in a more efficient manner.

2.2.4 Lagrange Dual Problem

Another important function that stems from the Lagrange function is the Lagrange dual function. This function is the infimum value taken by the Lagrange function $L(x, \lambda, \nu)$ with respect to x, the original variables of the problem. This is, for any value of (λ, ν), the Lagrange dual function is

$$g(\lambda, \nu) = \inf_{x \in \mathbb{R}^n} L(x, \lambda, \nu) = \inf_{x \in R} \left(f_0(x) + \sum_{i=1}^{m} \lambda_i f_i(x) + \sum_{i=1}^{p} \nu_i h_i(x) \right). \tag{2.5}$$

Note that this function may diverge if it is unbounded with respect to x. In this case, the Lagrange dual function takes the value $-\infty$. The Lagrange dual function also satisfies the following property: for any vector $\lambda \succeq 0$ (with all its components positive),

$$g(\lambda, \nu) \leq f_0(x^*). \tag{2.6}$$

Here x^* is the optimum solution of the original problem. This result is straight forward, as x* satisfies the problem's constraints $f_i(x^*) \leq 0$ and $h_i(x^*) = 0$, which, along with the condition $\lambda \succeq 0$, yields $L(x^*, \lambda, \nu) \leq f_0(x^*)$. Furthermore, since $g(\lambda, \nu)$ takes the infimum with respect to x of $L(x, \lambda, \nu)$, it follows that

$$g(\lambda, \nu) = \inf_{x \in \mathbb{R}^n} L(x, \lambda, \nu) \leq L(x^*, \lambda, \nu) \leq f_0(x^*). \tag{2.7}$$

Thus, the Lagrange dual function serves as a lower bound of the optimum objective value of the original problem. Since $g(\lambda, \nu)$ is a lower bound of the optimum value of the objective function $f_0(x^*)$, it is also a lower bound of any feasible point x of the original problem

$$g(\lambda, \nu) \leq f_0(x). \tag{2.8}$$

This relationship is known as "weak duality". Considering that the lower bound holds for any $\lambda \succeq 0$ and $\nu \in \mathbb{R}^p$, one can solve the alternative problem

$$\begin{aligned} \max_{\lambda \in \mathbb{R}^m,\, \nu \in \mathbb{R}^p} \quad & g(\lambda, \nu) \\ \text{s. t.} \quad & \lambda \succeq 0, \end{aligned} \tag{2.9}$$

which is referred to as the Lagrange dual problem of the optimization problem. Its optimal solution (λ^*, ν^*) also satisfies $g(\lambda^*, \nu^*) \leq f_0(x^*)$, which can even lead to equality in certain cases. When equality can be achieved for some optimization problem, it is said that "strong duality" holds for that problem.

2.2.5 Particular Case: Conic Constrained Problems

Before introducing conic constrained problems, here are some definitions:

1. A set C is a **cone** if, for every element $x \in C$ and any non-negative scalar $\theta \geq 0$, it holds that $\theta x \in C$. In other words, a set is a cone if all non-negative scalings of its elements also belong to it.

2. A set C is a **convex cone** if, for every pair of elements $x_1, x_2 \in C$ and any pair of non-negative scalars $\theta_1, \theta_2 \geq 0$, it holds that $\theta_1 x_1 + \theta_2 x_2 \in C$.

3. A set C is a **proper cone** if it is a convex cone that is closed, has a non-empty interior, and does not contain any lines.

4. The **dual cone** of a cone C is a set C^* whose elements y satisfy the condition that the inner product between y and any element $x \in C$ is always non-negative. This is

$$C^* = \{y : \langle x, y \rangle \geq 0, \forall x \in C\}, \tag{2.10}$$

   where $\langle x, y \rangle = x \cdot y = \sum_i x_i y_i$ is the **inner product** between vectors x and y.

In order to extend the definitions from the previous subsections to encompass more types of problems, we introduce **conic constrained problems** in the form

$$\begin{aligned} \min_{x \in \mathbb{R}^n} \quad & f_0(x) \\ \text{s. t.} \quad & -f_i(x) \in K_i, \quad i = 1, \ldots, m \\ & h_i(x) = 0, \qquad i = 1, \ldots, p, \end{aligned} \tag{2.11}$$

where $f_i : \mathbb{R}^n \to \mathbb{R}^{k_i}$ and $K_i \subseteq \mathbb{R}^{k_i}$ are proper cones.

Similar to the problems presented in the previous subsections, the Lagrange function for this type of conic constrained problem can be defined as

$$L(x, \lambda, \nu) = f_0(x) + \sum_{i=1}^{m} \langle \lambda_i, f_i(x) \rangle + \sum_{i=1}^{p} \nu_i h_i(x), \tag{2.12}$$

The difference compared to the definition (2.4) is that each $f_i(x)$ is associated with a vector $\lambda_i \in K_i^*$, where the star symbol denotes the dual cone of K_i. Therefore, if x is a feasible point of problem (2.11), it satisfies $-\langle \lambda_i, f_i(x) \rangle \geq 0$. Following the same arguments as in subsection 2.2.3, for feasible points of the conic constrained problem, the relaxation (2.12) is a lower bound of the objective function of this problem at the same point. Consequently, everything derived from the Lagrange relaxation, such as the dual Lagrange problem, is analogous for conic constrained problems.

2.3 Large-Scale Optimization

Just as handling large datasets poses a challenge in various ways for different types of problems, a large number of variables or constraints in an optimization problem increases its complexity, particularly in computational terms. Solution methods to address large-scale problems involve developing techniques and algorithms that provide a solution that is sufficiently close or identical to the optimal solution of the original problem within a reasonable execution time.

Some techniques for solving large-scale problems include decomposition methods, heuristics, approximation algorithms, among others. Naturally, there exists a trade-off between the quality of the solution of these methods and their computational efficiency. This paper will focus solely on decomposition methods.

Decomposition methods are optimization techniques that involve breaking down a large-scale optimization problem into smaller subproblems, which can, in principle, be solved more efficiently than the original problem. Each of these subproblems is solved separately, and the solutions they generate serve as additional information for the rest. These different solutions together form a framework capable of providing the solution to the large-scale optimization problem.

2.3.1 Decomposition Methods for Linear Problems

Decomposition methods for solving large-scale linear optimization problems have been studied for decades. The following subsections present two important decomposition methods.

2.3.1.1 Benders Decomposition

The method proposed by Benders (1962) is a technique for solving large-scale problems that follow a block structure. That is, the variables and constraints can be separated into two general groups, as can be seen in the formulation

$$\begin{aligned} \min_{x,y} \quad & c'x + d'y \\ \text{s. t.} \quad & Ax + By \geq b \\ & y \in Y \\ & x \geq 0 \end{aligned} \qquad , \tag{2.13}$$

where A and B are matrices with the same number of rows, and the same number of columns as the dimension of x and y, respectively, while the vector b is a bound for the relationship between the two variables. Y is the set that determines the nature of y, and c and d are cost vectors. The only term that links both variables is the first constraint, so without it,

they could be treated as two independent problems, each with its own variable. A similar idea is applied in this method, solving two types of problems in different iterations, each associated with a different variable, which provide lower and upper bounds for the original problem. Generally, the two types of problems are a master problem and a subproblem. In each iteration of the method, a new subproblem is solved, and its solution determines a new constraint to add to the master problem.

Benders Decomposition is commonly used in stochastic programming, for instance, when there is a vector of random variables for which there are different possible values (scenarios), and another vector of deterministic variables, whose optimal value is affected by the different possible scenarios of the random variables. Following this same line, it is a method widely used to solve multilevel optimization problems (Vicente & Calamai, 1994).

2.3.1.2 Dantzing-Wolfe

The method proposed by Dantzig & Wolfe (1961) is a column generation method for large-scale linear problems, taking advantage of the fact that in an optimal solution, many variables often take on zero values. Therefore, by selecting only a subgroup of variables initially and then iteratively adding them in an intelligent manner, the algorithm can typically attain the solution in less time. Similar to the previous method, there is a master problem that solves the problem with all the constraints, but only considering a subset of variables. It uses different subproblems to choose new solutions with new variables to enter the solution's base if they improve the objective function of the problem. Given its form, this algorithm is useful for problems with a large number of variables, where a considerable number of them are zero in the optimal solution, similar to what is required in a feature selection method. Because of this, the method proposed in this paper has significant similarities with this decomposition method.

2.3.2 Decomposition Methods for Nonlinear Problems

There are decomposition methods to solve nonlinear problem which are pretty similar to those used in linear optimization problems. In fact, the same Benders Decomposition method has been extended to solve nonlinear problems (Karbowski, 2021). In addition to this method, there are alternatives such as decomposition methods that make use of the Lagrange relaxation of a large-scale optimization problem, as reviewed in the book by Nowak (2005). As an alternative for various types of nonlinear optimization problems, a new method of column generation and Lagrange relaxation for large-scale optimization problems with conic constraints is described in subsection 2.4.3, and is the one that is used through all of the present work.

2.4 State of the Art

2.4.1 Feature Selection Methods with Exact Optimization

This section details some of the most common feature selection methods with exact optimization. These approaches share a common logic and, in certain instances, leverage the Lagrange relaxation of the problem rather than the optimization problem with constraints.

2.4.1.1 LASSO

The first method under review is the well-known LASSO, proposed by Tibshirani (1996), which remains one of the most widely employed techniques for regularization and feature selection in regression problems. The original optimization problem aims to minimize the squared error, just like Ordinary Least Squares, while constraining the coefficients using a predetermined parameter. Given a matrix $X \in \mathbb{R}^{n \times m}$ of m features and n observations, a vector $y \in \mathbb{R}^n$ of n independent variables to predict, and a vector of coefficients to estimate $\beta \in \mathbb{R}^m$, LASSO addresses the problem

$$\begin{aligned} \min_{\beta \in \mathbb{R}^m} \quad & \|y - X\beta\|_2^2 \\ \text{s. t.} \quad & \|\beta\|_1 \leq t, \end{aligned} \tag{2.14}$$

where $\|\cdot\|_p$ is the ℓ^p norm: $\|\beta\|_p = \left(\sum_{j=1}^m |\beta_j|^p\right)^{1/p}$.

An equally used alternative form of the LASSO method involves solving an equivalent problem with a penalization parameter λ for the ℓ^1 norm of the coefficients, formulated as

$$\min_{\beta \in \mathbb{R}^m} \quad \|y - X\beta\|_2^2 + \lambda \|\beta\|_1. \tag{2.15}$$

This problem directly derives from the Lagrange relaxation of (2.14) by fixing the values of the coefficient t.

2.4.1.2 Elastic Net

With a similar formulation, inspired by both LASSO and Ridge regression, Zou & Hastie (2005) propose the Elastic Net, which consists of an unconstrained optimization with the same form as 2.15, with an additional penalty term for the squared ℓ^2 norm of the coefficient vector β. In other words, it solves

$$\min_{\beta \in \mathbb{R}^m} \quad \|y - X\beta\|_2^2 + \lambda_1 \|\beta\|_1 + \lambda_2 \|\beta\|_2^2. \tag{2.16}$$

The penalty term associated with the ℓ^2 norm aims to decrease the coefficient values, emphasizing those with larger magnitudes, as it penalizes the square of each coefficient value.

2.4.2 Feature Selection with Conic Optimization

There are various approaches to linear models with conic constraints in contemporary research. Bertsimas et al. (2016) propose a mixed integer formulation for the "Best Subset Selection Problem", aiming to minimize the squared error of a linear model. The inclusion of constraints with binary variables facilitates the selection of linear model coefficients, imposing a maximum limit on the number of coefficients chosen, akin to the original concept of LASSO. In the same article, the authors introduce different conic variations like those of the aforementioned problem to derive lower bounds for the original model. In Chapter 3, this model is examined as an initial reference for addressing features selection problems with conic constraints.

With a relatively similar group of constraints as the aforementioned model, Kucukyavuz et al. (2020) propose a Mixed Integer Second-Order Cone Program (MISOCP), modifying the constraints that bound the coefficients of the linear model with a property that allows them to be written as second-order cones. Additionally, they incorporate a regularization parameter in the objective function, similar to the relaxation of the LASSO model, and another for penalizing each extra variable added to the model. This strategy is also reviewed in the methodology outlined in Chapter 3.

Finally, Schwendinger et al. (2022) propose a package for solving various types of linear models, "Generalized Holistic Linear Models", with a wide variety of possible conic constraints, such as second-order cones and exponential cones. These models deviate from the focus of this work and are therefore not reviewed.

2.4.3 Decomposition Method for Conic Optimization Problems

Chicoisne (2023) proposes a decomposition method for large-scale nonlinear problems of the form

$$\begin{aligned} (P(\mathcal{X})) \quad \omega(\mathcal{X}) = \min_{x,y} \quad & f(x,y) \\ \text{s. t.} \quad & x \in \mathcal{X}, \\ & -g(x,y) \in \mathcal{C}, \end{aligned} \tag{2.17}$$

where $x \in \mathcal{X}$ are the main variables of the problem, with $\mathcal{X}$ being a high-dimensional set, the vector y is an auxiliary variable, $\mathcal{C}$ is a cone in some Euclidean space, and f and g are

some functions from a vector space to scalar values. The initial assumption of the proposed decomposition method is that the difficulty of problem $P(\mathcal{X})$ derives from two sources: first, the existence of conic constraints $-g(x, y) \in \mathcal{C}$, and second, the high dimensionality of $\mathcal{X}$. This means that either the problem can be efficiently solved on a low-dimensional subset $\mathcal{S} \subseteq \mathcal{X}$, or it can be efficiently solved by eliminating the conic constraints $-g(x, y) \in \mathcal{C}$.

Based on the above idea, two types of alternative problems can be generated. The first is $P(\mathcal{S})$, which has the same form as (2.17), but with $x \in \mathcal{S} \subseteq \mathcal{X}$. This problem, being an optimization problem on a subset of the feasible region of $P(\mathcal{X})$, necessarily yields an optimal objective value that is greater than or equal to that of the latter, $\omega(\mathcal{X}) \leq \omega(\mathcal{S})$. On the other hand, the second type of problem can be generated with the Lagrange relaxation of the conic constraints, in the form

$$\begin{aligned} (L(\mathcal{X}, \lambda)) \quad \omega(\mathcal{X}, \lambda) = \min_{x,y} \quad & f(x, y) + \langle \lambda, g(x, y) \rangle \\ \text{s. t.} \quad & x \in \mathcal{X}, \end{aligned} \tag{2.18}$$

with $\lambda \in \mathcal{C}^*$, where $\mathcal{C}^*$ is the dual cone of $\mathcal{C}$. Thus, for feasible solutions x, y of $P(\mathcal{X})$, it holds that $\langle \lambda, g(x, y) \rangle \leq 0$, $\forall \lambda \in \mathcal{C}^*$. Therefore, since $\omega(\mathcal{X}, \lambda)$ is the optimal value obtained from the relaxation by minimizing over x, y, in particular, this value will be less than or equal to the optimal objective value of $(P(\mathcal{X}))$; that is, $\omega(\mathcal{X}, \lambda) \leq \omega(\mathcal{X})$, $\forall \lambda \in \mathcal{C}^*$. With this, a lower bound and an upper bound for problem (2.17) is obtained. These bounds are obtained by solving $P(\mathcal{S})$ and $L(\mathcal{X}, \lambda)$, which are easier to solve if the initial assumptions of this problem are met.

An iterative algorithm can be constructed to leverage the bounds provided by these two problems that bound the original problem $P(\mathcal{X})$. This algorithm can generate various subsets $\mathcal{S}^k$ that, in turn, produce new solutions (which may not necessarily be feasible in the original problem) derived through $L(\lambda, \mathcal{X})$. By solving $P(\mathcal{S}^k)$ for each of these subsets, feasible solutions can be obtained, potentially leading to the optimal solution of $P(\mathcal{X})$. This is precisely the idea of Algorithm 1, as proposed by Chicoisne (2023).

The stopping criteria of the algorithm can be summarized in the idea that the new solution $\bar{x}^k$, which should improve the objective function, did not make any difference in the optimal solution of the subproblem $P(\mathcal{S}^k)$. Consequently, the optimal solution derived from $P(\mathcal{S}^k)$ cannot be further improved. Therefore, it is the optimal solution of the problem $P(\mathcal{X})$.

Algorithm 1: Column Generation

Data: A problem $P(\mathcal{X})$
Result: An optimal solution for $P(\mathcal{X})$

Set $\lambda^0 = 0$, $\mathcal{S}^1 \subseteq \mathcal{X}$ contains at least one feasible solution for $P(\mathcal{X})$, and $k = 1$;
while *True* **do**
    Solve $P(\mathcal{S}^k)$. Let (x^k, y^k) be an optimal solution ;
    Let λ^k be an optimal dual vector corresponding to the constraints $-g(x, y) \in \mathcal{C}$;
    if $\lambda^k = \lambda^{k-1}$ **then**
        return (x^k, y^k) ;
    Solve $L(\mathcal{X}, \lambda^k)$. Let $(\bar{x}^k, \bar{y}^k)$ be an optimal solution ;
    if $\bar{x}^k \in \mathcal{S}^k$ **then**
        return (x^k, y^k) ;
    Choose a set $\mathcal{S}^{k+1} \subseteq \mathcal{X}$ containing $\bar{x}^k$;
    $k \leftarrow k + 1$;

Chapter 3

Methodology

For the following sections, it is necessary to consider different aspects of notation.

1. To represent data sets with n observations and $m \gg 1$ attributes or columns, the notation will be:
   (a) $y \in \mathbb{R}^n$: Target vector of observations to be predicted with the linear model.
   (b) $X \in \mathbb{R}^{n \times m}$: Matrix with different observations in its rows and attributes in its columns.
2. If a model does not explicitly state the set to which a variable x belongs, it is assumed to belong to $\mathbb{R}^q$, where q is some scalar coherent with the operations applied to that variable. In general, the variable $\beta \in \mathbb{R}^m$ is used in all models to represent the coefficients associated with each attribute of the linear model.
3. A sufficiently large parameter $M \gg 1$ is used in constraints that require bounds for the model's variables in certain scenarios, while in others, such bounds are not required.
4. Regarding the notation for the vectors used, the column vectors are denoted as $x = \begin{pmatrix} x_1 \\ \vdots \\ x_q \end{pmatrix} \in \mathbb{R}^q$. The apostrophe is used to represent the transpose of a column vector, a row vector, in the form $x' = \begin{pmatrix} x_1 & \dots & x_q \end{pmatrix} \in \mathbb{R}^{1 \times q}$. Consequently, the inner product between two vectors x and y in a Euclidean space can be written as $\langle x, y \rangle = x'y = \sum_{i=1}^{q} x_i y_i$. The same symbol is used for transposing matrices.

3.1 Baseline Model: Mixed Integer Quadratic Program (MIQP)

To establish the foundation for the models presented in the following subsections, we revisit the mixed integer quadratic problem proposed by Bertsimas et al. (2016), mentioned in 2.4.2.

In this model, the best subset of attributes is chosen, restricting the number of coefficients $\beta_i \neq 0$ to a maximum of k possible in the final solution, as follows

$$\begin{aligned}
\min_{\beta,z} \quad & \|y - X\beta\|_2^2 && && (3.1a)\\
\text{s.t.} \quad & -Mz_i \leq \beta_i \leq Mz_i, && i = 1, \ldots, m, && (3.1b)\\
& \sum_{i=1}^{m} z_i \leq k, && && (3.1c)\\
& z \in \{0,1\}^m, && && (3.1d)
\end{aligned}$$

Because of the binary nature of z, the constraint (3.1c) can be rewritten as $\|z\|_1 \leq k$, which is the same form of the original constraint of LASSO (2.14). Hence, it can be relaxed in the same way as in (2.15), and the problem of choosing the best subset can be rewritten equivalently as

$$\begin{aligned}
\min_{\beta,z} \quad & \|y - X\beta\|_2^2 + \tau\|z\|_1 && && (3.2a)\\
\text{s.t.} \quad & -Mz_i \leq \beta_i \leq Mz_i, && i = 1, \ldots, m, && (3.2b)\\
& z \in \{0,1\}^m. && && (3.2c)
\end{aligned}$$

The equivalence between both problems depends on the values of the parameters k and τ, similar to the two alternatives of LASSO.

Considering the non-negativity of the components of the vector z, the ℓ^1 norm of this vector will be represented as a summation for better readability in the subsequent models.

Similar to the MISOCP proposed by Kucukyavuz et al. (2020), also mentioned in subsection 2.4.2, the two bounds imposed on the coefficients in (3.2b) can be rewritten in a single quadratic form constraint, resulting in problem

$$\begin{aligned}
\min_{\beta,z} \quad & \|y - X\beta\|_2^2 + \tau \sum_{i=1}^{m} z_i && && (3.3a)\\
\text{s.t.} \quad & \beta_i^2 \leq M^2 z_i, && i = 1, \ldots, m, && (3.3b)\\
& z \in \{0,1\}^m, && && (3.3c)
\end{aligned}$$

where $z_i = z_i^2$, as they are binary variables. In this model, the value of the parameter $M \gg 1$ can be adjusted to set a certain bound on the maximum value that the coefficients β_i can take, depending on the context of the problem. Slightly akin to the aforementioned MISOCP, this parameter can also be a new variable of the model rather than a parameter, to which some penalty κ can be assigned to bound the values of the coefficients β_i. Thus, an alternative model can be written as

$$\begin{aligned}
\min_{\beta,z,u} \quad & \|y - X\beta\|_2^2 + \tau \sum_{i=1}^{m} z_i + \kappa \sum_{i=1}^{m} u_i && && (3.4a)\\
\text{s.t.} \quad & \beta_i^2 \leq u_i z_i, && i = 1, \ldots, m, && (3.4b)\\
& u_i \geq 0, && i = 1, \ldots, m, && (3.4c)\\
& z \in \{0,1\}^m. && && (3.4d)
\end{aligned}$$

3.2 Relaxation: Quadratic Program (QP)

Given the mixed integer quadratic program, MIQP (3.4), it is possible to establish a relaxed version of this problem by removing the integer constraint on the vector of variables z, resulting in the model

$$\begin{aligned}
\min_{\beta,z,u} \quad & \|y - X\beta\|_2^2 + \tau \sum_{i=1}^{m} z_i + \kappa \sum_{i=1}^{m} u_i && && \text{(3.5a)}\\
\text{s.t.} \quad & \beta_i^2 \leq u_i z_i, && i = 1, \ldots, m, && \text{(3.5b)}\\
& u_i \geq 0, && i = 1, \ldots, m, && \text{(3.5c)}\\
& z_i \geq 0, && i = 1, \ldots, m. && \text{(3.5d)}
\end{aligned}$$

This problem still maintains the idea of constraining the magnitudes of the coefficients β and attempting to minimize their values through the penalty parameters τ and κ. The difference from the model with binary variables is that it is no longer a problem of selecting the best subset, as in the original MIQP (3.1).

Upon closer examination of the relaxed problem, it is easy to see that if $\tau = \kappa$, then the problem is symmetric with respect to z and u, as they have the same constraints and terms in the objective function. Furthermore, Proposition 1 establishes an equivalence between different problems, where this condition is met for one of them.

Proposition 1 $\forall \tau, \kappa > 0$, parameters of the relaxed QP (3.5), there exists an equivalent problem with the same structure, and parameters $\tilde{\tau}$ and $\tilde{\kappa}$, such that $\tilde{\tau} = \tilde{\kappa} = \sqrt{\tau\kappa}$.

Proof. Note that it is possible to express $\kappa = \alpha^2\tau$, with $\alpha = \sqrt{\kappa/\tau} > 0$, in problem (3.5). Therefore, the problem in question can be written as follows

$$\begin{aligned}
(P1) \quad \min_{\beta,z,u} \quad & \|y - X\beta\|_2^2 + \tau \sum_{i=1}^{m} z_i + \alpha^2\tau \sum_{i=1}^{m} u_i && \\
\text{s.t.} \quad & \beta_i^2 \leq u_i z_i, && i = 1, \ldots, m,\\
& u_i \geq 0, && i = 1, \ldots, m,\\
& z_i \geq 0, && i = 1, \ldots, m.
\end{aligned}$$

The candidate for an equivalent problem of the same form with equal penalty parameters is the problem

$$\begin{aligned}
(P2) \quad \min_{\beta,z,u} \quad & \|y - X\beta\|_2^2 + \alpha\tau \sum_{i=1}^{m} z_i + \alpha\tau \sum_{i=1}^{m} u_i && \\
\text{s.t.} \quad & \beta_i^2 \leq u_i z_i, && i = 1, \ldots, m,\\
& u_i \geq 0, && i = 1, \ldots, m,\\
& z_i \geq 0, && i = 1, \ldots, m.
\end{aligned}$$

It is straightforward to see that a feasible point of $(P1)$ is also a feasible point of $(P2)$, and vice versa, as both problems have the same feasible region. Only the objective function changes.

Let (β^*, z^*, u^*) and $(\tilde{\beta}^*, \tilde{z}^*, \tilde{u}^*)$ be optimal solutions of $(P1)$ and $(P2)$, respectively. Taking the new vectors $\bar{u} = \alpha u^*$ and $\bar{z} = \frac{1}{\alpha} z^*$, the objective function of $(P1)$ evaluated at its optimal point can be expressed as

$$\|y - X\beta^*\|_2^2 + \tau \sum_{i=1}^{m} z_i^* + \alpha^2 \tau \sum_{i=1}^{m} u_i^* = \|y - X\beta^*\|_2^2 + \alpha\tau \sum_{i=1}^{m} \bar{z}_i + \alpha\tau \sum_{i=1}^{m} \bar{u}_i.$$

It follows that $(\beta^*, \bar{z}, \bar{u})$ is also a feasible point of $(P2)$, since $\beta_i^* \leq u_i^* z_i^* = \bar{u}_i \bar{z}_i$ for every component of these vectors, and $\bar{z}, \bar{u}$ maintain the non-negativity of their components when multiplied by positive values. Thus, being a feasible point of the second problem, the following relationship holds

$$\begin{aligned} \|y - X\tilde{\beta}^*\|_2^2 + \alpha\tau \sum_{i=1}^{m} \tilde{z}_i^* + \alpha\tau \sum_{i=1}^{m} \tilde{u}_i^* &\leq \|y - X\beta^*\|_2^2 + \alpha\tau \sum_{i=1}^{m} \bar{z}_i + \alpha\tau \sum_{i=1}^{m} \bar{u}_i \\ &= \|y - X\beta^*\|_2^2 + \tau \sum_{i=1}^{m} z_i^* + \alpha^2 \tau \sum_{i=1}^{m} u_i^*, \end{aligned}$$

with the optimal solution $(\tilde{\beta}^*, \tilde{z}^*, \tilde{u}^*)$. That is, the optimal value of the objective function of $(P2)$ is a lower bound on the objective value of $(P1)$.

Similarly, taking the vectors $\hat{u} = \frac{1}{\alpha} \tilde{u}^*$ and $\hat{z} = \alpha \tilde{z}^*$, the objective function of $(P2)$ can be rewritten as

$$\|y - X\tilde{\beta}^*\|_2^2 + \alpha\tau \sum_{i=1}^{m} \tilde{z}_i^* + \alpha\tau \sum_{i=1}^{m} \tilde{u}_i^* = \|y - X\tilde{\beta}^*\|_2^2 + \tau \sum_{i=1}^{m} \hat{z}_i + \alpha^2 \tau \sum_{i=1}^{m} \hat{u}_i.$$

Again, it is easy to see that $(\tilde{\beta}^*, \hat{z}, \hat{u})$ is a feasible point in $(P1)$ for the same reason as before. Hence, the following inequality holds

$$\begin{aligned} \|y - X\beta^*\|_2^2 + \tau \sum_{i=1}^{m} z_i^* + \alpha^2 \tau \sum_{i=1}^{m} u_i^* &\leq \|y - X\tilde{\beta}^*\|_2^2 + \tau \sum_{i=1}^{m} \hat{z}_i + \alpha^2 \tau \sum_{i=1}^{m} \hat{u}_i \\ &= \|y - X\tilde{\beta}^*\|_2^2 + \alpha\tau \sum_{i=1}^{m} \tilde{z}_i^* + \alpha\tau \sum_{i=1}^{m} \tilde{u}_i^*. \end{aligned}$$

Therefore, the optimal value of the objective function of $(P1)$ is a lower bound on the objective value of $(P2)$.

In conclusion, problems $(P1)$ and $(P2)$ are equivalent. Since the values of τ and κ are arbitrary in $\mathbb{R}_{++}$, it follows that $\forall \tau, \kappa > 0$, parameters of the relaxed QP (3.5), there exists an equivalent problem with the same structure, and parameters $\tilde{\tau}$ and $\tilde{\kappa}$, such that $\tilde{\tau} = \tilde{\kappa} = \sqrt{\tau\kappa}$.

□

It is easy to note from the previous procedure that $(\tilde{\beta}^*, \hat{z}, \hat{u})$ is an optimal point of $(P1)$ and that $(\beta^*, \bar{z}, \bar{u})$ is an optimal point of $(P2)$, thereby yielding the same solution for the coefficients β_i in both problems.

Proposition 1 allows problem (3.5) to be written equivalently with parameters $\tilde{\tau} = \tilde{\kappa}$ in any scenario. Thus, one can just take this alternative and, due to the fact that the problem is symmetric with respect to to z and u, work with only one variable representing both and only one penalty parameter. However, there could exist some computational benefits when treating the problem in the exact same form as in (3.5). Therefore, in what follows, both parameters and variables are still used independently.

Another important point to note is that the problem bounds the coefficients of problem (3.5) similarly to the relaxed version of LASSO (2.15), with a first-degree penalty for the magnitudes of the components of β. The equivalence between these two problems is established in the following Proposition.

Proposition 2 $\forall \tau, \kappa > 0$, parameters of the relaxed QP (3.5), there exists a penalty parameter $\lambda = 2\sqrt{\tau\kappa}$ for the relaxed version of LASSO (2.15), such that QP and LASSO are equivalent.

Proof. It is sufficient to consider the equivalent problem of the QP (3.5), with parameters $\tilde{\tau} = \tilde{\kappa} = \sqrt{\tau\kappa}$. Given the symmetry of the problem with respect to each z_i and u_i, one can remove variables u_i and show that the objective function of problem (3.5) can be expressed as $\|y - X\beta\|_2^2 + 2\sqrt{\tau\kappa}\sum_{i=1}^m z_i$, and the constraint (3.5b) as $\beta_i^2 \leq z_i^2$, $\forall i \in \{1, \ldots, m\}$. Then, note that at the optimum solution the equality $|\beta_i| = z_i$ holds for all $i \in \{1, \ldots, m\}$, allowing the equivalent problem to be written with the objective function $\|y - X\beta\|_2^2 + 2\sqrt{\tau\kappa}\sum_{i=1}^m |\beta_i|$, which is precisely the Lagrange relaxation of LASSO with a penalty parameter $\lambda = 2\sqrt{\tau\kappa}$. □

3.3 Conic Model: Second Order Cone Program (SOCP)

Once again, as described for the MISOCP proposed by Kucukyavuz et al. (2020), one can make use of the result

$$a^2 \leq bc \iff \left\| \begin{pmatrix} b - c \\ 2a \end{pmatrix} \right\|_2 \leq b + c, \ \forall\ a \in \mathbb{R},\ b, c \in \mathbb{R}_+, \tag{3.8}$$

in the constraints (3.5b) of the relaxed QP, to obtain a model with second-order conic (SOC) constraints, resulting in a conic problem where the decomposition method described in subsection 2.4.3 can be used on. The second-order cone program is formulated as

$$\min_{\beta, z, u} \quad \|y - X\beta\|_2^2 + \tau \sum_{i=1}^m z_i + \kappa \sum_{i=1}^m u_i \tag{3.9a}$$

$$\text{s.t.} \quad \left\| \begin{pmatrix} u_i - z_i \\ 2\beta_i \end{pmatrix} \right\|_2 \leq u_i + z_i, \qquad i = 1, \ldots, m, \tag{3.9b}$$

$$u_i \geq 0, \qquad i = 1, \ldots, m, \tag{3.9c}$$

$$z_i \geq 0, \qquad i = 1, \ldots, m. \tag{3.9d}$$

The structure of the problem can be exploited to include even more possibilities in terms of the conic constraints to be relaxed with the decomposition method. The quadratic error term on the objective function can be treated as a new constraint with an auxiliary variable $\xi \in \mathbb{R}$, that reaches equality with the error term at the optimal solution. In other words, consider $\|y - X\beta\|_2 \leq \xi$, and minimize the new variable as follows

$$\begin{aligned}
\min_{\beta,z,u,\xi} \quad & \xi^2 + \tau \sum_{i=1}^{m} z_i + \kappa \sum_{i=1}^{m} u_i && && \text{(3.10a)}\\
\text{s.t.} \quad & \|y - X\beta\|_2 \leq \xi, && && \text{(3.10b)}\\
& \left\| \begin{pmatrix} u_i - z_i \\ 2\beta_i \end{pmatrix} \right\|_2 \leq u_i + z_i, && i = 1, \ldots, m, && \text{(3.10c)}\\
& u_i \geq 0, && i = 1, \ldots, m, && \text{(3.10d)}\\
& z_i \geq 0, && i = 1, \ldots, m. && \text{(3.10e)}
\end{aligned}$$

This introduces a new second-order conic constraint, which can also be relaxed in the decomposition method, as described in the subsequent subsections.

3.4 Decomposition Method for the Second-Order Cone Program

Recapping what was discussed in subsection 2.4.3, the problems to which the conic decomposition method of Chicoisne (2023) is applied have the form

$$\begin{aligned}
\min_{x,y} \quad & f(x, y) && \text{(3.11)}\\
\text{s. a} \quad & x \in \mathcal{X}, &&\\
& -g(x, y) \in \mathcal{C}, &&
\end{aligned}$$

where $\mathcal{C}$ is a conic constraint that is challenging to solve. In the case of the SOCP (3.10), this constraint can be interpreted in various ways since any of the second-order cone constraints can be considered for relaxation, or even various intersections of them can be taken, as the intersection of cones with a shared vertex is also a cone. To further clarify the possibilities of conic sets that can be relaxed, we can rewrite the second-order conic problem (3.10) from the previous subsection in the form

$$\begin{aligned}
\min_{\beta,z,u,\xi} \quad & \xi^2 + \tau \sum_{i=1}^{m} z_i + \kappa \sum_{i=1}^{m} u_i && && \text{(3.12a)}\\
\text{s.t.} \quad & \begin{pmatrix} y - X\beta \\ \xi \end{pmatrix} \in \mathcal{L}_{n+1}, && && \text{(3.12b)}\\
& \begin{pmatrix} u_i - z_i \\ 2\beta_i \\ u_i + z_i \end{pmatrix} \in \mathcal{L}_3, && i = 1, \ldots, m, && \text{(3.12c)}\\
& u \in \mathbb{R}^m_+, && && \text{(3.12d)}\\
& z \in \mathbb{R}^m_+. && && \text{(3.12e)}
\end{aligned}$$

Here, $\mathcal{L}_{k+1} = \{(x, y) \in \mathbb{R}^{k+1} : \|x\|_2 \leq y\}$ is a second-order cone of dimension $k+1$, and $\mathbb{R}^m_+$ is the cone that contains all vectors in $\mathbb{R}^m$ with non-negative components.

Regarding the general assumptions of the decomposition method described in subsection 2.4.3, two considerations will be made for this particular case:

1. Concerning the high dimensionality of the variables $\mathcal{X}$, the only variables that add complexity in this regard are $\beta, z, u \in \mathbb{R}^m$, as ξ adds only one extra dimension, independent of the value of m. With that being said, ξ is only an auxiliary variable that is fully defined by y, X, and β at the optimal point, and it is used purely as an extra variable throughout.
2. With respect to the complexity of the conic constraints in the SOCP (3.12), since all the constraints of the problem are conic in nature, it is not clear whether all the constraints complicate the problem or if only certain constraints make it difficult to solve. In other words, different combinations of conic constraints can be considered for relaxation (different sets $\mathcal{X}$ and $\mathcal{C}$). However, the non-negative constraints will not be considered as candidates for the conic relaxation.

3.4.1 Decomposition Algorithm for Second-Order Cone Program

In general terms, considering the information mentioned in the previous subsection, the Algorithm 1 presented in subsection 2.4.3 can be rewritten in a way that is adapted to the form of the SOCP (3.12). Generally, the algorithm takes the form shown in Algorithm 2. In this algorithm, the optimal solutions of the master problem $P(\mathcal{S}^k)$ are denoted as x^k, while the solutions of the relaxation $L(\mathcal{X}, \lambda^k)$ are denoted as $\bar{x}^k$.

Algorithm 2: General Column Generation for SOCP

Data: An instance of the SOCP (3.12)
Result: An optimal solution of the problem
Set $\lambda^0 = 0$, $\mathcal{S}^1 \subseteq \mathcal{X}$ contains at least one feasible solution for $P(\mathcal{X})$, and $k = 1$;
while *True* **do**
  Solve $P(\mathcal{S}^k)$. Let $(\beta^k, z^k, u^k; \xi^k)$ be an optimal solution;
  Let λ^k be an optimal dual vector corresponding to the constraints $-g(\beta, z, u; \xi) \in C$;
  if $\lambda^k = \lambda^{k-1}$ **then**
    return $(\beta^k, z^k, u^k; \xi^k)$;
  Solve $L(\mathcal{X}, \lambda^k)$. Let $(\bar{\beta}^k, \bar{z}^k, \bar{u}^k; \bar{\xi}^k)$ be an optimal solution ;
  if $(\bar{\beta}^k, \bar{z}^k, \bar{u}^k) \in \mathcal{S}^k$ **then**
    return $(\beta^k, z^k, u^k; \xi^k)$;
  Choose a set $\mathcal{S}^{k+1} \subseteq \mathcal{X}$ containing $(\bar{\beta}^k, \bar{z}^k, \bar{u}^k)$;
  $k \leftarrow k + 1$;

3.4.2 Feasible Region $\mathcal{S}^k$

The first element to consider in the decomposition method are the feasible regions $\mathcal{S}^k \subseteq \mathcal{X}$; in other words, the feasible region of the master problem on each iteration. These subsets should be such that the instances $P(\mathcal{S}^k)$ of the master problem can be solved more efficiently than $P(\mathcal{X})$, while in each iteration of the method, they incorporate optimal solutions from the Lagrange relaxation of line 7 of the algorithm. Bearing this in mind, it is natural to think of $\mathcal{S}^k$ as some kind of combination of the solutions obtained in the previous $k-1$ iterations. Chicoisne (2023) reviews this same idea with different types of combinations, including convex, conic, and linear combinations to generate the feasible region.

For the decomposition method of the SOCP (3.12), these three combination options will be taken into account in the construction of $\mathcal{S}^k$, along with the affine combination. Each of these alternatives will have an initial set $\mathcal{S}^1$, which will contain a predefined number v_0 (parameter) of initial feasible points of $P(\mathcal{X})$, denoted as $\{\bar{x}^s\}_{s=1}^{v_0} = \{(\bar{\beta}^s, \bar{z}^s, \bar{u}^s)\}_{s=1}^{v_0}$. The problem $P(\mathcal{S}^k)$ will seek to find the optimal weights for this problem, denoted as $\pi \in \mathbb{R}^{v_0+k-1}$ for the v_0 initial solutions and $k-1$ solutions obtained in each iteration $\{\bar{x}^s\}_{s=1}^{v_0+k-1} = \{(\bar{\beta}^s, \bar{z}^s, \bar{u}^s)\}_{s=1}^{v_0+k-1}$.

1. Convex combination:

$$\mathcal{S}^k := \left\{ x \in \mathcal{X} : x = \sum_{s=1}^{v_0+k-1} \pi_s \bar{x}^s, \quad \pi \in \mathbb{R}_+^{v_0+k-1}, \quad \sum_{s=1}^{v_0+k-1} \pi_s = 1 \right\} = \mathcal{X} \cap \text{conv}(\{\bar{x}^s\}_{s=1}^{v_0+k-1})$$

2. Affine combination:

$$\mathcal{S}^k := \left\{ x \in \mathcal{X} : x = \sum_{s=1}^{v_0+k-1} \pi_s \bar{x}^s, \quad \pi \in \mathbb{R}^{v_0+k-1}, \quad \sum_{s=1}^{v_0+k-1} \pi_s = 1 \right\} = \mathcal{X} \cap \text{aff}(\{\bar{x}^s\}_{s=1}^{v_0+k-1})$$

3. Conic combination:

$$\mathcal{S}^k := \left\{ x \in \mathcal{X} : x = \sum_{s=1}^{v_0+k-1} \pi_s \bar{x}^s, \quad \pi \in \mathbb{R}_+^{v_0+k-1} \right\} = \mathcal{X} \cap \text{cone}(\{\bar{x}^s\}_{s=1}^{v_0+k-1})$$

4. Linear combination:

$$\mathcal{S}^k := \left\{ x \in \mathcal{X} : x = \sum_{s=1}^{v_0+k-1} \pi_s \bar{x}^s, \quad \pi \in \mathbb{R}^{v_0+k-1} \right\} = \mathcal{X} \cap \text{lin}(\{\bar{x}^s\}_{s=1}^{v_0+k-1})$$

The choice of initial feasible points $\{\bar{x}^s\}_{s=1}^{v_0}$ as the elements determining the set $\mathcal{S}^1$ can also be done in various ways. However, for simplicity, this study does not focus on finding the best initial set. With that said, some simple alternatives are proposed for the choice of this initial set as follows.

1. Random selection of v_0 initial feasible solutions of $P(\mathcal{X})$. This can be done with either canonical vectors or completely random feasible solutions.
2. A single solution vector $\bar{x}^1 = \mathbf{e}$, where $\mathbf{e}$ is the vector with all components equal to 1.

Both methods for selecting the initial set can be used together or separately.

3.4.3 Master Problem $P(\mathcal{S}^k)$

Given a feasible region $\mathcal{S}^k$ constructed using one of the alternatives from the previous subsection, the combination associated with each of the variables β, z, and u can be summarized as the multiplication of a matrix, containing the solution vectors of the previous iterations as columns, and different weights for each of them. In other words, one can define the following matrices

$$\mathbf{B}^k = \begin{pmatrix} \bar{\beta}_1^1 & \dots & \bar{\beta}_1^{v_0+k-1} \\ \vdots & \ddots & \vdots \\ \bar{\beta}_m^1 & \dots & \bar{\beta}_m^{v_0+k-1} \end{pmatrix} \in \mathbb{R}^{m\times(v_0+k-1)},$$

$$\mathbf{U}^k = \begin{pmatrix} \bar{u}_1^1 & \dots & \bar{u}_1^{v_0+k-1} \\ \vdots & \ddots & \vdots \\ \bar{u}_m^1 & \dots & \bar{u}_m^{v_0+k-1} \end{pmatrix} \in \mathbb{R}^{m\times(v_0+k-1)},$$

$$\mathbf{Z}^k = \begin{pmatrix} \bar{z}_1^1 & \dots & \bar{z}_1^{v_0+k-1} \\ \vdots & \ddots & \vdots \\ \bar{z}_m^1 & \dots & \bar{z}_m^{v_0+k-1} \end{pmatrix} \in \mathbb{R}^{m\times(v_0+k-1)},$$

where $\mathbf{B}_s^k$ represents the s-th column of the matrix $\mathbf{B}^k$, while $\mathbf{b}_i^k$ represents its i-th row, and the same notation is used for the other two matrices. Using this notation, the master problem $P(\mathcal{S}^k)$ of the SOCP (3.12) can be written as follows

$$\min_{\pi,\xi} \quad \xi^2 + \tau \sum_{i=1}^{m} \mathbf{z}_i^k \pi + \kappa \sum_{i=1}^{m} \mathbf{u}_i^k \pi \tag{3.13a}$$

$$\text{s.t.} \quad \begin{pmatrix} y - X\mathbf{B}^k\pi \\ \xi \end{pmatrix} \in \mathcal{L}_{n+1}, \tag{3.13b}$$

$$\begin{pmatrix} \mathbf{u}_i^k - \mathbf{z}_i^k \\ 2\mathbf{b}_i^k \\ \mathbf{u}_i^k + \mathbf{z}_i^k \end{pmatrix} \pi \in \mathcal{L}_3, \qquad i = 1, \dots, m, \tag{3.13c}$$

$$\pi \in \Pi, \tag{3.13d}$$

where Π is the set of all additional constraints on variable π required for the chosen combination (non-negative or normalized values). Non-negativity constraints on the components of $\mathbf{Z}^k\pi$ and $\mathbf{U}^k\pi$ are not necessary, as the solutions obtained at each iteration already meet these conditions.

3.4.4 Lagrange Relaxation $L(\mathcal{X}, \lambda^k)$

In general, after solving an instance of the master problem $P(\mathcal{S}^k)$, the dual variables of the conic constraints λ^k associated with the optimum of this problem are obtained. Based on these solutions and the relaxation of the complex conic constraints, an optimization problem of the minimization form (2.18) can be constructed. However, as mentioned at the beginning of the section, there are various ways to relax the SOCP (3.12). Therefore, the following three possibilities of relaxation will be considered.

1. Relaxation of the second-order cone constraint $\mathcal{L}_{n+1}$ in (3.13b). For simplicity, this constraint will be denoted as $\mathcal{C}_1$.
2. Relaxation of the m second-order cone constraints $\mathcal{L}_3$ in (3.13c). This intersection of constraints will be denoted as $\mathcal{C}_2$, which is also a cone.
3. Relaxation of both cones $\mathcal{C}_1$ and $\mathcal{C}_2$.

To understand the aforementioned relaxations, the Lagrange relaxation of both sets of constraints is presented as follows.

$$L(x,\lambda) = \underbrace{\xi^2 + \tau\sum_{i=1}^{m} z_i + \kappa\sum_{i=1}^{m} u_i}_{\text{objective function}} \underbrace{- \begin{pmatrix}\psi' & \mu\end{pmatrix}\begin{pmatrix} y - X\beta \\ \xi \end{pmatrix}}_{\mathcal{C}_1} \underbrace{- \sum_{i=1}^{m} \begin{pmatrix}\alpha_i & \gamma_i & \delta_i\end{pmatrix}\begin{pmatrix} u_i - z_i \\ 2\beta_i \\ u_i + z_i \end{pmatrix}}_{\mathcal{C}_2},$$

where $x = (\beta, z, u, \xi)$ is the vector representing the primal variables of the conic problem (3.12), $\lambda = (\psi, \mu, \alpha, \gamma, \delta)$ is the vector representing all dual variables associated with the relaxed constraints. To maintain the lower bound of this function in a feasible point x in the SOCP, it is necessary for the vectors (ψ, μ) and $(\alpha_i, \gamma_i, \delta_i)$, $\forall i \in \{1, \ldots, m\}$, to belong to the dual cones of the relaxed cones, respectively. However, second-order cones of any dimension q are self-dual; that is, $\mathcal{L}_q^* = \mathcal{L}_q$. Therefore, they must satisfy

$$\begin{pmatrix}\psi' \\ \mu\end{pmatrix} \in \mathcal{L}_{n+1} \quad \text{and} \quad \begin{pmatrix}\alpha_i \\ \gamma_i \\ \delta_i\end{pmatrix} \in \mathcal{L}_3, \forall i \in \{1, \ldots, m\}.$$

However, the decomposition method yields a fixed value λ^k for the dual variables once the master problem $P(\mathcal{S}^k)$ is solved, which already satisfy these conditions, so it is not necessary to impose these constraints on the values of λ^k. Nonetheless, the structure of each cone relaxation as shown in the Lagrange relaxation remains the same regardless.

3.4.4.1 Relaxation of $\mathcal{C}_1$

In this version, just the constraint $\mathcal{C}_1$ is relaxed. This is a second-order cone in $\mathbb{R}^{n+1}$, and as mentioned earlier, its dual cone is the same second-order cone ($\mathcal{L}_{n+1}^* = \mathcal{L}_{n+1}$). Therefore, to

relax this constraint, it is sufficient to take the optimal dual values (ψ^k, μ^k) obtained from the master problem $P(\mathcal{S}^k)$ and write the Lagrange relaxation associated with $\mathcal{C}_1$ as follows.

$$\begin{aligned} L_{\mathcal{C}_1}(x, \lambda^k) &= \xi^2 + \tau \sum_{i=1}^{m} z_i + \kappa \sum_{i=1}^{m} u_i - \begin{pmatrix} \psi^{k\prime} & \mu^k \end{pmatrix} \begin{pmatrix} y - X\beta \\ \xi \end{pmatrix} \\ &= \quad (\xi - \mu^k)\xi + \psi^{k\prime} X\beta + \tau \mathbf{e}' z + \kappa \mathbf{e}' u - \psi^{k\prime} y. \end{aligned}$$

Based on this function, a new objective function is constructed, while the remaining constraints of the problem $(\mathcal{X}_{\overline{\mathcal{C}_1}})$ remain the same as in the conic problem. Thus, the relaxed problem $L(\mathcal{X}_{\overline{\mathcal{C}_1}}, \lambda^k)$ is defined as

$$(L(\mathcal{X}_{\overline{\mathcal{C}_1}}, \lambda^k)) \quad \min_{\beta, z, u, \xi} \quad (\xi - \mu^k)\xi + \psi^{k\prime} X\beta + \tau \mathbf{e}' z + \kappa \mathbf{e}' u - \psi^{k\prime} y \tag{3.14a}$$

$$\text{s.t.} \quad \begin{pmatrix} u_i - z_i \\ 2\beta_i \\ u_i + z_i \end{pmatrix} \in \mathcal{L}_3, \qquad i = 1, \ldots, m, \tag{3.14b}$$

$$u \in \mathbb{R}^m_+, \tag{3.14c}$$

$$z \in \mathbb{R}^m_+, \tag{3.14d}$$

This problem can be simplified, as the variable ξ is completely independent of any other variable of the problem. Furthermore, it is merely an auxiliary variable of the original problem, so its value itself is not of interest once the constraint in which it participates is relaxed. Thus, the term $(\xi - \mu^k)\xi$ can be omitted from the objective function, and the same optimal solution will be attained. In case of needing to recover the value of this variable, it is sufficient to note that the objective function is a convex and unconstrained function with respect to this variable. Therefore, the first-order conditions can be used with this variable to obtain the value that minimizes this term (impose a saddle point). That is, it can be imposed that

$$\frac{\partial}{\partial \xi}(\xi - \mu^k)\xi = 0 \implies \xi = \frac{\mu^k}{2}.$$

3.4.4.2 Relaxation of $\mathcal{C}_2$

Analogously to the relaxation of the constraint $\mathcal{C}_1$, in this case, the dual cones are of the same type, so they are self-dual. The relaxation is written based on the optimal dual solutions $(\alpha^k, \gamma^k, \delta^k)$ obtained from the master problem $P(\mathcal{S}^k)$ as follows.

$$\begin{aligned} L_{\mathcal{C}_2}(x, \lambda^k) &= \xi^2 + \tau \sum_{i=1}^{m} z_i + \kappa \sum_{i=1}^{m} u_i - \sum_{i=1}^{m} \begin{pmatrix} \alpha_i^k & \gamma_i^k & \delta_i^k \end{pmatrix} \begin{pmatrix} u_i - z_i \\ 2\beta_i \\ u_i + z_i \end{pmatrix} \\ &= \xi^2 - 2\gamma^{k\prime}\beta + \left(\tau \mathbf{e} + \alpha^k - \delta^k\right)' z + \left(\kappa \mathbf{e} - \alpha^k - \delta^k\right)' u. \end{aligned}$$

And, again, the remaining constraints of the problem can be denoted as $\mathcal{X}_{\overline{\mathcal{C}_2}}$. Hence, the relaxed problem $L(\mathcal{X}_{\overline{\mathcal{C}_2}}, \lambda^k)$ is defined as

$$
\begin{aligned}
(L(\mathcal{X}_{\overline{\mathcal{C}_2}}, \lambda^k)) \quad \min_{\beta,z,u,\xi} \quad & \xi^2 - 2\gamma^{k\prime}\beta + \left(\tau\mathbf{e} + \alpha^k - \delta^k\right)' z + \left(\kappa\mathbf{e} - \alpha^k - \delta^k\right)' u && \text{(3.15a)}\\
\text{s.t.} \quad & \begin{pmatrix} y - X\beta \\ \xi \end{pmatrix} \in \mathcal{L}_{n+1}, && \text{(3.15b)}\\
& u \in \mathbb{R}^m_+, && \text{(3.15c)}\\
& z \in \mathbb{R}^m_+. && \text{(3.15d)}
\end{aligned}
$$

3.4.4.3 Relaxation of $\mathcal{C}_1$ and $\mathcal{C}_2$

The last alternative for relaxing the conic constraints that represent complexity when solving the conic problem 3.12, namely the relaxation of the sets $\mathcal{C}_1$ and $\mathcal{C}_2$, is presented in this subsection. The dual variables obtained in the master problem of the method are those mentioned in the two previous alternatives, which imply a Lagrange relaxation of the form

$$
\begin{aligned}
L_{\mathcal{C}_{1,2}}(x, \lambda^k) = \quad & \xi^2 + \tau\sum_{i=1}^m z_i + \kappa\sum_{i=1}^m u_i - \begin{pmatrix}\psi^{k\prime} & \mu^k\end{pmatrix}\begin{pmatrix} y - X\beta \\ \xi \end{pmatrix} - \sum_{i=1}^m \begin{pmatrix}\alpha_i^k & \gamma_i^k & \delta_i^k\end{pmatrix}\begin{pmatrix} u_i - z_i \\ 2\beta_i \\ u_i + z_i \end{pmatrix}\\
= \quad & (\xi - \mu^k)\xi + \left(\psi^{k\prime}X - 2\gamma^{k\prime}\right)\beta + \left(\tau\mathbf{e} + \alpha^k - \delta^k\right)' z + \left(\kappa\mathbf{e} - \alpha^k - \delta^k\right)' u - \psi^{k\prime}y.
\end{aligned}
$$

The relaxed optimization problem $L(\mathcal{X}_{\overline{\mathcal{C}}_{1,2}}, \lambda^k)$ can then be written as

$$
\begin{aligned}
(L(\mathcal{X}_{\overline{\mathcal{C}}_{1,2}}, \lambda^k)) \quad \min_{\beta,z,u,\xi} \quad & (\xi - \mu^k)\xi + \left(\psi^{k\prime}X - 2\gamma^{k\prime}\right)\beta && \text{(3.16a)}\\
& \quad + \left(\tau\mathbf{e} + \alpha^k - \delta^k\right)' z\\
& \quad + \left(\kappa\mathbf{e} - \alpha^k - \delta^k\right)' u - \psi^{k\prime}y\\
\text{s.t.} \quad & u \in \mathbb{R}^m_+, && \text{(3.16b)}\\
& z \in \mathbb{R}^m_+, && \text{(3.16c)}
\end{aligned}
$$

where $\mathcal{X}_{\overline{\mathcal{C}}_{1,2}}$ is the set of all the constraints not relaxed in the previous Lagrange function. This is a pretty simple problem since it only has non-negativity constraints and a linear objective function. Furthermore, if the structure of the problem admits a finite solution, then it can be found analytically. The optimal value of ξ can be obtained with the first order condition, as in the relaxation of $\mathcal{C}_1$, the optimal value of the term $\left(\psi^{k\prime}X - 2\gamma^{k\prime}\right)\beta$ must be always zero to meet the requirement of a finite solution, and the minimum value of z and u must also be zero for the same reason. However, the finite solution depends on the values of the dual variables, and this also true for the previous two problems. The following subsection delves deeper into this matter.

3.4.5 On the Boundedness of the Lagrange Relaxation $L(\mathcal{X}, \lambda^k)$

The decomposition method described in Algorithm 2 assumes that the optimization problem derived from the Lagrange relaxation $L(\mathcal{X}, \lambda^k)$ yields an optimal solution. However, this is not necessarily true, as the dual variables λ^k derive from the master problem $P(\mathcal{S}^k)$ described in (3.13), which can have a dual form very different from that of the original SOCP (3.12). To clarify this further, the dual problem $D(\mathcal{S}^k)$ of the master problem $P(\mathcal{S}^k)$ is presented, in the case where $\mathcal{S}^k$ is constructed through linear combinations of the previous method's solutions (π has no additional conditions of the form (3.13d)), as follows.

$$
\begin{aligned}
(D(\mathcal{S}^k)) \quad \max_{\psi,\mu,\alpha,\gamma,\delta} \quad & -\frac{\mu^2}{4} - \psi' y \\
\text{s.t.} \quad & (\tau\mathbf{e} + \alpha - \delta)'\mathbf{Z}^k + (\kappa\mathbf{e} - \alpha - \delta)'\mathbf{U}^k \\
& + (\psi' X - 2\gamma')\,\mathbf{B}^k = 0, && \text{(3.17a)} \\
& \begin{pmatrix} \psi \\ \mu \end{pmatrix} \in \mathcal{L}_{n+1}, && \text{(3.17b)} \\
& \begin{pmatrix} \alpha_i \\ \gamma_i \\ \delta_i \end{pmatrix} \in \mathcal{L}_3, \qquad i = 1, \ldots, m. && \text{(3.17c)}
\end{aligned}
$$

When considering the constraint on π belonging to $\mathbb{R}_+^{v_0+k-1}$ as in the conic and convex combination, the only thing that changes in the problem $D(\mathcal{S}^k)$ is that the nullity "$= 0$" of the expression in constraint (3.17a) becomes a "$\in \mathbb{R}_+^{v_0+k-1}$" (all its components must be non-negative). On the other hand, when considering the constraint of normalization of the elements of π as in the affine and convex combination, one can generate a new dual variable σ associated with this constraint, which adds an extra term "$-\sigma$" to the dual's objective function and an extra term "$+\sigma$" to each component of the resultant vector of the expression (3.17a). Therefore, the dual problem does not change much when considering those additional constraints.

In the following section, an analysis will be conducted on the necessary conditions for boundedness of the Lagrange relaxation of each of the three alternatives presented in the preceding subsection. This analysis aims to examine whether the dual problem (3.17) guarantees that the Lagrange relaxation is a bounded optimization problem or not.

3.4.5.1 Necessary conditions for boundedness of $L(\mathcal{X}_{\overline{\mathcal{C}_1}}, \lambda^k)$

The objective function (3.14a) of the problem $L(\mathcal{X}_{\overline{\mathcal{C}_1}}, \lambda^k)$ is written in a way that allows the analysis of each variable of the problem separately. First, it has been mentioned that the minimum can be found with respect to ξ using the first-order conditions. As for the rest of the variables, if the second-order cone constraints (3.14b) did not exist, ensuring boundedness of the objective function of the relaxation would require the following conditions.

1. $\psi^{k\prime} X = 0$, since β has no other constraints.

2. $\tau\mathbf{e}, \kappa\mathbf{e} \in \mathbb{R}^m+$, given that $z, u \in \mathbb{R}^m+$.

The second conditions is always met, since for the selection problem, τ, κ are always chosen from $\mathbb{R}_{++}$. However, the first condition is not guaranteed, as it is not a constraint on the dual variables λ^k of any of the possibilities of the master dual problem $D(\mathcal{S}^k)$. Therefore, the objective function may not be bounded on its own initially. However, with the existence of the second-order cone constraints (3.14b), and recalling that they are equivalent to $\beta_i^2 \leq z_i u_i$, the weights associated with z_i and u_i in the objective function can prevent the variable β_i from diverging in any direction, depending on their values.

Proposition 3 For any iteration k of Algorithm 2, the relaxed problem $L(\mathcal{X}_{\overline{\mathcal{C}_1}}, \lambda^k)$ (3.14) is bounded if and only if the dual variables λ^k satisfy the condition

$$|(X'\psi^k)_i| \leq 2\sqrt{\kappa\tau}, \quad \forall i \in \{1, \ldots, m\}.$$

Proof. If penalization values are chosen such that $\tau = \kappa$, then, due to the symmetry of problem $L(\mathcal{X}_{\overline{\mathcal{C}_1}}, \lambda^k)$ with respect to z and u in every iteration k of Algorithm 2, the optimum of the problem satisfies $z^* = u^*$ and achieves equality in the conic constraint $\beta_i^{*2} = z_i^* u_i^*$. Thus, the optimum satisfies $|\beta_i^*| = z_i^* = u_i^*$. Without loss of generality, a subproblem such that it has the equality constraint $|\beta_i| = z_i = u_i$, which achieves the same optimal value, can be used instead. This is because its objective function is necessarily greater than or equal to that of the original problem, being a subset of the feasible region, and the optimum of the problem $L(\mathcal{X}_{\overline{\mathcal{C}_1}}, \lambda^k)$ is feasible within it, resulting in the same objective function value at the optimum.

As mentioned in subsection 3.4.4.1, one can impose the first order condition on variable ξ to have a bound with respect to it on problem $L(\mathcal{X}_{\overline{\mathcal{C}_1}}, \lambda^k)$. The only variables that can lead to unboundedness then are β, z and u. The only terms that depend on these three variables in the objective function of the subproblem that satisfies $|\beta_i| = z_i = u_i$ are of the form

$$\begin{aligned}(X'\psi^k)_i\beta_i + \tau z_i + \kappa u_i &= (X'\psi^k)_i\beta_i + \tau|\beta_i| + \kappa|\beta_i| \\ &= (X'\psi^k)_i\beta_i + 2\tau|\beta_i|.\end{aligned}$$

These terms can be analyzed by cases: a subset of the feasible region with all the solutions that have a non-negative i-th component $\beta_i \geq 0$ and a subset of the feasible region with all the solutions with a negative i-th component $\beta_i < 0$. If $\beta_i \geq 0$, then these terms are of the form

$$\begin{aligned}(X'\psi^k)_i\beta_i + 2\tau|\beta_i| &= (X'\psi^k)_i\beta_i + 2\tau\beta_i \\ &= ((X'\psi^k)_i + 2\tau)\beta_i.\end{aligned}$$

Since $\beta_i \geq 0$, it must necessarily hold that $(X'\psi^k)_i + 2\tau \geq 0$ for the objective function to be bounded when minimizing with respect to β_i, which is equivalent to $-(X'\psi^k)_i \leq 2\tau$. In the opposite case, where $\beta_i < 0$, the terms of the objective function take the form

$$\begin{aligned}(X'\psi^k)_i\beta_i + 2\tau|\beta_i| &= (X'\psi^k)_i\beta_i - 2\tau\beta_i \\ &= ((X'\psi^k)_i - 2\tau)\beta_i.\end{aligned}$$

Now, because $\beta_i < 0$, it must hold that $(X'\psi^k)_i - 2\tau \leq 0$ for the objective function to be bounded, which is equivalent to $(X'\psi^k)_i \leq 2\tau$. The two conditions can be summarized in a single general condition as

$$|(X'\psi^k)_i| \leq 2\tau, \quad \forall i \in \{1, \ldots, m\}, \tag{3.18}$$

which applies to all the feasible region of the problem.

The only issue with the previous condition is that it only holds if $\tau = \kappa$. However, problem $L(\mathcal{X}_{\overline{\mathcal{C}_1}}, \lambda^k)$ and the quadratic problem (3.5) have the exact same structure for variables z and u. Thus, Proposition 1's proof also applies in this case. One can proof that for any problem $L(\mathcal{X}_{\overline{\mathcal{C}_1}}, \lambda^k)$ with penalty parameters $\tau \neq \kappa$, there exist an equivalent problem with the same structure such that its penalty parameters satisfy $\tilde{\tau} = \tilde{\kappa} = \sqrt{\tau\kappa}$. With this in mind, if one of the problems is unbounded, the same applies for the second one, and the same boundedness condition applies to both problems. Therefore, it suffices to transform condition (3.18) with weights $\tilde{\tau} = \tilde{\kappa} = \sqrt{\tau\kappa}$ into its equivalent version as

$$\begin{aligned} & |(X'\psi^k)_i| \leq 2\tilde{\tau}, && \forall i \in \{1, \ldots, m\} \\ \iff & |(X'\psi^k)_i| \leq 2\sqrt{\kappa\tau}, && \forall i \in \{1, \ldots, m\}, \end{aligned} \tag{3.19}$$

with the latter being the general boundedness condition for any value of τ and κ as in the proposition.

□

3.4.5.2 Necessary conditions for boundedness of $L(\mathcal{X}_{\overline{\mathcal{C}_2}}, \lambda^k)$

For the case of the relaxed problem $L(\mathcal{X}_{\overline{\mathcal{C}_2}}, \lambda^k)$ presented in (3.15), a similar behavior to the previously discussed case is observed. The necessary conditions can be summarized in the following proposition.

Proposition 4 For any iteration k of Algorithm 2, the relaxed problem $L(\mathcal{X}_{\overline{\mathcal{C}_2}}, \lambda^k)$ (3.15) is bounded if and only if the dual variables λ^k satisfy the conditions

$$\begin{aligned} \tau\mathbf{e} + \alpha^k - \delta^k \in \mathbb{R}^m_+, \\ \kappa\mathbf{e} - \alpha^k - \delta^k \in \mathbb{R}^m_+. \end{aligned}$$

Proof. There are three sets of variables that are completely independent in this problem: $\{z\}$, $\{u\}$, and $\{\xi, \beta\}$. Given this, boundedness conditions can be analyzed on a case-by-case basis. For the case of ξ and β, the existence of the auxiliary variable ξ can be ignored by removing the conic constraint (3.15b) and returning to the original form of the norm $\|y - X\beta\|_2^2$ in the objective function. Thus, the term dependent on the variable β in the objective function becomes

$$\|y - X\beta\|_2^2 - 2\gamma^{k\prime}\beta = y'y - 2y'X\beta + \beta'X'X\beta - 2\gamma^{k\prime}\beta,$$

which is a convex function with respect to β, as $X'X \in S^m_+$ (a positive semidefinite matrix in $\mathbb{R}^{m\times m}$). Therefore, without any constraints on this variable, a saddle point can be imposed

to find a minimum of this function using the first-order conditions, resulting in the optimum solution $\beta^* = (X'X)^{-1}(X'y + \gamma^k)$, assuming $X'X \in S^m_{++}$, a positive definite matrix. Given this, it is not necessary to impose conditions regarding this term in the objective function, as computing independently the optimal result for it is sufficient.

Regarding the variables z and u, the conditions are evident. As there are only constraints of non-negativity on the components of these vectors, any negative weighting of them in the objective function results in unbounded solutions. That is, the only necessary conditions for the convergence of the relaxed problem $L(\mathcal{X}_{\overline{\mathcal{C}_2}}, \lambda^k)$ are

$$\tau\mathbf{e} + \alpha^k - \delta^k \in \mathbb{R}^m_+,$$
$$\kappa\mathbf{e} - \alpha^k - \delta^k \in \mathbb{R}^m_+.$$

□

None of the conditions presented are guaranteed by the dual problem $D(\mathcal{S}^k)$, so any component that does not satisfy them in any iteration of the decomposition method causes a breakdown, as it does not provide a numerical solution for the relaxation.

3.4.5.3 Necessary conditions for boundedness of $L(\mathcal{X}_{\overline{\mathcal{C}}_{1,2}}, \lambda^k)$

In this last case the relaxed model $L(\mathcal{X}_{\overline{\mathcal{C}}_{1,2}}, \lambda^k)$ presented in (3.16) has only independent variables, so boundedness conditions are straightforward.

Proposition 5 For any iteration k of Algorithm 2, the relaxed problem $L(\mathcal{X}_{\overline{\mathcal{C}}_{1,2}}, \lambda^k)$ (3.16) is bounded if and only if the dual variables λ^k satisfy the conditions

$$\tau\mathbf{e} + \alpha^k - \delta^k \in \mathbb{R}^m_+,$$
$$\kappa\mathbf{e} - \alpha^k - \delta^k \in \mathbb{R}^m_+.$$

Proof. Regarding the auxiliary variable ξ, it forms a convex function with the term $(\xi - \mu^k)\xi$, for which it was previously mentioned that a saddle point can be imposed. The variable β is entirely free, so it can diverge in any direction, requiring the coefficient accompanying it to be zero to allow boundedness with respect to it. Lastly, regarding the variables z and u, the situation is analogous to the previous case, as the terms are identical. Given this, the necessary conditions for boundedness of this problem are given by

$$X'\psi^k - 2\gamma^k = 0, \tag{3.20}$$
$$\tau\mathbf{e} + \alpha^k - \delta^k \in \mathbb{R}^m_+, \tag{3.21}$$
$$\kappa\mathbf{e} - \alpha^k - \delta^k \in \mathbb{R}^m_+. \tag{3.22}$$

□

Since, once again, none of these conditions for the dual variables λ^k are guaranteed by the dual problem $D(\mathcal{S}^k)$, if any component of any of these conditions does not satisfy them, the relaxed problem has a non-finite solution, which leaves the decomposition method with no solution vector to add to the solutions matrix for the next iteration.

Table 3.1: Summary of the necessary boundedness conditions of the three versions of relaxation of the second-order cone problem at each iteration of the decomposition method.

Relaxed Problem	Boundedness Conditions
$L(\mathcal{X}_{\overline{\mathcal{C}_1}}, \lambda^k)$	$\lvert(X'\psi^k)_i\rvert \leq 2\sqrt{\kappa\tau}, \quad \forall i \in \{1, \ldots, m\}$
$L(\mathcal{X}_{\overline{\mathcal{C}_2}}, \lambda^k)$	$\tau\mathbf{e} + \alpha^k - \delta^k \in \mathbb{R}^m_+$, $\kappa\mathbf{e} - \alpha^k - \delta^k \in \mathbb{R}^m_+$
$L(\mathcal{X}_{\overline{\mathcal{C}}_{1,2}}, \lambda^k)$	$X'\psi^k - 2\gamma^k = 0$, $\tau\mathbf{e} + \alpha^k - \delta^k \in \mathbb{R}^m_+$, $\kappa\mathbf{e} - \alpha^k - \delta^k \in \mathbb{R}^m_+$

3.4.6 Artificial Solutions for Unbounded Relaxed Problems

As discussed in the previous subsection, there may be cases where the variables of the dual problem $D(\mathcal{S}^k)$ (3.17) do not satisfy the boundedness conditions, since the dual problem does not guarantee any of them, resulting in the diverging of the specific relaxed problems. Naturally, if this happens, the decomposition method loses its purpose, as obtaining a finite solution from an unbounded problem becomes impossible. Consequently, the set $\mathcal{S}^{k+1}$ cannot be updated for the next iteration of the method. Therefore, there is a need to generate solutions in an alternative way to ensure the functionality of the method in such cases.

A simple solution is to generate canonical solutions associated with the components of the main variable β randomly, i.e., assigning a uniform probability to each $i \in \{1, \ldots, m\}$ to obtain some index j, generating an artificial solution of the form $\bar{\beta}^k = \mathbf{e}_j$, where the only nonzero component is the j-th one. For the other two main variables, it is sufficient to take the same vector $\bar{z}^k = \bar{u}^k = \mathbf{e}_j$ when $\tau = \kappa$, or their equivalents in proportion, $\bar{z}^k = \sqrt{\kappa/\tau}\mathbf{e}_j$ and $\bar{u}^k = \sqrt{\tau/\kappa}\mathbf{e}_j$, when $\tau \neq \kappa$, to maintain the structure of the optimal solutions.

The above idea can be further developed by considering that all the boundedness conditions from the previous subsection come from weights of the form $h(\lambda^k)'x$ in the objective function of each problem, where $h(\lambda^k)$ is some transformation of the dual variables λ^k (constant in the relaxed problem), and x is the vector of the variables in the relaxed problem. If the problem were the minimization of the function $L(x, \lambda^k) = h(\lambda^k)'x$ without any constraints, then the direction of the steepest descent of this function with respect to x is given by $-\nabla_x L(x, \lambda^k) = -h(\lambda^k)$. In cases where $x_j \to \pm\infty$ upon solving the problem, i.e., the j-th components of the vector $h(\lambda^k)$ that do not satisfy the boundedness conditions, these values should diverge precisely in the directions $-\nabla_x L(x, \lambda^k)_j = -h(\lambda^k)_j$, so one might consider taking the components of the steepest descent direction in the diverging components as outlined in the previous subsection's boundedness conditions. However, the solution $\bar{x}^k$ with the diverging components equal to those of $-h(\lambda^k)$ and the rest of the components not necessarily is not a good solution for the method as it is a mixture of components of maximum descent and zero values. Therefore, it is proposed to take multiple canonical vectors in the diverging components, weighted by the sign of $-h(\lambda)^k_j$ if they are free variables in the problem, and simply canonical vectors in the diverging components for non-negative variables. This gives more freedom to the method to find the best direction among the diverging components and the rest of the previous solutions, in addition to being a step beyond the random choice

of canonical vectors, while making the same matching of components of β, z, and u described with the random canonical solutions.

The problem that can arise from this method is that too many synthetic canonical solutions are added due to the possibility of having many diverging components, which can complicate the master problem of the next iteration by increasing the dimensionality of $\mathcal{S}^{k+1}$ too much. To avoid this, a parameter v is added to the method, dictating the number of canonical solutions to add, selecting the v with the greatest absolute value in its weight on the objective function in each iteration where the relaxed problem is unbounded.

If $\mathcal{J}_v^k$ denotes the set of the v components of the greatest magnitude among those that do not meet the boundedness condition of their respective relaxed problem, the construction procedure of artificial solutions $\bar{x}^k = (\bar{\beta}^k, \bar{z}^k, \bar{u}^k)$ for each proposed problem can be explicitly presented in the summary of Table 3.2.

Table 3.2: Summary of boundedness conditions and the artificial solutions constructed in each case when the relaxed problem is unbounded. All components of the artificial solutions not explicitly stated, i.e., $j \notin \mathcal{J}_v^k$, take on zero values.

Relaxed Problem	Boundedness Conditions	Artificial Solutions
$L(\mathcal{X}_{\overline{\mathcal{C}_1}}, \lambda^k)$	$\lvert(X'\psi^k)_i\rvert \leq 2\sqrt{\kappa\tau}, \quad \forall i \in \{1, \ldots, m\}$	$\sqrt{\tau/\kappa}\bar{z}_j^k = \sqrt{\kappa/\tau}\bar{u}_j^k = 1,$ $\bar{\beta}_j^k = -\frac{(X'\psi^k)_j}{\lvert(X'\psi^k)_j\rvert}, \forall j \in \mathcal{J}_v^k$
$L(\mathcal{X}_{\overline{\mathcal{C}_2}}, \lambda^k)$	$\tau\mathbf{e} + \alpha^k - \delta^k \in \mathbb{R}_+^m$	$\sqrt{\tau/\kappa}\bar{z}_j^k = \sqrt{\kappa/\tau}\bar{u}_j^k = 1,$ $\bar{\beta}_j^k = \pm 1, \forall j \in \mathcal{J}_v^k$
	$\kappa\mathbf{e} - \alpha^k - \delta^k \in \mathbb{R}_+^m$	$\sqrt{\tau/\kappa}\bar{z}_j^k = \sqrt{\kappa/\tau}\bar{u}_j^k = 1,$ $\bar{\beta}_j^k = \pm 1, \forall j \in \mathcal{J}_v^k$
$L(\mathcal{X}_{\overline{\mathcal{C}_{1,2}}}, \lambda^k)$	$X'\psi^k - 2\gamma^k = 0$	$\sqrt{\tau/\kappa}\bar{z}_j^k = \sqrt{\kappa/\tau}\bar{u}_j^k = 1,$ $\bar{\beta}_j^k = -\frac{(X'\psi^k - 2\gamma^k)_j}{\lvert(X'\psi^k - 2\gamma^k)_j\rvert}, \forall j \in \mathcal{J}_v^k$
	$\tau\mathbf{e} + \alpha^k - \delta^k \in \mathbb{R}_+^m$	$\sqrt{\tau/\kappa}\bar{z}_j^k = \sqrt{\kappa/\tau}\bar{u}_j^k = 1,$ $\bar{\beta}_j^k = \pm 1, \forall j \in \mathcal{J}_v^k$
	$\kappa\mathbf{e} - \alpha^k - \delta^k \in \mathbb{R}_+^m$	$\sqrt{\tau/\kappa}\bar{z}_j^k = \sqrt{\kappa/\tau}\bar{u}_j^k = 1,$ $\bar{\beta}_j^k = \pm 1, \forall j \in \mathcal{J}_v^k$

After constructing the vectors $\bar{\beta}^k$ of artificial solutions, their non-zero components are separated into different canonical-like vectors, each representing a new column in the master problem.

Since the aforementioned conditions do not depend on any results from relaxed problems, the attempts to solve these problems can be omitted when any component does not meet the boundedness criterion, saving the time of resolution in the decomposition method.

Chapter 4

Results

All the results presented in this chapter were obtained using a computer with the following specifications.

1. Processor: AMD Ryzen 7 4800HS, 2.90 GHz, with 8 cores and 16 threads.
2. RAM: 16.0 GB (15.4 GB in use).
3. OS: Windows 11 22H2, 64-bit.

4.1 Data

Concerning the data, synthetic instances were generated with multiple size options. Given a quantity n of observations and m desired features, the matrix of independent variables X is generated, where its columns are m different arrays of random variables following a normal distribution $X_i \sim \mathcal{N}(\mu_i, \sigma_i^2)$, with its parameters once again random variables distributed uniformly $\mu_i \sim U(-10, 10)$ and $\sigma_i \sim U(0, 5)$. The actual coefficients β are also generated with a uniform distribution $U(0, 1)$, similar to the aforementioned parameters, while the target vector y is obtained using the equation $y = X\beta$.

Another detail about the data is that after obtaining the target vector y, ten percent of the columns of X are transformed into noise. This noise follows a standard normal distribution $\mathcal{N}(0, 1)$, resulting in a new matrix X, which is the one used as input for all the models tested in the following sections results. The goal of this procedure is to resemble a more realistic source of data.

4.2 Comparison of the Decomposition Method by the Relaxed Problem

In the following subsections, the term $CG - L_{\mathcal{C}}$ is used for the instances of the Column Generation method described in Chapter 3, where $\mathcal{C}$ is one of the cones to relax. The configuration of parameters of the method is detailed in each subsection. The term SOCP represents the results of instances of the second-order cone problem (3.12). All these methods are modeled in *Python* 3.9.7, using the package CVXPY version 1.2.2 (Diamond & Boyd, 2016), and solved through the Python API for MOSEK version 10.0.29 (MOSEK ApS, 2023). All convergence tolerances, including the solver and the decomposition method, are set to $1e-6 = 1 \times 10^{-6}$.

4.2.1 Preliminary Execution Times by the Relaxed Problem

The results for execution times of the conic model and the three types of decomposition methods for each of the three relaxed problems presented in subsection 3.4.4, incorporating also the artificial solutions in cases of unbounded problems as described in subsection 3.4.6, are shown in Figures 4.1, 4.2, and 4.3. For all three figures, the parameter configurations are considered as $\tau = \kappa = \|y - X\beta_{OLS}\|_2^2$, $v_0 = 6$, $v = \max\{\lfloor 0.012m \rfloor, 5\}$. Here, β_{OLS} is the solution of ordinary least squares $\beta_{OLS} = (X'X)^{-1}X'y$. A 30-minute execution time limit is set for all methods for these results.

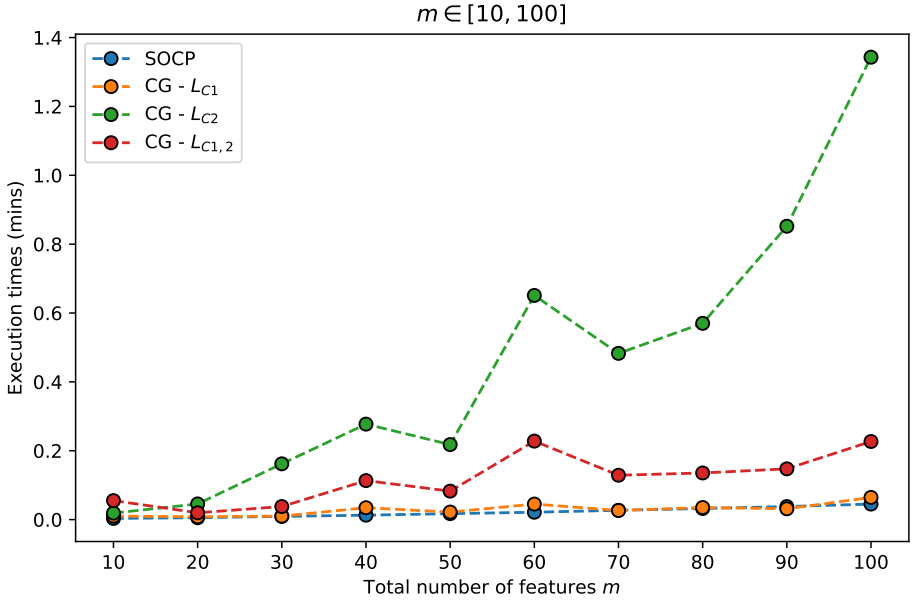

Figure 4.1: Execution times with synthetic data for the decomposition method for each of the three possibilities of relaxation and the second order cone problem (3.12), for different numbers of features $m \in [10, 100]$, and $n = 10000$ observations.

It can be observed that the decomposition method that relaxes the conic constraints

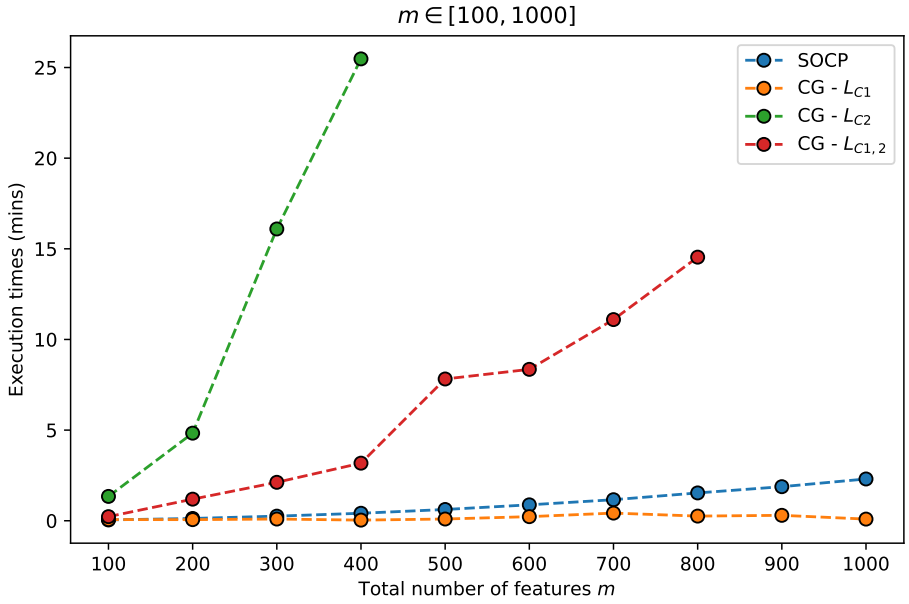

Figure 4.2: Execution times with synthetic data for the decomposition method for each of the three possibilities of relaxation and the second order cone problem (3.12), for different numbers of features $m \in [100, 1000]$, and $n = 10000$ observations.

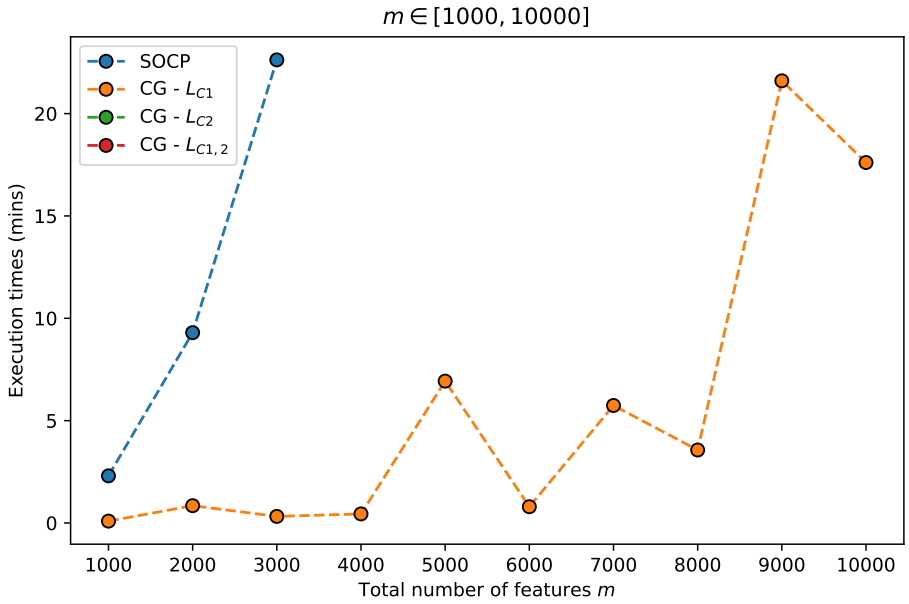

Figure 4.3: Execution times with synthetic data for the decomposition method for each of the three possibilities of relaxation and the second order cone problem (3.12), for different numbers of features $m \in [1000, 10000]$, and $n = 10000$ observations.

$\mathcal{C}_2$ is the slowest of all, reaching times exceeding 30 minutes for synthetic data instances with $m = 500$ or more features, which is well above the time of the SOCP (3.12), which is

able to solve the synthetic instance with $m = 500$ features in approximately 37 seconds. A similar situation occurs with the decomposition method that relaxes all second-order cone constraints $C_{1,2}$, which exceeds 30 minutes for instances with $m = 900$ or more features, while the SOCP is capable of solving the instance with $m = 900$ features in approximately 1 minute and 53 seconds. Neither of these two methods is able to improve the execution times of the original SOCP in any instance, rendering the decomposition method with these relaxations unnecessary.

Regarding the decomposition method that relaxes the second-order cone constraint $\mathcal{C}_1$, it can be seen that for instances with $m = 4000$ features or more, the decomposition method is systematically faster to solve than the SOCP, managing to solve an instance with $m = 10000$ features in less than 20 minutes, while the original conic problem exceeds 30 minutes of execution time for instances with $m = 4000$ features or more. It is necessary to highlight that in the instance with $m = 3000$ features, the decomposition method relaxing the constraints of $\mathcal{C}_1$ solves the problem in approximately 19 seconds, while the SOCP solves this instance in an approximate time of 22 minutes and 37 seconds. In other words, in this instance the original problem is 71.42 times slower than the decomposition method, showing that the decomposition method is capable, with this relaxation, of solving large instances of the original conic problem in considerably shorter times.

4.2.2 Optimal Objective Value by the Relaxed Problem

To confirm that the decomposition method is indeed delivering the optimal result for the different instances of the problem (3.12), it is also necessary to verify that in the previous instances, the method is delivering the same optimal objective value, given the tolerance. To do this, the counterparts of the figures from the previous subsection are presented for relative differences in the optimal objective values of each version of the decomposition method with respect to the second-order conic problem, in Figures 4.4, 4.5, and 4.6.

The relative difference of the optimal objective values ($RDOV$) of the decomposition method with a certain relaxation of conic constraints sets $\mathcal{C}$ ($CG - L_{\mathcal{C}}$) and the SOCP is defined as

$$RDOV_{CG-L_{\mathcal{C}}} = \frac{f(x^*_{CG-L_{\mathcal{C}}}) - f(x^*_{SOCP})}{f(x^*_{SOCP})}, \tag{4.1}$$

where $f(x^*_{CG-L_{\mathcal{C}}})$ is the optimal objective value of the $CG - L_{\mathcal{C}}$ method and $f(x^*_{SOCP})$ is the optimal objective value of the original conic problem.

It can be observed that the greatest deviation from the optimal objective value of the second-order cone problem occurs in Figure 4.4, with the decomposition method that relaxes the $\mathcal{C}_2$ constraint set in the instance with $m = 20$ attributes, where the relative difference is -8%, meaning the objective value of the $CG - L_{\mathcal{C}_2}$ method is 8% greater than that of the original problem. However, this is a small instance, so it is not expected that any decomposition method is needed for it. The instances of interest are those of large size, with $m = 1000$ or more attributes, and, as seen in Figure 4.6, the greatest relative difference among the three large instances is approximately -2×10^{-7}%, which, in practical terms,

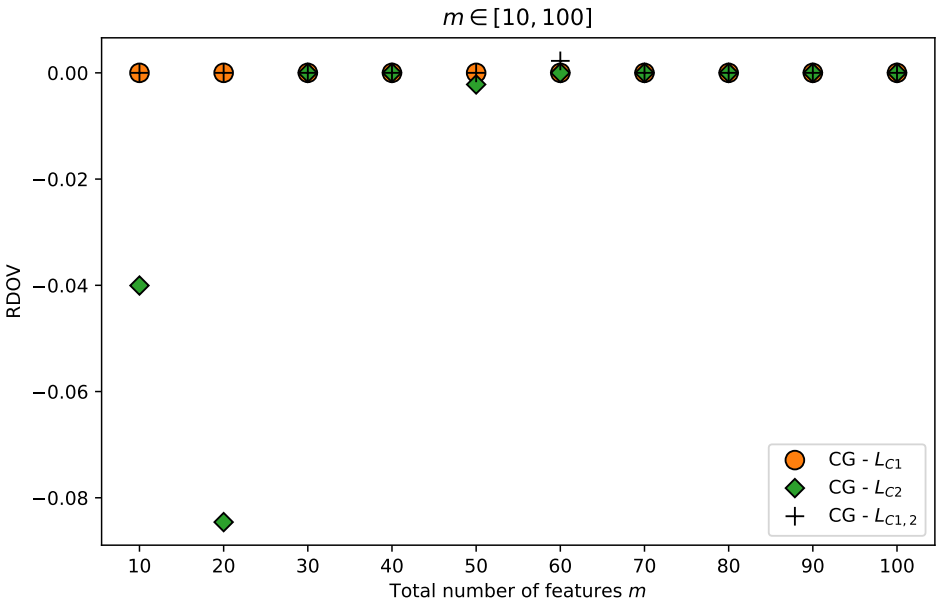

Figure 4.4: Relative difference of the optimal objective values with synthetic data for the decomposition method for each of the three possibilities of relaxation and the second order cone problem (3.12), for different numbers of features $m \in [10, 100]$, and $n = 10000$ observations.

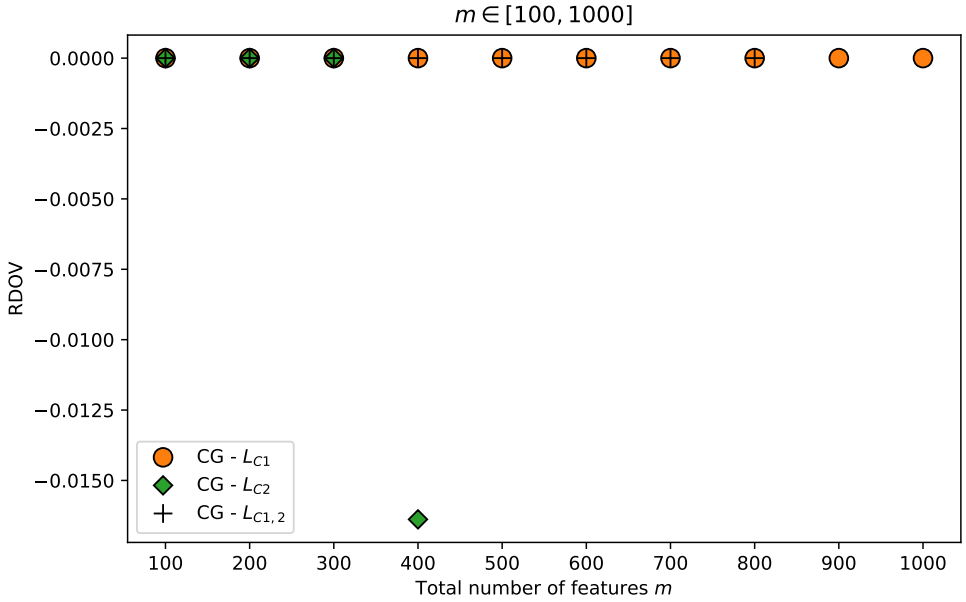

Figure 4.5: Relative difference of the optimal objective values with synthetic data for the decomposition method for each of the three possibilities of relaxation and the second order cone problem (3.12), for different numbers of features $m \in [100, 1000]$, and $n = 10000$ observations.

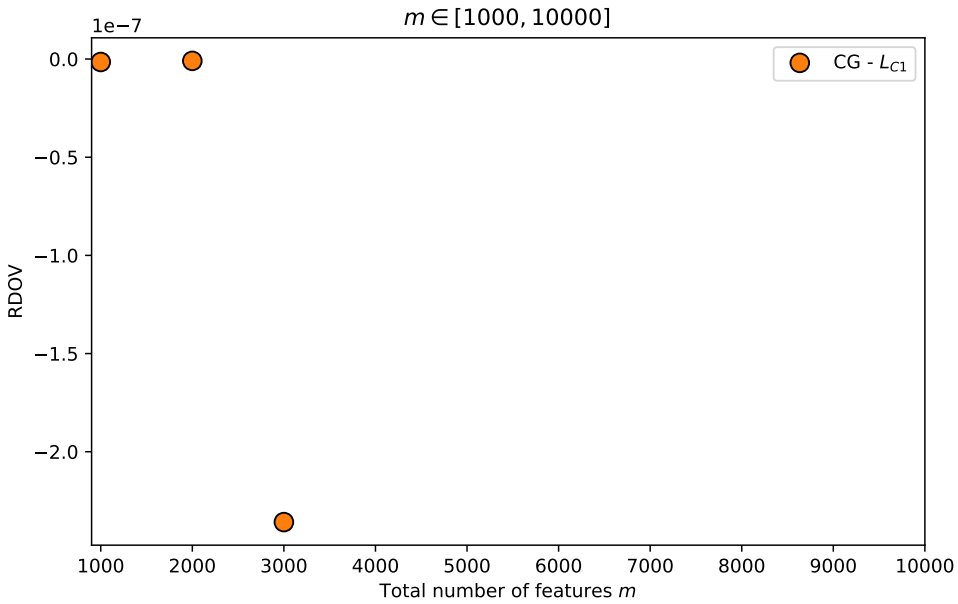

Figure 4.6: Relative difference of the optimal objective values with synthetic data for the decomposition method for each of the three possibilities of relaxation and the second order cone problem (3.12), for different numbers of features $m \in [1000, 10000]$, and $n = 10000$ observations.

given the predefined tolerance, is zero, so the optimal objective function is the same as the optimal objective value of the SOCP. It can be concluded that the optimal objective value of the decomposition methods in these instances are the same as that of the second-order conic problem, and therefore, in these instances, they are equivalent problems.

4.3 Review of Convergence Criteria in Relaxed Models

Regarding the convergence criteria of the different decomposition methods, there is an opportunity for improvement in terms of execution times. By knowing the boundedness conditions and the artificial solutions that should be implemented in each case, as shown in Table 3.1, it is possible to review these conditions prior to solving the respective relaxed problem. This approach involves avoiding the resolution of the relaxed problems when there is no finite solution and focuses only on generating the artificial solutions.

For the results presented in this subsection, the same aforementioned parameters are used, except for the penalty weights, which change to $\tau = \kappa = \|y - X\beta_{OLS}\|_2^2 / M^{0.6}$. This adjustment results in a smaller penalty depending on the instance size. The purpose of this adjustment is to show instances of the problem with more iterations, providing more opportunities to observe what happens with the relaxed problems.

4.3.1 Behavior of the Lagrange Relaxation over Iterations

To assess how often the boundedness conditions are not met, the optimal objective values and execution times of the master problem $P(\mathcal{S}^k)$ and the relaxed problem $L(\mathcal{X}_{\overline{\mathcal{C}_1}}, \lambda^k)$ are shown throughout the iterations of the $CG - L_{\mathcal{C}_1}$ method in Figures 4.7 and 4.8.

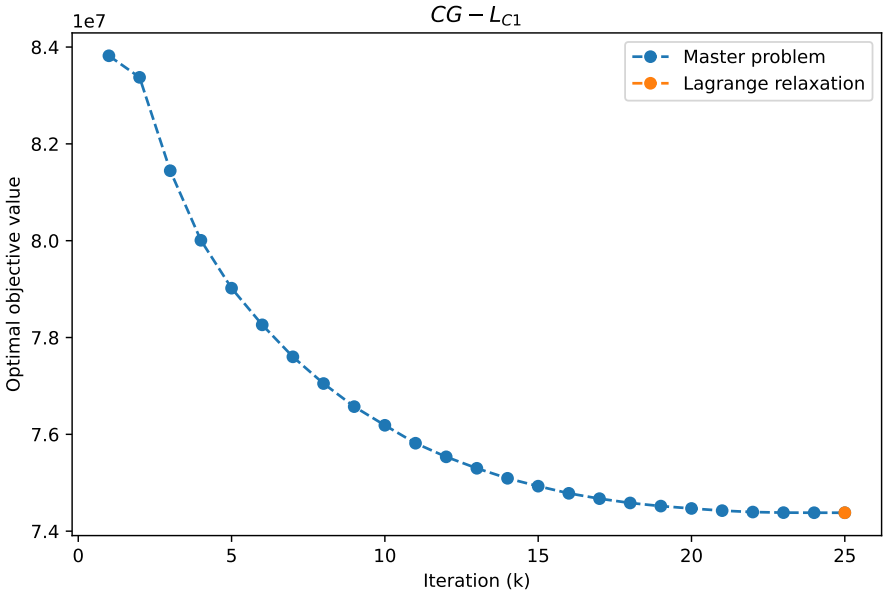

Figure 4.7: Optimal objective values of the subproblems by iteration of the $CG - L_{\mathcal{C}_1}$ method, without prior checking of boundedness conditions.

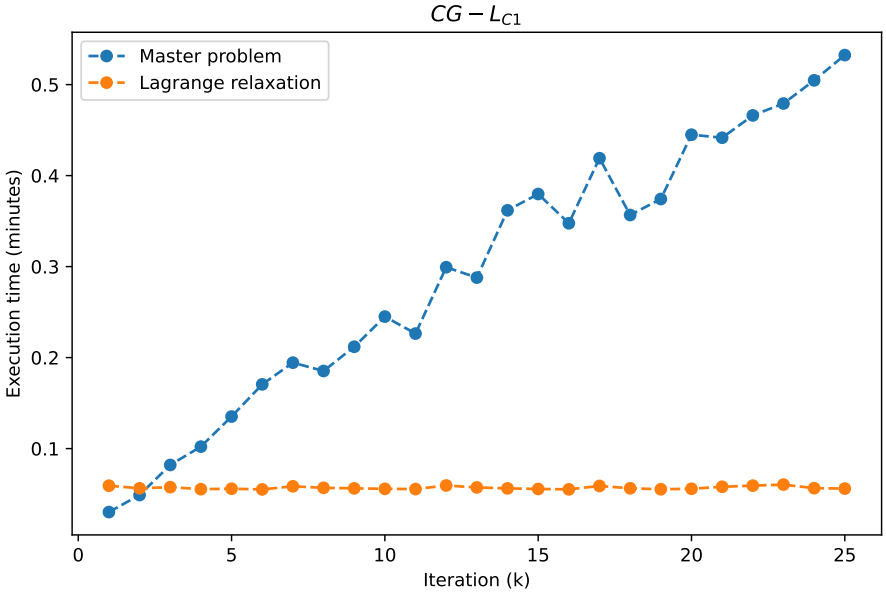

Figure 4.8: Execution times (in minutes) of the subproblems by iteration of the $CG - L_{\mathcal{C}_1}$ method, without prior checking of boundedness conditions.

These figures indicate that there is always an execution time for the Lagrange relaxation in each iteration, but not necessarily for its optimal value. This implies that the problem was solved, but the objective is unbounded in all iterations except the last one, rendering most of the time spent attempting to solve the relaxed problem futile. Execution times in each iteration, and the fraction invested in solving relaxed problems, are shown in detail in Table 4.1.

Table 4.1: Execution times (in minutes) and objective value for the decomposition method and its subproblems, without prior checking of boundedness conditions.

	t	Objective value
Decomposition Method	9.11	74328058.97
Master problem	7.41	74328058.97
Lagrange relaxation	1.45	74328058.98

It can be seen that the resolutions of the relaxed problems account for 15.91% of the total time of the method, making it worthwhile to attempt to reduce this time.

4.3.2 Omission of the Relaxed Problem with Boundedness Conditions

The results shown in Figures 4.9 and 4.10 are obtained for the same instance as in the previous subsection, but with the implementation of the prior checking of the boundedness conditions and omitting the relaxed problem when the conditions are not met. In this case, the boundedness condition correctly identifies cases where the relaxed problem is unbounded and attempts to solve it only in the last iteration, where the optimal value if finite.

For an aggregated comparison, the execution times in each subproblem of the method per iteration are presented in Table 4.2. In this table, it can be observed that the optimal objective value is the same as in Table 4.1, indicating that the same problem was solved, and in the same number of iterations as well. The time spent on solving the Lagrange relaxation of the problem decreased from 1.45 minutes to approximately 3.34 seconds. This value can be influenced by various factors, being only a single instance, but the difference is still substantial, clearly saving almost all the time of solving the Lagrange relaxation with this adjustment.

Table 4.2: Execution times (in minutes) and objective value for the decomposition method and its subproblems, with prior boundedness condition checking and omission of the relaxed problem.

	t	Objective value
Decomposition method	7.36	74328058.97
Master problem	7.24	74328058.97
Lagrange relaxation	0.06	74328058.98

To verify more instances and also to examine what happens when applying the same method with the remaining two methods, the execution times of the three decomposition

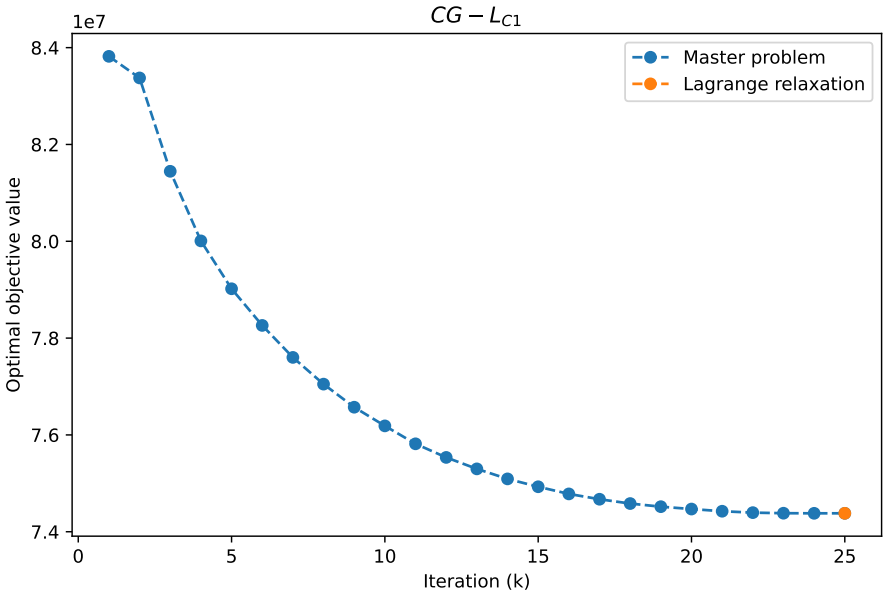

Figure 4.9: Optimal objective values of the subproblems by iteration of the $CG-L_{\mathcal{C}_1}$ method, with prior boundedness condition checking and omission of the relaxed problem.

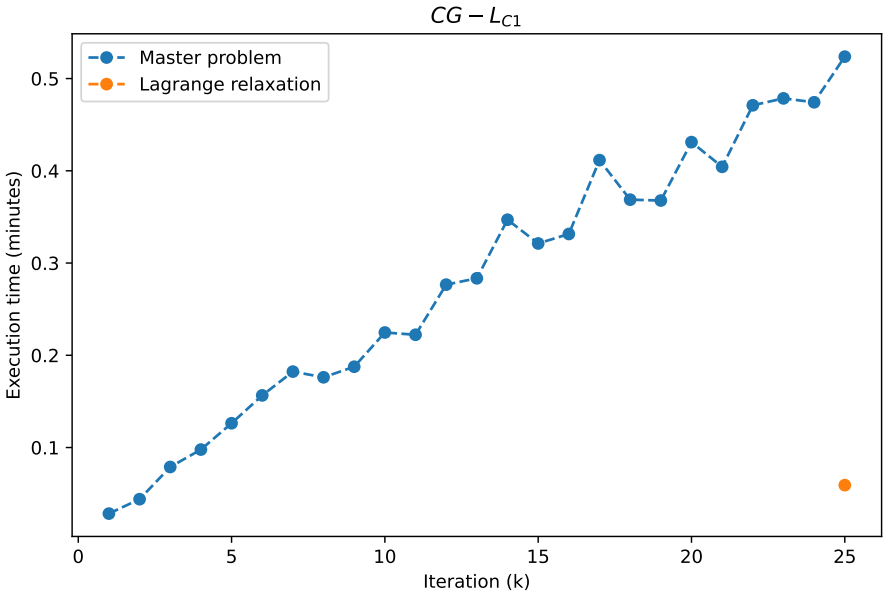

Figure 4.10: Execution times (in minutes) of the subproblems by iteration of the $CG - L_{\mathcal{C}_1}$ method, with prior boundedness condition checking and omission of the relaxed problem.

methods for different instance sizes are presented in Table 4.3. The table also includes the *RDOV* between the two alternatives of decomposition method, to show that they are equivalent.

It is clear that even with the omission of the Lagrange relaxation, the alternatives

Table 4.3: Execution times comparison between the different decomposition methods alternatives, with and without prior boundedness condition checking and $L(\mathcal{X}_{\overline{\mathcal{C}}}, \lambda^k)$ omission.

Relaxation	m	t normal	t pre-check	$RDOV$
$L(\mathcal{X}_{\overline{\mathcal{C}_1}}, \lambda^k)$	100	0.30	0.21	0
	200	0.72	0.52	0
	300	2.07	1.57	0
	400	4.58	3.64	0
	500	4.77	3.76	0
	600	4.47	3.65	0
	700	5.29	4.09	0
	800	6.25	4.89	0
	900	6.79	5.31	0
$L(\mathcal{X}_{\overline{\mathcal{C}_2}}, \lambda^k)$	100	1.03	1.02	0
	200	4.9	4.94	0
	300	13.81	13.45	0
	400	–	–	–
	500	–	–	–
	600	–	–	–
	700	–	–	–
	800	–	–	–
	900	–	–	–
$L(\mathcal{X}_{\overline{\mathcal{C}_{1,2}}}, \lambda^k)$	100	2.44	2.41	0
	200	3.43	3.38	0
	300	4.63	4.61	0
	400	–	–	–
	500	–	–	–
	600	–	–	–
	700	–	–	–
	800	–	–	–
	900	–	–	–

$L(\mathcal{X}_{\overline{\mathcal{C}_2}}, \lambda^k)$ and $L(\mathcal{X}_{\overline{\mathcal{C}_{1,2}}}, \lambda^k)$ cannot surpass in efficiency the decomposition method with the relaxation problem $L(\mathcal{X}_{\overline{\mathcal{C}_1}}, \lambda^k)$.

4.4 Parameter Tuning of the Method

The parameters of the decomposition method that can be varied to solve the same problem are mostly parameters that affect the construction of the solution sets $\mathcal{S}^k$. The focus will be on analyzing the results for different values of these parameters, in order to select those parameters that decrease execution times without significantly affecting the performance of the method.

4.4.1 Initial Solutions $\mathcal{S}^0$

The initial solution set, as mentioned in the construction of the $\mathcal{S}^k$ sets in subsection 3.4.2, has different options for its construction, which are divided into the following two subsections of results.

4.4.1.1 Form of the Initial solutions

In first place, the differences in the decomposition method with the relaxation $L(\mathcal{X}_{\overline{\mathcal{C}_1}}, \lambda^k)$ are analyzed with different ways of constructing the initial solutions, or the initial feasible set $\mathcal{S}^0$. As mentioned in the Methodology Chapter 3, the following two options are analyzed for its construction.

1. Adding to the initial matrix $\mathbf{B}^0$ a column with a vector of constant solutions with coefficients of absolute value one. That is, adding a solution such that $|\beta_i^0| = 1, \forall i \in \{1, \ldots, m\}$, and for the matrices $\mathbf{Z}^0$ and $\mathbf{U}^0$, taking vectors z^0, u^0 with all their components equal to 1. Since a sign must be chosen for the components of β^0, but it is not known which sign will behave better in each component, the sign of each of these components is randomly chosen, but the positive value is maintained in the components of the other two solution vectors.
2. Adding to the initial matrix $\mathbf{B}^0$ a quantity v_0 of vectors with a single component of absolute value one, but with different signs in the component associated with β_0, as in the previous case, and the rest of the components being zero. For the case of matrices $\mathbf{Z}^0$ and $\mathbf{U}^0$, canonical vectors with a one in the same non-zero component of the β^0 are added. To have baseline for this type of solutions, the option of constructing solutions with random components with values 0, 1, and -1 for the β^0 vectors will also be considered, while the other two vectors are filled with ones in the non-zero components of β^0.

In Table 4.4, the average execution time results, in minutes, for different instances are presented, with 10 attempts for each parameter configuration and number of attributes. The table also presents the number of successful instances, the execution time deviation, the average number of selected attributes, the average number of iterations, and the average objective value, in order to analyze the behavior of the different configurations more accurately. All the parameters no shown in the table are the same as stated in the last results, except for $\tau = \kappa = \|y - X\beta_{OLS}\|_2^2 / M^{0.4}$ and $v_0 = \max\{\lfloor 0.02m \rfloor, 5\}$.

It is observed that the canonical solutions speed up the method in any case, while the constant solution seems to increase execution times, which is seen even more clearly in instances with canonical initial solutions. However, instances without a constant solution in the canonical solutions results seem to be less stable, as they are opportunities where the number of features selected has the most difference with the rest of the configurations. The solution seems to stabilize by adding a constant solution, as the number of selected attributes remains similar in any case, while in instances without a constant solution, there

Table 4.4: Average execution time and other metrics for different configurations of the initial solutions form of the method for different data sizes, with $n = 10000$.

Configuration			Metrics					
m	Const. sol.	Canonical sol.	# success	Mean t	t std.	# feat.	# iter.	Objective value
1000	False	False	10	0.239	0.045	3.0	2.2	8.371744e+07
		True	10	0.147	0.015	3.1	2.2	8.371744e+07
	True	False	10	0.234	0.049	3.0	2.2	8.371744e+07
		True	10	0.265	0.068	3.0	2.7	8.371744e+07
2000	False	False	10	2.686	0.473	98.0	6.3	2.095731e+08
		True	10	0.813	0.084	98.5	6.0	2.095731e+08
	True	False	10	2.703	0.496	98.0	6.4	2.095731e+08
		True	10	2.369	0.436	98.0	6.1	2.095731e+08
3000	False	False	10	2.991	0.346	53.0	4.1	2.632429e+08
		True	10	0.644	0.023	57.3	3.0	2.632429e+08
	True	False	10	3.441	0.785	53.0	4.5	2.632429e+08
		True	10	1.852	0.241	54.5	3.1	2.632429e+08
4000	False	False	10	2.207	0.408	3.1	2.1	3.262692e+08
		True	10	0.673	0.013	28.7	2.0	3.262692e+08
	True	False	10	2.085	0.027	3.0	2.0	3.262692e+08
		True	10	1.773	0.019	3.1	2.0	3.262692e+08
5000	False	False	10	7.647	0.075	84.0	4.0	4.486613e+08
		True	10	1.172	0.026	90.3	3.0	4.486613e+08
	True	False	10	7.987	0.856	83.8	4.1	4.486613e+08
		True	10	4.272	0.026	84.3	3.0	4.486613e+08

is greater variability among configurations. However, the solutions appear to be numerically equivalent, as the objective function is the same in terms of tolerance.

In principle, the configuration with only canonical solutions is chosen as the best, prioritizing execution time, and in instances with greater convergence instability, the constant solution can also be added to the initial solution matrices.

4.4.1.2 Number of Initial Solutions

The second parameter to vary in terms of the construction of the initial feasible set $\mathcal{S}^0$ is the number of canonical initial solutions to consider. In other words, varying the size of the parameter v_0. To do this, it is changed in relation to the number of parameters in the problem. If $v_0 = m$ is chosen, then the original conic problem is obtained, so a considerably smaller quantity must be chosen, considering the assumption of large size m.

In Table 4.5, the results for instances with v_0 between 1 and $m/4+1$ are shown. It can be observed that as the fraction of canonical initial solutions $\frac{v_0}{m}$ increases, the method's resolution time also increases. However, choosing a single initial solution result in less stable solution, as some of these instances had some numerical errors, and the average number of selected attributes changes drastically (not the objective value). That being said, it is preferable to choose a small value but with greater consistency in the solutions. The procedure is then repeated with smaller intervals.

In the Table 4.6, it can be observed that the average execution times seem to be lower in instances with v_0 between 1 and $m/20 + 1$, with instances with a single solution again having many failed instances, while for v_0 close to between 1 and 2% of m, all instances are successful. However, the differences in execution times between these configurations are not

Table 4.5: Average execution time and other metrics for different configurations of the number of initial solutions of the method for different data sizes, with $n = 10000$ and $v_0 \in [1, \frac{m}{4} + 1]$.

Configuration		Metrics					
m	v_0	# success	Mean t	t std.	# feat.	# iter.	Objective value
1000	1	10	0.123	0.004	5.100	2.000	8.371673e+07
	51	10	0.185	0.028	3.200	2.100	8.371673e+07
	101	10	0.238	0.020	3.100	2.000	8.371673e+07
	151	10	0.304	0.046	3.100	2.100	8.371673e+07
	201	10	0.374	0.044	3.100	2.100	8.371673e+07
	251	10	0.465	0.070	3.900	2.100	8.371673e+07
2000	1	7	0.648	0.045	96.000	5.857	2.127228e+08
	101	10	0.839	0.025	96.800	5.000	2.127228e+08
	201	10	1.268	0.058	95.600	5.000	2.127228e+08
	301	10	1.761	0.051	95.600	5.000	2.127228e+08
	401	10	2.416	0.078	94.300	5.000	2.127228e+08
	501	10	3.106	0.226	95.000	4.900	2.127228e+08
3000	1	7	0.595	0.075	62.857	3.429	2.658655e+08
	151	10	0.814	0.025	52.900	3.000	2.658656e+08
	301	10	1.274	0.054	52.800	3.000	2.658656e+08
	451	10	1.874	0.066	52.400	3.000	2.658656e+08
	601	10	2.657	0.100	53.300	3.000	2.658656e+08
	751	10	3.641	0.120	53.000	3.000	2.658656e+08
4000	1	9	0.571	0.007	35.889	2.000	3.262936e+08
	201	10	0.827	0.023	11.600	2.000	3.262936e+08
	401	10	1.304	0.074	9.700	2.000	3.262936e+08
	601	9	1.980	0.066	8.111	2.000	3.262936e+08
	801	10	2.870	0.097	10.300	2.000	3.262936e+08
	1001	10	4.092	0.171	13.600	2.000	3.262936e+08
5000	1	5	1.140	0.123	122.000	3.600	4.558796e+08
	251	10	1.640	0.042	87.200	3.000	4.558797e+08
	501	10	2.783	0.069	90.900	3.000	4.558797e+08
	751	9	4.418	0.063	110.889	3.000	4.558797e+08
	1001	10	6.781	0.721	94.100	3.100	4.558797e+08
	1251	10	9.028	0.196	123.300	3.000	4.558797e+08

significantly large, with instances close to 2% of m appearing to take more time in general. In consequence, any value within this parameter range will be used in the following results.

4.4.2 Feasible Region $\mathcal{S}^k$

For the construction of the feasible region of the master problem, $\mathcal{S}^k$, various options are considered, as previously described.

1. The four different types of solution combinations described in subsection 3.4.2.
2. The number of artificial solutions to be added in iterations where the Lagrange relaxation is unbounded, as described in subsection 3.4.6.

Table 4.6: Average execution time and other metrics for different configurations of the number of initial solutions of the method for different data sizes, with $n = 10000$ and $v_0 \in [1, \frac{m}{20} + 1]$.

Configuration		Metrics					
m	v_0	# success	Mean t	t std.	# feat.	# iter.	Objective value
1000	1	10	0.122	0.004	5.100	2.000	8.371673e+07
	11	10	0.132	0.014	3.100	2.100	8.371673e+07
	21	10	0.135	0.006	3.100	2.000	8.371673e+07
	31	10	0.149	0.003	3.300	2.000	8.371673e+07
	41	10	0.160	0.011	3.200	2.000	8.371673e+07
	51	10	0.170	0.012	3.200	2.000	8.371673e+07
2000	1	7	0.649	0.014	98.286	6.000	2.127228e+08
	21	10	0.590	0.012	95.900	5.000	2.127228e+08
	41	10	0.668	0.039	96.400	5.100	2.127228e+08
	61	10	0.712	0.023	96.900	5.000	2.127228e+08
	81	10	0.779	0.048	97.000	5.100	2.127228e+08
	101	10	0.842	0.044	96.200	5.000	2.127228e+08
3000	1	9	0.621	0.066	58.778	3.556	2.658655e+08
	31	10	0.581	0.015	53.200	3.000	2.658656e+08
	61	10	0.618	0.015	53.100	3.000	2.658656e+08
	91	10	0.677	0.014	53.000	3.000	2.658656e+08
	121	10	0.740	0.026	52.700	3.000	2.658656e+08
	151	10	0.816	0.015	53.200	3.000	2.658656e+08
4000	1	8	0.579	0.005	40.750	2.000	3.262936e+08
	41	10	0.618	0.007	15.300	2.000	3.262936e+08
	81	10	0.671	0.019	8.900	2.000	3.262936e+08
	121	10	0.727	0.026	15.300	2.000	3.262936e+08
	161	10	0.784	0.021	9.800	2.000	3.262936e+08
	201	10	0.839	0.023	9.000	2.000	3.262936e+08
5000	1	3	1.136	0.141	116.333	3.667	4.558797e+08
	51	10	1.058	0.014	89.300	3.000	4.558797e+08
	101	10	1.156	0.018	86.800	3.000	4.558797e+08
	151	9	1.268	0.025	91.111	3.000	4.558797e+08
	201	10	1.426	0.039	88.300	3.000	4.558797e+08
	251	10	1.554	0.035	88.100	3.000	4.558797e+08

4.4.2.1 Solutions Combination

There are four options of combination to use: convex combination, affine combination, conic combination, and linear combination. These combinations can be summarized with two distinct constraints on the variables π of the master problem $P(\mathcal{S}^k)$ (3.13): $\sum_j \pi_j = 1$ and $\pi \geq 0$.

Table 4.7 presents the same results as the previous tables, but for the different configurations of the two combination constraints mentioned above. Two critical behaviors can be observed with respect to the convex and affine combinations, which include the constraint $\sum_j \pi_j = 1$. In these two combinations, there are many instances that fail, and some configurations even have no instances with results. The second peculiar behavior is that in these same combinations, the optimal value of the objective function is different from that of the conic and linear combinations. Upon closer examination of the successful instances of the convex and affine combinations, it is observed that the artificial solutions created in the last iterations are repeated throughout them. In other words, no new artificial solutions are added, but the same vectors are repeated, causing the master problem to have the same feasible region in the final iterations of the instance ($\mathcal{S}^{k+1} = \mathcal{S}^k$). This situation does not occur with the conic and linear combinations, where there are always new solutions among the artificial solutions.

The aforementioned problem can be largely explained by the nature of the artificial solutions, as these solutions consist of vectors with unitary or null magnitude components. A convex combination among them can only generate coefficient values $\beta^k = \mathbf{B}^k\pi$ with components between -1 and 1 ($-1 \leq \beta_i^k \leq 1$) and components between 0 and 1 for z^k and u^k. In other words, with this type of combination and artificial solutions, results for general problems cannot be obtained, quickly leading the problem to be stuck in the best possible solution of its feasible region, which is certainly not the optimum of the original problem, which has an optimum with coefficients of different magnitudes. As for the affine combination, it is not the same problem, but it is very similar since it also restricts the value of the coefficients with each other, where, for a coefficient to have a large magnitude, there must be enough coefficients different from it, with opposite signs, so that these magnitudes are counterbalanced in the constraint $\sum_j \pi_j = 1$. Consequently, the complete space of possible solutions is distorted, and the same with the optimal result.

Table 4.7: Average execution time and other metrics for different combinations of previous solutions of the method for different data sizes, with $n = 10000$.

Configuration			Metrics					
m	$\sum_j \pi_j = 1$	$\pi \geq 0$	# success	Mean t	t std.	# feat.	# iter.	Objective value
1000	False	False	10	0.142	0.018	3.200	2.200	8.371673e+07
		True	10	0.133	0.006	3.400	2.000	8.371673e+07
	True	False	10	0.271	0.021	3.000	3.100	8.371673e+07
		True	10	2.073	0.167	3.900	13.600	8.371673e+07
2000	False	False	10	0.530	0.013	94.800	4.000	2.127228e+08
		True	10	0.586	0.119	94.600	4.200	2.127228e+08
	True	False	0	–	–	–	–	–
		True	1	0.477	–	43.000	5.000	1.238704e+09
3000	False	False	10	0.615	0.076	55.400	3.000	2.658656e+08
		True	10	0.613	0.008	61.200	3.000	2.658656e+08
	True	False	10	0.516	0.075	42.900	3.800	4.611430e+08
		True	8	0.542	0.076	39.000	3.875	4.611430e+08
4000	False	False	10	0.675	0.010	20.500	2.000	3.262936e+08
		True	10	0.680	0.013	26.100	2.000	3.262936e+08
	True	False	10	1.794	0.408	6.500	2.800	3.262688e+08
		True	0	–	–	–	–	–
5000	False	False	10	1.162	0.032	110.300	3.000	4.558797e+08
		True	10	1.155	0.015	114.700	3.000	4.558797e+08
	True	False	3	1.129	0.373	95.000	4.333	1.129547e+09
		True	2	1.032	0.417	74.500	4.000	1.129547e+09

Given that combinations with the constraint $\sum_j \pi_j = 1$ are not suitable for problems of this type, one can continue analyzing Table 4.7 regarding the conic ($\pi \geq 0$) and linear combinations. It can be observed that the execution times for these two types of combinations are quite similar, so it cannot be concluded that one combination is better than the other. Initially, to minimize the number of constraints on the master problem, the linear combination, without constraints on π, is chosen as the default for the decomposition method.

4.4.2.2 Number of Artificial Solutions

The last parameter that influences the construction of the feasible set $\mathcal{S}^k$ is the number of artificial solutions v added in each iteration where the Lagrange relaxation is unbounded.

Similar to v_0, the results for different configurations of v are presented, with values between 1 and $m/4 + 1$ solutions in Table 4.8, where an increase in execution time can be observed as the value of v increases relative to m, but with a decrease between the single solution and 5% of the total attribute count m. It can be seen that the best performance in terms of execution time efficiency is obtained between 1 and 10%, with less consistent solutions in configurations with a single solution.

Table 4.8: Average execution time and other metrics for different configurations of the number of artificial solutions added in iterations with unbounded relaxation for different data sizes, with $n = 10000$ and $v \in [1, \frac{m}{4} + 1]$.

Configuration		Metrics					
m	v	# success	Mean t	t std.	# feat.	# iter.	Objective value
1000	1	10	0.191	0.004	3.000	4.0	8.371673e+07
	51	10	0.158	0.026	3.100	2.1	8.371673e+07
	101	10	0.180	0.016	3.100	2.0	8.371673e+07
	151	10	0.216	0.022	3.000	2.0	8.371673e+07
	201	10	0.284	0.080	3.100	2.1	8.371673e+07
	251	10	0.295	0.025	3.000	2.0	8.371673e+07
2000	1	10	8.542	0.193	92.100	91.7	2.127228e+08
	101	10	0.484	0.013	98.500	3.0	2.127228e+08
	201	10	0.638	0.163	99.200	2.7	2.127228e+08
	301	10	0.667	0.261	103.800	2.3	2.127228e+08
	401	10	1.001	0.678	102.200	2.4	2.127228e+08
	501	10	1.032	0.481	110.900	2.2	2.127228e+08
3000	1	10	6.226	0.085	51.300	51.7	2.658656e+08
	151	10	0.528	0.013	65.600	2.0	2.658656e+08
	301	10	0.682	0.031	65.500	2.0	2.658656e+08
	451	10	0.979	0.316	61.900	2.1	2.658656e+08
	601	10	1.160	0.038	54.500	2.0	2.658656e+08
	751	10	1.497	0.052	56.800	2.0	2.658656e+08
4000	1	10	0.852	0.061	3.100	3.8	3.262936e+08
	201	10	0.736	0.015	26.900	2.0	3.262936e+08
	401	9	0.995	0.029	3.667	2.0	3.262936e+08
	601	10	1.373	0.091	4.500	2.0	3.262936e+08
	801	9	1.840	0.050	7.222	2.0	3.262936e+08
	1001	10	2.370	0.096	6.000	2.0	3.262936e+08
5000	251	10	1.312	0.308	184.000	2.5	4.558797e+08
	501	10	1.412	0.032	189.900	2.0	4.558797e+08
	751	10	1.961	0.054	297.500	2.0	4.558797e+08
	1001	10	2.638	0.075	201.000	2.0	4.558797e+08
	1251	10	3.564	0.102	132.700	2.0	4.558797e+08

To further analyze the aforementioned range in detail, Table 4.9 shows the results for different smaller intervals between 1 and approximately 10% of the available m attributes in the dataset. This table shows that the shortest times are obtained with a quantity of artificial solutions v ranging from approximately $2 - 5\%$ of m, with seemingly shorter times at 3%, but with differences not significant enough to conclude that this percentage is the best value for v. That being said, the values within the $2 - 5\%$ interval of the total number of attributes m are considered sufficiently effective for the decomposition method.

4.5 Model Penalty Variation and Final Comparisons

In this section, the behavior of solutions and the execution times of the decomposition method will be analyzed as the penalty parameters of the problem, τ and κ, are varied. These parameters modify the problem by assigning different weights to the objective function for

Table 4.9: Average execution time and other metrics for different configurations of the number of artificial solutions added in iterations with unbounded relaxation for different data sizes, with $n = 10000$ and $v \in [1, \frac{m}{10} + 1]$.

Configuration		Metrics					
m	v	# success	Mean t	t std.	# feat.	# iter.	Objective value
1000	1	10	0.198	0.007	3.000	4.000	8.371673e+07
	17	10	0.152	0.027	3.100	2.200	8.371673e+07
	33	10	0.146	0.014	3.100	2.100	8.371673e+07
	49	10	0.160	0.012	3.100	2.000	8.371673e+07
	65	10	0.159	0.009	3.100	2.000	8.371673e+07
	81	10	0.170	0.009	3.200	2.000	8.371673e+07
	97	10	0.189	0.019	3.000	2.000	8.371673e+07
2000	1	9	8.659	0.192	92.444	92.000	2.127228e+08
	34	10	0.525	0.010	97.200	4.000	2.127228e+08
	67	10	0.482	0.090	97.500	3.100	2.127228e+08
	100	10	0.490	0.016	100.900	3.000	2.127228e+08
	133	10	0.607	0.114	98.000	3.100	2.127228e+08
	166	10	0.596	0.115	97.800	2.800	2.127228e+08
	199	10	0.566	0.172	102.100	2.500	2.127228e+08
3000	1	10	6.408	0.091	51.100	51.900	2.658656e+08
	51	10	0.605	0.005	53.600	3.000	2.658656e+08
	101	10	0.547	0.101	60.800	2.300	2.658656e+08
	151	10	0.556	0.092	62.400	2.100	2.658656e+08
	201	10	0.742	0.202	67.900	2.400	2.658656e+08
	251	10	0.682	0.153	70.800	2.100	2.658656e+08
	301	10	0.697	0.026	64.000	2.000	2.658656e+08
4000	1	10	0.891	0.008	3.000	4.000	3.262936e+08
	67	10	0.648	0.015	7.000	2.000	3.262936e+08
	133	10	0.693	0.018	26.600	2.000	3.262936e+08
	199	10	0.757	0.011	7.000	2.000	3.262936e+08
	265	10	0.818	0.018	34.400	2.000	3.262936e+08
	331	10	0.914	0.030	10.000	2.000	3.262936e+08
	397	10	1.014	0.032	7.800	2.000	3.262936e+08
5000	1	0	–	–	–	–	–
	84	10	1.128	0.017	97.300	3.000	4.558797e+08
	167	10	1.094	0.193	163.000	2.400	4.558797e+08
	250	9	1.152	0.254	152.222	2.222	4.558797e+08
	333	9	1.313	0.346	186.333	2.222	4.558797e+08
	416	9	1.273	0.048	201.778	2.000	4.558797e+08
	499	8	1.394	0.020	112.500	2.000	4.558797e+08

the variables that bound the coefficients β. Consequently, the number of selected attributes should change as these values are adjusted, leading the method to select fewer attributes as the values of τ and κ increase.

The rest of the model parameters are chosen among the best alternatives explored in the previous subsections, so it can have the best performance possible in average when comparing the decomposition method with its two equivalent problems. That is, the original second-order cone problem (3.12) and the state of the art feature selection model, LASSO (2.15). The latter is the relaxed version of the LASSO model that uses Coordinate Descent to solve the problem as proposed by Friedman et al. (2010), which is implemented in the *Python* package Scikit-Learn version 0.24.2 for Machine Learning (Pedregosa et al., 2011).

The different instances to compare the three models consider changes on the penalty parameters $\tau = \kappa$, and also on the number of original features m to have a wider range of results. The results of the mean execution times of the three models with $\tau, \kappa \in [\|y - X\beta_{OLS}\|_2^2/M^{0.7}, \|y - X\beta_{OLS}\|_2^2/M^{0.1}]$, $m \in [1000, 5000]$, $v = \max\{\lfloor 0.03m \rfloor, 5\}$, $v_0 = \max\{\lfloor 0.012m \rfloor, 5\}$, $n = 10000$ observations, linear combination of solutions, no constant initial solution, and

canonical initial solutions are shown in Figures 4.11, 4.12, 4.13, 4.14, and 4.15.

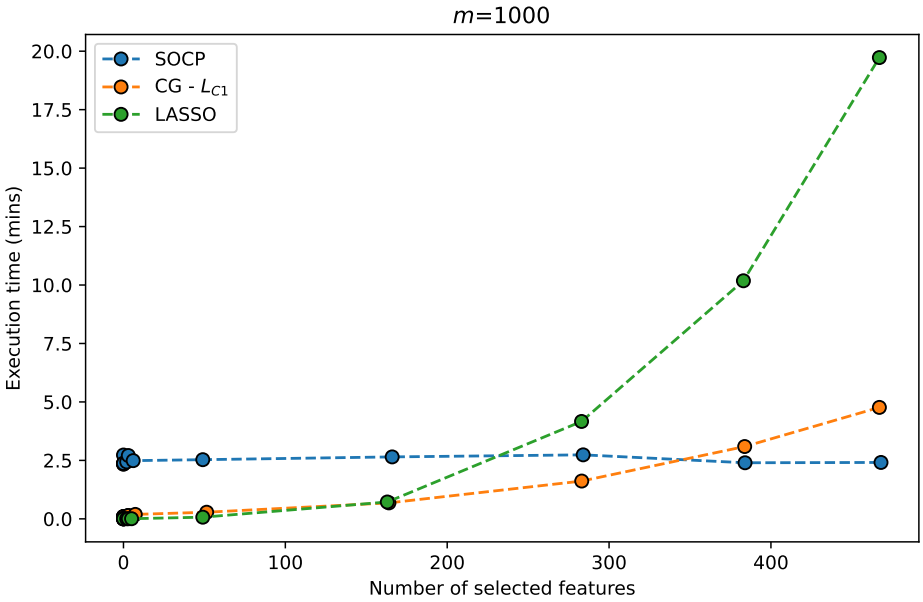

Figure 4.11: Average execution times of the three equivalent models in terms of the number of selected features in each instance, for $m = 1000$. Each instance is executed 10 times.

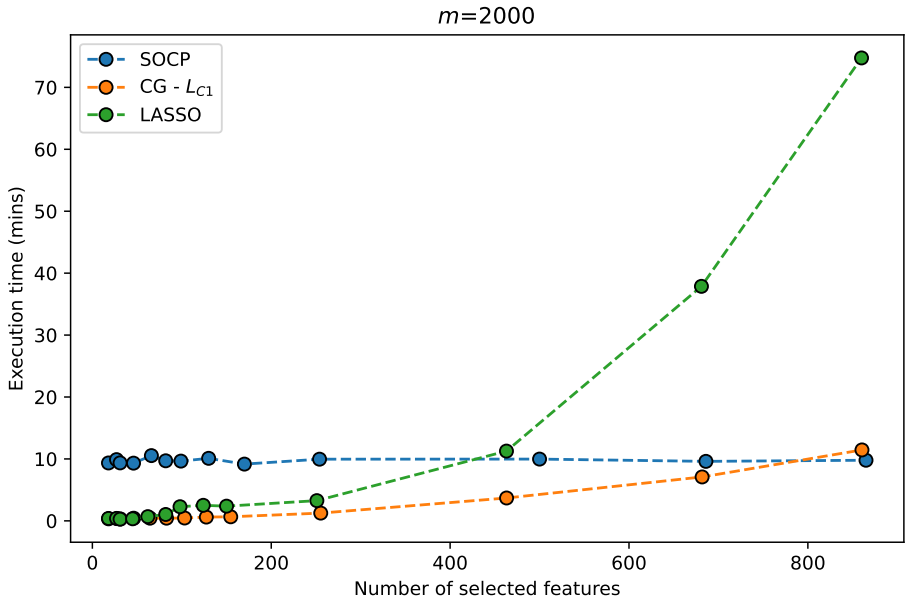

Figure 4.12: Average execution times of the three equivalent models in terms of the number of selected features in each instance, for $m = 2000$. Each instance is executed 10 times.

Given the results shown in these five Figures, is easy to see that the fastest method in the instances where just a few features are selected is the LASSO model solved with Coordinate Descent. When the number of features selected starts to grow, the fastest method is the

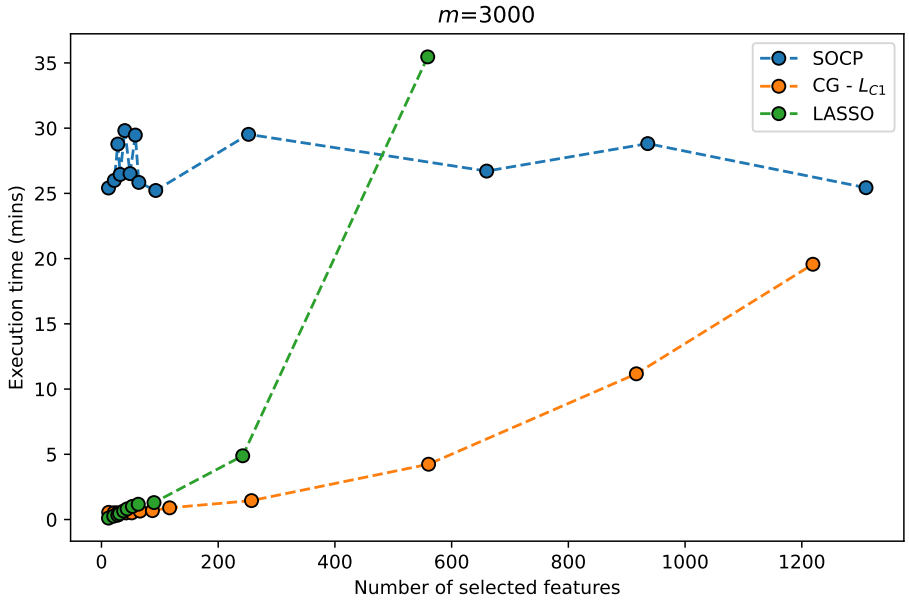

Figure 4.13: Average execution times of the three equivalent models in terms of the number of selected features in each instance, for $m = 3000$. Each instance is executed 10 times.

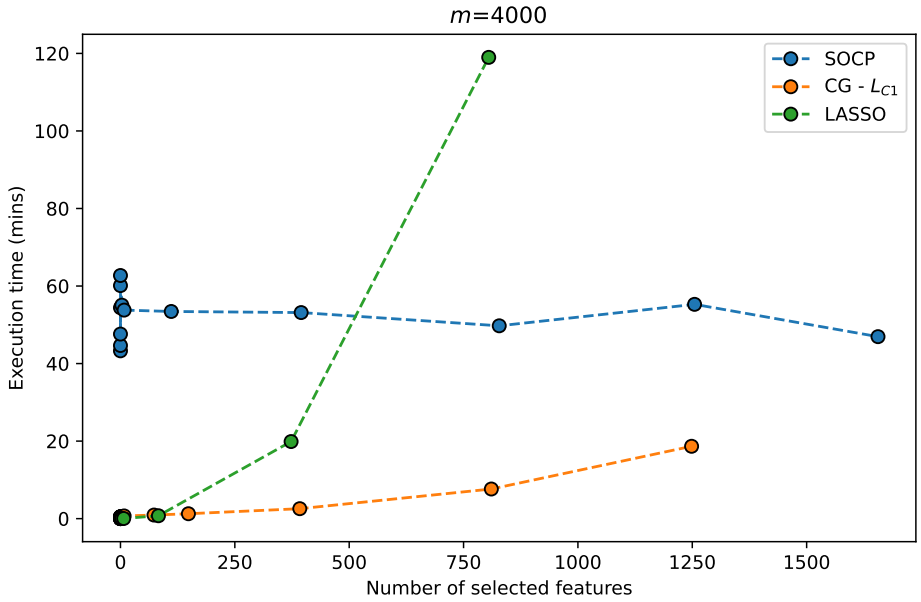

Figure 4.14: Average execution times of the three equivalent models in terms of the number of selected features in each instance, for $m = 4000$. Each instance is executed 10 times.

decomposition method proposed in this work. On the contrary, for instances where the number of features selected is around 40% or more of the total m, the original second-order cone problem appears to be the fastest. This threshold cannot be confirmed for instances with m greater than 3000, because the method starts to have numerical issues when that

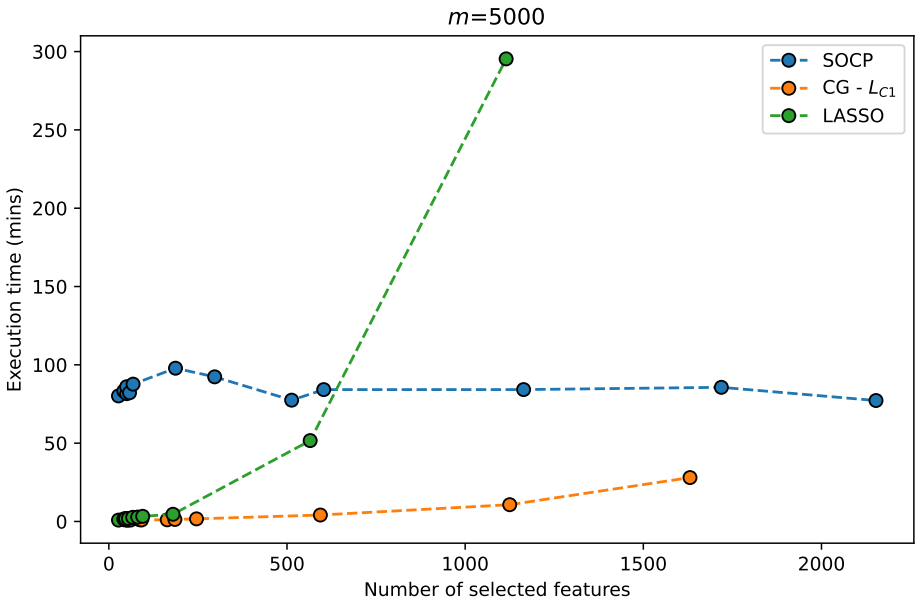

Figure 4.15: Average execution times of the three equivalent models in terms of the number of selected features in each instance, for $m = 5000$. Each instance is executed 10 times.

percentage of features is selected in bigger instances.

Figures 4.16 and 4.17 show the aggregated results for the same instances as the previous figures, and also the logarithm of execution times on the second figure. It is clear that the execution times of the Coordinate Descent method and the conic decomposition method on these instances depend only on the number of features selected, while the execution time of the original problem only increases with the number of original features m.

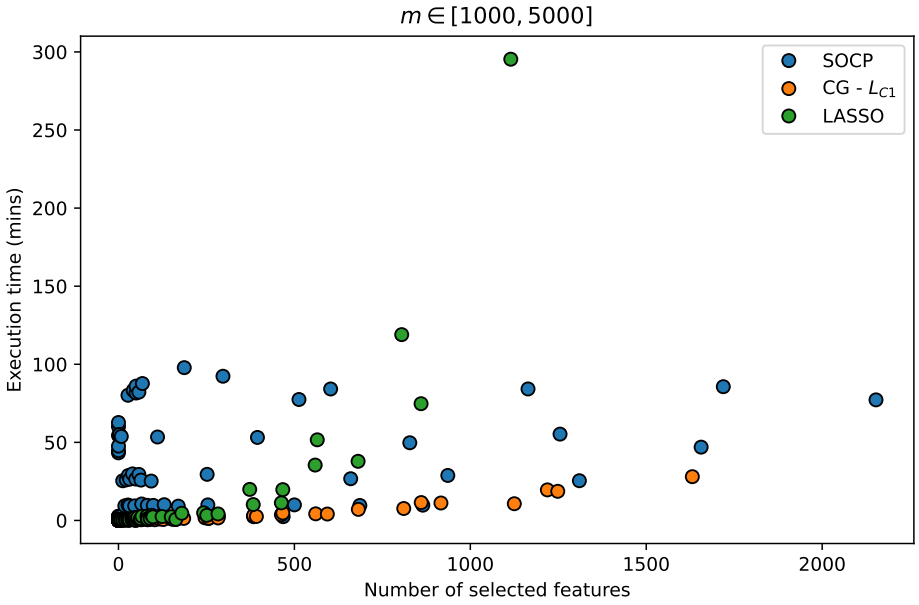

Figure 4.16: Average execution times of the three equivalent models in terms of the number of selected features in each instance, for $m \in [1000, 5000]$. Each instance is executed 10 times.

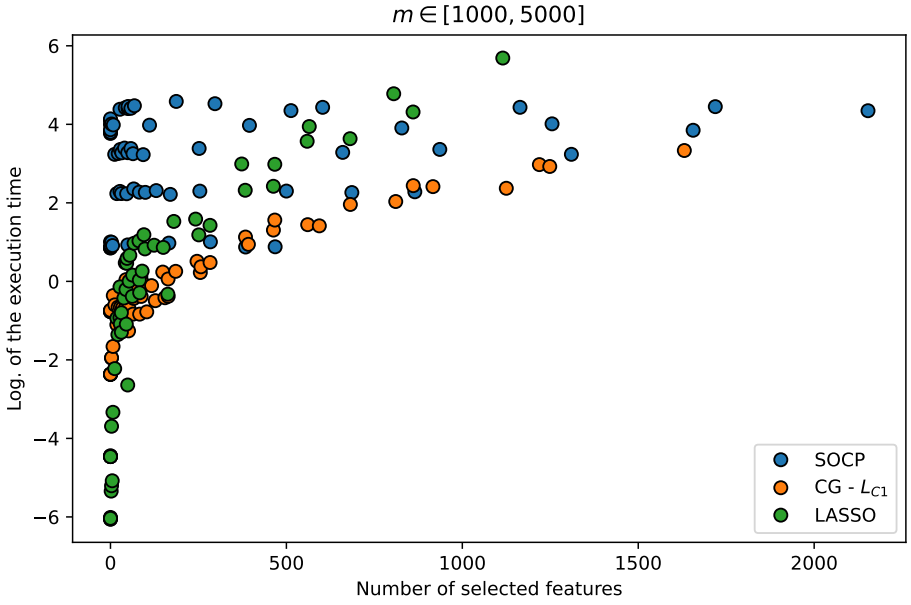

Figure 4.17: Logarithm of the average execution times of the three equivalent models in terms of the number of selected features in each instance, for $m \in [1000, 5000]$. Each instance is executed 10 times.

Chapter 5

Conclusions

This research has successfully achieved a significant breakthrough in the development of a novel feature selection model with conic constraints, capable of effectively handling instances of varying number of total features. Notably, the model has been proven as being equivalent to the widely recognized LASSO method (see Proposition 2).

The implementation of a state-of-the-art decomposition method for conic problems has yielded promising results, demonstrating the method's ability to match the optimal solution of the original conic problem in lower execution times, with instances were the decomposition method took 1% of the SOCP execution time to retrieve a solution (see Figure 4.15). Nonetheless, there is still room for improvements for this method to address some issues like the numerical problems for big instances with more than 40% of the features selected.

The $\mathcal{C}_1$ conic constraint relaxation has emerged as the fastest option among the three possible relaxation alternatives (see Figures 4.1, 4.2, and 4.3). This could be because of the relationship between the main three variables β, z and u, which only appears in the relaxed problem $L(\mathcal{X}_{\overline{\mathcal{C}_1}}, \lambda^k)$, not just as a constraint, also in the convergence condition of the problem. The other two relaxed problems treat each one of these variables as independent ones, possibly leading to poor solutions in many of the iterations of the method.

The identification of convergence conditions to check before trying to solve the different Lagrange relaxations has significantly improved the execution times of the $L(\mathcal{X}_{\overline{\mathcal{C}_1}}, \lambda^k)$ relaxation (see Tables 4.1 and 4.2). The rest of the relaxed problem alternatives did not show any significant decrease in the execution times (see Table 4.3). The Lagrange relaxation of these two alternatives only have positiveness constraints, so they are not hard-to-solve problems. While these advancements have been instrumental, there remains room for further enhancements in the pre-checking process of problematic behavior of the relaxed problems and in the artificial solution techniques.

A deeper exploration of the initial feasible region and solutions can be explored in the future. Seeing the fast results with techniques like Coordinate Descent in the LASSO method when just a few features are selected, as shown in Figures 4.11 to 4.15, a smart selection of an initial region holds the potential to refine the overall effectiveness of the method. Additionally, comparative analyses of the decomposition method, the original SOCP and the

LASSO method with more heterogeneous datasets and real-world instances are essential to fully evaluate the method's efficiency and applicability.

Further modifications to the method, such as solving the master's dual $D(\mathcal{S}^k)$ instead of the master problem $P(\mathcal{S}^k)$, trying different construction method of the feasible regions $\mathcal{S}^k$, and so on, offer promising avenues for decreasing execution times enhancing the method's robustness. Continued exploration of these modifications is crucial to ensure the method's reliability and consistency across various scenarios. This research marks a significant initial step, laying the foundation for continued advancements and refinements in the field of feature selection methods.

Bibliography

Aha, D. W. and Bankert, R. L. (1996). *A Comparative Evaluation of Sequential Feature Selection Algorithms*, pages 199–206. Springer New York, New York, NY.

Aizawa, A. (2003). An information-theoretic perspective of tf–idf measures. *Information Processing & Management*, 39(1):45–65.

Akman, D. V., Malekipirbazari, M., Yenice, Z. D., Yeo, A., Adhikari, N., Wong, Y. K., Abbasi, B., and Gumus, A. T. (2023). k-best feature selection and ranking via stochastic approximation. *Expert Systems with Applications*, 213:118864.

Bellman, R. E. (1957). *Dynamic Programming*. Princeton University Press, Princeton, New Jersey.

Benders, J. F. (1962). Partitioning procedures for solving mixed-variables programming problems. *Numerische Mathematik*, 4(1):238–252.

Bertsimas, D., King, A., and Mazumder, R. (2016). Best subset selection via a modern optimization lens. *The Annals of Statistics*, 44(2):813 – 852.

Boyd, S. P. and Vandenberghe, L. (2004). *Convex optimization*. Cambridge University Press.

Chen, R.-C., Dewi, C., Huang, S.-W., and Caraka, R. E. (2020). Selecting critical features for data classification based on machine learning methods. *Journal of Big Data*, 7(1).

Chicoisne, R. (2023). Computational aspects of column generation for nonlinear and conic optimization: classical and linearized schemes. *Computational Optimization and Applications*, 84(3):789–831.

Cui, Y., Jin, J. S., Zhang, S., Luo, S., and Tian, Q. (2010). Correlation-based feature selection and regression. In Qiu, G., Lam, K. M., Kiya, H., Xue, X.-Y., Kuo, C.-C. J., and Lew, M. S., editors, *Advances in Multimedia Information Processing - PCM 2010*, pages 25–35, Berlin, Heidelberg. Springer Berlin Heidelberg.

Curry, H. B. (1944). The method of steepest descent for non-linear minimization problems. *Quarterly of Applied Mathematics*, 2(3):258–261.

Dantzig, G. B. and Wolfe, P. (1961). The decomposition algorithm for linear programs. *Econometrica*, 29(4):767.

Diamond, S. and Boyd, S. (2016). CVXPY: A python-embedded modeling language for convex optimization. *J. Mach. Learn. Res.*, 17(1):2909–2913.

Ding, C. and Peng, H. (2005). Minimum redundancy feature selection from microarray gene expression data. *Journal of Bioinformatics and Computational Biology*, 03(02):185–205.

Fletcher, R. (2013). Newton-like methods. In *Practical Methods of Optimization*, pages 44–79. John Wiley & Sons, Ltd.

Friedman, J. H., Hastie, T., and Tibshirani, R. (2010). Regularization paths for generalized linear models via coordinate descent. *Journal of Statistical Software*, 33(1):1–22.

Guyon, I., Weston, J., Barnhill, S., and Vapnik, V. (2002). Gene selection for cancer classification using support vector machines. *Machine Learning*, 46(1/3):389–422.

Holland, J. H. (1975). Adaptation in natural and artificial systems: An introductory analysis with applications to biology, control, and artificial intelligence. *University of Michigan Press, Ann Arbor. (2nd Edition, MIT Press, 1992.)*.

Hou, Z., Hu, Q., and Nowinski, W. (2006). On minimum variance thresholding. *Pattern Recognition Letters*, 27(14):1732–1743.

Karbowski, A. (2021). Generalized benders decomposition method to solve big mixed-integer nonlinear optimization problems with convex objective and constraints functions. *Energies*, 14(20):6503.

Karush, W. (1939). Minima of functions of several variables with inequalities as side conditions. Master's thesis, Department of Mathematics, University of Chicago, Chicago, IL, USA.

Kenney, A., Chiaromonte, F., and Felici, G. (2021). MIP-BOOST: Efficient and Effective L0 Feature Selection for Linear Regression. *Journal of Computational and Graphical Statistics*, 30(3):566–577.

Khaleel, S. (2011). Feature selection using k-means clustering for data mining. In *International Symposia in Advance Computing (ISAC-2K11)*, Lucknow, Uttar Pradesh. Shri Ramswaroop Memorial College of Engineering and Management.

Kohavi, R. and John, G. H. (1997). Wrappers for feature subset selection. *Artificial Intelligence*, 97(1):273–324.

Kucukyavuz, S., Shojaie, A., Manzour, H., Wei, L., and Wu, H.-H. (2020). Consistent second-order conic integer programming for learning bayesian networks. *arXiv:2005.14346*.

Lemaréchal, C. (2001). *Lagrangian Relaxation*, pages 112–156. Springer Berlin Heidelberg, Berlin, Heidelberg.

Miao, J. and Niu, L. (2016). A survey on feature selection. *Procedia Computer Science*, 91:919–926. Promoting Business Analytics and Quantitative Management of Technology: 4th International Conference on Information Technology and Quantitative Management (ITQM 2016).

MOSEK ApS (2023). *The MOSEK Optimizer API for Python. Version 10.1.*

Nowak, I. (2005). *Relaxation and Decomposition Methods for Mixed Integer Nonlinear Programming*. Birkhäuser Basel.

Pearson, K. (1900). X. On the criterion that a given system of deviations from the probable in the case of a correlated system of variables is such that it can be reasonably supposed to have arisen from random sampling. *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science*, 50(302):157–175.

Pearson, K. (1901). LIII. On lines and planes of closest fit to systems of points in space. *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science*, 2(11):559–572.

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., et al. (2011). Scikit-learn: Machine learning in python. *Journal of machine learning research*, 12(Oct):2825–2830.

Quinlan, J. R. (1986). Induction of decision trees. *Machine Learning*, 1(1):81–106.

Rahmat, F., Zulkafli, Z., Ishak, A. J., Rahman, R. Z. A., Stercke, S. D., Buytaert, W., Tahir, W., Rahman, J. A., Ibrahim, S., and Ismail, M. (2023). Supervised feature selection using principal component analysis. *Knowledge and Information Systems*, pages 1–41.

Richards, J. A. (2022). *Supervised Classification Techniques*, pages 263–367. Springer International Publishing, Cham.

Rothlauf, F. (2011). *Optimization Methods*, pages 45–102. Springer Berlin Heidelberg, Berlin, Heidelberg.

Sarker, I. H. (2021). Machine learning: Algorithms, real-world applications and research directions. *SN Computer Science*, 2(3).

Schwendinger, B., Schwendinger, F., and Vana, L. (2022). Holistic generalized linear models. *arXiv:2205.15447*.

Sheikhpour, R., Sarram, M. A., Gharaghani, S., and Chahooki, M. A. Z. (2017). A survey on semi-supervised feature selection methods. *Pattern Recognition*, 64:141–158.

Tibshirani, R. (1996). Regression Shrinkage and Selection via the Lasso. *Journal of the Royal Statistical Society. Series B (Methodological)*, 58(1):267–288.

Vicente, L. N. and Calamai, P. H. (1994). Bilevel and multilevel programming: A bibliography review. *Journal of Global Optimization*, 5(3):291–306.

Xu, X., Gu, H., Wang, Y., Wang, J., and Qin, P. (2019). Autoencoder based feature selection method for classification of anticancer drug response. *Frontiers in Genetics*, 10.

Zou, H. and Hastie, T. (2005). Regularization and Variable Selection Via the Elastic Net. *Journal of the Royal Statistical Society Series B: Statistical Methodology*, 67(2):301–320.