The Solution of a Class of Limited Diversification Portfolio Selection Problems

Gwyneth Owens Butera

CRPC-TR97724-S
May 1997

Center for Research on Parallel Computation
Rice University
6100 South Main Street
CRPC - MS 41
Houston, TX 77005

Submitted in November 1997; Also available as Rice University Ph.D. Thesis

RICE UNIVERSITY

The Solution of a Class of Limited Diversification Portfolio Selection Problems

by

Gwyneth Owens Butera

A Thesis Submitted
in Partial Fulfillment of the
Requirements for the Degree
Doctor of Philosophy

Approved, Thesis Committee:

Robert E. Bixby, Co-chairman
Professor of Computational & Applied Mathematics

John E. Dennis, Jr., Co-chairman
Noah Harding Professor of Computational & Applied Mathematics

---

William Cook
Noah Harding Professor of Computational & Applied Mathematics

---

David Applegate
Associate Professor of Computational & Applied Mathematics

Barbara Ostdiek
Assistant Professor of Administrative Science, Jones Graduate School of Adminstration

Houston, Texas
May, 1997

Abstract

The Solution of a Class of Limited Diversification Portfolio Selection Problems

by

Gwyneth Owens Butera

A branch-and-bound algorithm for the solution of a class of mixed-integer nonlinear programming problems arising from the field of investment portfolio selection is presented. The problems in this class are characterized by the inclusion of the fixed transaction costs associated with each asset, a constraint that explicitly limits the number of distinct assets in the selected portfolio, or both. Modeling either of these forms of limiting the cost of owning an investment portfolio involves the introduction of binary variables, resulting in a mathematical programming problem that has a nonconvex feasible set. Two objective functions are examined in this thesis; the first is a positive definite quadratic function which is commonly used in the selection of investment portfolios. The second is a convex function that is not continuously differentiable; this objective function, although not as popular as the first, is, in many cases, a more appropriate objective function. To take advantage of the structure of the model, the branch-and-bound algorithm is not applied in the standard fashion; instead, we generalize the implicit branch-and-bound algorithm introduced by Bienstock [3]. This branch-and-bound algorithm adopts many of the standard techniques from mixed-integer linear programming, including heuristics for finding

feasible points and cutting planes. Implicit branch-and-bound involves the solution of a sequence of subproblems of the original problem, and thus it is necessary to be able to solve these subproblems efficiently. For each of the two objective functions, we develop an algorithm for solving its corresponding subproblems; these algorithms exploit the structure of the constraints and the objective function, simplifying the solution of the resulting linear systems. Convergence for each algorithm is proven. Results are provided for computational experiments performed on investment portfolio selection problems for which the cardinality of the universe of assets available for inclusion in the selected portfolio ranges in size from 52 to 1140.

Acknowledgments

I would like to thank the members of my committee for giving me the freedom to work on this problem and to discover for myself the trials and the joys of research. I would especially like to thank Dr. Bixby for believing in my abilities even when I did not. To Dr. Dennis I owe many thanks for supporting me and having faith in my desire to finish while I was so far from Rice.

Thanks must also go to my friends Maeve McCarthy, Marielba Rojas, and Laurie Feinswog for their incredible patience. Without friends supporting me through the good times and bad, I'm not sure that I could have seen this through.

Thanks to Daria Lawrence and Maeve McCarthy for helping me with the administrative details of being a thousand miles away from the Rice community. You both helped make what could have been a very difficult situation into one that was endurable.

To Dr. Karla Hoffman and Dr. Dianne O'Leary I owe many thanks for opening their doors to me when I was in need of guidance. To Mike Priwer, I owe thanks for his infinite patience and understanding that a thesis does not occur overnight. Also, thanks to Jake Ostdiek for arriving only one day ahead of schedule.

Thanks to my family for understanding the ups and downs of being a graduate student. To my mom, for all her encouragement and for teaching me about base 2 when I was 6 years old. I guess she knew it would one day be useful for me. To my dad, for fostering my love of mathematics and for suggesting that I apply to Rice. To my sister, Susannah, for living through many dinners of geek speak – here's to more to come! And to Carl and Rerun for keeping me company those many, many days that I was home alone working on my thesis.

My husband Rob deserves many thanks for supporting me along the way. Listening to me get excited about a new idea and even trying to understand what must sound like gibberish is a task that few could endure for very long. I love you!

Contents

Illustrations

Tables

Chapter 1

Introduction

In 1952, investment portfolio selection and mathematical programming were linked by the pioneering work of Harry Markowitz [36]. Markowitz argued that the probable success of an investment does not depend solely on the expected rate of return of that investment, but that it is also a function of the level of risk taken. Furthermore, he showed that to properly measure the risk of a collection of assets, called a *portfolio*, one should consider the pairwise correlations of returns among those assets. Markowitz developed a mathematical programming model and accompanying algorithm for selecting portfolios with high expected rates of return and low risk. Because Markowitz described risk as a function of the variances and covariances of the individual investments, this optimization problem is referred to as the *mean-variance portfolio selection problem.*

The risk of a portfolio can be reduced by including additional assets that are not perfectly correlated with the existing portfolio. The process of adding assets to reduce risk is known as *diversification.* This benefit of diversification implies that a Markowitz-optimal mean-variance portfolio will tend to include many of the available assets. Because the costs of transacting in many distinct assets are nontrivial, these costs could outweigh the benefits gained by optimization. To ensure that the costs of a portfolio are kept low, either an upper limit could be placed on the number of assets that appear in the selected portfolio or the fixed costs that are independent of volume could be included in the model, or both. Although these methods for limiting diversification are not new, much of the work done on such models has resulted in approximate algorithms or in algorithms that are not applicable when additional

constraints are placed on the portfolio [4, 7, 12, 28, 43]. In this thesis we present algorithms for the solution of two classes of limited diversification mean-variance portfolio selection problems with linear constraints.

1.1 The Class of Limited Diversification Problems

The class of problems we focus on are formulated as

$$\begin{aligned} \text{minimize} \quad & f(x,y) \\ \text{subject to} \quad & H(x,y) \geq h, \\ & x \in \mathbb{R}^n, y \in \mathbb{B}^n, \end{aligned} \tag{1.1}$$

where n is the number of available assets, m is the number of constraints in $H(x,y) \geq h$, $f : \mathbb{R}^{2n} \rightarrow \mathbb{R}$ is a continuous convex function, $H : \mathbb{R}^{2n} \rightarrow \mathbb{R}^m$ is a linear transformation, $h \in \mathbb{R}^m$, $x, y, \in \mathbb{R}^n$, and $\mathbb{B} = \{0,1\}$. As in many mixed-integer programming problems, the variables are divided into two groups: the continuous variables x and the binary decision variables y; the term *mixed-integer* means that there may be both integer and continuous variables in the formulation. These variables are coupled through a subset of the linear inequality constraints, $H(x,y) \geq h$. Denoting the jth component of a vector x by x_j, each asset $j \in \{1, \ldots, n\}$, has both a decision variable y_j and a continuous variable x_j associated with it. The decision variable y_j is a binary variable that indicates whether or not a particular asset is included in the portfolio; the continuous variable x_j indicates the proportion of total capital to be invested in the jth asset. The goal is to find a global minimizer of $f(x,y)$ which satisfies the constraints of (1.1).

Each of the two nonlinear objective functions $f(x,y)$ examined in this thesis is comprised of a linear combination of the expected rate of return and a penalty term based on the variance of return. In the first objective function the penalty term is the

variance; in the second objective function, the penalty term is the square root of the variance, more commonly referred to as the *standard deviation*. Although the first has become quite popular in the literature [44, 45, 51] since it was first introduced in 1962 by Farrar [14], use of the standard deviation evolved from a more precise strategy. Baumol [2] argued that an investor should not be concerned solely with Markowitz's definition of optimality, but also with the degree that the actual rate of return could be below the expected rate of return of the selected investment portfolio. Given two portfolios A and B from the set of Markowitz optimal portfolios, if portfolio A has an expected rate of return of 5% and a standard deviation of .2% and portfolio B has an expected rate of return of 8% and a standard deviation of 1%, any investor but the most conservative would prefer portfolio B. Fortunately, Chebyshev's theorem [15] helps to quantify this strategy; given a $\theta > 0$, this theorem provides a lower bound on the probability that the actual rate of return will be within θ standard deviations of the expected rate of return. The second objective function we examine allows the investor to choose a θ based on his level of risk aversion and maximize the "floor" of his confidence interval; this floor is the level of θ standard deviations below the expected rate of return.

1.2 The Algorithm

Although there have been algorithms proposed in the past for solving other mixed-integer nonlinear programming problems [3, 10, 18, 31, 32], we have developed an algorithm that exploits the structure of the objective functions and the constraints of (1.1).

Our algorithm can be divided into two separate algorithms: the outer branch-and-bound algorithm that works to satisfy the binary conditions on the decision variables

and an algorithm for solving the linearly constrained subproblems created by relaxing or forcing the binary conditions on the decision variables.

Branch-and-bound is a technique that has been used for almost 40 years in the solution of mixed-integer programming problems with linear objective functions [30]. To find the optimal solution of a mixed-integer linear programming problem, a sequence of subproblems of the original problem are solved. A similar algorithm could be applied to (1.1); although the objective functions are not linear, they are convex and, as we will show, the convergence results for standard branch-and-bound still hold. Bienstock [3] introduced an efficient implicit branch-and-bound algorithm for the class of portfolio selection problems in which the binary variables are incident only on the $2n$ coupling constraints and the constraint which enforces an upper limit on the number of assets in the final portfolio; his algorithm involves removing the binary variables and approximating the limited diversification constraint using only the continuous variables. We have extended this method to the allow binary variables to appear in any number of constraints. Furthermore, we show that the presence of the binary variables is redundant, and (1.1) can be reformulated such that these variables are unnecessary.

The performance of the implicit branch-and-bound algorithm can be improved by the addition of linear inequalities, called cutting planes, that are valid for the feasible points of (1.1); three classes of cutting planes valid for (1.1) are provided. We introduce other methods for improving the performance of the algorithm, including heuristics for finding feasible points of the problem and for creating the next subproblems in the sequence of subproblems.

One of the most important aspects of a good branch-and-bound algorithm is the ability to use the solution of a previously solved subproblem to aid in solving the current subproblem. The two algorithms we have developed for solving the subprob-

lems created in branch-and-bound are dual algorithms based on the necessary and sufficient conditions for optimality of the subproblem. Dual algorithms, which can use information from previous solutions to determine a good starting point, have been studied in the past for the first objective function. Goldfarb and Idnani [22] proposed a dual algorithm for the solution of linear inequality constrained quadratic programming problems. We have extended this method to exploit the structure of our constraints, particularly the simple bound constraints, $l_j \leq x_j \leq u_j$, on the continuous variables. For the second objective function, we introduce a dual algorithm similar to that of the Goldfarb-Idnani algorithm. Convergence results are proven and numerical stability is discussed for these dual algorithms.

1.3 Organization

In Chapter 2, the results from mathematical statistics needed to build the two objective functions are introduced and a summary of the history of the mean-variance portfolio selection problem is provided. A thorough description of the constraints and objective functions is also presented in Chapter 2. In Chapter 3, our generalized implicit branch-and-bound algorithm is introduced along with the proof that it is guaranteed to converge to the global minimum. The procedures and heuristics developed for determining cutting planes, finding feasible points, and creating and choosing the problems in the sequence of subproblems are discussed in Chapter 4. Chapters 5 and 6 introduce the dual algorithms that we developed to solve the subproblems created in branch-and-bound for the first and second objective functions, respectively. Chapter 7 presents results from computational experiments on portfolio selection problems where n, the number of available assets, ranges from 52 to 1140.

Chapter 2

The Mean-Variance Model

In this chapter, we review the mathematical statistics associated with investment portfolio selection and present a condensed history of the mean-variance model for this problem. Following the background discussion, the two optimization problems examined in the remainder of the thesis are introduced.

2.1 Results from Mathematical Statistics

In the *ex ante* evaluation of an investment portfolio, it is necessary to determine the expected rate of return and the risk of that portfolio; risk of a portfolio can be characterized by the portfolio's expected rate of return and variance of return of that portfolio. The basic statistical tools needed to compute these values are introduced in this section.

First, however, the concept of a portfolio must be formalized. A portfolio is a collection of assets; the percentage of total capital invested in each asset uniquely describes the portfolio. Given a collection of n assets, a portfolio can be represented by $x \in \mathbb{R}^n$ where, for each $j \in \mathcal{N} = \{1, \ldots, n\}$, x_j is the proportion of total capital invested in asset j. The capital constraint $\sum_{j \in \mathcal{N}} x_j \leq 1$ must be satisfied.

The rate of return of asset $j \in \mathcal{N}$ is a random variable, R_j; the expected rate of return of asset j is represented by $E(R_j)$ and the variance of R_j is represented by σ_j^2. The covariance of R_i and R_j, where $i, j \in \mathcal{N}, i \neq j$, is represented by σ_{ij}; since covariance is a symmetric function, σ_{ij} must be equivalent to σ_{ji}. The matrix formed by placing σ_j^2 in the jth diagonal position $\forall j \in \mathcal{N}$ and placing σ_{ij} in the ith row and

jth column $\forall i, j \in \mathcal{N}, i \neq j$, is called the *variance-covariance matrix* and is denoted by V; a variance-covariance matrix is always symmetric.

Approximations of the necessary variances, covariances, and expected rates of return can be estimated using data describing the past performance of each asset. The variance-covariance matrix calculated from past data is necessarily positive semi-definite [33]. Substantial research [19, 35, 38] indicates that past performance is often not a good indicator of future performance and that the investor's perceptions of the future should also be taken into account. Throughout this thesis it is assumed that the investor has not relied solely on past data but has determined the expected rates of return and the variance-covariance matrix to accurately reflect what he believes the future holds. It is also assumed that the variance-covariance matrix is positive definite. Although this may not be a good assumption in general, in §2.2 we show that in many cases the full variance-covariance matrix is not calculated but is instead approximated by a positive definite diagonal matrix V.

The rate of return of portfolio x is a random variable $R(x)$ such that

$$R(x) = \sum_{j=1}^{n} R_j x_j.$$

Because expected value is a linear function, the expected rate of return of portfolio x is a weighted sum of the expected rates of return of the available assets:

$$E(R(x)) = E\left(\sum_{j=1}^{n} R_j x_j\right) = \sum_{j=1}^{n} E(R_j) x_j.$$

Letting $\mu_j \equiv E(R_j)$, the expected rate of return of portfolio x is $\mu^T x$. The variance of return of portfolio x is

$$\mathrm{Var}(R(x)) = \sum_{j=1}^{n} \sigma_{jj}^2 x_j^2 + 2 \sum_{j=1}^{n} \sum_{k=1}^{j-1} \sigma_{jk} x_j x_k = x^T V x.$$

The *standard deviation* of return is simply the square root of the variance of return.

2.2 History of the Mean-Variance Model

For the past 45 years, the mean-variance model for investment portfolio selection has been studied by many researchers in both the finance and the mathematical programming communities. Some have embraced and extended this model, while others have criticized it. Here we present a short summary of the literature from both sides. Detail is provided only where necessary for later sections of the thesis.

Before 1952, spreading capital among many assets, called *diversification*, was known to be a method for reducing risk. However, no one had developed a theory that explained this phenomenon until the pioneering paper of Markowitz [36]. Markowitz proposed that variance is an appropriate measure of risk and that reducing variance can be accomplished by diversification.

Markowitz defined the *efficient frontier* as the set of portfolios with minimum variance for a given expected rate of return or more and maximum expected rate of return for a given variance or less. In his seminal paper [36], Markowitz provided a geometric description of the critical line method for solving this problem when the assets in the portfolio must have nonnegative holdings. It was not until 1956 that he published a computational technique for determining the efficient frontier of the portfolios that satisfy general linear equality and inequality constraints supplied by the investor [37].

Farrar [14] introduced a parametric objective function for the mean-variance portfolio selection problem. Using the necessary and sufficient conditions for optimality, which are reviewed in Chapter 5, it is straightforward to show that for any $\theta > 0$, the optimal solution to

$$\text{maximize} \qquad f(x) = \mu^T x - \theta x^T V x, \tag{2.1}$$

is on the efficient frontier as defined by Markowitz. The function $f(x)$ is called a *quadratic function* and is the objective function we study in Chapter 5.

Concurrently with Markowitz, Roy [48] also determined that the objective of maximizing the expected rate of return does not explain the benefits of diversification. His rule of Safety First states that an investor is concerned that the actual rate of return, denoted by $\bar{\mu}^T x$, should not fall at or below a given disaster level, ϱ; thus the investor should minimize the probability that the actual rate of return is at most ϱ.

In 1963, Baumol [2] developed an objective function that, similar to Roy's Safety First rule, is based on the probability that the actual rate of return will be above a certain level. Given μ and V associated with the available assets and a probability $0 < \bar{P} < 1$, Baumol's model determines a feasible portfolio x^* and a scalar δ^* such that the actual rate of return, $\bar{\mu}^T x^*$, is greater than δ^* with probability at least $\bar{P}$, and the value of δ^* is maximized over all $\delta \in \mathbb{R}$. For example, Baumol considers a portfolio for which the actual rate of return will be above 7% with probability at least 96% to be better than one for which the actual rate of return will be above 5% with probability at least 96%.

Fortunately, the scalar δ can be represented as a function of x that is dependent on the values of the constants μ, V, and $\bar{P}$. Given any scalar $\theta > 0$, Chebyshev's theorem [15] states that the probability that the actual rate of return will be within θ standard deviations of the expected rate of return is at least $1 - 1/\theta^2$:

$$P(|\mu^T x - \bar{\mu}^T x| < \theta\sigma) \geq 1 - \frac{1}{\theta^2}.$$

This implies that

$$P(\bar{\mu}^T x > \mu^T x - \theta\sigma) \geq 1 - \frac{1}{\theta^2};$$

i.e., the actual rate of return will be greater than θ standard deviations below the expected rate of return with probability at least $1 - 1/\theta^2$. We can replace δ with

$\mu^T x - \theta\sigma$, where $\theta = (1 - \bar{P})^{-1/2}$. Alternatively, if a joint normal distribution is assumed, a standard normal distribution table gives the value of θ such that the actual rate of return will be greater than $\mu^T x - \theta\sigma$ with probability $\bar{P}$, independent of x.

Whether a joint normal distribution of return is assumed or Chebyshev's inequality is used to determine θ, an optimal portfolio is a solution to the optimization problem

$$\begin{aligned} \text{maximize} \quad & \mu^T x - \theta\sqrt{x^T V x} \\ \text{subject to} \quad & \sum_{j=1}^{n} x_j \leq 1, \end{aligned}$$

where θ is chosen by the investor. In the case that the investor wishes to determine the optimal portfolio for a single value of $\theta \geq 0$ and not for all $\theta \geq 0$, Baumol's formulation is clearer than Farrar's because the value of the risk-aversion parameter θ has a precise and important meaning. According to Sortino and van der Meer [55], Baumol's objective function was adopted by Salomon Brothers in 1989. We study Baumol's objective function in Chapter 6.

Markowitz, Roy, and Baumol were concerned with quantifying an objective for the portfolio selection problem. However, their models include the dense variance-covariance matrix, which is not easy to manipulate and requires the investor to determine approximately $n^2/2$ covariances. To reduce the amount of work in collecting the data, Markowitz [38] suggested the single-factor model. In this model, the rate of return and risk of an asset is divided into two parts: the first, called unsystematic, captures the idiosyncratic operations of the company and has no relation to the risk and rate of return of the other assets, and the second, called systematic, is due to the correlation between that asset and some outside factor, such as the stock market. Given this division, the expected rate of return of asset j can be written as

$$R_j = \hat{\mu}_j + \beta_j F + \xi_j,$$

where $\hat{\mu}_j$ is the expected unsystematic rate of return of asset j, β_j is the measure of the effect of the outside factor on asset j, ξ_j is a random variable with expected value zero and variance, $\hat{\sigma}_j^2$, which is equal to the unsystematic variance of asset j, and F is the level of the factor. The future level of F can be described as

$$F = \hat{\mu}_{n+1} + \xi_{n+1},$$

where $\hat{\mu}_{n+1}$ is the expected rate of return of the outside factor and ξ_{n+1} is a random variable with expected value zero and variance equal to the variance of the factor, which is denoted by $\hat{\sigma}_{n+1}^2$. In this model of rates of return and variances, the covariance of ξ_i and ξ_j where $i, j \in \mathcal{N} \cup \{n+1\}, i \neq j$, is zero.

It is easy to show that given this model, the following relations hold:

- $E(R_j) = \hat{\mu}_j + \beta_j \hat{\mu}_{n+1}$ for all $j \in \mathcal{N}$,
- $\sigma_j^2 = \beta_j^2 \hat{\sigma}_{n+1}^2 + \hat{\sigma}_j^2$ for all $j \in \mathcal{N}$, and
- $\sigma_{ij} = \beta_i \beta_j \hat{\sigma}_{n+1}^2$ for all $i, j \in \mathcal{N}, i \neq j$.

Markowitz suggested that the vectors $\hat{\mu}$, β, and $\hat{\sigma}$ be entered into a computer which could then easily calculate the expected rate of return of each asset and the full variance-covariance matrix.

However, manipulating the dense variance-covariance matrix is difficult if n is large. Sharpe [51] showed that instead of forming the full variance-covariance matrix, it is possible to add a variable x_{n+1} to represent the factor F and then to transform the problem into one with a diagonal variance-covariance matrix. Adding the constraint $x_{n+1} = \sum_{j=1}^{n} \beta_j x_j$, the expected rate of return of a portfolio x is $\sum_{j=1}^{n+1} \hat{\mu}_j x_j$ and the variance is $\sum_{j=1}^{n+1} \hat{\sigma}_j^2 x_j^2$. These alternate expressions can be used in place of the dense variance-covariance matrix formulation in any of the aforementioned mean-variance portfolio selection models.

Cohen and Pogue [6] also suggested that the mean-variance model was not popular because of the effort required to determine the input data, especially the covariances. They proposed a multi-factor approach which Markowitz had alluded to in [38]. Unlike Sharpe's model that has a single factor, their model allows that there may be multiple factors, such as capitalization weighting and industry type, that determine the covariance between two assets. Instead of adding a single variable, a variable is added for each factor. Since these factors may be correlated, the transformed variance-covariance matrix is not diagonal, but it is diagonal in the assets and dense in the factors.

The single-factor, multi-factor, and dense matrix models place no limit on the number of assets included in the final portfolio. The first step towards limiting diversification was made in 1970 by Mao [34]. He stated that it is economical to limit the number of assets in the portfolio, since the management and transaction costs are not zero. He did assume that the investor is willing to place capital in a sufficient number of assets to reduce the unsystematic risk to a negligible amount. In 1974, Jacob [28] proposed that Mao's strategy may not work for the small investor who may prefer to take on some unsystematic risk if it helps to keep the transaction costs down. She modified Sharpe's linear programming relaxation [53] of the single-factor model to allow the investor to limit the number of assets in the final portfolio. Her algorithm forces the final solution to have the capital distributed equally among the assets chosen. Faaland [12] presented a model similar to Jacob's, but allowed the investor to choose positive integers κ and ϑ, $\vartheta \geq \kappa$, such that the final portfolio has ϑ equal parts distributed among κ assets. To solve this model, he developed a dynamic programming algorithm that becomes impractical as ϑ gets large. The algorithms proposed by Mao, Jacob, and Faaland did not allow for fixed transaction costs to be included.

Much work has been done since 1974 on algorithms for mean-variance portfolio selection; however, almost all such work has assumptions such as no linear constraints other than simple bounds on the variables and the capital constraint [4, 7, 11, 42, 43] or no fixed transaction costs [44]. The only work we are aware of that allows any number of linear constraints, minimum transaction levels, and transaction costs, both linear and fixed, is Bienstock [3]. A few of the ideas in this thesis are similar to his algorithm; however, he limits the appearance of the control variables y in the formulation and examines solely the quadratic objective function (2.1).

A discussion of mean-variance models for investment portfolio selection would not be complete without a short synopsis of the criticisms of these models. Breen and Savage [5] state that mean-variance models, when viewed as an investor maximizing the expected utility of return, requires that "the probability distribution of total portfolio return is one which is completely described by two parameters, usually mean and variance, or that the investor's utility function is quadratic in return." Hanoch and Levy [25] claim that a cubic utility, which includes skewness, may be preferable to the quadratic utility. However, Tsiang [57] argued that mean-variance analysis is a useful approximate method when the risk is a small fraction of total wealth.

Another criticism of the mean-variance model is that variance may be a poor definition of risk. Volatility alone does not define risk; it may be preferable to consider the possible consequences corresponding to loss when defining risk [29]. To model the consequences of loss, some authors have proposed measures of downside risk. We have already discussed one such model, called shortfall risk or Safety First, which assumes that the investor's goal is to minimize the probability of falling below a minimum acceptable rate of return [24, 48]. Other authors believe that neither variance nor shortfall risk may be an appropriate measure of risk since both fail to measure the consequences of the degree to which the return falls below the minimum acceptable

rate of return. To model the different consequences of falling short of the minimum acceptable rate of return by different levels, they use lower partial moment models, such as semi-variance [27, 26, 38, 46, 55].

Other authors have proposed that random walks may better describe the behavior of asset returns because past performance may not be a good indicator of future performance. In 1965, Fama [13] presented empirical evidence that the past cannot be used to predict the future and that a random-walk model is valid for choosing a portfolio. Sharpe [52] agreed with Fama's findings and stated that this implies that it may be difficult to detect incorrectly priced assets.

Despite these criticisms, many investment managers continue to use some form of the mean-variance model for portfolio selection. In the words of Shell, "the [mean-variance] model has been around, is well understood and is relatively well-suited to computation" [54].

2.3 The Model

The portfolio selection problems studied in this thesis are in the general form

$$\begin{aligned} \text{minimize} \quad & f(x,y) \\ \text{subject to} \quad & H(x,y) \geq h, \\ & x \in \mathbb{R}^n, y \in \mathbb{B}^n, \end{aligned} \tag{2.2}$$

where n is the number of available assets, m is the number of constraints in $H(x,y) \geq h$, $H : \mathbb{R}^{2n} \rightarrow \mathbb{R}^m$ is a linear transformation, $h \in \mathbb{R}^m$, $x \in \mathbb{R}^n$ represents the portfolio, and $y \in \mathbb{B}^n$ is the vector of binary decision variables such that the jth component of y, y_j, indicates if there is a nonzero holding in the jth asset. The objective function $f(x,y)$ is either Farrar's formulation, $f(x,y) = -\mu^T x + \theta x^T V x$, or Baumol's objective function, $f(x,y) = -\mu^T x + \theta\sqrt{x^T V x}$, where $\mu \in \mathbb{R}^n$ is the vector

of the expected rates of return of the assets, V is the variance-covariance of returns matrix, and θ is the risk-aversion parameter whose value is chosen by the investor. If the problem is a single-factor or multi-factor model, the expected rates of return and the variance-covariance matrix may be in a modified form.

The variable x_j represents the proportion of capital invested in asset j; this proportion is bounded above by u_j and below by l_j. It is assumed without loss of generality that $l_j \leq 0 \leq u_j$ and at least one of l_j and u_j is nonzero. If $l_j < 0$, then *short-selling* is allowed for the jth asset; a short sale is the sale of an asset not owned by the investor but borrowed in anticipation that the rate of return will be negative. The decision variables y_j are such that whenever x_j is nonzero y_j must be one and whenever y_j is zero x_j must be zero. To enforce these conditions, the linking constraints

$$x_j - l_j y_j \geq 0 \quad \text{and} \quad u_j y_j - x_j \geq 0$$

must be included in the constraints $H(x, y) \geq h$ for each asset j. The linking constraints enforce the simple bound constraints $l_j \leq x_j \leq u_j$ for each asset j.

If short-selling is allowed and there are minimum transaction levels, the model must be modified. Instead of a single continuous variable indicating the proportion of asset j in the portfolio, there must be two continuous variables associated with each asset; one, x_j, indicates the long position of the asset in the portfolio and the other, x_{j+n}, indicates the short position of asset j in the portfolio. Likewise, two binary variables must be included for each asset j. In effect, we double the number of variables from $2n$ to $4n$. Because all of the algorithms developed in this thesis are easily extended to the case of minimum transaction levels, we do not discuss these extensions explicitly, but develop the algorithms for the case in which there are not minimum transaction levels.

Other constraints that may be included in $H(x,y) \geq h$ are the minimum allowable yield constraint, $\mu^T x \geq \varrho$, and the capital constraint, $\sum_{j=1}^{n} x_j \leq 1$. The limited diversification constraint may also be included. For example, if the investor requires that no more than κ assets to be nonzero in the final portfolio, he would include the constraint $\sum_{j=1}^{n} y_j \leq \kappa$. This inequality can be represented in $H(x,y) \geq h$ as $\sum_{j=1}^{n} -y_j \geq -\kappa$. Although an equality constraint can be represented in $H(x,y) \geq h$ by two inequality constraints, our algorithms are designed to take advantage of the properties of equality constraints.

If any transaction costs are included in the model, the capital constraint must be modified, since the amount of capital available to be invested in assets is reduced by the total transaction cost. Two types of transaction costs are considered: volume related costs and fixed costs. The volume related cost of asset j is $\$c_j$ per \$1 of asset j, and the fixed cost of holding any amount of asset j is $\$g_j$. The fixed costs must be scaled by the total capital available; if the investor has $\$\chi$ capital, then the capital constraint is $\sum_{j=1}^{n} [(1 + c_j)x_j + g_j y_j/\chi] \leq 1$.

Although the results in this thesis are derived for the general form of the model, the computational experiments are performed on real-world data of a specific form which arises in index-tracking. An investment performance index, such as the Standard & Poor's 500, provides a benchmark against which to evaluate any portfolio of assets [17]. An investment manager whose performance is being compared to a specific index wishes to avoid falling behind that index; one way to ensure he does not fall behind the index is, assuming no transaction costs, to invest in the index portfolio. However, active managers are not willing to forego the possibility of having higher returns than the index nor are they willing to pay the transaction costs associated with owning a great number of assets [19]. Instead, the manager would like the portfolio to consistently outperform the index while only containing a subset of the

assets in the index. To guard against loss, he also would like his portfolio to deviate from the index in a manner that minimizes the possibility of it performing worse than the index. In other words, he is aiming for an expected rate of return that is greater than the expected rate of return of the index, but he also would like to minimize the variance of the difference between his portfolio and the index portfolio [39, 47, 49].

To formulate this mathematically, the variance of this difference is $(x-x^0)^T V(x-x^0)$, where x_j^0 is the proportion of the total index capital that is invested in asset j. Using Farrar's objective function, the model is

$$\begin{aligned} \text{minimize} \quad & -\mu^T x + \theta(x-x^0)^T V(x-x^0) \\ \text{subject to} \quad & \textstyle\sum_{j=1}^n [(1+c_j)x_j + g_j y_j/\chi] \le 1, \\ & \mu^T x \ge \mu^T x^0 + \varrho, \\ & \textstyle\sum_{i=1}^n y_j \le \kappa, \\ & l_j y_j \le x_j \le u_j y_j \ \forall j, \\ & x \in \mathbb{R}^n, \quad y \in \mathbb{B}^n, \end{aligned} \tag{2.3}$$

where θ is the risk-aversion parameter chosen by the manager and $\mu^T x^0 + \varrho$ is the minimum acceptable expected rate of return. Given a $\theta \ge 0$, the difference of the optimal portfolio, x^*, and the index portfolio, x^0, is on the efficient frontier of differences, $x - x^0$, subject to the constraints of (2.3).

The index-tracking data used in computing the results for Chapter 7 is in a modified multi-factor form; the variance-covariance matrix V is a diagonal matrix for which the jth diagonal entry is the variance that can be attributed specifically to the jth company. However, variables are not added to represent the factors; instead, constraints are added to mimic the factors. For example, the *capitalization weighting* of asset j is the total value of the outstanding shares; if one of the factors is whether or not an asset is in the group of assets with large index capitalization weightings (big-cap), the portfolio should mimic the index in this factor by having the same

proportion of capital invested in the big-cap assets as the index does. The same can be done for other factors, such as an industry factor like technology.

The constraints that mimic the factors of the index may not be strict equalities but may be "soft"; a slack variable that is bounded above and below is added to each such constraint. If a term is added to the objective function to penalize these slack variables for being away from zero, there must be a nonzero coefficient on the diagonal of V for this variable.

Throughout the thesis, we do not assume a diagonal matrix V in the objective function, nor do we assume an index tracking model.

Chapter 3

The Implicit Branch-and-Bound Algorithm

In this chapter, we review the standard branch-and-bound algorithm for solving mixed-integer linear programming problems and prove that this algorithm can be modified to solve the limited diversification portfolio selection problem. We then introduce our implicit branch-and-bound algorithm, along with improvements that make the algorithm a more efficient method than standard branch-and-bound for solving the mixed-integer nonlinear programming problems examined in this thesis.

3.1 Introduction to Branch-and-Bound

The standard linear programming based branch-and-bound algorithm was introduced in 1960 by Land and Doig [30] for solving problems of the form

$$\begin{aligned} \text{minimize} \quad & f(x,y) = f_x^T x + f_y^T y \\ \text{subject to} \quad & H_x x + H_y y \geq h, \\ & x \in \mathbb{R}^{n_x},\ y \in \mathbb{B}^{n_y}, \end{aligned} \tag{3.1}$$

where n_x is the number of continuous variables, $f_x \in \mathbb{R}^{n_x}$, n_y is the number of binary variables, $f_y \in \mathbb{R}^{n_y}$, H_x is an m by n_x matrix, H_y is an m by n_y matrix, m is the number of linear constraints, and $h \in \mathbb{R}^m$. The *feasible set* of such a problem is the set of points that satisfy the constraints including the integrality restrictions; the feasible set of (3.1) is the set $\mathcal{T} = \{(x,y) : H_x x + H_y y \geq h,\ x \in \mathbb{R}^{n_x},\ y \in \mathbb{B}^{n_y}\}$. A *feasible point* is an element of the feasible set. The *objective function* of this problem is the function that is to be minimized; in (3.1), the objective function is the linear function $f(x,y) = f_x^T x + f_y^T y$. An *optimal solution*, or *minimizer* of (3.1) is a feasible

point (x^*, y^*) such that for all other (x, y) in the feasible set, $f(x^*, y^*) \leq f(x, y)$. If the feasible set $\mathcal{T}$ is not empty, then there must exist at least one optimal solution. Although there may be multiple optimal solutions, they must evaluate to the same objective function value.

Standard branch-and-bound is a binary-tree search algorithm. Each node in the tree is a subproblem of (3.1) such that a subset $\{y_j : j \in \mathcal{Z}\}$ of the binary variables are fixed to zero and a subset $\{y_j : j \in \mathcal{P}\}$, where $\mathcal{P}$ is disjoint from $\mathcal{Z}$, of the binary variables are fixed to one. The root node of the tree is the mixed-integer programming problem (3.1); the sets $\mathcal{Z}$ and $\mathcal{P}$ associated with the root node are empty. The leaf nodes are subproblems of (3.1) for which every binary variable is fixed to zero or one; the union of the sets $\mathcal{Z}$ and $\mathcal{P}$ associated with a leaf node contains the indices of all of the binary variables. Every node that is not a leaf node has two child nodes. Child

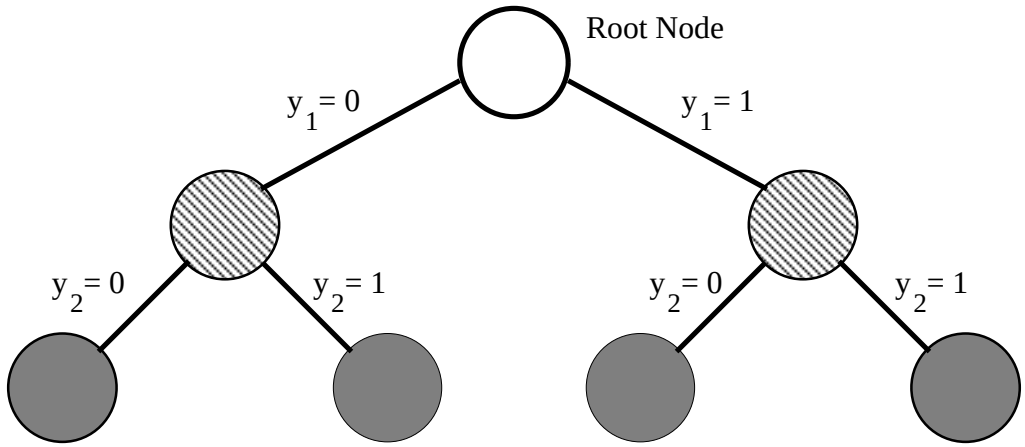

Figure 3.1 Example of a search tree for a mixed-integer programming problem with two binary variables

nodes are mixed-integer programming problems that are subproblems of their parent such that in one child, called the *down-branch*, a binary variable that was not fixed in the parent is fixed to zero and in the other child, called the *up-branch*, that same

variable is fixed to one. Moreover, that binary variable is referred to as the *branching variable*. If a node is not a leaf node, then each feasible point of its subproblem is a feasible point of exactly one of its children. An optimal solution to the subproblem of that node must then be an optimal solution to one of its two children.

An optimal solution to the original mixed-integer programming problem is a minimizer of the optimal solutions of the two children of the root node. Furthermore, if these children are not leaf nodes, an optimal solution to a child of the root node is a minimizer of the optimal solutions of its two children, and so on down to the leaf nodes. The subproblem corresponding to a leaf node is the linear programming problem

$$\begin{aligned} \text{minimize} \quad & f_x^T x + f_y^T \bar{y} \\ \text{subject to} \quad & H_x x \geq h - H_y \bar{y}, \\ & x \in \mathbb{R}^{n_x}, \end{aligned}$$

where $\bar{y} \in \mathbb{B}^{n_y}$ is the vector of zeros and ones based on the sets $\mathcal{Z}$ and $\mathcal{P}$ that characterize this leaf node. Although there are efficient algorithms for calculating an optimal solution to a linear programming problem, solving the 2^{n_y} linear programs of the leaf nodes in a branch-and-bound tree is impractical if n_y is large. For example, if $n_y = 52$ and we could solve 10,000 linear programs per second, it would take over 140 thousand years to solve the entire set of leaf nodes. Fortunately, it is possible to reduce the size of the search tree by determining in advance that there is no need to examine the leaf nodes that are descendants of a given node. A node and all of its descendants is called a *subtree*, and the process of eliminating a subtree is known as *pruning*. We can prune the subtree of a given node by determining that the node has no feasible points or that the node has no feasible point with an objective function value smaller than that of some known feasible point of (3.1).

Definition 3.1.1 A *lower bound* of a function $f : \mathbb{R}^n \to \mathbb{R}$ over a set $\mathcal{F} \subseteq \mathbb{R}^n$ is a scalar such that for every $v \in \mathcal{F}$, the value of $f(v)$ is no smaller than that scalar.

Pruning can occur when the lower bound determined for the optimal solution of a node of (3.1) is greater than the value of some known feasible solution of (3.1). Given the node defined by the sets $\mathcal{Z}$ and $\mathcal{P}$, we can determine a lower bound of the node by solving the *standard relaxation*,

$$\begin{aligned} \text{minimize} \quad & f(x,y) = f_x^T x + f_y^T y \\ \text{subject to} \quad & H_x x + H_y y \geq h, \\ & 0 \leq y_j \leq 1 \quad \forall j \notin \mathcal{Z} \cup \mathcal{P}, \\ & y_j = 0 \quad \forall j \in \mathcal{Z}, \\ & y_j = 1 \quad \forall j \in \mathcal{P}, \\ & x \in \mathbb{R}^{n_x}, \ y \in \mathbb{R}^{n_y}, \end{aligned} \tag{3.2}$$

of the corresponding subproblem of (3.1); we denote the feasible set of this relaxation $\mathcal{F}_{\mathcal{Z},\mathcal{P}}$. The feasible set of (3.2) contains the feasible set of the given node. If this linear program has no feasible points ($\mathcal{F}_{\mathcal{Z},\mathcal{P}} = \emptyset$), then neither does the given node. If this linear program has a feasible solution, then any lower bound of $f(x,y)$ over $\mathcal{F}_{\mathcal{Z},\mathcal{P}}$ is also a lower bound for the node. Given a lower bound on the optimal objective function value of (3.2), if it is no smaller than the value of the best known feasible point of (3.1), we can prune the node. It is important to note that although we may need to calculate the optimal solutions for some of the linear programs at the leaf nodes, we need only determine lower bounds for the other nodes.

The following algorithm is the simplest form of the branch-and-bound algorithm.

Algorithm 3.1.1 Standard Branch-and-Bound

Step 1: Let $\mathcal{Z} = \emptyset$, $\mathcal{P} = \emptyset$, and $\mathcal{BT} = \{(\mathcal{Z}, \mathcal{P})\}$. If possible, find a feasible point (x^{IP}, y^{IP}) of (3.1) and set $\phi^{IP} = f(x^{IP}, y^{IP})$. Otherwise, set $\phi^{IP} = \infty$.

Step 2: If $\mathcal{BT} = \emptyset$, goto Step 4. Otherwise, remove some ordered pair $(\mathcal{Z}, \mathcal{P})$ from the list $\mathcal{BT}$ and find a minimizer (x^*, y^*) of $f(x, y)$ over the feasible set $\mathcal{F}_{\mathcal{Z},\mathcal{P}}$. If $\mathcal{F}_{\mathcal{Z},\mathcal{P}} = \emptyset$ or $f(x^*, y^*) \geq \phi^{IP}$, repeat Step 2. Otherwise, goto Step 3.

Step 3: If $y_j^* \in \{0, 1\}$ for all $j \in \mathcal{N}$ set $(x^{IP}, y^{IP}) \leftarrow (x^*, y^*)$ and $\phi^{IP} \leftarrow \phi^*$ and goto Step 2. Otherwise, choose $k \in \mathcal{N}$ such that $0 < y_k^* < 1$ and set $\mathcal{BT} \leftarrow \mathcal{BT} \cup (\mathcal{Z} \cup \{k\}, \mathcal{P}) \cup (\mathcal{Z}, \mathcal{P} \cup \{k\})$; goto Step 2.

Step 4: If $\phi^{IP} = \infty$, then (3.1) has no feasible points. Otherwise, an optimal solution to (3.1) is (x^{IP}, y^{IP}).

At each step in the algorithm, the best known feasible point (x^{IP}, y^{IP}) is called the *upper bound*, since the optimal objective function value of (3.1) can be no greater than $\phi^{IP} = f(x^{IP}, y^{IP})$. Furthermore, the minimal optimal objective function value of the relaxations of the parents of the problems in the set $\mathcal{BT}$ is called the lower bound and is denoted ϕ^L. Once the *relative optimality gap*,

$$\frac{|\phi^{IP} - \phi^L|}{|\phi^{IP}| + 1},$$

where ϕ^L is a lower bound of the root node, is smaller than a predetermined tolerance, the algorithm terminates.

Since there are at most $2^{n_y+1} - 1$ nodes in a standard branch-and-bound tree and associated with each node is a linear programming problem that can be solved in finite time, this algorithm will terminate in finite time. In fact, we have proven

that standard branch-and-bound will determine an optimal solution in finite time for any mathematical programming problem with a nonempty feasible set of the form of the feasible set of (3.1), provided there exists a finite algorithm for determining optimal solutions of the resulting leaf nodes. In Chapters 5 and 6, we provide efficient algorithms for solving the leaf nodes and the relaxations of the subproblems associated with each of the objective functions examined in this thesis. For the remainder of this chapter, it is assumed that these efficient algorithms exist.

3.2 Implicit Branch-and-Bound

The general form of the problem that we examine in this thesis is

$$\begin{array}{rl} \text{minimize} & f(x,y) \\ \text{subject to} & H_x x + H_y y \geq h, \\ & x \in \mathbb{R}^n,\ y \in \mathbb{B}^n, \end{array} \tag{3.3}$$

where the constraints $H_x x + H_y y \geq h$ include the linking constraints $l_j y_j \leq x_j \leq u_j y_j$ for each $j \in \mathcal{N} = \{1, \ldots, n\}$. Branch-and-bound may be applied directly to this problem as long as an algorithm for determining the optimal solution to a leaf node exists. Instead of using the standard relaxation, however, we have extended a relaxation due to Bienstock [3] that can be solved more efficiently than the standard relaxation. Bienstock examined a problem of the form (3.3) where the binary variables appear only in the linking constraints and the limited diversification constraint $\sum_{j \in \mathcal{N}} y_j \leq \kappa$. He suggested replacing the linking constraints with the simple bound constraints $l_j \leq u_j$ for each $j \in \mathcal{N}$, approximating the limited diversification constraint with a surrogate constraint, and branching implicitly on the continuous variables. Although we remove the binary variables and use a branching algorithm similar to that proposed by Bienstock, we propose a relaxation that allows us to approximate any constraint

on which a binary variable is incident. Furthermore, we prove that, using our approximations, the optimal objective function to the standard relaxation of a subproblem can be determined.

The *implicit relaxation* of a node defined by the sets $\mathcal{Z}$ and $\mathcal{P}$ in the branch-and-bound tree is

$$\begin{aligned} \text{minimize} \quad & f(x,y) \\ \text{subject to} \quad & \hat{H}_x x \geq \hat{h}, \\ & 0 \leq y_j \leq 1 \quad \forall j \in \mathcal{N}, \\ & x_j = y_j = 0 \quad \forall j \in \mathcal{Z}, \\ & y_j = 1 \quad \forall j \in \mathcal{P}, \\ & x \in \mathbb{R}^n, \; y \in \mathbb{R}^n, \end{aligned} \tag{3.4}$$

where $\mathcal{N} = \{1, \ldots, n\}$ and the linear constraints $\hat{H}_x x \geq \hat{h}$ are the constraints of $H_x x + H_y y \geq h$ for which no binary variable has a nonzero coefficient. The simple bounds $l_j \leq x_j \leq u_j$ for all $j \in \mathcal{N}$ are also included in $\hat{H}_x x \geq \hat{h}$; it can be assumed without loss of generality that $l_j \leq 0 \leq u_j$ and at least one bound is nonzero for each $j \in \mathcal{N}$. The feasible set of (3.4), denoted $\hat{\mathcal{F}}_{\mathcal{Z},\mathcal{P}}$, contains $\mathcal{F}_{\mathcal{Z},\mathcal{P}}$, and thus it also contains the feasible set of the node defined by the sets $\mathcal{Z}$ and $\mathcal{P}$. A lower bound of $f(x,y)$ over $\hat{\mathcal{F}}_{\mathcal{Z},\mathcal{P}}$ is a lower bound on the optimal solution for that node. Since y does not appear in either objective function considered in this thesis and, in the implicit relaxation, the binary variables are either fixed or are constrained solely to be on the interval $[0,1]$, their presence has no effect on the solution time of (3.4). Thus we can think of the binary variables as being "implicit".

The implicit branch-and-bound algorithm is similar to the standard branch-and-bound algorithm introduced in the first section of this chapter. At each node that is not a leaf node, a branching variable x_j can be chosen such that $j \notin \mathcal{Z} \cup \mathcal{P}$; in the down-branch branch, j is added to $\mathcal{Z}$ and in the up-branch, j is added to $\mathcal{P}$. At

a leaf node, the set $\mathcal{Z} \cup \mathcal{P}$ contains the index of every continuous variable, and, for these nodes, all of the binary variables and the linear constraints $H_x x \geq h - H_y \bar{y}$ are included as before: if $j \in \mathcal{Z}$, then $\bar{y}_j = 0$, otherwise $\bar{y}_j = 1$. The leaf node defined by the index sets $\mathcal{Z}$ and $\mathcal{P}$ in the implicit branch-and-bound tree is identical to the corresponding leaf node in the standard branch-and-bound tree. We now present the implicit branch-and-bound algorithm and then prove that it will determine an optimal solution of (3.3).

Algorithm 3.2.1 Implicit Branch-and-Bound

Step 1: Let $\mathcal{Z} = \emptyset$, $\mathcal{P} = \emptyset$, and $\mathcal{BT} = \{(\mathcal{Z}, \mathcal{P})\}$. If possible, find a feasible point (x^{IP}, y^{IP}) of (3.3) and set $\phi^{IP} = f(x^{IP}, y^{IP})$. Otherwise, set $\phi^{IP} = \infty$.

Step 2: If $\mathcal{BT} = \emptyset$, goto Step 5. Otherwise, remove some ordered pair $(\mathcal{Z}, \mathcal{P})$ from the list $\mathcal{BT}$. If $\mathcal{Z} \cup \mathcal{P} = \mathcal{N}$, goto Step 3. Otherwise, find a lower bound (x^*, y^*) of $f(x, y)$ over the feasible set $\hat{\mathcal{F}}_{\mathcal{Z},\mathcal{P}}$. If $\hat{\mathcal{F}}_{\mathcal{Z},\mathcal{P}} = \emptyset$ or $f(x^*, y^*) \geq \phi^{IP}$, repeat Step 2. Otherwise, goto Step 4.

Step 3: Determine $\bar{y}$ based on $\mathcal{Z}$ and $\mathcal{P}$ and determine the optimal solution $(x^*, \bar{y})$ of $f(x, \bar{y})$ over the convex set $\hat{\mathcal{F}}_{\mathcal{Z},\mathcal{P}}$. If $f(x^*, \bar{y}) < \phi^{IP}$, set $(x^{IP}, y^{IP}) \leftarrow (x^*, \bar{y})$ and $\phi^{IP} \leftarrow \phi^*$. Goto Step 2.

Step 4: Choose $k \in \mathcal{N}$ such that $k \notin \mathcal{Z} \cup \mathcal{P}$ and set $\mathcal{BT} \leftarrow \mathcal{BT} \cup (\mathcal{Z} \cup \{k\}, \mathcal{P}) \cup (\mathcal{Z}, \mathcal{P} \cup \{k\})$; goto Step 2.

Step 5: If $\phi^{IP} = \infty$, then (3.3) has no feasible points. Otherwise, an optimal solution to (3.3) is (x^{IP}, y^{IP}).

Theorem 3.2.1 The optimal objective function value of the mixed-integer nonlinear programming problem (3.3) found by the implicit branch-and-bound algorithm is the same as that found by the standard branch-and-bound algorithm.

Proof: Since the leaf nodes are the same in both algorithms, the objective function values of the optimal leaf nodes are the same in both algorithms. It only remains to be shown that the implicit branch-and-bound algorithm will not prune the subtree of an ancestor of an optimal leaf node unless an optimal solution to (3.3) is already known. The implicit relaxation of an ancestor of a leaf node in the new formulation has a subset of the constraints of that leaf. Also, a lower bound of the ancestor can be no greater that the optimal objective function value of that leaf. Thus an ancestor of an optimal leaf node will be pruned only if an optimal solution to (3.3) is already known.□

The implicit branch-and-bound method works well when the optimal solution of the implicit relaxation of a node is not much smaller than the optimal solution of the standard relaxation of that node. We expect this method to perform well when the binary variables appear in few constraints other than the $2n$ linking constraints. The implicit relaxations have half the number of variables as do the standard relaxations, since the values of the binary variables can be determined before solving the subproblem (3.4). This, along with the removal of the linking constraints, has enabled us to solve the implicit relaxations in less than a quarter of the time it takes to solve the standard relaxations.

The freedom in the choice of the y^* of the optimal solution of the implicit relaxation of a node allows for an additional improvement. Given the optimal solution (x^*, y^*) determined for the implicit relaxation of the node defined by the sets $\mathcal{Z}$ and $\mathcal{P}$, we

set $\bar{y} \in \mathbb{B}^n$ such that $\bar{y}_j = 1$ for every $j \in \mathcal{N} \backslash \mathcal{Z}$ and $\bar{y}_j = 0$ for all $j \in \mathcal{Z}$. If $(x^*, \bar{y})$ is a feasible point of (3.3), then the upper bound of (3.3) can be updated and the subtree of this node can be pruned.

3.3 Approximating Constraints

In the implicit branch-and-bound algorithm, the feasible set of the standard relaxation of the subproblem characterized by the sets $\mathcal{Z}$ and $\mathcal{P}$ may be strictly contained in the feasible set of the implicit relaxation of that node. This implies that the lower bound determined for a node may be greater in the standard branch-and-bound algorithm than in the implicit branch-and-bound algorithm, allowing the possibility that more nodes may be created in the later algorithm. Instead of completely removing the constraints which have at least one nonzero coefficient for a binary variable, Bienstock [3] introduced a surrogate constraint which approximates the limited diversification constraint using only continuous variables. Since we have placed no limitation on the sign of the coefficients of the binary variables nor in what constraints they might appear, we have derived valid approximations of the general linear inequality constraints $H_x x + H_y y \geq h$. These approximations are intended to reduce the size of the feasible sets of the implicit relaxations, but they are such that the branch-and-bound algorithm is guaranteed to find the optimal solution to (3.3).

> **Definition 3.3.1** Given $\alpha \in \mathbb{R}$ and $a, b \in \mathbb{R}^n$, the inequality $a^T x + b^T y \geq \alpha$ is *valid* for a node in the implicit branch-and-bound tree of (3.3) if this constraint is satisfied by every feasible point of that node.

If there is a constraint in $H_x x + H_y y \geq h$ for which there is a binary variable that has a nonzero coefficient, this inequality can be written as $a^T x + \overline{b}^T y - \underline{b}^T y \geq \alpha$, where $\alpha \in \mathbb{R}$, $a, \overline{b}, \underline{b} \in \mathbb{R}^n$, $\overline{b}_j, \underline{b}_j \geq 0$, and $\overline{b}^T \underline{b} = 0$. For any index $j \in \mathcal{N}$, the

constraints $l_j y_j \leq x_j \leq u_j y_j$ and $0 \leq y_j \leq 1$ are valid at every node in the implicit branch-and-bound tree. We know that either $l_j < 0$ or $u_j > 0$. If $l_j < 0$, then the inequality $y_j \geq x_j / l_j$ is valid; if $u_j > 0$, the inequality $y_j \geq x_j / u_j$ is valid. If $u_j > 0$, we set $r_j = u_j$; otherwise, we set $r_j = l_j$.

Theorem 3.3.1 Under the above conditions, the *approximating constraint*

$$a^T x + \sum_{j=1}^{n} \left(\overline{b}_j - \frac{\underline{b}_j x_j}{r_j} \right) \geq \alpha \tag{3.5}$$

is a valid inequality for every node of the search tree of (3.3).

Proof: The inequalities

$$\begin{aligned} a^T x + \overline{b}^T y - \underline{b}^T y &\geq \alpha, \\ \overline{b}^T y \equiv \textstyle\sum_{j=1}^{n} \overline{b}_j y_j &\leq \textstyle\sum_{j=1}^{n} \overline{b}_j, \quad \text{and} \\ \underline{b}^T y \equiv \textstyle\sum_{j=1}^{n} \underline{b}_j y_j &\geq \textstyle\sum_{j=1}^{n} \frac{\underline{b}_j x_j}{r_j} \end{aligned}$$

are valid for every node of the search tree for (3.3), thus the inequality (3.5) is valid for every node.□

Approximating constraints can be strengthened at the node defined by the sets $\mathcal{Z}$ and $\mathcal{P}$. If j is in $\mathcal{Z}$, then $y_j = 0$ for every node in the subtree of that node. Likewise, if j is in $\mathcal{P}$, then $y_j = 1$ for every node in the subtree of that node.

Corollary 3.3.1 The *updated approximating constraint*

$$a^T x + \sum_{j \in \mathcal{N} \setminus \mathcal{Z}} \overline{b}_j - \sum_{j \in \mathcal{P}} \underline{b}_j - \sum_{j \in \mathcal{N} \setminus (\mathcal{Z} \cup \mathcal{P})} \frac{\underline{b}_j x_j}{r_j} \geq \alpha.$$

is a valid inequality for every node in the subtree of the node defined by the sets $\mathcal{Z}$ and $\mathcal{P}$.

Example 3.3.1 If the limited diversification constraint $\sum_{j\in\mathcal{N}} y_j \leq \kappa$ is in the set of constraints , then, for the subtree of the node defined by the sets $\mathcal{Z}$ and $\mathcal{P}$, the limited diversification constraint can be approximated by the inequality

$$\sum_{j\in\mathcal{N}\setminus(\mathcal{Z}\cup\mathcal{P})} -x_j/r_j \geq -\kappa + |\mathcal{P}|.$$

An updated approximating constraint of a node is not unique, if, for an index $j \in \mathcal{N}\setminus(\mathcal{Z}\cup\mathcal{P})$, $l_j < 0 < u_j$ and $\underline{b}_j$ is strictly greater than zero. If, for the optimal solution determined for that node, (x^*, y^*), the values of x_j^* and r_j are of opposite sign, it may be possible to determine an approximating constraint violated by (x^*, y^*). For all $j \notin \mathcal{Z}\cup\mathcal{P}$ such that $l_j < 0 < u_j$ and the values of x_j^* and r_j are of opposite sign, we can set r_j to the opposite bound. If any of these new updated approximating constraints are violated, a new, possible larger lower bound for the node can be calculated.

3.4 Improvement for Approximating Constraints

The implicit branch-and-bound method works better than standard branch-and-bound when the optimal objective function value of the implicit relaxation of a node is not much smaller than the optimal objective function value of the standard relaxation of that node. We prove in this section that it is possible to generate valid inequalities that guarantee that the optimal solutions of the implicit and standard relaxations have the same optimal objective function values.

In the following theorem, we let $\mathcal{F}_{\mathcal{Z},\mathcal{P}}$ represent the feasible set of the standard relaxation of the node defined by the index sets $\mathcal{Z}$ and $\mathcal{P}$. Moreover, since the binary variables y do not appear in either objective function examined in this thesis, the objective function $f(x, y)$ can be denoted as $\hat{f}(x)$.

Lemma 3.1 The point $x^* \in \mathbb{R}^n$ is a minimizer of $\hat{f}(x)$ over the set $\{x : (x,y) \in \mathcal{F}_{\mathcal{Z},\mathcal{P}}\}$ if and only if there exists a $y^* \in \mathbb{R}^n$ such that the point (x^*, y^*) is a minimizer of $f(x,y)$ over the feasible set $\mathcal{F}_{\mathcal{Z},\mathcal{P}}$.

Proof: Let $x^* \in \mathbb{R}^n$ be a minimizer of $\hat{f}(x)$ over the set $\{x : (x,y) \in \mathcal{F}_{\mathcal{Z},\mathcal{P}}\}$. Because $x^* \in \{x : (x,y) \in \mathcal{F}_{\mathcal{Z},\mathcal{P}}\}$, there exists a $y^* \in \mathbb{R}^n$ such that $(x^*, y^*) \in \mathcal{F}_{\mathcal{Z},\mathcal{P}}$. Assume by way of contradiction that there exists a point $(\bar{x}, \bar{y}) \in \mathcal{F}_{\mathcal{Z},\mathcal{P}}$ such that $f(\bar{x}, \bar{y}) < f(x^*, y^*)$. Since this implies $\bar{x} \in \{x : (x,y) \in \mathcal{F}_{\mathcal{Z},\mathcal{P}}\}$ and $\hat{f}(\bar{x}) < \hat{f}(x^*)$, the point x^* is not a minimizer of $\hat{f}(x)$ over the set $\{x : (x,y) \in \mathcal{F}_{\mathcal{Z},\mathcal{P}}\}$. Because this is a contradiction, there exists a $y^* \in \mathbb{R}^n$ such that (x^*, y^*) is a minimizer of $f(x,y)$ over the feasible set $\mathcal{F}_{\mathcal{Z},\mathcal{P}}$.

Let $(x^*, y^*) \in \mathbb{R}^{2n}$ be a minimizer of $f(x,y)$ over the set $\mathcal{F}_{\mathcal{Z},\mathcal{P}}$. Assume by way of contradiction that there exists a point $\bar{x} \in \{x : (x,y) \in \mathcal{F}_{\mathcal{Z},\mathcal{P}}\}$ such that $\hat{f}(\bar{x}) < \hat{f}(x^*)$. Since this implies there exists a $\bar{y} \in \mathbb{R}^n$ such that $(\bar{x}, \bar{y}) \in \mathcal{F}_{\mathcal{Z},\mathcal{P}}$ and $f(\bar{x}, \bar{y}) < f(x^*, y^*)$, the point (x^*, y^*) is not a minimizer of $f(x,y)$ over the set $\mathcal{F}_{\mathcal{Z},\mathcal{P}}$. Because this is a contradiction, x^* is a minimizer of $\hat{f}(x)$ over the set $\{x : (x,y) \in \mathcal{F}_{\mathcal{Z},\mathcal{P}}\}$. □

Given a set of valid linear inequalities $Cx \geq d$ such that $\bar{x} \in \{x : Cx \geq d\}$ if and only if $\bar{x} \in \{x : (x,y) \in \mathcal{F}_{\mathcal{Z},\mathcal{P}}\}$, then by Lemma 3.1 the minimum value of $\hat{f}(x)$ over the set $\{x : Cx \geq d\}$ equals the minimum value of $f(x,y)$ over $\mathcal{F}_{\mathcal{Z},\mathcal{P}}$. In the following theorem, the linear inequalities $Cx \geq d$ are the simple bounds of the continuous variables and all of the possible approximating constraints to the rows of $H_x x + H_y y \geq h$. Note that $Cx \geq d$ includes the constraints $\hat{H}_x x \geq \hat{h}$ from (3.4).

Theorem 3.4.1 If, for every $j \in \mathcal{N}$, the coefficient b_j is nonpositive for each non-linking constraint $a^T x + b^T y \geq \alpha$ of the constraints $H_x x + H_y y \geq h$, then the vector $\bar{x} \in \mathbb{R}^n$ satisfies the inequalities $Cx \geq d$ if and only if there exists a $\bar{y} \in \mathbb{R}^n$ such that $(\bar{x}, \bar{y}) \in \mathcal{F}_{\emptyset,\emptyset}$.

Proof: Let $\bar{x} \in \mathbb{R}^n$ satisfy the inequalities $Cx \geq d$. For each $j \in \mathcal{N}$, if $\bar{x}_j = 0$, set $\bar{y}_j = 0$ and r_j to a nonzero bound of x_j; otherwise set $\bar{y}_j = \bar{x}_j / r_j$, where $r_j = u_j$ if $\bar{x}_j$ is strictly positive and $r_j = l_j$ if $\bar{x}_j$ is strictly negative. The point $(\bar{x}, \bar{y})$ clearly satisfies the bound constraints $0 \leq y_j \leq 1$ on the binary variables and the linking constraints $l_j y_j \leq x_j \leq u_j y_j$. We only need to show that $(\bar{x}, \bar{y})$ satisfies each non-linking constraint $a^T x + b^T y \geq \alpha$ of $H_x x + H_y y \geq h$. Since $\bar{x}$ satisfies the approximating constraint $a^T x + \sum_{j \in \mathcal{N}} b_j x_j / r_j \geq \alpha$ of $a^T x + b^T y \geq \alpha$ and $\bar{y}_j = \bar{x}_j / r_j$, the point $(\bar{x}, \bar{y})$ must satisfy $a^T x + b^T y \geq \alpha$. Thus $(\bar{x}, \bar{y})$ is in $\mathcal{F}_{\emptyset,\emptyset}$.

Let $(\bar{x}, \bar{y})$ satisfy $H_x x + H_y y \geq h$ and $0 \leq y_j \leq 1$ for all $j \in \mathcal{N}$. By the validity of the approximating constraints, $\bar{x}$ must be valid for $Cx \geq d$.□

When the binary variables do not all have nonpositive coefficients in the non-linking constraint rows of $H_x x + H_y y \geq h$, more work must be done to generate a set of valid inequalities $Cx \geq d$ such that $\bar{x} \in \{x : (x, y) \in \mathcal{F}_{\mathcal{Z},\mathcal{P}}\}$ if and only if $C\bar{x} \geq d$. The Fourier-Motzkin elimination method [50] generates such a set of inequalities. We present this algorithm and a proof that it generates the appropriate set of inequalities below.

Before presenting this algorithm, we first introduce some notation. Let C_{ij} represent the coefficient in the ith row and jth column of C. The symbol $C_{:j}$ represents the jth column of C, and $C_{i:}$ represents the ith row of C. The symbol $C_{i,1:k}$ represents the

first k columns of the ith row of C. For vectors, $y_{1:k}$ represents the first k components of y.

Algorithm 3.4.1 Fourier-Motzkin Elimination

Step 1: Let $k = n$ and $C^k x + D^k y_{1:k} \geq d^k$ represent the rows of $H_x x + H_y y \geq h$, including the linking constraints and the bound constraints $0 \leq y_j \leq 1$ on the binary variables. The superscript k represents the iteration, and m^k is the number of constraints in $C^k x + D^k y_{1:k} \geq d^k$. Note that m^k does not mean m to the kth power.

Step 2: If $k = 0$, set $Cx \geq d$ to $C^0 x \geq d^0$; STOP.

Step 3: For each $i \in \{1, \ldots, m^k\}$, if $|D_{ik}| > 0$, scale the constraint $C^k_{i:} x + D^k_{i:} y_{1:k} \geq d^k_i$ by $1/|D_{ik}|$ so that the coefficient of the kth binary variable is 0, 1, or -1. Represent these new m^k constraints by $\hat{C}^k x + \hat{D}^k y_{1:k} \geq \hat{d}^k$.

Step 4: For each constraint of $\hat{C}^k x + \hat{D}^k y_{1:k} \geq \hat{d}^k$ that has a positive coefficient for the kth binary variable, add it to each constraint that has a negative coefficient for the kth binary variable. These new constraints, along with the constraints of $\hat{C}^k x + \hat{D}^k y_{1:k} \geq \hat{d}^k$ that have a coefficient of zero for the kth binary variable, make up $C^{k-1} x + D^{k-1} y_{1:k-1} \geq d^{k-1}$. Set $k = k - 1$; goto Step 2.

Theorem 3.4.2 Algorithm 3.4.1 yields a set of constraints $Cx \geq d$ such that $\bar{x} \in \{x : Cx \geq d\}$ if and only if there exists a $\bar{y}$ such that $(\bar{x}, \bar{y}) \in \mathcal{F}_{\emptyset,\emptyset}$.

Proof: First, let $(\bar{x}, \bar{y}) \in \mathcal{F}_{\emptyset,\emptyset}$. Since $(\bar{x}, \bar{y})$ satisfies $H_x x + H_y y \geq h$ and $0 \leq y_j \leq 1$ for all $j \in \mathcal{N}$, then it must also satisfy all linear combinations of these constraints. Thus $\bar{x}$ satisfies the final constraints $Cx \geq d$.

To prove the other direction, we need only prove that one iteration of the algorithm works; this implies that the whole algorithm works. Let the point $(\bar{x}, \bar{y})$ satisfy $C^{k-1}x + D^{k-1}y_{1:k-1} \geq d^{k-1}$. We need to show that there exists $\hat{y} \in \mathbb{R}^n$ such that $(\bar{x}, \hat{y})$ satisfies $C^k x + D^k y_{1:k} \geq d^k$. Assume by way of contradiction that there does not exist a $\hat{y}$ such that $(\bar{x}, \hat{y})$ satisfies $C^k x + D^k y_{1:k} \geq d^k$. Thus there does not exist a $\hat{y_k} \in \mathbb{R}$ such that

$$D^k_{:k}\hat{y}_k \geq d^k - C^k\bar{x} - D^k_{:,1:k-1}\bar{y}_{1:k-1}. \tag{3.6}$$

For each $1 \leq i \leq m^k$, if $|D^k_{ik}| > 0$, scale the ith constraint by $1/|D^k_{ik}|$. Then the infeasibility of (3.6) implies that there does not exist a $\hat{y_k}$ such that

$$\mathrm{sgn}(D^k_{ik})\hat{y}_k \geq \hat{d}^k_i - \hat{C}^k_{i:}\bar{x} - \hat{D}^k_{i,1:k-1}\bar{y}_{1:k-1} \quad \forall i \in \{1, \ldots, m^k\}.$$

This in turn implies that either

1. there is an $i \in \{1, \ldots, m^k\}$ such that $D^k_{ik} = 0$ and $\hat{d}^k_i - \hat{C}^k_{i:}\bar{x} - \hat{D}^k_{:,1:k-1}\bar{y}_{1:k-1} > 0$ or
2. there are $i, j \in \{1, \ldots, m^k\}$ such that $|D^k_{ik}| < 0$, $|D^k_{jk}| > 0$, and $(\hat{d}^k_i - \hat{C}^k_{i:}\bar{x} - \hat{D}^k_{i,1:k-1}\bar{y}_{1:k-1}) + (\hat{d}^k_j - \hat{C}^k_{j:}\bar{x} - \hat{D}^k_{j,1:k-1}\bar{y}_{1:k-1}) > 0$.

The first case leads to a contradiction because we know that $(\bar{x}, \bar{y})$ satisfies $\hat{C}^k_i x + \hat{D}^k_{i,1:k-1}y_{1:k-1} \geq \hat{d}^k_i$, since this constraint is included in $C^{k-1}x + D^{k-1}y_{1:k-1} \geq d^{k-1}$. The second case also leads to a contradiction because we know that $(\bar{x}, \bar{y})$ satisfies

$$\hat{C}^k_{i:}x + \hat{D}^k_{i,1:k-1}y_{1:k-1} + \hat{C}^k_{j:}x + \hat{D}^k_{j,1:k-1}y_{1:k-1} \geq \hat{d}^k_i + \hat{d}^k_j$$

since this constraint is included in $C^{k-1}x + D^{k-1}y_{1:k-1} \geq d^{k-1}$.

Thus there exists a $\hat{y}\mathbb{R}^n$ such that $(\bar{x}, \hat{y})$ satisfies $C^k x + D^k y_{1:k} \geq d^k$, and we are done.□

It is easy to extend Algorithm 3.4.1 and Theorems 3.4.1 and 3.4.2 to other nodes in the implicit branch-and-bound tree.

Although Algorithm 3.4.1 provides a method for us to remove the binary variables from the formulation without losing any of their benefits, it is not a practical algorithm since it possibly could generate an exponential number of constraints! In the next section we provide a practical algorithm to generate valid inequalities based on the approximating constraints.

3.5 Practical Improvement of Approximating Constraints

In the standard branch-and-bound algorithm, an inequality of the form $\pi^T H_x x + \pi^T H_y y \geq \pi^T h$ is valid at every node, where $\pi \in \mathbb{R}^m$, $\pi \geq \mathbf{0}$, and $\mathbf{0}$ is the zero vector of the appropriate dimension. Furthermore, this constraint is satisfied by the optimal solution of the standard relaxation of every node in the search tree. However, the approximating constraint of this valid inequality may be violated by the optimal solution (x^*, y^*) determined for the implicit relaxation of this node. Since the approximating constraints are derived based on the signs of the coefficients of the binary variables, we insist that π satisfies $\pi^T H_y \leq \mathbf{0}$. Under this condition, an approximating constraint of the valid inequality $\pi^T H_x x + \pi^T H_y y \geq \pi^T h$ is $\pi^T H_x x + \pi^T \hat{H}_y x \geq \pi^T h$, where the jth column of $\hat{H}_y$ is the jth column of H_y scaled by $1/r_j$, and r_j is a nonzero simple bound of x_j.

Theorem 3.5.1 If $\pi \geq \mathbf{0}$ and $\pi^T H_y \leq \mathbf{0}$, then the inequality $\pi^T(H_x + \hat{H}_y)x \geq \pi^T h$ is a valid inequality for the entire implicit branch-and-bound search tree. Furthermore, if ϵ^* is strictly positive for the optimal solution

(ϵ^*, π^*) of the linear programming problem

$$\begin{aligned} \text{maximize} \quad & \epsilon \\ \text{subject to} \quad & \pi^T H_y \leq \mathbf{0} \\ & \pi^T(H_x x^* + \hat{H}_y x^* - h) + \epsilon \leq 0, \\ & \pi \geq \mathbf{0}, \\ & 0 \leq \epsilon \leq 1, \\ & \epsilon \in \mathbb{R},\ \pi \in \mathbb{R}^m, \end{aligned}$$

where (x^*, y^*) is an optimal solution to the implicit relaxation of the current node, then $\pi^{*T}(H_x + \hat{H}_y)x \geq \pi^{*T}h$ is a valid inequality that is violated by (x^*, y^*).

Proof: Since $\pi^* \geq \mathbf{0}$, the inequality $\pi^{*T}(H_x x + H_y y) \geq \pi^{*T}h$ is a valid in equality for the current node. Since $\pi^{*T}H_y \leq \mathbf{0}$, by Theorem 3.3.1, the constraint $\pi^{*T}(H_x + \hat{H}_y)x \geq \pi^{*T}h$ is valid for the current node. If $\epsilon^* > 0$, then $\pi^{*T}(H_x + \hat{H}_y)x \geq \pi^{*T}h$ is violated by (x^*, y^*).□

Example 3.5.1 If the feasible set of a mixed-integer programming problem is

$$\begin{aligned} -y_1 - y_2 - y_3 &\geq -2, \\ -y_1 + y_2 &\geq 0, \\ 0 \leq x_j \leq \frac{2}{3}y_j \quad &\text{and} \quad y_j \in \{0,1\} \quad \forall j \in \{1,2,3\}, \end{aligned}$$

then the implicit relaxation of the root node of the branch-and-bound tree has the feasible set defined by

$$\begin{aligned} -\frac{3}{2}x_1 - \frac{3}{2}x_2 - \frac{3}{2}x_3 &\geq -2, \\ -\frac{3}{2}x_1 &\geq -1, \text{ and} \end{aligned}$$

$$0 \leq x_j \leq \frac{2}{3} \qquad \forall j \in \{1,2,3\}.$$

Given the objective to minimize $-x_1 - x_3$, an optimal solution to the implicit relaxation of the root node is $x_1^* = \frac{2}{3}$, $x_2^* = 0$, $x_3^* = \frac{2}{3}$, and $y_j^* = 0 \ \ \forall j$.

Setting $\pi_1^* = 1$ and $\pi_2^* = 1$ gives the valid inequality $-2y_1 - y_3 \geq -2$. The approximating constraint of this valid inequality is $-3x_1 - \frac{3}{2}x_3 \geq -2$, which is violated by (x^*, y^*).

Chapter 4

Improvements to Implicit Branch-and-Bound

This chapter is devoted to the procedures and heuristics developed to reduce the total number of nodes that are formed in the implicit branch-and-bound search tree. We first discuss a variety of classes of valid inequalities designed to allow us to determine better lower bounds for each node in the branch-and-bound tree. We then describe the procedures we have derived for finding feasible points; together the valid inequalities and upper bound heuristics aid in our ability to prune larger subtrees. We also discuss the branching variable selection heuristics that we have designed to create child nodes that hopefully can be pruned quickly.

4.1 Valid Inequalities

The approximating constraints derived in §3.3 are beneficial in determining a tighter formulation of the feasible set of the implicit relaxation of a subproblem in the branch-and-bound tree. In general, a tighter formulation can be created by deriving a valid inequality that is not valid for some feasible point of the relaxation of the current formulation of the subproblem. This type of valid inequality is called a *cutting plane.* Adding a cutting plane and repeating the optimization of the relaxation, the value of the largest known lower bound of this node may increase enough to prune the subtree of this node.

To describe this process more fully, we must first develop some terminology. Given the node characterized by $\mathcal{Z}$ and $\mathcal{P}$, the set $\mathcal{T}_{\mathcal{Z},\mathcal{P}} \subseteq \mathbb{R}^{2n}$ contains the points feasible for that node.

Definition 4.1.1 Given a set $\mathcal{T} \subseteq \mathbb{R}^n$, for any finite set of points $v^i \in \mathcal{T}$ $(i = \{1, \ldots, t\})$ and scalars $\lambda_i \geq 0$ such that $\sum_{i=1}^{t} \lambda_i = 1$, the point $v = \sum_{i=1}^{t} \lambda_i v^i$ is a *convex combination* of points of $\mathcal{T}$. The *convex hull* of $\mathcal{T}$ is the set of all points that are convex combinations of points in $\mathcal{T}$; it is denoted $conv(\mathcal{T})$.

At the node defined by the sets $\mathcal{Z}$ and $\mathcal{P}$, the convex hull of $\mathcal{T}_{\mathcal{Z},\mathcal{P}}$ is the smallest convex set that contains $\mathcal{T}_{\mathcal{Z},\mathcal{P}}$. Furthermore, the feasible set of the implicit relaxation of the node, $\hat{\mathcal{F}}_{\mathcal{Z},\mathcal{P}}$, must contain the set $\mathcal{T}_{\mathcal{Z},\mathcal{P}}$. The closer the set $\hat{\mathcal{F}}_{\mathcal{Z},\mathcal{P}}$ approximates the convex hull of $\mathcal{T}_{\mathcal{Z},\mathcal{P}}$, the greater the lower bound calculated for the node will be, increasing the possibility of being able to prune the subtree of that node. Adding a valid inequality to $\hat{\mathcal{F}}_{\mathcal{Z},\mathcal{P}}$ may help to determine a better approximation of $conv(\mathcal{T}_{\mathcal{Z},\mathcal{P}})$.

Example 4.1.1 We would like to find the optimal portfolio that satisfies the constraints

$$\begin{aligned} \frac{4}{3}x_1 + 2x_2 &\leq 1, \\ 0 \leq x_1 &\leq \frac{1}{2}, \\ 0 \leq x_2 &\leq \frac{3}{4}, \end{aligned} \tag{4.1}$$

and has no more than one asset. At the start of the branch-and-bound algorithm, the feasible set of the implicit relaxation of the root node, $\hat{\mathcal{F}}_{\emptyset,\emptyset}$, is the set of points (x, y) that satisfy the constraints (4.1) and the approximating constraint of $y_1 + y_2 \leq 1$, which is

$$2x_1 + \frac{4}{3}x_2 \leq 1.$$

Given the objective function $f(x, y) = -0.09x_1 - 0.09x_2 + 0.01x_1^2 + 0.01x_2^2 + 0.01x_1x_2$, an optimal solution is $x_1^* = 0.3$, $x_2^* = 0.3$, $y_1^* = 1$, and $y_2^* = 1$

with objective function value $f(x^*, y^*) = -0.05373$. Examining Figure 4.1, we see that the inequality

$$0 \leq x_2 \leq \frac{1}{2}$$

is valid for the set $\mathcal{T}_{\mathcal{Z},\mathcal{P}}$ of the root node. Thus, the approximating constraint can be replaced with

$$2x_1 + 2x_2 \leq 1,$$

and the optimal objective function value of $f(x, y)$ over the new feasible set $\hat{\mathcal{F}}_{\mathcal{Z},\mathcal{P}}$ is -0.04481.

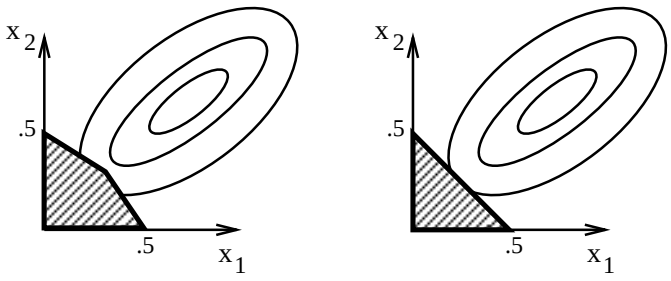

Figure 4.1 Example 4.1.1: Feasible values of x for $\hat{\mathcal{F}}_{\mathcal{Z},\mathcal{P}}$ vs. $conv(\mathcal{T}_{\mathcal{Z},\mathcal{P}})$

For the remainder of this chapter we let $C_{\mathcal{Z},\mathcal{P}}x \geq d_{\mathcal{Z},\mathcal{P}}$ denote all of the valid inequalities generated for the node defined by the sets $\mathcal{Z}$ and $\mathcal{P}$. The feasible set of the implicit relaxation of that node is the set

$$\hat{\mathcal{F}}_{\mathcal{Z},\mathcal{P}} = \{(x, y) : \hat{H}_x x \geq \hat{h},\ C_{\mathcal{Z},\mathcal{P}}x \geq d_{\mathcal{Z},\mathcal{P}},\ 0 \leq y_j \leq 1 \ \ \forall j \in \mathcal{N},$$
$$x_j = y_j = 0 \ \ \forall j \in \mathcal{Z},\ y_j = 1 \ \ \forall j \in \mathcal{P}\},$$

where $\hat{H}_x$ and $\hat{h}$ are as defined in (3.4).

4.1.1 Disjunctive Cuts

Disjunctive cuts are valid inequalities that were introduced by Balas [1] and discussed in [3]. In this section, we show how these cuts can be derived at a node in the implicit branch-and-bound tree.

At the node defined by the sets $\mathcal{Z}$ and $\mathcal{P}$ in the implicit branch-and-bound tree, an optimal solution (x^*, y^*) of $f(x, y)$ over the given feasible set $\hat{\mathcal{F}}_{\mathcal{Z},\mathcal{P}}$ is determined. If the subtree of this node cannot be pruned, then a branching variable, x_k is chosen, and $\hat{\mathcal{F}}_{\mathcal{Z}\cup\{k\},\mathcal{P}} \cup \hat{\mathcal{F}}_{\mathcal{Z},\mathcal{P}\cup\{k\}}$, the union of the feasible sets of the two children created by this branching variable, must contain the set $\mathcal{T}_{\mathcal{Z},\mathcal{P}}$. Instead of immediately branching to create the two child nodes, we first determine if the point x^* is in the set $\{x : (x, y) \in conv(\hat{\mathcal{F}}_{\mathcal{Z}\cup\{k\},\mathcal{P}} \cup \hat{\mathcal{F}}_{\mathcal{Z},\mathcal{P}\cup\{k\}})\}$. If it is not, then (x^*, y^*) is not in $conv(\mathcal{T}_{\mathcal{Z},\mathcal{P}})$ and Farka's Lemma [41] can be used to derive an inequality $a^T x \geq \alpha$ that is valid for this node and such that $a^T x^* < \alpha$. Adding this disjunctive cut to the formulation and solving the augmented subproblem, the value of the largest known lower bound of this node may increase enough to prune the subtree of this node. Throughout this section, the projection of the set $\hat{\mathcal{F}}_{\mathcal{Z},\mathcal{P}} \subseteq \mathbb{R}^{2n}$ onto the continuous variables is denoted $\hat{\mathcal{F}}_{\mathcal{Z},\mathcal{P}}(x) = \{x : (x, y) \in \hat{\mathcal{F}}_{\mathcal{Z},\mathcal{P}}\}$.

The following derivation determines a system of linear inequalities that are satisfied by every $x \in \{x : (x, y) \in conv(\hat{\mathcal{F}}_{\mathcal{Z}\cup\{k\},\mathcal{P}} \cup \hat{\mathcal{F}}_{\mathcal{Z},\mathcal{P}\cup\{k\}})\}$. Note that $\{x : (x, y) \in conv(\hat{\mathcal{F}}_{\mathcal{Z}\cup\{k\},\mathcal{P}} \cup \hat{\mathcal{F}}_{\mathcal{Z},\mathcal{P}\cup\{k\}})\}$ is equivalent to $conv(\hat{\mathcal{F}}_{\mathcal{Z}\cup\{k\},\mathcal{P}}(x) \cup \hat{\mathcal{F}}_{\mathcal{Z},\mathcal{P}\cup\{k\}}(x))$.

1. A point $x \in \mathbb{R}^n$ is in the set $\{x : (x, y) \in conv(\hat{\mathcal{F}}_{\mathcal{Z}\cup\{k\},\mathcal{P}} \cup \hat{\mathcal{F}}_{\mathcal{Z},\mathcal{P}\cup\{k\}})\}$ if and only if there exist $v \in \hat{\mathcal{F}}_{\mathcal{Z}\cup\{k\},\mathcal{P}}(x)$, $w \in \hat{\mathcal{F}}_{\mathcal{Z},\mathcal{P}\cup\{k\}}(x)$, and $0 \leq \lambda \leq 1$ such that $x = \lambda v + (1 - \lambda)w$.

Equivalently, a point $x \in \mathbb{R}^n$ is in the set $\{x : (x,y) \in conv(\hat{\mathcal{F}}_{\mathcal{Z}\cup\{k\},\mathcal{P}} \cup \hat{\mathcal{F}}_{\mathcal{Z},\mathcal{P}\cup\{k\}})\}$ if and only if there exist $v \in \lambda\hat{\mathcal{F}}_{\mathcal{Z}\cup\{k\},\mathcal{P}}(x)$, $w \in (1-\lambda)\hat{\mathcal{F}}_{\mathcal{Z},\mathcal{P}\cup\{k\}}(x)$, and $0 \leq \lambda \leq 1$ such that $x = v + w$.

2. A point $v \in \mathbb{R}^n$ is an element of $\lambda\hat{\mathcal{F}}_{\mathcal{Z}\cup\{k\},\mathcal{P}}(x)$ if and only if

   (a) $\hat{H}_x v \geq \lambda\hat{h}$,

   (b) $C_{\mathcal{Z}\cup\{k\},\mathcal{P}} v \geq \lambda d_{\mathcal{Z}\cup\{k\},\mathcal{P}}$, and

   (c) $v_j = 0 \ \forall j \in \mathcal{Z} \cup \{k\}$.

3. A point $w \in \mathbb{R}^n$ is an element of $(1-\lambda)\hat{\mathcal{F}}_{\mathcal{Z},\mathcal{P}\cup\{k\}}(x)$ if and only if

   (a) $\hat{H}_x w \geq (1-\lambda)\hat{h}$,

   (b) $C_{\mathcal{Z},\mathcal{P}\cup\{k\}} w \geq (1-\lambda) d_{\mathcal{Z},\mathcal{P}\cup\{k\}}$, and

   (c) $w_j = 0 \ \forall j \in \mathcal{Z}$.

4. Substituting $x - v$ for w, the point x is in $\{x : (x,y) \in conv(\hat{\mathcal{F}}_{\mathcal{Z}\cup\{k\},\mathcal{P}} \cup \hat{\mathcal{F}}_{\mathcal{Z},\mathcal{P}\cup\{k\}})\}$ if and only if there exist $v \in \mathbb{R}^n$ and $\lambda \in \mathbb{R}$ which satisfy

$$\begin{aligned}
\hat{H}_x v - \lambda\hat{h} &\geq 0 \\
C_{\mathcal{Z}\cup\{k\},\mathcal{P}} v - \lambda d_{\mathcal{Z}\cup\{k\},\mathcal{P}} &\geq 0 \\
-\hat{H}_x v + \lambda\hat{h} &\geq -\hat{H}_x x + \hat{h} \\
-C_{\mathcal{Z},\mathcal{P}\cup\{k\}} v + \lambda d_{\mathcal{Z},\mathcal{P}\cup\{k\}} &\geq -C_{\mathcal{Z},\mathcal{P}\cup\{k\}} x + d_{\mathcal{Z},\mathcal{P}\cup\{k\}} \qquad (4.2) \\
v_j &= 0 \ \ \forall j \in \mathcal{Z} \cup \{k\} \\
v_j &= -x_j \ \ \forall j \in \mathcal{Z} \\
-\lambda &\geq 1 \\
\lambda &\geq 0.
\end{aligned}$$

We write system (4.2) as

$$G \begin{bmatrix} v \\ \lambda \end{bmatrix} \leq g(x), \tag{4.3}$$

where G is a $\hat{m}$ by $n+1$ matrix, $g(x) : \mathbb{R}^n \rightarrow \mathbb{R}^{\hat{m}}$, and $\hat{m}$ is the number of linear inequalities in (4.2). The point $\bar{x} \in \mathbb{R}^n$ is an element of $\{x : (x,y) \in conv(\hat{\mathcal{F}}_{\mathcal{Z}\cup\{k\},\mathcal{P}} \cup \hat{\mathcal{F}}_{\mathcal{Z},\mathcal{P}\cup\{k\}})\}$ if and only if system (4.3) with right-hand side $g(\bar{x})$ is feasible.

Theorem 4.1.1 Let (x^*, y^*) be the optimal solution calculated for the implicit relaxation of the node defined by the index sets $\mathcal{Z}$ and $\mathcal{P}$ and $k \in \mathcal{N}\backslash(\mathcal{Z} \cup \mathcal{P})$. Let $\gamma^* \in \mathbb{R}^{\hat{m}}$ be an optimal solution to the linear programming problem

$$\begin{aligned} \text{minimize} \quad & g(x^*)^T\gamma \\ \text{subject to} \quad & G^T\gamma = \mathbf{0}, \\ & g(x^*)^T\gamma \geq -1, \\ & \gamma \geq \mathbf{0}, \\ & \gamma \in \mathbb{R}^{\hat{m}}, \end{aligned} \tag{4.4}$$

where $\mathbf{0}$ represents the vector of zeros of appropriate dimension and G and $g(x)$ are as defined in (4.3). Then $g(x)^T\gamma^* \geq 0$ is a valid inequality for the given node.

Proof: Let $\bar{x} \in \mathcal{T}_{\mathcal{Z},\mathcal{P}}(x)$. Since every feasible point of a node must be valid in any relaxation of the node, $\bar{x} \in \{x : (x,y) \in conv(\hat{\mathcal{F}}_{\mathcal{Z}\cup\{k\},\mathcal{P}} \cup \hat{\mathcal{F}}_{\mathcal{Z},\mathcal{P}\cup\{k\}})\}$. Thus there are $\bar{v} \in \mathbb{R}^n$ and $\bar{\lambda} \in \mathbb{R}$ such that

$$G \begin{bmatrix} \bar{v} \\ \bar{\lambda} \end{bmatrix} \leq g(\bar{x}).$$

Then,

$$\gamma^{*T} g(\bar{x}) \geq \gamma^{*T} G \begin{bmatrix} \bar{v} \\ \bar{\lambda} \end{bmatrix} = 0.$$

Since, for any $\bar{x} \in \mathcal{T}_{\mathcal{Z},\mathcal{P}}(x)$, the inequality $g(\bar{x})^T \gamma^* \geq 0$ holds, then for the given node $g(x)^T \gamma^* \geq 0$ is a valid inequality.□

If the point $x^* \in \mathbb{R}^n$ is not in $\{x : (x,y) \in conv(\hat{\mathcal{F}}_{\mathcal{Z}\cup\{k\},\mathcal{P}} \cup \hat{\mathcal{F}}_{\mathcal{Z},\mathcal{P}\cup\{k\}})\}$ then system (4.3) with right-hand side $g(x^*)$ is infeasible. By Farkas' Lemma we know this implies that the optimal solution γ^* to (4.4) is such that $g(x^*)^T \gamma^* = -1$. Thus $g(x)^T \gamma^* \geq 0$ is a valid inequality that is violated by x^*.

At a node in the branch-and-bound algorithm, a branching variable is chosen and the system (4.3) is created. Solving the linear programming problem (4.4) for an optimal solution γ^*, if $g(x^*)^T \gamma^* < 0$, then we have determined a valid inequality that is violated. This procedure can be repeated with every possible choice of branching variable. Adding all of the disjunctive cuts to $C_{\mathcal{Z},\mathcal{P}} x \geq d_{\mathcal{Z},\mathcal{P}}$, a new optimal solution (x^*, y^*) of the relaxation of this node can be calculated. We include these cuts in the formulation of the relaxation of every node in the subtree of this node; however, in order to maintain numerical stability, we use only the original inequalities $H(x,y) \geq h$ or their updated approximations when calculating disjunctive cuts.

Example 4.1.2 For this example only, to simplify the discussion, when we refer the feasible set of the implicit relaxation of a given node, we are referring to the projected set $\{x : (x,y) \in \hat{\mathcal{F}}_{\mathcal{Z},\mathcal{P}}\}$. Returning to Example 4.1.1 from above, we have the constraints (4.1) and, at the root node, the approximating constraint

$$2x_1 + \frac{4}{3} x_2 \leq 1.$$

As before, the optimal solution is $x_1^* = 0.3$ and $x_2^* = 0.3$.

Any point x is in the convex hull of the feasible sets of the implicit relaxations of the children created by branching on x_1 if and only if there exists a $0 \leq \lambda \leq 1$ and points $v, w \in \mathbb{R}^n$ such that v is in the feasible set of the implicit relaxation of the down-branch scaled by λ, w is in the feasible set of the implicit relaxation of the up-branch scaled by $(1-\lambda)$, and $x = v + w$. These conditions can be represented by the system of linear inequalities

$$\begin{array}{rcrcrcl}
\frac{4}{3}v_1 & + & 2v_2 & - & \lambda & \leq & 0, \\
v_1 & & & - & \frac{1}{2}\lambda & \leq & 0, \\
 & & \frac{4}{3}v_2 & - & \lambda & \leq & 0, \\
 & & -v_2 & & & \leq & 0, \\
-\frac{4}{3}v_1 & - & 2v_2 & + & \lambda & \leq & 1 - \frac{4}{3}x_1 - 2x_2, \\
v_1 & & & & & \leq & x_1, \\
-v_1 & + & & & \frac{1}{2}\lambda & \leq & \frac{1}{2} - x_1, \\
 & & -v_2 & + & \frac{3}{4}\lambda & \leq & \frac{3}{4} - x_2, \\
 & & 0 & \leq & v_1 & \leq & 0, \\
 & & x_2 & \leq & v_2 & \leq & x_2, \\
 & & 0 & \leq & \lambda & \leq & 1,
\end{array}$$

which is solvable if and only if x is in the convex hull of the feasible sets of the implicit relaxations of the child nodes.

These constraints are unsolvable for x^* because x^* violates the valid inequality $x_1 + x_2 \leq \frac{1}{2}$ which is produced by taking the linear combination of the four constraints $\frac{4}{3}v_1 + 2v_2 - \lambda \leq 0$, $v_1 \leq 0$, $-v_1 + \frac{1}{2}\lambda \leq \frac{1}{2} - x_1$, and $-v_2 \leq -x_2$ with the multipliers 5, 10/3, 10, and 10, respectively. With the addition of this new constraint, we have a complete description of the set $\{x : (x, y) \in conv(\mathcal{T}_{\mathcal{Z},\mathcal{P}})\}$.

4.1.2 Cover Inequalities

Given a constraint $a^T x + b^T y \geq \alpha$ of (3.3), where $\alpha > 0$ and $a, b \in \mathbb{R}^n$, then at least one binary variable y_j where $j \in \mathcal{V} = \{j \in \mathcal{N} : a_j \neq 0 \text{ or } b_j > 0\}$ must be nonzero for every feasible point of (3.3). Furthermore, it is possible to determine a lower bound on the number of binary variables that must be nonzero in order to *cover*, or satisfy, that constraint. For each $j \in \mathcal{V}$, we calculate the maximum value of $a_j x_j + b_j y_j$ based on the simple bounds on x_j, and set η_j to this maximum value. Sorting the values η_j, $j \in \mathcal{V}$, from largest to smallest, we easily can calculate a lower bound on the number of nonzero binary variables required to make $a^T x + b^T y \geq \alpha$. The constraint $\sum_{j \in \mathcal{V}} y_j \geq \hat{\alpha}$, where $\hat{\alpha}$ is the minimum number of nonzero binary variables needed to cover the constraint, is valid for every node of the implicit branch-and-bound tree. The cover inequalities can be included in the original constraints $H(x, y) \geq h$.

Inside the implicit branch-and-bound tree, if any of the indices of $\mathcal{V}$ are in $\mathcal{P}$ or $\mathcal{Z}$, then new cover inequalities that are valid for the subtree of the current node can be determined based on the sets $\mathcal{Z}$ and $\mathcal{P}$.

4.1.3 Knapsack Cuts

The discussion in this section is a derivation of a similar discourse in Bienstock [3], and it is valid only if the limited diversification constraint $\sum_{j \in \mathcal{N}} -y_j \geq -\kappa$ is present. Given a constraint $a^T x + b^T y \geq \alpha$ of (3.3), where $\alpha > 0$ and $a, b \in \mathbb{R}^n$, we examine the system

$$\begin{aligned} a^T x + b^T y &\geq \alpha, \\ \textstyle\sum_{j \in \mathcal{N}} -y_j &\geq -\kappa, \\ l_j y_j \leq x_j \leq u_j y_j &\quad \forall j \in \mathcal{N}, \\ y \in \mathbb{B}^n. & \end{aligned}$$

It is assumed without loss of generality that for each $j \in \mathcal{N}$, $l_j \leq 0 \leq u_j$ and at least one of these simple bounds is nonzero. For each $j \in \mathcal{N}$ let η_j be the maximum value of $a_j x_j + b_j y_j$, based on the simple bounds on x_j.

Definition 4.1.2 Let $\bar{\mathcal{V}} = \{j \in \mathcal{N} : \eta_j > 0\}$. A subset $\mathcal{V}$ of $\bar{\mathcal{V}}$ is *critical* if $\sum_{j \in \mathcal{J}} \eta_j < \alpha$ for every subset $\mathcal{J}$ of $\mathcal{V}$ such that $|\mathcal{J}| = \kappa$.

Given a set $\mathcal{V}$ that is critical, there must be at least one binary variable y_j, $j \notin \mathcal{V}$, that is nonzero, thus the inequality $\sum_{j \in \mathcal{V}} y_j \leq \kappa - 1$ is valid.

Given the optimal solution (x^*, y^*) determined for the current node, we have derived the following heuristic to determine a critical set $\mathcal{V}$ associated with the constraint $a^T x + b^T y \geq \alpha$ for which the approximating constraint of $\sum_{j \in \mathcal{V}} y_j \leq \kappa - 1$ is violated.

Algorithm 4.1.1

Step 1: For each $j \in \mathcal{N}$, determine the associated value of η_j, and set r_j to u_j if $x_j^* \geq 0$ or l_j if $x_j^* < 0$. Let $\bar{\mathcal{V}}$ be the set $\{j : \eta_j > 0\}$.

Step 2: Let $\mathcal{V}$ be the set of indices of the κ smallest values of η_j for which x_j^* is nonzero; as a tie-breaker, choose the j with the larger value of x_j^*/r_j. If $\sum_{j \in \mathcal{V}} \eta_j \geq \alpha$, STOP. Otherwise, $\mathcal{V}$ is a critical set. Let $\delta = \sum_{j \in \mathcal{V}} x_j^*/r_j - \kappa + 1$. If $\delta > 0$, goto Step 5.

Step 3: Let $k \in \mathcal{V}$ be the index of the smallest η_j, $j \in \mathcal{V}$. If there exists an $i \in \mathcal{N} \backslash \mathcal{V}$ such that

$$\eta_i + \sum_{j \in \mathcal{V} \backslash \{k\}} \eta_j < \alpha$$

and $\delta + x_i^*/r_i > 0$, then $\mathcal{V} \leftarrow \mathcal{V} \cup \{i\}$ is a critical set; goto Step 5.

Step 4: Let $k_1, k_2 \in \mathcal{V}$ be the indices of the two smallest η_j, $j \in \mathcal{V}$. If there exist $i_1, i_2 \in \mathcal{N} \backslash \mathcal{V}$ such that

$$\eta_{i_1} + \eta_{i_2} + \sum_{j \in \mathcal{V} \setminus \{k_1, k_2\}} \eta_j < \alpha$$

and $\delta + x^*_{i_1}/r_{i_1} + x^*_{i_2}/r_{i_2} > 0$, then $\mathcal{V} \leftarrow \mathcal{V} \cup \{i_1, i_2\}$ is a critical set; otherwise STOP.

Step 5: The set $\mathcal{V}$ is a critical set for which $\sum_{j \in \mathcal{V}} x^*_j/r_j > \kappa - 1$. For each index k in $\mathcal{N} \setminus \mathcal{V}$ such that $\eta_j > 0$, if the sum of η_k and the $\kappa - 1$ largest η_j, where $j \in \mathcal{V}$, is less than α, let $\mathcal{V} \leftarrow \mathcal{V} \cup \{k\}$; repeat Step 5 until $\mathcal{V}$ cannot be augmented further.

Similar to the cover inequalities, new knapsack cuts can be determined at a node based on the sets $\mathcal{Z}$ and $\mathcal{P}$ that are valid for the subtree of that node.

4.1.4 Symmetric Dominance Inequalities

Although the symmetric dominance inequalities can be calculated before the implicit branch-and-bound procedure has begun, these inequalities can be derived only in the case that the variance-covariance matrix V is diagonal. Indices $j, k \in \mathcal{N}$ are said to be *symmetric* if the jth and kth columns of H_x are identical and the jth and kth columns of H_y are identical, other than in the linking constraint rows. This type of symmetry is prevalent in the index-tracking data when only a few factors are being mimicked. Removing the symmetry, if possible, reduces the number of nodes created in the implicit branch-and-bound procedure.

For symmetric indices j and k, index j *dominates* index k if it is possible to determine from the input data that if there is an optimal solution with y_k nonzero, there must be one with y_j nonzero. If indices j and k are symmetric, the simple bounds on the continuous variable x_k are tighter than on x_j ($l_j \leq l_k, \quad u_k \leq u_j$), the linear objective term for the continuous variable x_j is larger ($\mu_j \geq \mu_k$), and the

quadratic term for the continuous variable x_j is smaller ($V_{jj} \leq V_{kk}$), then j dominates k and the inequality $y_j \geq y_k$ leaves at least one optimal solution feasible in the implicit branch-and-bound tree. This may not be a valid inequality because there may be a point in $\mathcal{T}_{\emptyset,\emptyset}$ that violates this inequality. We do not add these cuts to the formulation; instead, if a symmetric dominance cut is violated by fixed binary variables at a node, we prune the subtree of that node.

4.2 Upper Bound Heuristics

In the implicit branch-and-bound search tree, many unnecessary nodes may be created if a good upper bound is not known. We have developed two methods for determining a good upper bound.

The first upper bound heuristic involves calculating an upper bound to a mixed-integer programming problem with a linear approximation to the objective function $f(x, y)$. Branch-and-bound algorithms for solving mixed-integer linear programming problems are highly sophisticated and software packages for solving these problems are numerous. A "good" feasible point of the mixed-integer linear programming problem

$$\begin{aligned} \text{minimize} \quad & \bar{f}(x, y) \\ \text{subject to} \quad & H_x x + H_y y \geq h, \\ & x \in \mathbb{R}^n, \quad y \in \mathbb{B}^n, \end{aligned} \tag{4.5}$$

where $\bar{f}(x, y)$ is a linear function and H_x, H_y, and h are as in (3.3), can be found quickly using a mixed-integer linear programming problem solver. In determining a linearization that approximates $f(x, y)$, we see that if short-selling is not allowed, and either the fixed transaction costs are large, or the limited diversification constraint has a small κ, then the continuous variables in a feasible solution to (3.3) will tend to be close to or at their upper bounds. Based on this conjecture, for the first

objective function, $f(x,y) = -\mu^T x + \theta x^T V x$, a possible linearization is $\bar{f}(x,y) = (-\mu + \theta V u)^T x$. For the second objective function, $f(x,y) = -\mu^T x + \theta\sqrt{x^T V x}$, two possible linearizations are $\bar{f}(x,y) = -\mu^T x$ and $\bar{f}(x,y) = (\theta V u - \mu)^T x$.

If the mixed-integer linear programming solver finds no feasible point of (4.5), then (3.3) is infeasible as well. Any feasible point (x^{IP}, y^{IP}) of (4.5) is also feasible for (3.3) and could be good upper bound. An optimal solution, (x^*, y^{IP}), to the leaf node of (3.3) defined by the sets $\mathcal{Z}$ and $\mathcal{P}$ which are described by y^{IP} is probably a better upper bound.

The second upper bound heuristic is based on the implicit branch-and-bound algorithm, and it is performed after the first upper bound heuristic has terminated. In this heuristic, implicit branch-and-bound is performed on a reduced set of the variables until a feasible point of the original problem (3.3) is found. In order to ensure that this heuristic encounters an upper bound, the value of ϕ^{IP} is assumed to be ∞ at the start. The reduced set of variables on which the heuristic is performed is chosen based on the upper bound (x^{IP}, y^{IP}) determined by the first upper bound heuristic and the optimal solution (x^*, y^*) determined for the root node of the implicit branch-and-bound tree. Any index j such that $y_j^{IP} = 0$ and the absolute value of x_j^* is sufficiently near zero is placed in the set $\mathcal{Z}$ and the set $\mathcal{P}$ is set to $\emptyset$. Implicit branch-and-bound is performed on the subtree of the node defined by the sets $\mathcal{Z}$ and $\mathcal{P}$ until an upper bound is encountered. Once this upper bound is found, all of the information other than the upper bound itself is discarded and the implicit branch-and-bound procedure is restarted on the full index set. Restarting the algorithm is important because the branching variables were chosen based on the fact that a good upper bound was not available.

In practice we have found that it is useful not to terminate the second heuristic immediately upon encountering a feasible solution of (3.3). Instead, it may be wise

to continue this depth-first search branching procedure for a fixed number of nodes after the first upper bound is encountered.

4.3 Branching Variable Selection

The method for deciding the branching variable at a node is crucial to the success of the branch-and-bound algorithm. A good choice of branching variable may make it possible to prune both child nodes; a series of poor choices may result in the creation of a large subtree of the current node.

Example 4.3.1 The optimal solution to the relaxation of the root node in the standard branch-and-bound algorithm of the integer programming problem

$$\begin{array}{lrcrl} \text{maximize} & -y_1 & + & 2y_2 & \\ \text{subject to} & -2y_1 & + & 5y_2 & \leq 1.5, \\ & -3y_1 & + & 5y_2 & \leq 1, \\ & y_1, y_2 & \in & \mathbb{B}, & \end{array}$$

is $y_1^* = 0.5,\ y_2^* = 0.5$, with objective function value 0.5. As seen in Figures 4.2 and 4.3, branching on y_1 first forces the standard branch-and-bound algorithm to solve six more nodes, but branching on y_2 first allows the optimal solution to be determined after solving the relaxations of only two more nodes.

Ideally, as in the example above, the choice of branching variable leads to two children that can be pruned. However, since the knowledge of a "good" upper bound is helpful in determining that a node can be pruned, the method for choosing a branching variable in our algorithm is dependent on the value of the current upper bound. During the implicit branch-and-bound procedure, if the current upper bound has

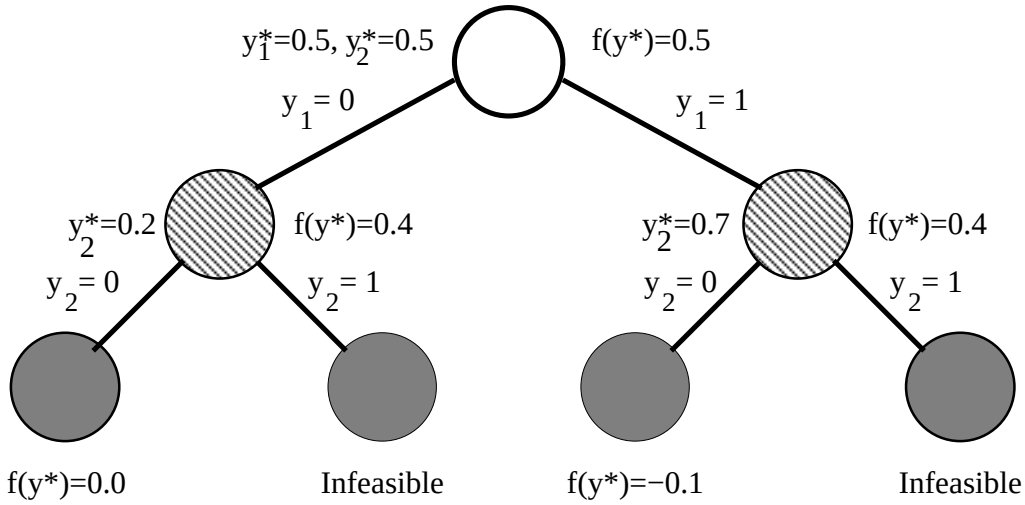

Figure 4.2 Branch on y_1 first

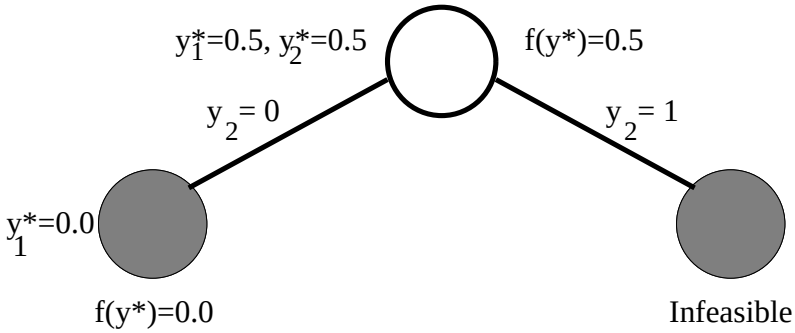

Figure 4.3 Branch on y_2 first

objective function value ϕ^{IP} and the current lower bound is ϕ^L, the *relative optimality gap* is defined to be

$$\frac{|\phi^{IP} - \phi^L|}{|\phi^{IP}| + 1};$$

if no upper bound is known, the relative optimality gap is considered to be ∞. An upper bound is said to be "good" if the relative optimality gap is smaller than some predetermined tolerance.

If no feasible point of the original problem is known, or if the relative optimality gap is larger than the tolerance, finding a good upper bound is the most important task. Given the optimal solution, (x^*, y^*), to a node defined by the sets $\mathcal{Z}$ and $\mathcal{P}$, a good guess for the sets $\bar{\mathcal{Z}}$ and $\bar{\mathcal{P}}$ of the optimal leaf node of the subtree of the current node is that the indices $j \in \mathcal{N} \backslash \mathcal{P}$ which currently have a small value of $|x_j^*|$ will be in $\bar{\mathcal{Z}}$. Thus, the method of branching on the variable x_k, where k minimizes the absolute value of x_j^* over all $j \in \mathcal{N} \backslash (\mathcal{Z} \cup \mathcal{P})$, and choosing the down-branch as the next node to be examined performs well in practice for finding a good upper bound.

If the relative optimality gap is smaller than the tolerance, then pruning the subtrees of one or both of the children of the current node defined by the sets $\mathcal{Z}$ and $\mathcal{P}$ will keep the tree small. Given the optimal solution (x^*, y^*) for this node, the method of branching on the variable x_k, where k maximizes the absolute value of x_j^* over all $j \in \mathcal{N} \backslash (\mathcal{Z} \cup \mathcal{P})$, works well for raising the optimal objective function vale of the down-branch.

Instead of forcing the algorithm to choose the branching variable solely on the values of x^*, we have implemented a version of a branching variable selection heuristic called *strong branching*. In this heuristic, a subset $\mathcal{V}$ of the indices $j \in \mathcal{N} \backslash (\mathcal{Z} \cup \mathcal{P})$ is selected. Each such index defines a tentative down-branch, $DOWN(j)$, and up-branch, $UP(j)$, for this node. The optimal objective function values for all $2|\mathcal{V}|$ of these possible child nodes are calculated, and we let $(\underline{x}^j, \underline{y}^j)$ represent the optimal solution to $DOWN(j)$ and $(\overline{x}^j, \overline{y}^j)$ represent the optimal solution to $UP(j)$. In the best case, there is a $k \in \mathcal{V}$ such that the subtrees of both child nodes created by branching on x_k can be pruned. If there is no such $k \in \mathcal{V}$, then, of the variables for which the subtree of one child node can be pruned, the one that maximizes the optimal objective function value of the other child is chosen as the branching variable. If no choice of variable leads to a child that can be pruned, then x_k is chosen as the

branching variable, where $k \in \mathcal{V}$ maximizes the value of

$$f(\overline{x}^j, \overline{y}^j) + f(\underline{x}^j, \underline{y}^j) \quad \forall j \in \mathcal{V}.$$

Because of the nature of the problem, the optimal objective function values of $UP(j)$ $\forall j \in \mathcal{V}$ tend to be the same as the optimal objective function value of the current node. Thus we have also chosen to implement a one-sided strong branching routine which calculates only the optimal solutions to the down-branches $DOWN(j)$ $\forall j \in \mathcal{V}$.

4.4 Node Selection

In the implicit branch-and-bound algorithm, the relaxation of a node and the relaxation of its up-branch will very often have the same optimal objective function value. Thus, a *depth-first search* method which chooses the up-branch of the current node as the next node to examine closely resembles a *breadth-first search* method that chooses the node whose parent has the objective function value equal to ϕ_L. Unfortunately, the stack of open problems grows much larger when we examine the up-child first than when we examine the down-child first. Furthermore, if the upper bound heuristic has determined as upper bound that has an objective function value close to the optimal objective function value, the method for choosing the next node to evaluate has little or no effect on the number of nodes that are created. This is because our branching variable selection routines do not depend on the order in which the nodes are processed. Thus, if a node cannot be pruned, we work on its down-branch first, and the up-branch is placed on a last-in first-out stack. This minimizes the memory requirements and the amount of time needed to store the necessary information about the node.

Chapter 5

The Quadratic Objective Function

In this chapter we present our algorithm for finding the optimal solution to a subproblem of (3.3) in the implicit branch-and-bound tree where $f(x,y)$ is the objective function introduced by Farrar. We first review the necessary and sufficient conditions for optimality. We then extend the Goldfarb-Idnani [22] algorithm for solving strictly convex quadratic programming problems in order to handle the simple bounds on the variables in an efficient manner. Convergence results are proven and numerical stability is discussed for this algorithm.

5.1 The Quadratic Programming Problem

At a node in the implicit branch-and-bound tree, the optimal solution to the implicit relaxation of the subproblem defined by the sets $\mathcal{Z}$ and $\mathcal{P}$ must be determined; this relaxation is

$$\begin{aligned} \text{minimize} \quad & f(x,y) = -\mu^T x + \theta x^T V x \\ \text{subject to} \quad & Cx \geq d, \\ & 0 \leq y_j \leq 1 \quad \forall j \in \mathcal{N}, \\ & x_j = y_j = 0 \quad \forall j \in \mathcal{Z}, \\ & y_j = 1 \quad \forall j \in \mathcal{P}, \\ & x \in \mathbb{R}^n, \; y \in \mathbb{R}^n, \end{aligned} \tag{5.1}$$

where $\mathcal{N} = \{1, \ldots, n\}$, θ is a strictly positive scalar, V is a symmetric positive definite matrix, and the linear constraints $Cx \geq d$ include the simple bounds $l_j \leq x_j \leq u_j$ for all $j \in \mathcal{N}$, the constraints of $H_x x + H_y y \geq h$ for which no binary variable has a nonzero coefficient, and other valid inequalities derived in Chapter 4.

To simplify the notation in this chapter, (5.1) is formulated to include the equality constraints explicitly, and the continuous variables that are fixed to zero are removed. Since we predetermine the values of the binary variables at a node in the implicit branch-and-bound tree, the binary variables are all removed. The optimization subproblem is

$$\begin{aligned} \text{minimize} \quad & \hat{f}(x) = -\mu^T x + \theta x^T V x \\ \text{subject to} \quad & Hx = h, \\ & Cx \geq d, \\ & l \leq x \leq u, \\ & x \in \mathbb{R}^n, \end{aligned} \tag{5.2}$$

where n is the number of remaining variables, $\mathcal{N} = \{1, \ldots, n\}$, m_1 is the number of equality constraints, $\mathcal{M}_1 = \{1, \ldots, m_1\}$, m_2 is the number of inequality constraints (excluding the simple bounds on the variables), $\mathcal{M}_2 = \{1, \ldots, m_2\}$, H is an m_1 by n matrix, $h \in \mathbb{R}^{m_1}$, C is an m_2 by n matrix, and $d \in \mathbb{R}^{m_2}$. Without loss of generality, we can assume the matrix H has full row rank, and no inequality constraint or remaining simple bound can be represented as a linear combination of the rows of H. The feasible set of this problem is denoted by $\hat{\mathcal{F}}$.

As stated in the second chapter, the matrix V is positive definite. Since $\hat{f}(x)$ is twice continuously differentiable and the Hessian matrix, $\nabla^2 \hat{f}(x) \equiv 2\theta V$, is positive definite for all $x \in \hat{\mathcal{F}}$, $\hat{f}(x)$ is a strictly convex function over $\hat{\mathcal{F}}$. If $\hat{\mathcal{F}}$ is not empty, this implies that (5.2) has a unique optimal solution. It is interesting to note that the objective function $f(x, y)$ may not be strictly convex over the feasible set of (5.1), and thus $f(x, y)$ may not have a unique minimizer over the constraints of (5.1).

5.2 The Optimality Conditions

Before reviewing the necessary and sufficient conditions for optimality, we must first define the terms *active* and *nullspace*.

Definition 5.2.1 Given a point $\bar{x}$, the inequality $a^T x \geq \alpha$ is *active* if $a^T \bar{x} = \alpha$.

An equality constraint $a^T x = \alpha$ is considered active for any point $\bar{x}$ such that $a^T \bar{x} = \alpha$.

Definition 5.2.2 The *nullspace* of a matrix W is the vector space comprised of all vectors v such that $Wv = \mathbf{0}$, where $\mathbf{0}$ is the vector of zeros of the appropriate dimension. The matrix Z is a *nullspace matrix* of W if for every vector v in the nullspace of W, there exists a vector w such that $Zw = v$.

The *Lagrangian function* of (5.2) is

$$\mathcal{L}(x, \pi, \tau, \gamma, \rho) = \hat{f}(x) - \pi^T(Hx - h) - \tau^T(Cx - d) - \gamma^T(x - l) - \rho^T(u - x),$$

where $\pi \in \mathbb{R}^{m_1}$, $\tau \in \mathbb{R}^{m_2}$ and $\gamma,\ \rho \in \mathbb{R}^n$. The four vectors π, τ, γ, and ρ are called *Lagrange multipliers* or *dual variables*. Differentiating this function with respect to x results in the function

$$\nabla_x \mathcal{L}(x, \pi, \tau, \gamma, \rho) = -\mu + 2\theta V - H^T\pi - C^T\tau - \gamma + \rho.$$

The following theorem is adapted from pages 430 and 441 of Nash and Sofer [41]; the inequality $\tau \geq \mathbf{0}$ means that every component of the vector τ is nonnegative.

Theorem 5.2.1 In problem (5.2), if x^* is a minimizer of $\hat{f}(x)$ over the feasible set $\hat{\mathcal{F}}$, then there exist vectors π^*, τ^*, γ^*, and ρ^* of Lagrange multipliers such that

- $\nabla_x \mathcal{L}(x^*, \pi^*, \tau^*, \gamma^*, \rho^*) = \mathbf{0}$,
- $\tau^* \geq \mathbf{0}$, $\gamma^* \geq \mathbf{0}$, and $\rho^* \geq \mathbf{0}$,
- $\tau^{*T}(Cx^* - d) = 0$, $\gamma^{*T}(x^* - l) = 0$, and $\rho^{*T}(u - x^*) = 0$, and
- $Z^T \nabla^2 \hat{f}(x^*) Z$ is positive semi-definite,

where Z is a nullspace matrix for the active constraints at x^*.

Since V is a positive definite matrix, the condition that $Z^T \nabla^2 \hat{f}(x^*) Z$ be positive semi-definite is satisfied by any matrix Z with n rows.

Because (5.2) has a convex objective function, the above necessary conditions for optimality are also sufficient. The following theorem and proof are adapted from Theorem 19 on page 90 of Fiacco and McCormick [16].

Theorem 5.2.2 The sufficient conditions for $x^* \in \hat{\mathcal{F}}$ to be an optimal solution of the convex programming problem (5.2) are that there exist vectors π^*, τ^*, γ^*, and ρ^* such that

- $\nabla_x \mathcal{L}(x^*, \pi^*, \tau^*, \gamma^*, \rho^*) = \mathbf{0}$,
- $\tau^* \geq \mathbf{0}$, $\gamma^* \geq \mathbf{0}$, and $\rho^* \geq \mathbf{0}$, and
- $\tau^{*T}(Cx^* - d) = 0$, $\gamma^{*T}(x^* - l) = 0$, and $\rho^{*T}(u - x^*) = 0$.

Proof: Let $\bar{x}$ be any other point satisfying the constraints of (5.2). Since $\hat{f}(x)$ is convex, $\hat{f}(\bar{x}) \geq \hat{f}(x^*) + (\bar{x} - x^*)^T \nabla \hat{f}(x^*)$ [16]. Then

$$\begin{aligned}
\hat{f}(\bar{x}) &\geq \hat{f}(\bar{x}) - \pi^{*T}(H\bar{x} - h) - \tau^{*T}(C\bar{x} - d) - \gamma^{*T}(\bar{x} - l) - \rho^{*T}(u - \bar{x}) \\
&\geq \hat{f}(x^*) + (\bar{x} - x^*)^T \nabla \hat{f}(x^*) - (\bar{x} - x^*)^T H^T \pi^* - (\bar{x} - x^*)^T C^T \tau^* \\
&\quad -(\bar{x} - x^*)^T \gamma^* + (\bar{x} - x^*)^T \rho^* \\
&= \hat{f}(x^*) + (\bar{x} - x^*)^T \nabla_x \mathcal{L}(x^*, \pi^*, \tau^*, \gamma^*, \rho^*) \\
&= \hat{f}(x^*). \square
\end{aligned}$$

The necessary and sufficient conditions are typically divided into three categories:

Primal feasibility: $Hx = h,\ Cx \geq d,\ l \leq x \leq u$

Dual feasibility: $-\mu + 2\theta Vx - H^T\pi - C^T\tau - \gamma + \rho = \mathbf{0},\ \tau \geq \mathbf{0},\ \gamma, \rho \geq \mathbf{0}$

Complementary slackness: $\tau^T(Cx - d) = 0,\ \gamma^T(x - l) = 0,\ \rho^T(u - x) = 0.$

The method we have chosen to solve (5.2) is a dual method; at every iteration, the dual feasibility conditions and the complementary slackness conditions are satisfied, along with the primal equality constraints. We have chosen this method because it allows for a quick "restart" after a new constraint is added. The optimal solution to the parent problem or to the current problem before the addition of a new valid inequality satisfies dual feasibility, complementary slackness and the primal equality conditions, so no procedure to regain feasibility is necessary. This is an active set method that may force many iterations to regain optimality; fortunately, this has not been the case in practice.

5.3 The Extended Goldfarb-Idnani Algorithm

The Goldfarb-Idnani[22] algorithm is a dual algorithm for solving quadratic programming problems; it calculates the optimal solution to the problem

$$\begin{array}{lrcl} \text{minimize} & -\mu^T x & + & \theta x^T V x \\ \text{subject to} & Cx & \geq & d \end{array}$$

by solving a series of subproblems which have the same objective function but are subject to only a subset of the inequality constrains. The dual feasibility and complementary slackness conditions are satisfied at every iteration.

Unfortunately, this method forces the simple bounds to be included in the constraint matrix as the $2n$ constraints $-Ix \geq -u$ and $Ix \geq l$, where I is the n by n

identity matrix. Furthermore, as part of the implicit branch-and-bound procedure, the storage requirements, which are on the order of n^2 for Goldfarb-Idnani, are unreasonable. Our algorithm attempts to reduce the storage requirements to the order of $(m_1 + m_2)^2$.

In order to describe our extensions to the Goldfarb-Idnani algorithm, it is necessary to introduce some notation. Given a matrix C whose rows are indexed on $\mathcal{M}_2$ and columns are indexed on $\mathcal{N}$, if $\mathcal{A} \subseteq \mathcal{M}_2$ and $\mathcal{J} \subseteq \mathcal{N}$, the matrix $C_{\mathcal{A}\mathcal{J}}$ is comprised of the rows of C that correspond to the indices in $\mathcal{A}$ and the columns of C that correspond to the indices in $\mathcal{J}$. The entire index set is represented as ":"; for example, $C_{\mathcal{M}_2\mathcal{J}}$ is written as $C_{:\mathcal{J}}$. Moreover, $C^T_{\mathcal{A}\mathcal{J}} \equiv (C_{\mathcal{A}\mathcal{J}})^T$.

Our algorithm, like the Goldfarb-Idnani algorithm, is based on the quadratic programming subproblem $\mathcal{D}(\mathcal{A},\mathcal{L},\mathcal{U})$ which is

$$\begin{aligned}
\text{minimize} \quad \hat{f}(x) &= -\mu^T x + \theta x^T V x \\
\text{subject to} \quad Hx &= h, \\
C_{\mathcal{A}:}x &\geq d_{\mathcal{A}}, \\
x_{\mathcal{L}} &\geq l_{\mathcal{L}}, \\
-x_{\mathcal{U}} &\geq -u_{\mathcal{U}}, \\
x &\in \mathbb{R}^n,
\end{aligned}$$

where $\mathcal{A} \subseteq \mathcal{M}_2$ and $\mathcal{L},\mathcal{U} \subseteq \mathcal{N}$. If this subproblem of (5.2) is not infeasible, there exists a feasible solution $\bar{x}$ and associated Lagrange multipliers $\bar{\pi}$, $\bar{\tau}$, $\bar{\gamma}$, and $\bar{\rho}$ such that the dual feasibility and complementary slackness conditions are satisfied. Given working sets $\bar{\mathcal{A}} \subseteq \mathcal{A}$, $\bar{\mathcal{L}} \subseteq \mathcal{L}$, and $\bar{\mathcal{U}} \subseteq \mathcal{U}$ such that $C_{\bar{\mathcal{A}}:}\bar{x} = d_{\bar{\mathcal{A}}}$, $\bar{\tau}_{\mathcal{M}_2\setminus\bar{\mathcal{A}}} = \mathbf{0}$, $\bar{x}_{\bar{\mathcal{L}}} = l_{\bar{\mathcal{L}}}$, $\bar{\gamma}_{\mathcal{N}\setminus\bar{\mathcal{L}}} = \mathbf{0}$, $\bar{x}_{\bar{\mathcal{U}}} = u_{\bar{\mathcal{U}}}$, $\bar{\rho}_{\mathcal{N}\setminus\bar{\mathcal{U}}} = \mathbf{0}$, and the rows of H, $C_{\bar{\mathcal{A}}:}$, $I_{\bar{\mathcal{L}}:}$, and $-I_{\bar{\mathcal{U}}:}$ are linearly independent, the ordered list $(\bar{x},\bar{\mathcal{A}},\bar{\mathcal{L}},\bar{\mathcal{U}})$ is called an *S-list*, and $\bar{x}$ is the optimal solution to the subproblem $\mathcal{D}(\bar{\mathcal{A}},\bar{\mathcal{L}},\bar{\mathcal{U}})$. We prove the following theorem in the next section.

Theorem 5.3.1 If the subproblem $\mathcal{D}(\mathcal{A},\mathcal{L},\mathcal{U})$ is not infeasible, then any optimal solution $\bar{x}$ to $\mathcal{D}(\mathcal{A},\mathcal{L},\mathcal{U})$ has an associated S-list.

At each step in the algorithm, we start with an S-list $(x^*,\mathcal{A},\mathcal{L},\mathcal{U})$. If x^* is feasible for (5.2), then x^* is the optimal solution to (5.2). Otherwise, there is some constraint or simple bound that is violated. Selecting one such violated constraint, a new S-list $(\bar{x},\bar{\mathcal{A}},\bar{\mathcal{L}},\bar{\mathcal{U}})$ is determined such that the new objective function value, $\hat{f}(\bar{x})$, is strictly greater than $\hat{f}(x^*)$. Furthermore, each working set $\bar{\mathcal{A}}$, $\bar{\mathcal{L}}$, and $\bar{\mathcal{U}}$ of the new S-list is a subset of the corresponding working set of the previous S-list, except that the selected constraint violated by x^* is satisfied at equality by $\bar{x}$ and its index is a member of the appropriate working set. Since the objective function value increases at each iteration and there are finitely many subsets $\mathcal{A}$ of $\mathcal{M}_2$ and $\mathcal{L}$ and $\mathcal{U}$ of $\mathcal{N}$, the algorithm will terminate with the optimal solution after a finite number of iterations.

This is similar to the algorithm discussed in [22]. Our algorithm takes advantage of the structure of simple bounds by observing that setting $\mathcal{B} \equiv \mathcal{N}\backslash(\mathcal{L}\cup\mathcal{U})$, the rows of H, $C_{\mathcal{A}:}$, $I_{\mathcal{L}:}$, and $-I_{\mathcal{U}:}$ are linearly independent if and only if the rows of $H_{:\mathcal{B}}$ and $C_{\mathcal{A}\mathcal{B}}$ are linearly independent.

5.3.1 The Algorithm

We first prove the theorem introduced in the last section.

Theorem 5.3.1 If the subproblem $\mathcal{D}(\mathcal{A},\mathcal{L},\mathcal{U})$ is not infeasible, then any optimal solution $\bar{x}$ to $\mathcal{D}(\mathcal{A},\mathcal{L},\mathcal{U})$ has an associated S-list.

Proof: Since $\bar{x}$ is the optimal solution to the quadratic programming problem $\mathcal{D}(\mathcal{A},\mathcal{L},\mathcal{U})$, there are associated Lagrange multipliers $\bar{\pi}$, $\bar{\tau}$, $\bar{\gamma}$, and $\bar{\rho}$ such that the dual feasibility and complementarity slackness conditions are satisfied. Let $\bar{\mathcal{L}} = \{j \in \mathcal{L} : \bar{x}_j = l_j,\ \bar{\gamma}_j > 0\}$, $\bar{\mathcal{U}} = \{j \in \mathcal{U} : \bar{x}_j =$

$u_j,\ \bar{\rho}_j > 0\}$, and $\bar{\mathcal{A}} = \{i \in \mathcal{A} : C_i\bar{x} = d_i,\ \bar{\tau}_i > 0\}$. If the rows of H, $C_{\bar{\mathcal{A}}:}$, $I_{\bar{\mathcal{L}}:}$, and $-I_{\bar{\mathcal{U}}:}$ are linearly independent, then $(\bar{x}, \bar{\mathcal{A}}, \bar{\mathcal{L}}, \bar{\mathcal{U}})$ is an S-list. Otherwise, we need to construct subsets of $\bar{\mathcal{A}}$, $\bar{\mathcal{L}}$ and $\bar{\mathcal{U}}$ that, along with $\bar{x}$, are an S-list such that $\bar{x}$ is feasible for this subproblem.

Case 1: There exists an $i \in \bar{\mathcal{A}}$ such that $C_{i:}$ is a linear combination of the rows of H, $C_{\bar{\mathcal{A}}\setminus\{i\}:}$, $I_{\bar{\mathcal{L}}:}$ and $-I_{\bar{\mathcal{U}}:}$. This implies that there exist $\Delta\pi \in \mathbb{R}^{m_1}$, $\Delta\tau_{\bar{\mathcal{A}}\setminus\{i\}} \in \mathbb{R}^{|\bar{\mathcal{A}}|-1}$, $\Delta\gamma_{\bar{\mathcal{L}}} \in \mathbb{R}^{|\bar{\mathcal{L}}|}$, and $\Delta\rho_{\bar{\mathcal{U}}} \in \mathbb{R}^{|\bar{\mathcal{U}}|}$, such that

$$H^T\Delta\pi + C^T_{\bar{\mathcal{A}}\setminus\{i\}:}\Delta\tau_{\bar{\mathcal{A}}\setminus\{i\}} + I^T_{\bar{\mathcal{L}}:}\Delta\gamma_{\bar{\mathcal{L}}} - I^T_{\bar{\mathcal{U}}:}\Delta\rho_{\bar{\mathcal{U}}} = -\bar{\tau}_i C^T_{i:}.$$

Let $\Delta\tau_i = \bar{\tau}_i$ and $\varpi = \min\{\frac{\bar{\tau}_k}{\Delta\tau_k} \quad (k \in \bar{\mathcal{A}},\ \Delta\tau_k > 0),$

$\frac{\bar{\gamma}_k}{\Delta\gamma_k} \quad (k \in \bar{\mathcal{L}},\ \Delta\gamma_k > 0),$

$\frac{\bar{\rho}_k}{\Delta\rho_k} \quad (k \in \bar{\mathcal{U}},\ \Delta\rho_k > 0)\}.$

Clearly $\varpi > 0$ and there is at least one dual variable $\bar{\tau}_k,\ k \in \bar{\mathcal{A}}$, such that $\bar{\tau}_k - \varpi\Delta\tau_k = 0$, or $\bar{\gamma}_k,\ k \in \bar{\mathcal{L}}$, such that $\bar{\gamma}_k - \varpi\Delta\gamma_k = 0$, or $\bar{\rho}_k,\ k \in \bar{\mathcal{U}}$, such that $\bar{\rho}_k - \varpi\Delta\rho_k = 0$. Since

$$\begin{aligned}
&2\theta V\bar{x} - \mu - H^T[\bar{\pi} - \varpi\Delta\pi] - C^T_{\bar{\mathcal{A}}:}[\bar{\tau}_{\bar{\mathcal{A}}} - \varpi\Delta\tau_{\bar{\mathcal{A}}}] - I^T_{\bar{\mathcal{L}}:}[\bar{\gamma}_{\bar{\mathcal{L}}} - \varpi\Delta\gamma_{\bar{\mathcal{L}}}] + \\
&\qquad I^T_{\bar{\mathcal{U}}:}[\bar{\rho}_{\bar{\mathcal{U}}} - \varpi\Delta\rho_{\bar{\mathcal{U}}}] \\
&= \quad 2\theta V\bar{x} - \mu - H^T\bar{\pi} - C^T_{\bar{\mathcal{A}}:}\bar{\tau}_{\bar{\mathcal{A}}} - I^T_{\bar{\mathcal{L}}:}\bar{\gamma} + I^T_{\bar{\mathcal{U}}:}\bar{\rho}_{\bar{\mathcal{U}}} + \\
&\qquad \varpi[H^T\Delta\pi + C^T_{\bar{\mathcal{A}}:}\Delta\tau_{\bar{\mathcal{A}}} + I^T_{\bar{\mathcal{L}}:}\Delta\gamma_{\bar{\mathcal{L}}} + (-I_{\bar{\mathcal{U}}:})^T\Delta\rho_{\bar{\mathcal{U}}}] \\
&= \quad \mathbf{0} + \varpi[\Delta\tau_i C^T_{i:} - \bar{\tau}_i C^T_{i:}] = \mathbf{0},
\end{aligned}$$

dual feasibility is maintained after updating $\bar{\pi} \leftarrow \bar{\pi} - \varpi\Delta\pi$, $\bar{\tau}_{\bar{\mathcal{A}}} \leftarrow \bar{\tau}_{\bar{\mathcal{A}}} - \varpi\Delta\tau_{\bar{\mathcal{A}}}$, $\bar{\gamma}_{\bar{\mathcal{L}}} \leftarrow \bar{\gamma}_{\bar{\mathcal{L}}} - \varpi\Delta\gamma_{\bar{\mathcal{L}}}$, and $\bar{\rho}_{\bar{\mathcal{U}}} \leftarrow \bar{\rho}_{\bar{\mathcal{U}}} - \varpi\Delta\rho_{\bar{\mathcal{U}}}$. Dropping the indices of all constraints for which the corresponding dual variable is now zero from the appropriate sets $\bar{\mathcal{A}}$, $\bar{\mathcal{L}}$, and $\bar{\mathcal{U}}$, we are left with $\bar{x}$, $\bar{\pi}$, $\bar{\tau}_{\bar{\mathcal{A}}}$, $\bar{\gamma}_{\bar{\mathcal{L}}}$, and $\bar{\rho}_{\bar{\mathcal{U}}}$ that satisfy primal feasibility, dual feasibility, and complementary slackness.

Case 2: There is no such $i \in C_{\bar{\mathcal{A}}:}$. Then there must be a $j \in \bar{\mathcal{L}}$ such that $I_{j:}^T$ is a linear combination of the rows of H, $C_{\bar{\mathcal{A}}:}$, $I_{\bar{\mathcal{L}}\setminus\{j\}:}$ and $-I_{\bar{\mathcal{U}}:}$ or a $j \in \bar{\mathcal{U}}$ such that $-I_{j:}^T$ is a linear combination of the rows of H, $C_{\bar{\mathcal{A}}:}$, $I_{\bar{\mathcal{L}}:}$ and $-I_{\bar{\mathcal{U}}\setminus\{j\}:}$. A similar process as in Case 1 can be done to reduce the size of at least one of the sets $\bar{\mathcal{A}}$, $\bar{\mathcal{L}}$ or $\bar{\mathcal{U}}$.

In either case, some constraint is dropped from a working set and the process can be repeated until the rows of H, $C_{\bar{\mathcal{A}}:}$, $I_{\bar{\mathcal{L}}:}$, and $-I_{\bar{\mathcal{U}}:}$ are linearly independent, resulting in the S-list $(\bar{x}, \bar{\mathcal{A}}, \bar{\mathcal{L}}, \bar{\mathcal{U}})$.□

To start the dual algorithm for solving (5.2), an S-list for (5.2) must be known. At the root node in the implicit branch-and-bound, an S-list is $(x^*, \emptyset, \emptyset, \emptyset)$ where x^* is the minimizer of $\hat{f}(x)$ over the set $\{x : Hx = h\}$. For other nodes, the optimal S-list of the parent of that current node is a valid S-list. Moreover, the optimal S-list to the same problem before the addition of a violated cutting plane is a valid S-list.

The process of determining the optimal solution to a subproblem of (5.2) is called an *iteration*. At each iteration, the algorithm starts with an S-list $(x^*, \mathcal{A}, \mathcal{L}, \mathcal{U})$ and its corresponding Lagrange multipliers π^*, τ^*, γ^*, and ρ^*. If x^* is primal feasible for (5.2), then x^* is the optimal solution to (5.2). Otherwise, a violated constraint is chosen; if the violated constraint is a lower bound $x_k \geq l_k$ where $k \in \mathcal{N}\setminus\mathcal{L}$, then we let $a = I_{k:}^T$ and $\alpha = l_k$; if it is an upper bound $-x_k \geq -u_k$ where $k \in \mathcal{N}\setminus\mathcal{U}$, then we let $a = -I_{k:}^T$ and $\alpha = -u_k$; else it is a constraint $C_{k:}x \geq d_k$ where $k \in \mathcal{M}_2\setminus\mathcal{A}$, then we let $a = C_{k:}^T$ and $\alpha = d_k$.

The algorithm then determines a primal step direction $\Delta x_{\mathcal{B}}$ towards the hyperplane $a_{\mathcal{B}}^T x_{\mathcal{B}} = \alpha - a_{\mathcal{L}}^T x_{\mathcal{L}}^* - a_{\mathcal{U}}^T x_{\mathcal{U}}^*$ in the nullspace of the matrix

$$W = \begin{bmatrix} H_{:\mathcal{B}} \\ C_{\mathcal{A}\mathcal{B}} \end{bmatrix},$$

where $\mathcal{B} \equiv \mathcal{N} \backslash (\mathcal{L} \cup \mathcal{U})$. It also determines corresponding dual step directions that keep the complementary slackness conditions and dual equality constraint

$$-\mu + 2\theta V x - H^T \pi - C^T \tau - \gamma + \rho = \mathbf{0}$$

satisfied. Since $\tau^*_{\mathcal{M}_2 \backslash \mathcal{A}} = \mathbf{0}$, $\gamma^*_{\mathcal{N} \backslash \mathcal{L}} = \mathbf{0}$, $\rho^*_{\mathcal{N} \backslash \mathcal{U}} = \mathbf{0}$, $Hx^* = h$, $C_{\mathcal{A}:} x^* = d_{\mathcal{A}}$, $x^*_{\mathcal{L}} = l_{\mathcal{L}}$, and $x^*_{\mathcal{U}} = u_{\mathcal{U}}$, this is equivalent to determining step directions $\Delta x_{\mathcal{B}}$, $\Delta \pi$, $\Delta \tau_{\mathcal{A}}$, $\Delta \gamma_{\mathcal{L}}$ and $\Delta \rho_{\mathcal{U}}$ such that

$$\begin{aligned} \Delta x_{\mathcal{L}} = \mathbf{0},\ \Delta x_{\mathcal{U}} = \mathbf{0},\ H_{:\mathcal{B}} \Delta x_{\mathcal{B}} = \mathbf{0},\ C_{\mathcal{A}\mathcal{B}} \Delta x_{\mathcal{B}} &= \mathbf{0}, \text{ and} \\ 2\theta V \Delta x + H^T \Delta \pi + C^T \Delta \tau + \Delta \gamma - \Delta \rho &= a. \end{aligned} \tag{5.3}$$

Simplifying (5.3), we see that it is equivalent to

$$\begin{aligned} 2\theta V_{\mathcal{B}\mathcal{B}} \Delta x_{\mathcal{B}} + H^T_{:\mathcal{B}} \Delta \pi + C^T_{\mathcal{A}\mathcal{B}} \Delta \tau_{\mathcal{A}} &= a_{\mathcal{B}}, \\ 2\theta V_{\mathcal{L}\mathcal{B}} \Delta x_{\mathcal{B}} + H^T_{:\mathcal{L}} \Delta \pi + C^T_{\mathcal{A}\mathcal{L}} \Delta \tau_{\mathcal{A}} + \Delta \gamma_{\mathcal{L}} &= a_{\mathcal{L}}, \text{ and} \\ 2\theta V_{\mathcal{U}\mathcal{B}} \Delta x_{\mathcal{B}} + H^T_{:\mathcal{U}} \Delta \pi + C^T_{\mathcal{A}\mathcal{U}} \Delta \tau_{\mathcal{A}} - \Delta \rho_{\mathcal{U}} &= a_{\mathcal{U}}. \end{aligned}$$

Thus the algorithm determines $\Delta x_{\mathcal{B}}$, $\Delta \pi$, and $\Delta \tau_{\mathcal{A}}$ that satisfy the system

$$\begin{bmatrix} 2\theta V_{\mathcal{B}\mathcal{B}} & W^T \\ W & \mathbf{0} \end{bmatrix} \begin{bmatrix} \Delta x_{\mathcal{B}} \\ \Delta \pi \\ \Delta \tau_{\mathcal{A}} \end{bmatrix} = \begin{bmatrix} a_{\mathcal{B}} \\ \mathbf{0} \end{bmatrix}, \tag{5.4}$$

where $\mathbf{0}$ is a matrix of zeros of the appropriate dimensions, and then it sets $\Delta \gamma_{\mathcal{L}} = a_{\mathcal{L}} - 2\theta V_{\mathcal{L}\mathcal{B}} \Delta x_{\mathcal{B}} - H^T_{:\mathcal{L}} \Delta \pi - C^T_{\mathcal{A}\mathcal{L}} \Delta \tau_{\mathcal{A}}$ and $\Delta \rho_{\mathcal{U}} = -a_{\mathcal{U}} + 2\theta V_{\mathcal{U}\mathcal{B}} \Delta x_{\mathcal{B}} + H^T_{:\mathcal{U}} \Delta \pi + C^T_{\mathcal{A}\mathcal{U}} \Delta \tau_{\mathcal{A}}$.

Since the matrix $V_{\mathcal{B}\mathcal{B}}$ is a principal submatrix of V, both it and its inverse are positive definite. Performing Gaussian elimination on (5.4), we see that

$$\begin{bmatrix} \Delta \pi \\ \Delta \tau_{\mathcal{A}} \end{bmatrix} = (W V^{-1}_{\mathcal{B}\mathcal{B}} W^T)^{-1} W V^{-1}_{\mathcal{B}\mathcal{B}} a_{\mathcal{B}} \tag{5.5}$$

and

$$\begin{aligned}\Delta x_B &= \frac{1}{2\theta}V_{\mathcal{BB}}^{-1}\left(a_{\mathcal{B}} - H^T\Delta\pi - C_{\mathcal{AB}}^T\Delta\tau_{\mathcal{A}}\right)\\ &= \frac{1}{2\theta}V_{\mathcal{BB}}^{-1}\left(I - W^T(WV_{\mathcal{BB}}^{-1}W^T)^{-1}WV_{\mathcal{BB}}^{-1}\right)a_{\mathcal{B}}\\ &= \frac{1}{2\theta}Za_{\mathcal{B}};\end{aligned}$$

the matrix

$$Z = V_{\mathcal{BB}}^{-1} - V_{\mathcal{BB}}^{-1}W^T(WV_{\mathcal{BB}}^{-1}W^T)^{-1}WV_{\mathcal{BB}}^{-1}$$

is a symmetric positive semi-definite nullspace matrix for W. Since $a_{\mathcal{B}}$ is in the range of W^T if and only if $Za_{\mathcal{B}} = \mathbf{0}$, $\Delta x_{\mathcal{B}} = \mathbf{0}$ implies that the matrix

$$\bar{W} = \begin{bmatrix} W \\ a_{\mathcal{B}}^T \end{bmatrix}$$

does not have full row rank. If $\Delta x_{\mathcal{B}} = \mathbf{0}$ and a step cannot be taken in the dual space such that a constraint can be dropped from $\mathcal{A}$, $\mathcal{L}$ or $\mathcal{U}$, then (5.2) is infeasible, as we show in Theorem 5.3.2.

If $\|\Delta x_{\mathcal{B}}\|_2 > 0$, then a step size such that the dual variables remain nonnegative is determined. If the step size is shorter than $(\alpha - a^Tx^*)/a_{\mathcal{B}}^T\Delta x_{\mathcal{B}}$, then the step is stopped short at a point that does not satisfy complementary slackness. However, this point does satisfies primal feasibility of the constraints associated with $\mathcal{A}$, $\mathcal{L}$, and $\mathcal{U}$ and dual feasibility. The constraint whose dual variable went to zero can be dropped from the appropriate working set and the nullspace of the new matrix

$$\bar{W} = \begin{bmatrix} H_{:\bar{\mathcal{B}}} \\ C_{\bar{\mathcal{A}}\bar{\mathcal{B}}} \end{bmatrix}$$

can be determined. The process of determining primal and dual steps and dropping active constraints from the working sets is repeated until a full primal step is taken and the constraint $a^Tx \geq \alpha$ is satisfied at equality. These inner "iterations",

where complementary slackness for the constraint $a^T x \geq \alpha$ is not satisfied, are called *subiterations.*

We now present a formal description of our dual algorithm for solving quadratic programming problems.

Algorithm 5.3.1

Step 1: **Initial Conditions:** Assume that some S-list $(x^*, \mathcal{A}, \mathcal{L}, \mathcal{U})$ is given, along with associated Lagrange multipliers π^*, $\tau^*_{\mathcal{A}}$, $\gamma^*_{\mathcal{L}}$, and $\rho^*_{\mathcal{U}}$.

Step 2: **Begin an iteration:** Set the Lagrange multiplier, ν, for the violated constraint to be 0. Choose $k \in \mathcal{N} \backslash \mathcal{L}$ such that $x^*_k < l_k$ and set $a^T x \geq \alpha$ to that violated constraint and goto Step 3. Otherwise, choose $k \in \mathcal{N} \backslash \mathcal{U}$ such that $x^*_k > u_k$ and set $a^T x \geq \alpha$ to that violated constraint and goto Step 3. Otherwise, choose $k \in \mathcal{M}_2 \backslash \mathcal{A}$ such that $C_{k:} x^* < d_k$ and set $a^T x \geq \alpha$ to that violated constraint and goto Step 3. Since we have determined that there are no violated constraints, x^* is optimal for (5.2); STOP.

Step 3: **Begin a subiteration:**

Step 3a: Let $\mathcal{B} \equiv \mathcal{N} \backslash (\mathcal{L} \cup \mathcal{U})$. Calculate the dual step directions

$$\begin{bmatrix} \Delta\pi \\ \Delta\tau_{\mathcal{A}} \end{bmatrix} = (W V^{-1}_{\mathcal{B}\mathcal{B}} W^T)^{-1} W V^{-1}_{\mathcal{B}\mathcal{B}} a_{\mathcal{B}}, \text{ where } W = \begin{bmatrix} H_{:\mathcal{B}} \\ C_{\mathcal{A}\mathcal{B}} \end{bmatrix},$$

and the primal step direction

$$\Delta x_{\mathcal{B}} = \frac{1}{2\theta} V^{-1}_{\mathcal{B}\mathcal{B}} \left(a_{\mathcal{B}} - H^T \Delta\pi - C^T_{\mathcal{A}\mathcal{B}} \Delta\tau_{\mathcal{A}} \right).$$

Step 3b: If $\mathcal{L}$ is empty, or if the dual step $\Delta\gamma_{\mathcal{L}} = a_{\mathcal{L}} - 2\theta V_{\mathcal{L}\mathcal{B}} \Delta x_{\mathcal{B}} - H^T_{:\mathcal{L}} \Delta\pi - C^T_{\mathcal{A}\mathcal{L}} \Delta\tau_{\mathcal{A}} \leq \mathbf{0}$, let $\varpi_1 = \infty$. Else let $\varpi_1 = \min\{\gamma_j / \Delta\gamma_j :$

$j \in \mathcal{L}, \Delta\gamma_k > 0\}$ and p_1 be the element of $\mathcal{L}$ that achieves this minimum.

Step 3c: If $\mathcal{U}$ is empty, or if the dual step $\Delta\rho_{\mathcal{U}} = -a_{\mathcal{U}} + 2\theta V_{\mathcal{U}\mathcal{B}}\Delta x_{\mathcal{B}} + H_{:\mathcal{U}}^T\Delta\pi + C_{\mathcal{A}\mathcal{U}}^T\Delta\tau_{\mathcal{A}} \leq \mathbf{0}$, let $\varpi_2 = \infty$. Else let $\varpi_2 = \min\{\rho_j/\Delta\rho_j : j \in \mathcal{U}, \Delta\rho_j > 0\}$ and p_2 be the element of $\mathcal{U}$ that achieves this minimum.

Step 3d: If $\mathcal{A}$ is empty, or if $\Delta\tau_{\mathcal{A}} \leq \mathbf{0}$, let $\varpi_3 = \infty$. Else let $\varpi_3 = \min\{\tau_i/\Delta\tau_i : i \in \mathcal{A}, \Delta\tau_i > 0\}$ and p_3 be the element of $\mathcal{A}$ that achieves this minimum.

Step 3e: If $\Delta x_{\mathcal{B}} = \mathbf{0}$, $\varpi_4 = \infty$. Else let $\varpi_4 = (\alpha - a^T x^*)/a_{\mathcal{B}}^T\Delta x_{\mathcal{B}}$.

Step 3f: Set $\varpi = \min(\varpi_1, \varpi_2, \varpi_3, \varpi_4)$.

Step 4: **Update primal and dual vectors:**

Step 4a: If $\varpi = \infty$ then (5.2) is infeasible. STOP.

Step 4b: Set $x_{\mathcal{B}}^* \leftarrow x_{\mathcal{B}}^* + \varpi\Delta x_{\mathcal{B}}$, $\pi^* \leftarrow \pi^* - \varpi\Delta\pi$, $\tau_{\mathcal{A}}^* \leftarrow \tau_{\mathcal{A}}^* - \varpi\Delta\tau_{\mathcal{A}}$, $\gamma_{\mathcal{L}}^* \leftarrow \gamma_{\mathcal{L}}^* - \varpi\Delta\gamma_{\mathcal{L}}$, $\rho_{\mathcal{U}}^* \leftarrow \rho_{\mathcal{U}}^* - \varpi\Delta\rho_{\mathcal{U}}$, $f \leftarrow f + \varpi a_{\mathcal{B}}^T\Delta x_{\mathcal{B}}(\nu + \varpi/2)$, and $\nu \leftarrow \nu + \varpi$.

Step 4c: **The subiteration is complete:**

If $\varpi = \varpi_3$, then set $\mathcal{A} \leftarrow \mathcal{A}\backslash\{k\}$; goto Step 3. Else if $\varpi = \varpi_2$, then set $\mathcal{U} \leftarrow \mathcal{U}\backslash\{k\}$; goto Step 3. Else if $\varpi = \varpi_1$, then set $\mathcal{L} \leftarrow \mathcal{L}\backslash\{k\}$; goto Step 3.

Step 4d: **The iteration is complete:**

If the constraint $a^T x \geq \alpha$ is a simple lower bound, then set $\mathcal{L} \leftarrow \mathcal{L}\cup\{k\}$; if it is a simple upper bound, then set $\mathcal{U} \leftarrow \mathcal{U}\cup\{k\}$; otherwise set $\mathcal{A} \leftarrow \mathcal{A} \cup \{k\}$. Goto Step 2.

The proofs of the next three theorems for our extended algorithm are modeled on the proofs of the theorems in Goldfarb and Idnani [22].

Theorem 5.3.2 If $\varpi = \infty$ in Step 4a of the algorithm, then (5.2) is infeasible.

Proof: Since $\varpi_4 = \infty$, we know that $\Delta x_{\mathcal{B}} = \mathbf{0}$ and the dual step directions $\Delta\pi \in \mathbb{R}^{m_1}$ and $\Delta\tau_{\mathcal{A}} \in \mathbb{R}^{|\mathcal{A}|}$ are such that $H_{:\mathcal{B}}^T \Delta\pi + C_{\mathcal{A}\mathcal{B}}^T \Delta\tau_{\mathcal{A}} = a_{\mathcal{B}}$. Since $\varpi_1 = \infty$ then either $a_{\mathcal{L}} \leq H_{:\mathcal{L}}^T \Delta\pi + C_{\mathcal{A}\mathcal{L}}^T \Delta\tau_{\mathcal{A}}$ or $\mathcal{L} = \emptyset$. Since $\varpi_2 = \infty$ then either $a_{\mathcal{U}} \geq H_{:\mathcal{U}}^T \Delta\pi + C_{\mathcal{A}\mathcal{U}}^T \Delta\tau_{\mathcal{A}}$ or $\mathcal{U} = \emptyset$. Since $\varpi_3 = \infty$, then either $\Delta\tau_{\mathcal{A}} \leq \mathbf{0}$ or $\mathcal{A} = \emptyset$.

Assume there exists an $\hat{x}$ such that $x^* + \hat{x}$ satisfies $H(x^* + \hat{x}) = h$, $C(x^* + \hat{x}) \geq d$, $l \leq (x^* + \hat{x}) \leq u$ and $a^T(x^* + \hat{x}) \geq \alpha$. In other words, $H\hat{x} = \mathbf{0}$, $C_{\mathcal{A}:}\hat{x} \geq \mathbf{0}$, $\hat{x}_{\mathcal{L}} \geq \mathbf{0}$, $\hat{x}_{\mathcal{U}} \leq \mathbf{0}$, and $a^T\hat{x} > 0$.

Given the existence of such a $\hat{x}$,

$$\begin{aligned} 0 &\geq (\Delta\pi^T H + \Delta\tau_{\mathcal{A}}^T C_{\mathcal{A}:})\hat{x} \\ &= (\Delta\pi^T H_{:\mathcal{B}} + \Delta\tau_{\mathcal{A}}^T C_{\mathcal{A}\mathcal{B}})\hat{x}_{\mathcal{B}} + (\Delta\pi^T H_{:\mathcal{L}}\hat{x}_{\mathcal{L}} + \Delta\tau_{\mathcal{A}}^T C_{\mathcal{A}\mathcal{L}})\hat{x}_{\mathcal{L}} \\ &\quad +(\Delta\pi^T H_{:\mathcal{U}}\hat{x}_{\mathcal{U}} + \Delta\tau_{\mathcal{A}}^T C_{\mathcal{A}\mathcal{U}})\hat{x}_{\mathcal{U}} \\ &\geq a_{\mathcal{B}}^T\hat{x}_{\mathcal{B}} + a_{\mathcal{L}}^T\hat{x}_{\mathcal{L}} + a_{\mathcal{U}}^T\hat{x}_{\mathcal{U}}. \end{aligned}$$

Since this contradicts the choice of $\hat{x}$, no such $\hat{x}$ exists and the quadratic programming problem (5.2) is infeasible.□

Theorem 5.3.3 If (5.2) is not infeasible, then starting with an S-list $(x^*, \mathcal{A}, \mathcal{L}, \mathcal{U})$ and having chosen a constraint $a^T x \geq \alpha$ violated by x^* in Step 2, the iteration will terminate with a new S-list such that the selected constraint violated by x^* is satisfied at equality and the index of that constraint is in the appropriate working set.

Proof: As long as (5.2) is feasible, at least one subiteration will occur. After each subiteration, there are updated dual variables ν, π^*, τ^*, γ^*, and ρ^*, a primal variable x^*, and working sets $\mathcal{A}$, $\mathcal{L}$, and $\mathcal{U}$ such that the dual feasibility constraint $-\mu + 2\theta V x^* - H^T\pi^* - C^T_{\mathcal{A}:}\tau^*_{\mathcal{A}} - I^T_{\mathcal{L}:}\gamma^*_{\mathcal{L}} - (-I_{\mathcal{U}:})^T\rho^*_{\mathcal{U}} - \nu a = \mathbf{0}$ and the nonnegativity of the dual variables $\tau^*_{\mathcal{A}}$, $\gamma^*_{\mathcal{L}}$, $\rho^*_{\mathcal{U}}$, and ν are satisfied. The complementary slackness condition of every primal constraint, except possibly of the constraint $a^T x \geq \alpha$, is satisfied. If the iteration is not done, then the complementary slackness condition $\nu(a^T x - \alpha) = 0$ may not be not satisfied since the full primal step was not taken. Thus the dual variable of some constraint previously in a working set $\mathcal{A}$, $\mathcal{L}$, or $\mathcal{U}$ reached zero and was dropped from the appropriate working set. Only a finite number of subiterations can be taken before the iteration is complete since the working sets are of finite cardinality.

Eventually a full primal step must be taken and $a^T x^* \geq \alpha$ will be satisfied at equality. Since a full primal step was taken, $\|\Delta x_{\mathcal{B}}\|_2 > 0$. Thus we know the rows of $H_{:\mathcal{B}}$ and $C_{\mathcal{A}\mathcal{B}}$ and the vector $a_{\mathcal{B}}$ are linearly independent. The complementary slackness condition of the new active constraint is satisfied so the updated $(x^*, \mathcal{A}, \mathcal{L}, \mathcal{U})$ is an S-list. □

Theorem 5.3.4 Let $(x^*, \mathcal{A}, \mathcal{L}, \mathcal{U})$ be an S-list for (5.2) and $a^T x \geq \alpha$ be a constraint of (5.2) such that $a^T x^* < \alpha$. If (5.2) is not infeasible, then after every subiteration the objective function value does not decrease and after the final subiteration, the objective function value strictly increases.

Proof: At the beginning of a subiteration, we have primal variables x^*, dual variables ν, π^*, τ^*, γ^*, and ρ^*, and working sets $\mathcal{A}$, $\mathcal{L}$, and $\mathcal{U}$ such that the dual feasibility constraint $-\mu + 2\theta V x^* - H^T\pi^* - C^T_{\mathcal{A}:}\tau^*_{\mathcal{A}} - I^T_{\mathcal{L}:}\gamma^*_{\mathcal{L}} -$

$(-I_{\mathcal{U}:})^T \rho^*_{\mathcal{U}} - \nu a = \mathbf{0}$ is satisfied. Let Z represent the appropriate nullspace matrix; then $ZH^T_{:\mathcal{B}} = \mathbf{0}$ and $ZC^T_{\mathcal{AB}} = \mathbf{0}$. Let $\Delta x_{\mathcal{B}} = \frac{1}{2\theta} Z a_{\mathcal{B}}$ be the primal step direction calculated during the subiteration. Then

$$\begin{aligned}
f(x^* + \varpi \Delta x) - f(x^*) &= \varpi \Delta x^T (2\theta V x^* - \mu) + \varpi^2 \theta \Delta x^T V \Delta x \\
&= \varpi \Delta x^T_{\mathcal{B}} (2\theta V_{\mathcal{B}:} x^* - \mu_{\mathcal{B}}) + \varpi^2 \theta \Delta x^T_{\mathcal{B}} V_{\mathcal{BB}} \Delta x_{\mathcal{B}} \\
&= \frac{\varpi}{2\theta} a^T_{\mathcal{B}} Z (2\theta V_{\mathcal{B}:} x^* - \mu_{\mathcal{B}}) + \frac{\varpi^2}{4\theta} a^T_{\mathcal{B}} Z V_{\mathcal{BB}} Z a_{\mathcal{B}} \\
&= \frac{\varpi}{2\theta} a^T_{\mathcal{B}} Z (C^T_{\mathcal{AB}} \tau^*_{\mathcal{A}} + H^T_{:\mathcal{B}} \pi^* + \nu a_{\mathcal{B}}) + \frac{\varpi^2}{4\theta} a^T_{\mathcal{B}} Z a_{\mathcal{B}} \\
&= \frac{\varpi}{2\theta} a^T_{\mathcal{B}} Z (\nu a_{\mathcal{B}}) + \frac{\varpi^2}{2} a^T_{\mathcal{B}} \Delta x_{\mathcal{B}} \\
&= \varpi \Delta x^T_{\mathcal{B}} a_{\mathcal{B}} (\nu + \frac{\varpi}{2}).
\end{aligned}$$

Since $\Delta x^T_{\mathcal{B}} a_{\mathcal{B}} = \frac{1}{2\theta} a^T_{\mathcal{B}} Z a_{\mathcal{B}} \geq 0$, $\nu \geq 0$ and $\varpi \geq 0$, we have that $f(x^* + \varpi \Delta x_{\mathcal{B}}) \geq f(x^*)$. Since $\|\Delta x_{\mathcal{B}}\|_2 > 0$ in the final subiteration, we know that $\varpi a^T_{\mathcal{B}} Z a_{\mathcal{B}} > 0$ and thus $f(x^* + \varpi \Delta x_{\mathcal{B}}) > f(x^*)$.□

5.3.2 Matrix Factorization Updates

In each subiteration of the algorithm, the working sets $\mathcal{A} \subseteq \mathcal{M}_2$ and $\mathcal{L}$,$\mathcal{U} \subseteq \mathcal{N}$ are given, and $\mathcal{B}$ is set to $\mathcal{N} \backslash (\mathcal{L} \cup \mathcal{U})$. To determine the dual steps

$$\begin{bmatrix} \Delta \pi \\ \Delta \tau_{\mathcal{A}} \end{bmatrix} = (W V^{-1}_{\mathcal{BB}} W^T)^{-1} W V^{-1}_{\mathcal{BB}} a_{\mathcal{B}} \quad \text{where } W = \begin{bmatrix} H_{:\mathcal{B}} \\ C_{\mathcal{AB}} \end{bmatrix},$$

a factorization of $W V^{-1}_{\mathcal{BB}} W^T$ must be calculated. Instead of factoring this matrix from scratch in every subiteration, we update the factorization whenever one of the working sets $\mathcal{A}$, $\mathcal{L}$, and $\mathcal{U}$ is changed. Goldfarb and Idnani [22] update an orthogonal factorization

$$Q \begin{bmatrix} \hat{R} \\ \mathbf{0} \end{bmatrix}$$

of $\Sigma^{-1/2}L^{-1}W^T$, where $L\Sigma L^T$ is the Cholesky factorization of V, Q is an $\mathcal{N}$ by $\mathcal{N}$ orthogonal matrix and $\hat{R}$ is an upper triangular matrix on the order of the number of active constraints. The space required to store the orthogonal matrix Q associated with this factorization is on the order of the square of the number of columns in W, so storing this matrix for each node of the implicit branch-and-bound tree may not be possible. Instead, we store and update a unit upper triangular matrix R and a diagonal matrix D such that $R^T DR$ is the Cholesky factorization of the current $WV_{\mathcal{BB}}^{-1}W^T$.

At the start of the current subiteration, we have a unit upper triangular matrix R and a diagonal matrix D such that $R^T DR = WV_{\mathcal{BB}}^{-1}W^T$. During a subiteration, either a row or column is added to or dropped from the matrix W. If a row is added or dropped, then there is a new set of active working constraints $\bar{\mathcal{A}}$; if a column is added or dropped, there is a new set of nonworking variables $\bar{\mathcal{B}}$. In either case, the matrix W is updated to $\bar{W}$, where $\bar{W}$ has full row rank. To calculate the dual steps in the next subiteration or iteration, we need a new unit upper triangular factor $\bar{R}$ and a new diagonal matrix $\bar{D}$ such that $\bar{R}\bar{D}\bar{R}^T = \bar{W}V_{\bar{\mathcal{B}}\bar{\mathcal{B}}}^{-1}\bar{W}^T$. These factorizations take on the order of $(m_1 + |\mathcal{A}|)^2$ storage, which is typically much less than the storage needed for the Goldfarb-Idnani algorithm.

Add an inequality $C_{k:}x \geq d_k$ to the working set of active constraints:

At the end of the subiteration, $\bar{\mathcal{A}} \leftarrow \mathcal{A} \cup \{k\}$ and $\bar{W}^T = [W^T \quad C_{k\mathcal{B}}^T]$. We need the Cholesky factorization of $\bar{W}V_{\mathcal{BB}}^{-1}\bar{W}^T$, which can be written as

$$\begin{bmatrix} W \\ C_{k\mathcal{B}} \end{bmatrix} V_{\mathcal{BB}}^{-1} \begin{bmatrix} W^T & C_{k\mathcal{B}}^T \end{bmatrix} = \begin{bmatrix} WV_{\mathcal{BB}}^{-1}W^T & WV_{\mathcal{BB}}^{-1}C_{k\mathcal{B}}^T \\ C_{k\mathcal{B}}V_{\mathcal{BB}}^{-1}W^T & C_{k\mathcal{B}}V_{\mathcal{BB}}^{-1}C_{k\mathcal{B}}^T \end{bmatrix}.$$

To calculate the new Cholesky factorization $\bar{R}^T\bar{D}\bar{R}$, we determine a vector $r \in \mathbb{R}^{m_1+|\bar{\mathcal{A}}|}$ and a constant $\eta > 0$ such that

$$\begin{bmatrix} WV_{\mathcal{BB}}^{-1}W^T & WV_{\mathcal{BB}}^{-1}C_{k\mathcal{B}}^T \\ C_{k\mathcal{B}}V_{\mathcal{BB}}^{-1}W^T & C_{k\mathcal{B}}V_{\mathcal{BB}}^{-1}C_{k\mathcal{B}}^T \end{bmatrix} = \begin{bmatrix} R^T & \mathbf{0} \\ r^T & 1 \end{bmatrix} \begin{bmatrix} D & \mathbf{0} \\ \mathbf{0} & \eta \end{bmatrix} \begin{bmatrix} R & r \\ \mathbf{0} & 1 \end{bmatrix} = \bar{R}^T\bar{D}\bar{R}.$$

The vector $r = D^{-1}R^{-T}WV_{\mathcal{BB}}^{-1}C_{k\mathcal{B}}^T$ was computed in the process of determining the dual steps and

$$\begin{aligned} \eta &= C_{k\mathcal{B}}V_{\mathcal{BB}}^{-1}C_{k\mathcal{B}}^T - r^TDr \\ &= C_{k\mathcal{B}}V_{\mathcal{BB}}^{-1}C_{k\mathcal{B}}^T - C_{k\mathcal{B}}V_{\mathcal{BB}}^{-1}W^TR^{-1}D^{-1}R^{-T}WV_{\mathcal{BB}}^{-1}C_{k\mathcal{B}}^T \\ &= C_{k\mathcal{B}}(V_{\mathcal{BB}}^{-1} - V_{\mathcal{BB}}^{-1}W^TR^{-1}D^{-1}R^{-T}WV_{\mathcal{BB}}^{-1})C_{k\mathcal{B}}^T \\ &= C_{k\mathcal{B}}ZC_{k\mathcal{B}}^T. \end{aligned}$$

Since $\eta = C_{k\mathcal{B}}ZVZC_{k\mathcal{B}}^T = 4\theta^2\Delta x_{\mathcal{B}}^TV\Delta x_{\mathcal{B}}$, we know that $\eta > 0$.

Drop an inequality $C_{k:}x \geq d_k$ from the working set of active constraints:

At the end of the subiteration, we set $\bar{\mathcal{A}} \leftarrow \mathcal{A}\backslash\{k\}$, thus we need to drop the row $C_{k\mathcal{B}}$ from W. If we were to remove the corresponding column from R and let $\bar{D} = D$, we would have

$$\bar{R}^T\bar{D}\bar{R} = \bar{W}V_{\mathcal{BB}}^{-1}\bar{W}^T.$$

However this $\bar{R}$ is not square nor upper triangular. Using a sequence of modified Givens rotations we first annihilate the nonzeros on the diagonal in the columns of R that follow the column of R corresponding to $C_{k\mathcal{B}}$.

Algorithm 5.3.2 Modified Givens Rotations

If $|\bar{R}_{j,j+1}\sqrt{\bar{D}_{jj}}| > |\bar{R}_{j+1,j+1}\sqrt{\bar{D}_{j+1,j+1}}|$ let

$$\xi = \frac{-\bar{R}_{j+1,j+1}\sqrt{\bar{D}_{j+1,j+1}}}{\bar{R}_{j,j+1}\sqrt{\bar{D}_{jj}}}; \quad s = 1/\sqrt{1+\xi^2}; \quad c = s\xi.$$

Otherwise let

$$\xi = \frac{-\bar{R}_{j,j+1}\sqrt{\bar{D}_{jj}}}{\bar{R}_{j+1,j+1}\sqrt{\bar{D}_{j+1,j+1}}}; \quad c = 1/\sqrt{1+\xi^2}; \quad s = c\xi.$$

Algorithm 5.3.3 Annihilate Off-Diagonal Elements

Step 1: Let j represent the position of the column of R that is to be deleted. Set $\bar{R} = R$ and $\bar{D} = D$.

Step 2: If $j = n$, STOP. Otherwise, calculate c and s based on $\bar{R}_{j,j+1}$ and $\bar{R}_{j+1,j+1}$ using the Modified Givens Rotation.

Step 3: For each $k = j+1, \ldots, n$ we let

$$\begin{aligned} w &= c\bar{R}_{jk} - s\bar{R}_{j+1,k}\sqrt{\bar{D}_{j+1,j+1}/\bar{D}_{jj}}, \\ \bar{R}_{j+1,k} &= c\bar{R}_{j+1,k} + s\bar{R}_{jk}\sqrt{\bar{D}_{jj}/\bar{D}_{j+1,j+1}}, \text{ and} \\ \bar{R}_{jk} &= w. \end{aligned}$$

Step 4: Scale $\bar{D}_{jj} = \bar{D}_{jj}\bar{R}^2_{j,j+1}$ and $\bar{R}_{j:} = \bar{R}_{j:}/\bar{R}_{j,j+1}$. Let $j = j+1$; goto Step 2.

The jth column of $\bar{R}$, the last row of $\bar{R}$ and the last column and row of $\bar{D}$ can be dropped, and we are left with the appropriate Cholesky factorization. Step 4 is necessary so that the diagonal elements of $\bar{R}$ are all equal to one.

Drop a column from some working set of active columns

At the end of the subiteration, $\bar{\mathcal{B}} \leftarrow \mathcal{B} \cup \{k\}$ and $\mathcal{A}$ has not changed.

We first need to update the Cholesky factorization $L\Sigma L^T$ of $V_{\mathcal{B}\mathcal{B}}$ to a Cholesky factorization $\bar{L}\bar{\Sigma}\bar{L}^T$ of $V_{\bar{\mathcal{B}}\bar{\mathcal{B}}}$. Since

$$V_{\mathcal{B}\mathcal{B}} = \begin{bmatrix} V_{\mathcal{B}\mathcal{B}} & V_{\mathcal{B}k} \\ V_{k\mathcal{B}} & V_{kk} \end{bmatrix} = \begin{bmatrix} L & \mathbf{0} \\ r^T & 1 \end{bmatrix} \begin{bmatrix} \Sigma & \mathbf{0} \\ \mathbf{0} & \eta \end{bmatrix} \begin{bmatrix} L^T & r \\ \mathbf{0} & 1 \end{bmatrix},$$

solving for $r = \Sigma^{-1}L^{-1}V_{\mathcal{B}k}$ and setting $\eta = V_{kk} - r^T\Sigma r$ results in the correct Cholesky factorization $\bar{L}\bar{\Sigma}\bar{L}^T$ of $V_{\bar{\mathcal{B}}\bar{\mathcal{B}}}$.

To determine the Cholesky factorization $\bar{R}^T\bar{D}\bar{R}$ of $\bar{W}V_{\bar{\mathcal{B}}\bar{\mathcal{B}}}^{-1}\bar{W}^T$, we first let

$$w = \begin{bmatrix} H_{:k} \\ C_{\mathcal{A}k} \end{bmatrix} \text{ so that } \bar{W} = \begin{bmatrix} W & w \end{bmatrix}.$$

Then we see that

$$\begin{aligned} \bar{W}V_{\bar{\mathcal{B}}\bar{\mathcal{B}}}^{-1}\bar{W}^T &= \begin{bmatrix} W & w \end{bmatrix} \begin{bmatrix} V_{\mathcal{B}\mathcal{B}} & V_{\mathcal{B}k} \\ V_{k\mathcal{B}} & V_{kk} \end{bmatrix}^{-1} \begin{bmatrix} W^T \\ w^T \end{bmatrix} \\ &= \begin{bmatrix} W & w \end{bmatrix} \begin{bmatrix} V_{\mathcal{B}\mathcal{B}}^{-1} + \eta^{-1}L^{-T}rr^TL^{-1} & -\eta^{-1}L^{-T}r \\ -\eta^{-1}r^TL^{-1} & \eta^{-1} \end{bmatrix} \begin{bmatrix} W^T \\ w^T \end{bmatrix} \\ &= WV_{\mathcal{B}\mathcal{B}}^{-1}W^T + \eta^{-1}WL^{-T}rr^TL^{-1}W^T - \eta^{-1}wr^TL^{-1}W^T \\ &\quad -\eta^{-1}WL^{-T}rw^T + \eta^{-1}ww^T \\ &= R^TDR + \eta^{-1}(WL^{-T}r - w)(WL^{-T}r - w)^T, \end{aligned}$$

which is a Cholesky rank-one update. Letting $v = \eta^{-1/2}(WL^{-T}r - w)$, then $\bar{W}V_{\bar{\mathcal{B}}\bar{\mathcal{B}}}^{-1}\bar{W}^T = R^T(D + pp^T)R$, where $R^Tp = v$. Algorithm 5.3.4 is taken from [21].

Algorithm 5.3.4 Cholesky Rank-One Update

Step 1: Let $t_0 = 1$ and $j = 1$.

Step 2: If $j > n$, STOP. Let $p_j = v_j$, $t_j = t_{j-1} + p_j^2/D_{jj}$, and $\bar{D}_{jj} = D_{jj}t_j/t_{j-1}$. Let $r = j + 1$.

Step 3: If $r > n$, let $j = j + 1$ and goto Step 2. Otherwise, let $v_r = v_r - p_j R_{jr}$ and $\bar{R}_{jr} = R_{jr} + p_j v_r/(D_{jj}t_j)$. Let $r = r + 1$ and repeat Step 3.

Add a column to a working set of active columns

At the end of the subiteration, $\bar{\mathcal{B}} \leftarrow \mathcal{B}\backslash\{k\}$ and $\mathcal{A}$ has not changed.

We assume that the index of the column being dropped from $\mathcal{B}$ is in the last position in $\mathcal{B}$. If it is not, then we can permute Σ and the columns of L and annihilate any nonzeros below the diagonal with modified Givens rotations.

We first need to update the Cholesky factorization $L\Sigma L^T$ of $V_{\mathcal{B}\mathcal{B}}$ to a Cholesky factorization $\bar{L}\bar{\Sigma}\bar{L}^T$ of $V_{\bar{\mathcal{B}}\bar{\mathcal{B}}}$.

$$V_{\mathcal{B}\mathcal{B}} = \begin{bmatrix} V_{\bar{\mathcal{B}}\bar{\mathcal{B}}} & V_{\bar{\mathcal{B}}k} \\ V_{k\bar{\mathcal{B}}} & V_{kk} \end{bmatrix} = \begin{bmatrix} \bar{L} & \mathbf{0} \\ r^T & 1 \end{bmatrix} \begin{bmatrix} \bar{\Sigma} & \mathbf{0} \\ \mathbf{0} & \eta \end{bmatrix} \begin{bmatrix} \bar{L}^T & r \\ \mathbf{0} & 1 \end{bmatrix}.$$

The correct Cholesky factorization of $V_{\bar{\mathcal{B}}\bar{\mathcal{B}}}$ is $\bar{L}\bar{\Sigma}\bar{L}^T$ and we set $r = \bar{\Sigma}^{-1}\bar{L}^{-1}V_{\bar{\mathcal{B}}k}$ and $\eta = V_{kk} - r^T\bar{\Sigma}r$.

To determine the Cholesky factorization $\bar{R}^T\bar{D}\bar{R}$ of $\bar{W}V_{\bar{\mathcal{B}}\bar{\mathcal{B}}}^{-1}\bar{W}^T$, we first let

$$w = \begin{bmatrix} H_{:k} \\ C_{\mathcal{A}k} \end{bmatrix} \text{ so that } W = \begin{bmatrix} \bar{W} & w \end{bmatrix}.$$

Then we see that

$$WV_{\mathcal{B}\mathcal{B}}^{-1}W^T = \begin{bmatrix} \bar{W} & w \end{bmatrix} \begin{bmatrix} V_{\bar{\mathcal{B}}\bar{\mathcal{B}}} & V_{\bar{\mathcal{B}}k} \\ V_{k\bar{\mathcal{B}}} & V_{kk} \end{bmatrix}^{-1} \begin{bmatrix} \bar{W}^T \\ w^T \end{bmatrix}$$

$$
\begin{aligned}
&= \begin{bmatrix} \bar{W} & w \end{bmatrix} \begin{bmatrix} V_{\bar{\mathcal{B}}\bar{\mathcal{B}}}^{-1} + \eta^{-1}\bar{L}^{-T}rr^T\bar{L}^{-1} & -\eta^{-1}\bar{L}^{-T}r \\ -\eta^{-1}r^T\bar{L}^{-1} & \eta^{-1} \end{bmatrix} \begin{bmatrix} \bar{W}^T \\ w^T \end{bmatrix} \\
&= \bar{W}V_{\bar{\mathcal{B}}\bar{\mathcal{B}}}^{-1}\bar{W}^T + \eta^{-1}\bar{W}\bar{L}^{-T}rr^T\bar{L}^{-1}\bar{W}^T - \eta^{-1}wr^T\bar{L}^{-1}\bar{W}^T \\
&\quad -\eta^{-1}\bar{W}\bar{L}^{-T}rw^T + \eta^{-1}ww^T \\
&= \bar{R}^T\bar{D}\bar{R} + \eta^{-1}(\bar{W}\bar{L}^{-T}r - w)(\bar{W}\bar{L}^{-T}r - w)^T,
\end{aligned}
$$

which is a Cholesky rank-one downdate. Letting $v = \eta^{-1/2}(\bar{W}\bar{L}^{-T}r - w)$, we see that $\bar{W}V_{\bar{\mathcal{B}}\bar{\mathcal{B}}}^{-1}\bar{W}^T = R^T(D - pp^T)R$, where $R^Tp = v$. Algorithm 5.3.5 is taken from [21].

Algorithm 5.3.5 Cholesky Rank-One Downdate

Step 1: Let $p = R^{-T}v$ and let $t_{n+1} = 1 - p^TD^{-1}p$; if $t_{n+1} \leq \epsilon$, set $t_{n+1} = \epsilon$, where ϵ is the machine precision. Let $j = n$.

Step 2: If $j < 1$, STOP. Let $v_j = p_j$, $t_j = t_{j+1} + p_j^2/D_{jj}$, and $\bar{D}_{jj} = D_{jj}t_{j+1}/t_j$. Let $r = j + 1$.

Step 3: If $r > n$, let $j = j - 1$ and goto Step 2. Otherwise, let $\bar{R}_{jr} = R_{jr} - p_jv_r/(D_{jj}t_{j+1})$ and $v_r = v_r + p_jR_{jr}$. Let $r = r + 1$ and goto Step 3.

5.3.3 Numerical Stability of Updates

Although in our experience these factorization updates perform well, in theory they may not be stable. In Step 3a of Algorithm 5.3.1, we solve (5.5) for the dual step directions. This can be seen as the zero residual full-rank least squares problem

$$\min \|V_{\mathcal{B}\mathcal{B}}^{-1/2}W^T \begin{bmatrix} \Delta\pi \\ \Delta\tau_{\mathcal{A}} \end{bmatrix} - \mathrm{V}_{\mathcal{B}\mathcal{B}}^{-1/2}\mathrm{a}_{\mathcal{B}}\|_2$$

such that $\Delta\pi \in \mathbb{R}^{m_1}$ and $\Delta\tau_{\mathcal{A}} \in \mathbb{R}^{|\mathcal{A}|}$. According to Golub and Van Loan [23], for the full-rank least squares problem, a normal equations method produces a solution

whose relative error depends on the square of the condition of $V_{\mathcal{BB}}^{-1/2}W^T$ and may result in a solution that is not as accurate as the orthogonal Householder approach. This is in part due to the fact that the matrix $WV_{\mathcal{BB}}^{-1}W^T$ has a condition number that is square the condition number of $V_{\mathcal{BB}}^{-1/2}W^T$. However, the normal equations approach involves about half the arithmetic and requires much less storage than the orthogonal approach. We have chosen to use a method comparable to the normal equations and have instituted checks to help ensure stability. The least stable update in the previous section is the Cholesky downdate used when adding a column to one of the working sets. The Cholesky rank-one downdating algorithm we described is derived in [20], and although not backwards stable, is preferable to the method of hyperbolic transformations [56].

After determining the optimal solution to (5.2), we check that the final values of the primal and dual variables satisfy all of the optimality conditions. If any of the optimality conditions are violated, we return to the S-list given at the start of Algorithm 5.3.1, calculate from scratch the Cholesky factorization of the appropriate matrix, and resolve the quadratic programming problem. If the optimality conditions are not satisfied by the final values of the primal and dual variables again, then we return to the original S-list and solve the quadratic programming problem using the more computationally expensive orthogonal factorization. The algorithms for updating the orthogonal factorization after adding or dropping an inequality or a row are straightforward and can be found in [9].

Chapter 6

The Confidence Interval Objective Function

In this chapter we present our algorithm for finding the optimal solution to a subproblem of the mixed-integer nonlinear programming problem (3.3) in the implicit branch-and-bound tree, where the objective function is the confidence interval objective function. We first examine the properties of this problem subject to only equality constraints, and then we extend the results to a more general problem.

6.1 The Nonlinear Programming Problem

At a node in the implicit branch-and-bound tree, the optimal solution to the implicit relaxation of the subproblem defined by the sets $\mathcal{Z}$ and $\mathcal{P}$ must be determined; this relaxation is

$$\begin{aligned} \text{minimize} \quad & f(x,y) = -\mu^T x + \theta\sqrt{x^T V x} \\ \text{subject to} \quad & Cx \geq d, \\ & 0 \leq y_j \leq 1 \quad \forall j \in \mathcal{N}, \\ & x_j = y_j = 0 \quad \forall j \in \mathcal{Z}, \\ & y_j = 1 \quad \forall j \in \mathcal{P}, \\ & x \in \mathbb{R}^n, \; y \in \mathbb{R}^n, \end{aligned} \tag{6.1}$$

where $\mathcal{N} = \{1, \ldots, n\}$, θ is a strictly positive scalar, V is a symmetric positive definite matrix, and the linear constraints $Cx \geq d$ include the simple bounds $l_j \leq x_j \leq u_j$ for all $j \in \mathcal{N}$, the constraints of $H_x x + H_y y \geq h$ for which no binary variable has a nonzero coefficient, and the valid inequalities derived in Chapter 4.

To simplify the notation in this chapter, (6.1) is formulated to include the equality constraints explicitly, and the continuous variables that are fixed to zero are removed. Since we predetermine the values of the binary variables at a node in the implicit branch-and-bound tree, the binary variables are removed. The optimization subproblem is

$$\begin{aligned} \text{minimize} \quad & \hat{f}(x) = -\mu^T x + \theta\sqrt{x^T V x} \\ \text{subject to} \quad & Hx = h, \\ & Cx \geq d, \\ & l \leq x \leq u, \\ & x \in \mathbb{R}^n, \end{aligned} \tag{6.2}$$

where n is the number of remaining variables, $\mathcal{N} = \{1, \ldots, n\}$, m_1 is the number of equality constraints, $\mathcal{M}_1 = \{1, \ldots, m_1\}$, m_2 is the number of inequality constraints (excluding the simple bounds on the variables), $\mathcal{M}_2 = \{1, \ldots, m_2\}$, H is an m_1 by n matrix, $h \in \mathbb{R}^{m_1}$, C is an m_2 by n matrix, and $d \in \mathbb{R}^{m_2}$. Without loss of generality, we can assume the matrix H has full row rank, and no inequality constraint or remaining simple bound can be represented as a linear combination of the rows of H. The feasible set of this problem is denoted by $\hat{\mathcal{F}}$.

The nonlinear programming problem (6.2) is more interesting than the quadratic programming problem (5.2) because the function $\nabla\hat{f}(x) : \mathbb{R}^n \to \mathbb{R}^n$ may not be defined over the entire feasible set, since $\nabla\hat{f}(x) = -\mu + \theta(x^T V x)^{-1/2} V x$ is not defined for $x = \mathbf{0}$. Another interesting aspect of (6.2) is that $\hat{f}(x)$ is not strictly convex, and thus (6.2) may not have a unique solution. Fortunately, $\hat{f}(x)$ is convex; so any local minimizer of (6.2) is a global minimizer of that problem as well.

Theorem 6.1.1 The function $\hat{f}(x) = -\mu^T x + \theta\sqrt{x^T V x}$ is convex over the convex feasible set $\hat{\mathcal{F}} \subseteq \mathbb{R}^n$.

Proof: We need to show that for any $v, w \in \hat{\mathcal{F}}$ and $0 \leq \lambda \leq 1$, the inequality $f(\lambda v + (1-\lambda)w) \leq \lambda f(v) + (1-\lambda)f(w)$ holds. For simplicity of notation, we note that $\sqrt{x^T V x} = \|x\|_V$ for any $x \in \mathbb{R}^n$, where $\|\cdot\|_V$ is the weighted 2-norm and that $\hat{f}(x) = -\mu^T x + \theta\|x\|_V$. By the triangle inequality,

$$\begin{aligned} & f(\lambda v + (1-\lambda)w) \\ &\quad = -\lambda\mu^T v - (1-\lambda)\mu^T w + \theta\|(\lambda v + (1-\lambda)w)\|_V \\ &\quad \leq -\lambda\mu^T v - (1-\lambda)\mu^T w + \theta\|\lambda v\|_V + \theta\|(1-\lambda)w\|_V \\ &\quad = -\lambda\mu^T v - (1-\lambda)\mu^T w + \theta\lambda\|v\|_V + \theta(1-\lambda)\|w\|_V \\ &\quad = \lambda f(v) + (1-\lambda)f(w), \end{aligned}$$

and we are done.□

6.2 The Equality Constrained Problem

As in Chapter 5, we solve the general problem (6.2) by solving a sequence of equality constrained problems. It is thus useful to examine the equality constrained problem

$$\begin{aligned} \text{minimize} \quad & -\mu^T x + \theta\sqrt{x^T V x} \\ \text{subject to} \quad & Wx = w, \end{aligned} \tag{6.3}$$

where $\|\mu\|_2 > 0$, W has full row rank, and m is the number equality constraints. For the rest this section, we let the matrix Z represent the n by n positive semi-definite matrix $V^{-1} - V^{-1}W^T(WV^{-1}W^T)^{-1}WV^{-1}$, which is a nullspace matrix for W. It is easy to show that given some $\bar{x} \in \bar{\mathcal{F}} = \{x : Wx = w\}$, the vector x is in $\bar{\mathcal{F}}$ if and only if there exists a $v \in \mathbb{R}^n$ such that $x = \bar{x} + Zv$. In this section, we prove the following four theorems which provide the conditions for the existence of a minimizer of (6.3), based on the relationship between the values of θ and $\sqrt{\mu^T Z \mu}$.

Theorem 6.2.1 If $\theta < \sqrt{\mu^T Z \mu}$, then $\hat{f}(x)$ is unbounded below on the feasible set $\bar{\mathcal{F}} = \{x : Wx = w\}$.

Theorem 6.2.2 If $\theta = \sqrt{\mu^T Z \mu}$ and $w = \mathbf{0}$, then the set of minimizers of $\hat{f}(x)$ over $\bar{\mathcal{F}} = \{x : Wx = w\}$ has infinite cardinality.

Theorem 6.2.3 If $\theta = \sqrt{\mu^T Z \mu}$ and $\|w\|_2 > 0$, then $\hat{f}(x)$ is bounded below on the feasible set $\bar{\mathcal{F}} = \{x : Wx = w\}$, but has no minimizer.

Theorem 6.2.4 If $\theta > \sqrt{\mu^T Z \mu}$, then $\hat{f}(x)$ has a unique minimizer on the feasible set $\bar{\mathcal{F}} = \{x : Wx = w\}$.

Once we have completed the proofs of these theorems, we present the necessary and sufficient conditions for an optimal solution of (6.3).

Theorem 6.2.1 If $\theta < \sqrt{\mu^T Z \mu}$, then $\hat{f}(x)$ is unbounded below on the feasible set $\bar{\mathcal{F}} = \{x : Wx = w\}$.

Proof: We first show that the point $\bar{x} = N^T w + \alpha Z \mu$, where $N = (WV^{-1}W^T)^{-1}WV^{-1}$ and $\alpha > 0$, is in $\bar{\mathcal{F}}$:

$$W\bar{x} = W(N^T w + \alpha Z \mu) = WN^T w + 0 = w.$$

For simplicity of notation, we note that $\sqrt{x^T V x} = \|x\|_V$ for any $x \in \mathbb{R}^n$, where $\|\cdot\|_V$ is the weighted 2-norm. Evaluating $\hat{f}(x)$ at $\bar{x}$ and using the triangle inequality, we get

$$\begin{aligned}
\hat{f}(\bar{x}) &= -\mu^T N^T w - \alpha \mu^T Z \mu + \theta \|N^T w + \alpha Z \mu\|_V \\
&\leq -\mu^T N^T w - \alpha \mu^T Z V Z \mu \\
&\quad + \theta \|N^T w\|_V + \theta |\alpha| \|Z \mu\|_V \\
&= \hat{f}(N^T w) + \alpha \|Z \mu\|_V \left(\theta - \|Z \mu\|_V\right).
\end{aligned}$$

Since $\|Z\mu\|_V = \mu^T ZVZ\mu = \mu^T Z\mu > \theta > 0$ and $\theta - \|Z\mu\|_V < 0$, the value of $\hat{f}(N^T w + \alpha Z\mu)$ approaches $-\infty$ as the scalar α approaches $+\infty$.□

Example 6.2.1 In the unconstrained case, Z is equivalent to V^{-1}. The function $\hat{f}(x) = -.4x_1 - .6x_2 + .5\sqrt{.25x_1^2 + x_2^2}$ is unbounded below because θ is strictly smaller than $\sqrt{\mu^T V^{-1}\mu}$. In this surface plot, the contour lines of the objective function are shown on the base for emphasis.

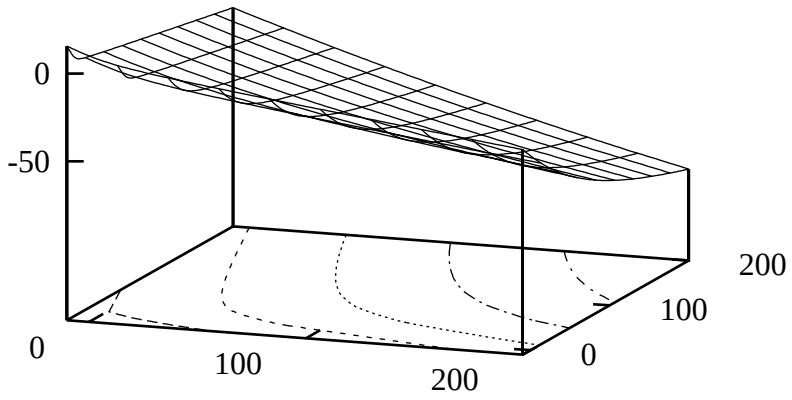

Figure 6.1 No minimizer

The case where $\theta = \sqrt{\mu^T Z\mu}$ is quite interesting. If $w = \mathbf{0}$, then $x^* = \mathbf{0}$ is a minimizer of $\hat{f}(x)$, and the set of minimizers of $\hat{f}(x)$ has infinite cardinality. However, if $\|w\|_2 > 0$, the function $\hat{f}(x)$ is bounded below on the feasible set, but it has no minimizer.

Theorem 6.2.2 If $\theta = \sqrt{\mu^T Z\mu}$ and $w = \mathbf{0}$, then the set of minimizers of $\hat{f}(x)$ over $\bar{\mathcal{F}} = \{x : Wx = w\}$ has infinite cardinality.

Proof: Let $x^* = \mathbf{0}$. We first show that x^* is a minimizer by showing that $\hat{f}(x^* + Zv) - \hat{f}(x^*) = \hat{f}(Zv) \geq 0$ for all $v \in \mathbb{R}^n$. If $\mu^T Zv \leq 0$, then

$\hat{f}(Zv) = -\mu^T Zv + \theta\sqrt{v^T Zv} \geq 0$. If, on the other hand, $\mu^T Zv > 0$, then by the Schwarz inequality

$$\begin{aligned} \hat{f}(Zv) &= -\mu^T Zv + \theta\|Zv\|_V \\ &\geq -\mu^T Zv + \theta\mu^T ZVZv\|Z\mu\|_V^{-1} \\ &= -\mu^T Zv + \mu^T Zv\|Z\mu\|_V \cdot \|Z\mu\|_V^{-1} \\ &= 0. \end{aligned} \tag{6.4}$$

Thus $x^* = \mathbf{0}$ is a minimizer of $\hat{f}(x)$ over $\bar{\mathcal{F}}$. Since $\theta = \sqrt{\mu^T Z\mu} = \|Z\mu\|_2$, we know that $\|Z\mu\|_2 > 0$. Then

$$\begin{aligned} \hat{f}(Z\mu) &= -\mu^T Z\mu + \theta\sqrt{\mu^T ZVZ\mu} \\ &= -\mu^T Z\mu + \|Z\mu\|_V^2 \\ &= 0, \end{aligned}$$

so $Z\mu$ is another minimizer. Since $\hat{f}(\alpha Z\mu) = \alpha\hat{f}(Z\mu) = 0$ for all $\alpha \geq 0$, the set of minimizers of $\hat{f}(x)$ over $\bar{\mathcal{F}}$ is of infinite cardinality.□

Example 6.2.2 The set of minimizers of the function $\hat{f}(x) = -.4x_1 - .6x_2 + \sqrt{.25x_1^2 + x_2^2}$ has infinite cardinality since both θ and $\sqrt{\mu^T V^{-1}\mu}$ are equal to 1.

The following plot is seen from the side to emphasize that the function never decreases below zero.

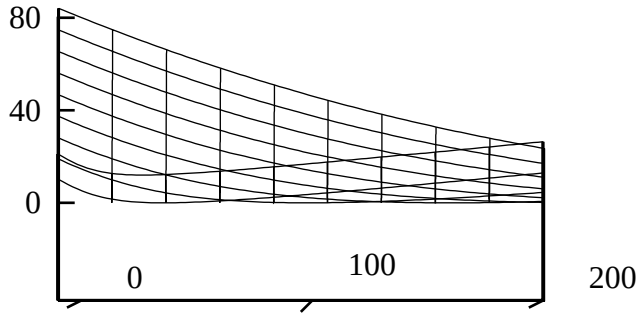

Figure 6.2 Infinitely many minimizers

Theorem 6.2.3 If $\theta = \sqrt{\mu^T Z \mu}$ and $\|w\|_2 > 0$, then $\hat{f}(x)$ is bounded below on the feasible set $\bar{\mathcal{F}} = \{x : Wx = w\}$, but has no minimizer.

Proof: We first show that $\hat{f}(x)$ is bounded below on the feasible set $\mathcal{F}$ by the value $-\mu^T N^T w$, where $N = (WV^{-1}W^T)^{-1}WV^{-1}$. Let $\bar{x} = N^T w$. Then for any $v \in \mathbb{R}^n$, such that $\|Zv\|_2 > 0$,

$$\hat{f}(\bar{x} + Zv) = -\mu^T \bar{x} - \mu^T Zv + \theta\sqrt{w^T(WV^{-1}W^T)^{-1}w + v^T Z v}.$$

If $\mu^T Zv \leq 0$, clearly $\hat{f}(\bar{x} + Zv) > -\mu^T N^T w$. If $\mu^T Zv > 0$, then, since $\|w\|_2 > 0$ and $WV^{-1}W^T$ is a positive definite matrix,

$$\begin{aligned} \hat{f}(\bar{x} + Zv) &= -\mu^T \bar{x} - \mu^T Zv + \sqrt{\theta^2 w^T(WV^{-1}W^T)^{-1}w + \mu^T Z \mu v^T Z v} \\ &> -\mu^T N^T w - \mu^T Zv + \sqrt{(\mu^T Zv)^2} \\ &= -\mu^T N^T w. \end{aligned}$$

Thus $-\mu^T N^T w$ is a lower bound for $\hat{f}(x)$ on the feasible set $\bar{\mathcal{F}}$, but no feasible point attains this bound.

We next show that for every $\epsilon > 0$, there exists a $v \in \mathbb{R}^n$ such that $\hat{f}(\bar{x} + Zv) + \mu N^T w \leq \epsilon$. Let $\epsilon > 0$ be given. Since $\theta = \|Z\mu\|_V$, $\mu^T Z\mu > 0$. Letting

$$\alpha = \frac{w^T(WV^{-1}W^T)^{-1}w}{2\epsilon} - \frac{\epsilon}{2\mu^T Z\mu}$$

and setting $v = \alpha\mu$, we get

$$\begin{aligned}
&\hat{f}(\bar{x} + \alpha Z\mu) + \mu^T N^T w \\
&= -\alpha\mu^T Z\mu + \theta\sqrt{w^T(WV^{-1}W^T)^{-1}w + \alpha^2\mu^T Z\mu} \\
&= -\alpha\mu^T Z\mu + \frac{\theta}{\|Z\mu\|_V}\left(\left(\frac{w^T(WV^{-1}W^T)^{-1}w\mu^T Z\mu}{2\epsilon} + \frac{\epsilon}{2}\right)^2\right)^{1/2} \\
&= -w^T(WV^{-1}W^T)^{-1}w\mu^T Z\mu/2\epsilon + \epsilon/2 \\
&\quad + w^T(WV^{-1}W^T)^{-1}w\mu^T Z\mu/2\epsilon + \epsilon/2 \\
&= \epsilon.
\end{aligned}$$

It is easy to see that $\hat{f}(\bar{x} + \alpha Z\mu)$ approaches $-\mu^T N^T w$ asymptotically as α approaches $+\infty$. This infimum is never attained since $\hat{f}(x) > -\mu^T N^T w$ for all $x \in \bar{\mathcal{F}}$.□

Theorem 6.2.4 If $\theta > \sqrt{\mu^T Z\mu}$, then $\hat{f}(x)$ has a unique minimizer on the feasible set $\bar{\mathcal{F}} = \{x : Wx = w\}$.

Proof: The point

$$x^* = \sqrt{\frac{w^T(WV^{-1}W^T)^{-1}w}{\theta^2 - \mu^T Z\mu}}Z\mu + V^{-1}W^T(WV^{-1}W^T)^{-1}w \qquad (6.5)$$

is an element of $\bar{\mathcal{F}}$. To prove that x^* is the unique minimizer of $\hat{f}$ over $\bar{\mathcal{F}}$ we only need to show that for any $v \in \mathbb{R}^n$ such that $\|Zv\|_2 > 0$, $\hat{f}(x^* + Zv) - \hat{f}(x^*) > 0$.

We first note that

$$\begin{aligned} x^{*T}Vx^* &= \frac{w^T(WV^{-1}W^T)^{-1}w}{\theta^2-\mu^TZ\mu}\mu^TZ\mu + w^T(WV^{-1}W^T)^{-1}w \\ &\quad +2\sqrt{\frac{w^T(WV^{-1}W^T)^{-1}w}{\theta^2-\mu^TZ\mu}}\mu^TZW^T(WV^{-1}W^T)^{-1}w \\ &= \theta^2\frac{w^T(WV^{-1}W^T)^{-1}w}{\theta^2-\mu^TZ\mu}. \end{aligned}$$

Then,

$$\begin{aligned} &\hat{f}(x^*+Zv)-\hat{f}(x^*) \\ &= -\mu^TZv+\theta\sqrt{\|x^*\|_V^2+2v^TZVx^*+v^TZv}-\theta\|x^*\|_V \\ &= -\mu^TZv-\theta\|x^*\|_V+\sqrt{\theta^2\|x^*\|_V^2+2\theta\|x^*\|_V\mu^TZv+\theta^2v^TZv} \\ &> -\mu^TZv-\theta\|x^*\|_V+\sqrt{\theta^2\|x^*\|_V^2+2\theta\|x^*\|_V\mu^TZv+\mu^TZ\mu v^TZv} \\ &\geq -\mu^TZv-\theta\|x^*\|_V+\sqrt{\theta^2\|x^*\|_V^2+2\theta\|x^*\|_V\mu^TZv+(\mu^TZv)^2} \\ &= -\mu^TZv-\theta\|x^*\|_V+|\theta\|x^*\|_V+\mu^TZv| \end{aligned}$$

If $\theta\|x^*\|_V+\mu^TZv \geq 0$, then $\hat{f}(x^*+Zv)-\hat{f}(x^*) > 0$. Furthermore, if $\theta\|x^*\|_V+\mu^TZv < 0$, then $\hat{f}(x^*+Zv)-\hat{f}(x^*) > 0$.□

Example 6.2.3 The function $\hat{f}(x) = -.4x_1 - .6x_2 + 2\sqrt{.25x_1^2+x_2^2}$ has a unique global minimizer at zero.

In this surface plot, the contour lines of the objective function are shown on the base for emphasis.

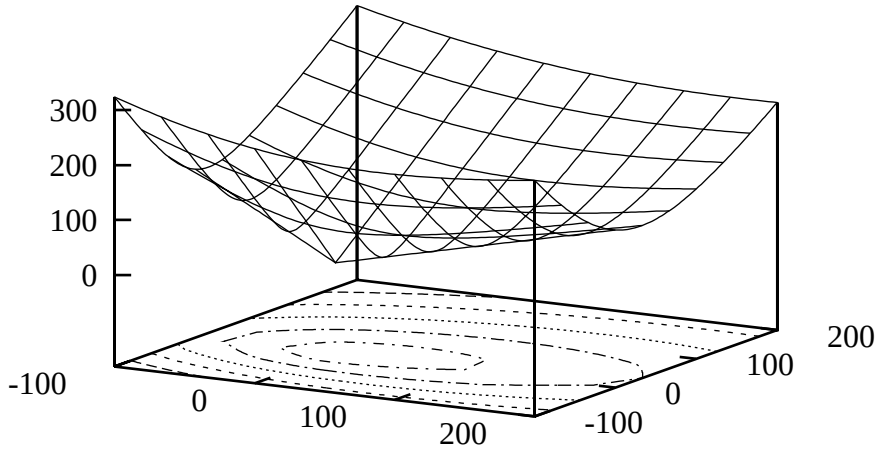

Figure 6.3 Unique minimizer

The choice for x^* in the proof of Theorem 6.2.4 is not surprising if one examines the necessary and sufficient conditions for optimality. If $x = \mathbf{0}$ is not in the feasible set $\bar{\mathcal{F}} = \{x \in \mathbb{R}^n : Wx = w\}$ of this problem, then, since $\nabla^2 \hat{f}(x)$ is positive semi-definite, Theorems 5.2.1 and 5.2.2, can be combined to give the following theorem.

Theorem 6.2.5 The necessary and sufficient conditions that $x^* \in \bar{\mathcal{F}}$, where $\mathbf{0} \notin \bar{\mathcal{F}}$, be an optimal solution of the convex programming problem (6.3) are that there exists a vector of Lagrange multipliers $\pi^* \in \mathbb{R}^m$ such that

$$\nabla_x \mathcal{L}(x^*, \pi^*) = -\mu + \frac{\theta}{\sqrt{x^{*T} V x^*}} V x^* - W^T \pi^* = \mathbf{0}.$$

In fact, if $\theta > \sqrt{\mu^T Z \mu}$, then solving the system of nonlinear equations

$$\begin{aligned} -\mu + \tfrac{\theta}{\sqrt{x^T V x}} V x - W^T \pi &= \mathbf{0}, \\ W x &= w, \end{aligned} \tag{6.6}$$

gives

$$\pi = -(WV^{-1}W^T)^{-1} W V^{-1} \mu + \frac{\theta}{\sqrt{x^T V x}} (WV^{-1}W^T)^{-1} w$$

and

$$x = \sqrt{\frac{w^T(WV^{-1}W^T)^{-1}w}{\theta^2 - \mu^T Z\mu}} Z\mu + V^{-1}W^T(WV^{-1}W^T)^{-1}w,$$

which is exactly (6.5). If $\theta \leq \sqrt{\mu^T Z\mu}$, then (6.6) has no solution. The following theorem gives the necessary and sufficient conditions for an optimal solution of (6.3) if $\mathbf{0} \in \bar{\mathcal{F}}$.

Theorem 6.2.6 Let $\mathbf{0} \in \bar{\mathcal{F}}$. Then a necessary and sufficient condition that $x^* \in \bar{\mathcal{F}}$ be an optimal solution of the convex programming problem (6.3) is that

either 1. $\theta = \sqrt{\mu^T Z\mu}$ and there exists an $\alpha > 0$ such that $x^* = \alpha Z\mu$,

or 2. $\theta > \sqrt{\mu^T Z\mu}$ and $x^* = \mathbf{0}$.

Proof: First we assume that x^* is a minimizer of $\hat{f}(x)$ over the set $\{x : Wx = \mathbf{0}\}$. This implies that $\theta \geq \sqrt{\mu^T Z\mu}$.

Case 1: Let $\theta = \sqrt{\mu^T Z\mu}$. For any feasible point $\bar{x}$, there exists $\bar{v} \in \mathbb{R}^n$ such that $\bar{x} = Z\bar{v}$. In Theorem 6.2.2 we showed that the minimum value of $\hat{f}(x)$ over $\{x : Wx = \mathbf{0}\}$ is zero. A sufficient condition for $\bar{v}$ to be a minimizer is that $\hat{f}(Z\bar{v}) = 0$. In (6.4) we see that $\hat{f}(Z\bar{v}) = 0$ if and only if $\mu^T Z\bar{v} > 0$ and $Z\bar{v}$ is a scalar multiple of $Z\mu$. Thus $Z\bar{v} = \alpha Z\mu$ for $\alpha > 0$.

Case 2: Let $\theta > \sqrt{\mu^T Z\mu}$. In Theorem 6.2.4, we showed that $x^* = \left(\frac{w^T(WV^{-1}W^T)^{-1}w}{\theta^2 - \mu^T Z\mu}\right)^{1/2} Z\mu + V^{-1}W^T(WV^{-1}W^T)^{-1}w = 0$ is the unique minimizer of $\hat{f}(x)$ over the set $\{x : Wx = \mathbf{0}\}$. Thus x^* must be equal to $\mathbf{0}$.

Now we assume that the two conditions hold and show that x^* is a minimizer of $\hat{f}(x)$ over the set $\{x : Wx = \mathbf{0}\}$.

Case 1: Let $\theta = \sqrt{\mu^T Z \mu}$ and $x^* = \alpha Z \mu$, where $\alpha > 0$. Since $\hat{f}(x^*) = 0$ and $\bar{x} = \mathbf{0}$ is a minimizer of $\hat{f}(x)$ over $\{x : Wx = \mathbf{0}\}$ by Theorem 6.2.2, we have that x^* is also a minimizer.

Case 2: Let $\theta > \sqrt{\mu^T Z \mu}$. In Theorem 6.2.4, we showed that

$$\bar{x} = \sqrt{\frac{w^T (W V^{-1} W^T)^{-1} w}{\theta^2 - \mu^T Z \mu}} Z \mu + V^{-1} W^T (W V^{-1} W^T)^{-1} w$$

is the unique minimizer of $\hat{f}(x)$ over the set $\{x : Wx = \mathbf{0}\}$. Thus $x^* = \mathbf{0}$ is the minimizer.□

6.3 The Inequality Constrained Problem

Given the necessary and sufficient conditions for the existence and optimality of a minimizer of the equality constrained problem, we now can extend the dual algorithm from Chapter 5 for solving quadratic programming problems to an algorithm for solving the inequality constrained problem

$$\begin{aligned} \text{minimize} \quad & -\mu^T x + \theta \sqrt{x^T V x} \\ \text{subject to} \quad & Cx \geq d, \end{aligned} \tag{6.7}$$

where $\|\mu\|_2 > 0$, $\mathcal{M}_2 = \{1, 2, \ldots, m_2\}$, and m_2 is the number of inequality constraints. We let $\mathcal{F}$ represent the feasible set $\{x : Cx \geq d\}$. In this section we assume that $\theta > \sqrt{\mu^T V^{-1} \mu}$; although this assumption is not true in general, it makes the derivation of our algorithm for solving (6.7) simpler. We discuss the extensions to the more general case in §6.3.4. Given that $\theta > \sqrt{\mu^T V^{-1} \mu}$, if the zero vector is an element of $\mathcal{F}$, then by Theorem 6.2.6 the point $x^* = \mathbf{0}$ is the optimal solution to (6.7). Thus we assume for §6.3 that the point $x = \mathbf{0}$ does not satisfy $Cx \geq d$.

6.3.1 The Optimality Conditions

Although the feasible set for (6.2) is closed and bounded, the algorithm we have developed to solve this problem solves a series of subproblems that have a subset of the constraints of (6.2). Since these subproblems may not have bounded feasible regions and since the confidence interval objective function is not strictly convex, we must prove that an optimal solution exists for any subproblem of (6.7).

Theorem 6.3.1 Let $\mathcal{A} \subseteq \mathcal{M}_2$. Then there exists an $x^* \in \{x : C_{\mathcal{A}:}x \geq d_{\mathcal{A}}\}$ such that $\hat{f}(x^*) \leq \hat{f}(x)$ for all $x \in \{x : C_{\mathcal{A}:}x \geq d_{\mathcal{A}}\}$.

Proof: We know that $\hat{f}(x)$ is bounded below by zero since $\hat{x} = \mathbf{0}$ is the unique unconstrained minimizer of $\hat{f}(x)$. Let $\tilde{x} \in \{x : C_{\mathcal{A}:}x \geq d_{\mathcal{A}}\}$, and let $\tilde{\mathcal{F}} = \{x : C_{\mathcal{A}:}x \geq d_{\mathcal{A}},\ \hat{f}(x) \leq \hat{f}(\tilde{x})\}$. Given some $v \in \mathbb{R}^n$ such that $\|v\|_2 > 0$ and an $\alpha > 0$, we know that $\hat{f}(v) > 0$ and that $\hat{f}(\alpha v) = \alpha \hat{f}(v)$. This implies that there exists a finite $\alpha_v > 0$ such that $\hat{f}(\alpha_v v) = \hat{f}(\tilde{x})$. Thus $\tilde{\mathcal{F}}$ is closed and bounded, and there must exist an $x^* \in \tilde{\mathcal{F}}$ such that $\hat{f}(x^*) \leq \hat{f}(x)$ for all $x \in \tilde{\mathcal{F}}$. But such an x^* is also a minimizer of $\hat{f}(x)$ over $\{x : C_{\mathcal{A}:}x \geq d_{\mathcal{A}}\}$.□

As in Chapter 5, the necessary and sufficient conditions for optimality of (6.7) involve the Lagrangian function; the *Lagrangian function* of (6.7) is

$$\mathcal{L}(x, \tau) = \hat{f}(x) - \tau^T(Cx - d),$$

where $\tau \in \mathbb{R}^{m_2}$. The vector τ is called the vector of *Lagrange multipliers* or *dual variables*. Differentiating with respect to x results in the function

$$\nabla_x \mathcal{L}(x, \tau) = -\mu + \frac{\theta}{\sqrt{x^T V x}} V x - C^T \tau.$$

The following theorem is similar to Theorem 5.2.1; the inequality $\tau \geq \mathbf{0}$ means that every component of the vector τ is nonnegative.

Theorem 6.3.2 In problem (6.7), if x^* is a minimizer of $\hat{f}(x)$ over the feasible set $\mathcal{F}$, then there exists a vector τ^* of Lagrange multipliers such that

- $\nabla_x \mathcal{L}(x^*, \tau^*) = \mathbf{0}$,
- $\tau^* \geq \mathbf{0}$,
- $\tau^{*T}(Cx^* - d) = 0$, and
- $Z^T \nabla^2 \hat{f}(x^*) Z$ is positive semi-definite,

where Z is a nullspace matrix for the active constraints at x^*.

Since the matrix

$$\nabla^2 \hat{f}(x^*) = \frac{\theta}{\sqrt{x^{*T} V x^*}} \left(V - \frac{V x^* x^{*T} V}{x^{*T} V x^*} \right)$$

is positive semi-definite, the condition that $Z^T \nabla^2 \hat{f}(x^*) Z$ be positive semi-definite is satisfied by any matrix Z with n rows.

Because (6.7) has a convex objective function, the necessary conditions for optimality are also sufficient. The following theorem is similar to Theorem 5.2.2.

Theorem 6.3.3 The sufficient conditions for $x^* \in \mathcal{F}$ to be an optimal solution of the convex programming problem (6.7) is that there exists a vector τ^* such that

- $\nabla_x \mathcal{L}(x^*, \tau^*) = \mathbf{0}$,
- $\tau^* \geq \mathbf{0}$, and
- $\tau^{*T}(Cx^* - d) = 0$.

Proof: Let $\bar{x}$ be any other point satisfying the constraints of (6.7). Since $\hat{f}(x)$ is convex, and the point $\mathbf{0} \notin \mathcal{F}$, we have $\hat{f}(\bar{x}) \geq \hat{f}(x^*) + (\bar{x} - x^*)^T \nabla \hat{f}(x^*)$ [16]. Then

$$\begin{aligned} \hat{f}(\bar{x}) &\geq \hat{f}(\bar{x}) - \tau^{*T}(C\bar{x} - d) \\ &\geq \hat{f}(x^*) + (\bar{x} - x^*)^T \nabla \hat{f}(x^*) - (\bar{x} - x^*)^T C^T \tau^* \\ &= \hat{f}(x^*) + (\bar{x} - x^*)^T \nabla_x \mathcal{L}(x^*, \tau^*) \\ &= \hat{f}(x^*). \square \end{aligned}$$

As in Chapter 5, the necessary and sufficient conditions can be divided into the following three categories:

Primal feasibility: $Cx \geq d$,

Dual feasibility: $-\mu + \frac{\theta}{\sqrt{x^T V x}} V x - C^T \tau = \mathbf{0}$, $\tau \geq \mathbf{0}$, and

Complementary slackness: $\tau^T (Cx - d) = 0$.

The method we have chosen to solve the inequality constrained problem (6.7) is a dual method; at every iteration, the dual feasibility conditions and the complementary slackness conditions are satisfied. We have chosen this method because it allows for a quick "restart" after a new constraint is added. The optimal solution to the parent problem or to the current problem before the addition of a new valid inequality satisfies dual feasibility and complementary slackness, so no procedure to regain feasibility is necessary.

Our dual algorithm is based on the nonlinear programming subproblem $\mathcal{D}(\mathcal{A})$, which is

$$\begin{aligned} \text{minimize} \quad & \hat{f}(x) = -\mu^T x + \theta \sqrt{x^T V x} \\ \text{subject to} \quad & C_{\mathcal{A}\cdot} x \geq d_{\mathcal{A}}, \\ & x \in \mathbb{R}^n, \end{aligned}$$

where $\mathcal{A} \subseteq \mathcal{M}_2$. If the feasible set of this subproblem of (6.7) is not empty and does not contain the point $x = \mathbf{0}$, then there exists a solution $\bar{x}$ and a vector of associated Lagrange multipliers $\bar{\tau}$ such that optimality conditions for subproblem $\mathcal{D}(\mathcal{A})$ are satisfied. Given the working set $\emptyset \neq \bar{\mathcal{A}} \subseteq \mathcal{A}$ such that $C_{\bar{\mathcal{A}}:}\bar{x} = d_{\bar{\mathcal{A}}}$, $\bar{\tau}_{\mathcal{M}_2 \setminus \bar{\mathcal{A}}} = \mathbf{0}$, $\|d_{\bar{\mathcal{A}}}\|_2 > 0$, and the rows of $C_{\bar{\mathcal{A}}:}$ are linearly independent, the ordered list $(\bar{x}, \bar{\mathcal{A}})$ is called an *S-list*. Not only is $\bar{x}$ is an optimal solution to the subproblem $\mathcal{D}(\bar{\mathcal{A}})$, but $\bar{x}$ and $\bar{\tau}$ are dual feasible and satisfy complementary slackness for (6.7).

We prove the following theorem in the next section.

> **Theorem 6.3.4** If the feasible set of the subproblem $\mathcal{D}(\mathcal{A})$ is not empty and does not contain the point $x = \mathbf{0}$, then any optimal solution $\bar{x}$ to $\mathcal{D}(\mathcal{A})$ has an associated S-list.

At each step in the algorithm, we start with an S-list $(x^*, \mathcal{A})$ such that $\mathcal{A} \neq \emptyset$ and $\|d_{\mathcal{A}}\|_2 > 0$. If x^* is feasible for (6.7), then x^* is an optimal solution to (6.7). Otherwise, there is some constraint violated by x^*. Selecting one such violated constraint, a new S-list $(\bar{x}, \bar{\mathcal{A}})$ is determined such that the objective function value of the new iterate, $\hat{f}(\bar{x})$, is strictly greater than $\hat{f}(x^*)$. Furthermore, the working set $\bar{\mathcal{A}}$ of the new S-list is a subset of $\mathcal{A}$ except that the selected constraint violated by x^* is satisfied at equality by $\bar{x}$ and its index is a member of $\bar{\mathcal{A}}$. Since the objective function value increases at each iteration and there are finitely many subsets $\mathcal{A}$ of $\mathcal{M}_2$, the algorithm will terminate with the optimal solution after a finite number of iterations.

6.3.2 The Dual Algorithm

We first prove the theorem introduced in the last section.

Theorem 6.3.4 If the feasible set of the subproblem $\mathcal{D}(\mathcal{A})$ is not empty and does not contain the point $x = \mathbf{0}$, then any optimal solution $\bar{x}$ to $\mathcal{D}(\mathcal{A})$ has an associated S-list.

Proof: The feasible set of $\mathcal{D}(\mathcal{A})$ does not contain $\mathbf{0}$, thus $\|d_{\mathcal{A}}\|_2 > 0$. Since $\bar{x}$ is the optimal solution to the nonlinear programming problem $\mathcal{D}(\mathcal{A})$, there exists a vector of associated Lagrange multipliers $\bar{\tau}$ such that the dual feasibility and complementarity slackness conditions are satisfied. Let $\bar{\mathcal{A}} = \{i \in \mathcal{A} : C_i\bar{x} = d_i,\ \bar{\tau}_i > 0\}$. If the rows of $C_{\bar{\mathcal{A}}:}$ are linearly independent, then $(\bar{x}, \bar{\mathcal{A}})$ is an S-list. Otherwise, we need to construct a subset of $\bar{\mathcal{A}}$ that, along with $\bar{x}$, is an S-list such that $\bar{x}$ is feasible for this subproblem. If the rows of $C_{\bar{\mathcal{A}}:}$ are not linearly independent, then there exists an $i \in \bar{\mathcal{A}}$ such that $C_{i:}$ is a linear combination of the rows of $C_{\bar{\mathcal{A}}\setminus\{i\}:}$. This implies that there exists a $\Delta\tau_{\bar{\mathcal{A}}\setminus\{i\}} \in \mathbb{R}^{|\bar{\mathcal{A}}|-1}$ such that $C^T_{\bar{\mathcal{A}}\setminus\{i\}:}\Delta\tau_{\bar{\mathcal{A}}\setminus\{i\}} = -\bar{\tau}_i C^T_{i:}$. Let $\Delta\tau_i = \bar{\tau}_i$ and $\varpi = \min\{\bar{\tau}_k/\Delta\tau_k : k \in \bar{\mathcal{A}},\ \Delta\tau_k > 0\}$.

Clearly $\varpi > 0$ and there is at least one dual variable $\bar{\tau}_k,\ k \in \bar{\mathcal{A}}$, such that $\bar{\tau}_k - \varpi\Delta\tau_k = 0$. Since

$$\begin{aligned}\frac{\theta}{\sqrt{\bar{x}^T V \bar{x}}}V\bar{x} - C^T_{\bar{\mathcal{A}}:}[\bar{\tau}_{\bar{\mathcal{A}}} - \varpi\Delta\tau_{\bar{\mathcal{A}}}] &= \frac{\theta}{\sqrt{\bar{x}^T V \bar{x}}}V\bar{x} - C^T_{\bar{\mathcal{A}}:}\bar{\tau}_{\bar{\mathcal{A}}} + \varpi C^T_{\bar{\mathcal{A}}:}\Delta\tau_{\bar{\mathcal{A}}} \\ &= \mathbf{0} + \varpi[\Delta\tau_i C^T_{i:} - \bar{\tau}_i C^T_{i:}] = \mathbf{0},\end{aligned}$$

dual feasibility is maintained after updating $\bar{\tau}_{\bar{\mathcal{A}}} \leftarrow \bar{\tau}_{\bar{\mathcal{A}}} - \varpi\Delta\tau_{\bar{\mathcal{A}}}$. Dropping from $\bar{\mathcal{A}}$ the indices of all constraints for which the corresponding dual variable is now zero, we are left with $\bar{x}$ and $\bar{\tau}_{\bar{\mathcal{A}}}$ that satisfy primal feasibility, dual feasibility, and complementary slackness.

This process can be repeated until the rows of $C_{\bar{\mathcal{A}}:}$ are linearly independent, resulting in the S-list $(\bar{x}, \bar{\mathcal{A}})$.□

To start the dual algorithm for solving (6.7), an S-list for (6.7) must be known. Since the point $x = \mathbf{0}$ is not feasible, there must be at least one inequality, $C_{k:}x \geq d_k$, that is violated by $x = \mathbf{0}$. Because $\theta > \sqrt{\mu^T V^{-1} \mu}$, there is a minimizer, x^*, of $\hat{f}(x)$ over the set $\{x : C_{k:}x = d_k\}$. Thus $(x^*, \{k\})$ is an S-list for (6.7). Within the branch-and-bound tree, if the optimal solution of the parent node is not $x^* = \mathbf{0}$, then the optimal S-list of the parent of that current node is a valid S-list. Moreover, an optimal S-list to the same problem before the addition of a violated cutting plane is a valid S-list.

The process of determining the optimal solution to a subproblem of (6.7) is called an *iteration*. In each iteration, the algorithm starts with an S-list $(x^*, \mathcal{A})$ and its corresponding Lagrange multipliers τ^*. If x^* is primal feasible for (6.7), then x^* is the optimal solution to (6.7). Otherwise, we can determine a violated constraint $C_{k:}x \geq d_k$ where $k \in \mathcal{M}_2 \backslash \mathcal{A}$. At the start of the iteration, the dual variable ν associated with this violated constraint is such that $\nu^* = 0$.

In each subiteration, a primal step towards the hyperplane $C_{k:}x = d_k$ in the nullspace of the matrix $C_{\mathcal{A}:}$ is determined. Corresponding dual steps associated with the working set $\mathcal{A}$ and the "entering" row k are determined to keep the complementary slackness conditions and dual equality constraint

$$-\mu + \frac{\theta}{\sqrt{x^T V x}} V x - C^T \tau = \mathbf{0}$$

satisfied. A step size ϖ that maintains the nonnegativity of the dual variables is also determined; if possible, a full step is taken such that $C_{k:}x = d_k$ is satisfied at equality by the new primal iterate. A full step is not possible if $C_{k:}^T$ is in the range of $C_{\mathcal{A}:}^T$, and a full step may not be possible if a dual variable becomes negative. When a full step is not possible, an index is dropped from the working set $\mathcal{A}$. Unlike the dual algorithm in Chapter 5 for solving quadratic programming problems, the step directions for this problem are dependent on the step size. The primal and dual steps functions are,

respectively, $\Delta x(\varpi)$ and $\Delta\tau_{\mathcal{A}}(\varpi)$. For any step size $\varpi > 0$ such that $C_{k:}\bar{x}(\varpi) \leq d_k$, we insist that the nonlinear system

$$\begin{aligned} -\mu + \frac{\theta}{\sqrt{\bar{x}(\varpi)^T V \bar{x}(\varpi)}} V\bar{x}(\varpi) - C_{\mathcal{A}:}^T \bar{\tau}_{\mathcal{A}}(\varpi) &= (\nu^* + \varpi) C_{k:}^T \\ C_{\mathcal{A}:}\bar{x}(\varpi) &= d_{\mathcal{A}} \end{aligned} \tag{6.8}$$

be satisfied.

If $\|d_{\mathcal{A}}\|_2 > 0$, solving system (6.8) gives the step direction functions

$$\Delta\tau_{\mathcal{A}}(\varpi) = \left(1 - \frac{\sqrt{x^{*T} V x^*}}{\sqrt{\bar{x}(\varpi)^T V \bar{x}(\varpi)}}\right)\left(N(\nu^* C_{k:}^T + \mu) + \tau_{\mathcal{A}}^*\right) + \varpi N C_{k:}^T, \tag{6.9}$$

and

$$\Delta x(\varpi) = \frac{\sqrt{\bar{x}(\varpi)^T V \bar{x}(\varpi)}}{\theta}\left((\nu^* + \varpi) Z C_{k:}^T + Z\mu\right) - \frac{\sqrt{x^{*T} V x^*}}{\theta}(\nu^* Z C_{k:}^T + Z\mu), \tag{6.10}$$

where

$$\begin{aligned} N &= \left(C_{\mathcal{A}:} V^{-1} C_{\mathcal{A}:}^T\right)^{-1} C_{\mathcal{A}:} V^{-1}, \\ Z &= V^{-1} - V^{-1} C_{\mathcal{A}:}^T \left(C_{\mathcal{A}:} V^{-1} C_{\mathcal{A}:}^T\right)^{-1} C_{\mathcal{A}:} V^{-1}, \text{ and} \\ \bar{x}(\varpi)^T V \bar{x}(\varpi) &= x^{*T} V x^* \cdot \frac{\theta^2 - (\nu^* C_{k:}^T + \mu)^T Z (\nu^* C_{k:}^T + \mu)}{\theta^2 - \left((\nu^* + \varpi) C_{k:}^T + \mu\right)^T Z \left((\nu^* + \varpi) C_{k:}^T + \mu\right)}. \end{aligned}$$

Letting $g(\varpi) = \theta^2 - \left((\nu^* + \varpi) C_{k:}^T + \mu\right)^T Z \left((\nu^* + \varpi) C_{k:}^T + \mu\right)$, we see that the system (6.8) has a solution if and only if $x^{*T} V x^* = 0$ or $g(0)/g(\varpi) \geq 0$. In Theorems A.1.6 and A.1.7 and Corollary A.2.1, we show that at the beginning of every subiteration, $g(0) > 0$. In Theorems A.1.2 and A.2.1 of Appendix A, we show that $g(0) > 0$ implies that $g(\varpi) > 0$ for all $\varpi > 0$ such that $C_{k:}\bar{x}(\varpi) \leq d_k$.

We first present the formal algorithm and then prove that these step directions satisfy the system (6.8).

Algorithm 6.3.1

Step 1: **Initial Conditions:** Assume that some S-list $(x^*, \mathcal{A})$ is given, along with associated Lagrange multipliers $\tau^*_{\mathcal{A}}$.

Step 2: **Begin an iteration:** Set the Lagrange multiplier, ν^*, for the violated constraint to be 0. Choose $k \in \mathcal{M}_2 \backslash \mathcal{A}$ such that $C_{k:}x^* < d_k$ and goto Step 3. Otherwise, there are no violated constraints, and x^* is optimal for (6.7). STOP.

Step 3: **Begin a subiteration:**

Step 3a: Let $Z = V^{-1} - V^{-1}C^T_{\mathcal{A}:}(C_{\mathcal{A}:}V^{-1}C^T_{\mathcal{A}:})^{-1}C_{\mathcal{A}:}V^{-1}$. If $ZC^T_{k:} = \mathbf{0}$, let $\varpi_1 = \infty$. Else determine the primal and dual step functions $\Delta x(\varpi)$ and $\Delta\tau_{\mathcal{A}}(\varpi)$, and let ϖ_1 be such that $C_{k:}(x^* + \Delta x(\varpi_1)) = d_k$.

Step 3b: If $\varpi_1 = \infty$, goto Step 3c. If $\tau^*_{\mathcal{A}} - \Delta\tau_{\mathcal{A}}(\varpi)$ has no roots smaller than ϖ_1, let $\varpi_2 = \infty$. Else let ϖ_2 be the smallest root of $\tau^*_{\mathcal{A}} - \Delta\tau_{\mathcal{A}}(\varpi)$ and p_2 be the element of $\mathcal{A}$ that achieves this minimum. Goto Step 3d.

Step 3c: If $NC^T_{k:} \leq \mathbf{0}$, let $\varpi_2 = \infty$. Else let $\varpi_2 = \min\{\tau_i/(N_{i:}C^T_{k:}) : i \in \mathcal{A}, N_{i:}C^T_{k:} > 0\}$ and p_2 be the element of $\mathcal{A}$ that achieves this minimum.

Step 3d: Set $\varpi = \min(\varpi_1, \varpi_2)$.

Step 4: **Update primal and dual vectors:**

Step 4a: If $\varpi = \infty$, then (6.7) is infeasible. STOP.

Step 4b: Set $x^* \leftarrow x^* + \Delta x(\varpi)$, $\tau^*_{\mathcal{A}} \leftarrow \tau^*_{\mathcal{A}} - \Delta\tau_{\mathcal{A}}(\varpi)$, and $\nu^* \leftarrow \nu^* + \varpi$.

Step 4c: **The subiteration is complete:**

If $\varpi < \varpi_1$, then set $\mathcal{A} \leftarrow \mathcal{A} \backslash \{k\}$. If $\mathcal{A} \neq \emptyset$ or $\|d_{\mathcal{A}}\|_2 > 0$, goto Step 3. Otherwise, this is the degenerate case. STOP. (Refer to §6.3.3).

Step 4d: **The iteration is complete:**

Set $\tau_k^* \leftarrow \nu^*$ and $\mathcal{A} \leftarrow \mathcal{A} \cup \{k\}$. Goto Step 2.

The algorithm may fail in the case that a degenerate working set is encountered; in §6.3.3, we discuss how to determine a valid S-list in this case. In Appendix A, we show how to determine the value of the step size variable ϖ_2, where ϖ_2 is the smallest positive root of $\bar{\tau}_i(\varpi) = \tau_i^* - \Delta\tau_i(\varpi)$ for all $i \in \mathcal{A}$. We also show that if $\|ZC_{k:}^T\|_2 > 0$, the value of ϖ_1, such that $C_{k:}(x^* + \Delta x(\varpi_1)) = d_k$, is strictly positive.

We now prove that the step direction functions (6.9) and (6.10) satisfy system (6.8), and then we prove that each iteration will terminate with a new S-list, with a degenerate working set $\mathcal{A}$, or with the conclusion that (6.7) is infeasible.

Theorem 6.3.5 Given that

$$-\mu + \frac{\theta}{\sqrt{x^{*T} V x^*}} V x^* - C_{\mathcal{A}:}^T \tau_{\mathcal{A}}^* = \nu^* C_{:k}^T \quad \text{and} \quad C_{\mathcal{A}:} x^* = d_{\mathcal{A}},$$

where $\|d_{\mathcal{A}}\|_2 > 0$, the step direction functions $\Delta\tau_{\mathcal{A}}(\varpi)$ and $\Delta x(\varpi)$ as given in (6.9) and (6.10), respectively, satisfy the system (6.8) for all $\varpi > 0$ such that $C_{k:}\bar{x}(\varpi) \leq d_k$.

Proof: In Theorems A.1.2 and A.2.1 of Appendix A, we show that

$$\theta^2 - \left((\nu^* + \varpi)C_{k:}^T + \mu\right)^T Z \left((\nu^* + \varpi)C_{k:}^T + \mu\right) > 0$$

for all $\varpi > 0$ such that $C_{k:}(x^* + \Delta x(\varpi)) \leq d_k$, so we know that $\sqrt{\bar{x}(\varpi)^T V \bar{x}(\varpi)}$ is in $\mathbb{R}^1$ and is strictly positive.

First, we show that $C_{\mathcal{A}:}\bar{x}(\varpi) = d_{\mathcal{A}}$ by proving that the primal step $\Delta x(\varpi)$ is in the nullspace of $C_{\mathcal{A}:}$. Since Z is a nullspace matrix for $C_{\mathcal{A}:}$,

$$\begin{aligned} C_{\mathcal{A}:}\Delta x(\varpi) &= \frac{\sqrt{\bar{x}(\varpi)^T V \bar{x}(\varpi)}}{\theta} C_{\mathcal{A}:}\left((\nu^* + \varpi) Z C_{k:}^T + Z\mu\right) \\ &\quad - \frac{\sqrt{x^{*T} V x^*}}{\theta} C_{\mathcal{A}:}(\nu^* Z C_{k:}^T + Z\mu) \\ &= \mathbf{0}. \end{aligned}$$

We now show that for all $\varpi > 0$ such that $C_{k:}(x^* + \Delta x(\varpi)) \leq d_k$,

$$-\mu + \frac{\theta}{\sqrt{\bar{x}(\varpi)^T V \bar{x}(\varpi)}} V\bar{x}(\varpi) - C_{\mathcal{A}:}^T \bar{\tau}_{\mathcal{A}}(\varpi) = (\nu^* + \varpi) C_{k:}^T.$$

Based on the step direction functions, we see that

$$\begin{aligned} C_{\mathcal{A}:}^T \bar{\tau}_{\mathcal{A}}(\varpi) &= (VZ - I)\left((\nu^* + \varpi) C_{k:}^T + \mu\right) \\ &\quad - \frac{\sqrt{x^{*T} V x^*}}{\sqrt{\bar{x}(\varpi)^T V \bar{x}(\varpi)}} (VZ - I)\left((\nu^* C_{k:}^T + \mu)\right) \\ &\quad + \frac{\sqrt{x^{*T} V x^*}}{\sqrt{\bar{x}(\varpi)^T V \bar{x}(\varpi)}} C_{\mathcal{A}:}^T \tau_{\mathcal{A}}^* \end{aligned}$$

and

$$\begin{aligned} \theta V\bar{x}(\varpi) &= \theta V x^* + \sqrt{\bar{x}(\varpi)^T V \bar{x}(\varpi)} VZ\left((\nu^* + \varpi) C_{k:}^T + \mu\right) \\ &\quad - \sqrt{x^{*T} V x^*} VZ\left(\nu^* C_{k:}^T + \mu\right). \end{aligned}$$

Thus,

$$\begin{aligned} &-\mu + \tfrac{\theta}{\sqrt{\bar{x}(\varpi)^T V \bar{x}(\varpi)}} V\bar{x}(\varpi) - C_{\mathcal{A}:}^T \bar{\tau}_{\mathcal{A}}(\varpi) \\ &\quad = -\mu + \tfrac{\theta}{\sqrt{\bar{x}(\varpi)^T \bar{x}(\varpi)}} V x^* + (\nu^* + \varpi) C_{k:}^T + \mu \\ &\qquad - \tfrac{\sqrt{x^{*T} V x^*}}{\sqrt{\bar{x}(\varpi)^T V \bar{x}(\varpi)}} \left(\nu^* C_{k:}^T + \mu + C_{\mathcal{A}:}^T \tau_{\mathcal{A}}^*\right) \\ &\quad = (\nu^* + \varpi) C_{k:}^T \\ &\qquad + \tfrac{\sqrt{x^{*T} V x^*}}{\sqrt{\bar{x}(\varpi)^T V \bar{x}(\varpi)}} \left(-\mu + \tfrac{\theta}{\sqrt{x^{*T} V x^*}} V x^* - C_{\mathcal{A}:}^T \tau_{\mathcal{A}}^* - \nu^* C_{k:}^T\right) \\ &\quad = (\nu^* + \varpi) C_{k:}^T, \end{aligned}$$

and we are done.□

The following three theorems prove that each iteration of Algorithm 6.3.1 will terminate either with the conclusion that (6.7) is infeasible, with an S-list such that the objective function value of the new iterate is strictly greater than the objective function value of the previous iterate, or with a degenerate working set $\mathcal{A}$.

Theorem 6.3.6 If $\varpi = \infty$ in Step 4a of the algorithm, then (6.7) is infeasible.

Proof: Since $\varpi_1 = \infty$, we know that $ZC_{k:}^T = \mathbf{0}$. Since $\varpi_1 = \infty$ and $\varpi_2 = \infty$, we have that $NC_{k:}^T \leq \mathbf{0}$.

Assume there exists an $\hat{x}$ such that $x^* + \hat{x}$ satisfies $C_{\mathcal{A}:}(x^* + \hat{x}) \geq d_{\mathcal{A}}$, and $C_{k:}(x^* + \hat{x}) \geq d_k$. In other words, $C_{\mathcal{A}:}\hat{x} \geq \mathbf{0}$ and $C_{k:}\hat{x} > 0$.

Given the existence of such a $\hat{x}$,

$$\begin{aligned} 0 &\geq C_{k:}N^T C_{\mathcal{A}:}\hat{x} \\ &= C_{k:}\hat{x} - C_{k:}ZV\hat{x} \\ &= C_{k:}\hat{x}, \end{aligned}$$

since $N^T C_{\mathcal{A}:}V^{-1} = V^{-1} - Z$. This contradicts the choice of $\hat{x}$, thus no such $\hat{x}$ exists and the nonlinear programming problem (6.7) is infeasible.□

Theorem 6.3.7 If (6.7) is not infeasible, then starting with an S-list $(x^*, \mathcal{A})$ and having chosen a constraint $C_{k:}x \geq d_k$ violated by x^*,the iteration will terminate with a degenerate working set $\mathcal{A}$ or a new S-list such that $C_{k:}x \geq d_k$ is satisfied at equality and k is in the new working set.

Proof: As long as (6.7) is feasible, at least one subiteration will occur. After each subiteration, there are updated dual variables ν^* and τ^*, a

primal variable x^*, and a working set $\mathcal{A}$ such that the dual feasibility constraint

$$-\mu + \frac{\theta}{\sqrt{x^{*T} V x^*}} V x^* - C_{\mathcal{A}:}^T \tau_{\mathcal{A}}^* - \nu^* C_{k:}^T = \mathbf{0}$$

and the nonnegativity of the dual variables $\tau_{\mathcal{A}}^*$ and ν^* are satisfied. The complementary slackness condition of every primal constraint, except possibly of the constraint $C_{k:}x \geq d_k$, is satisfied. If the iteration is not done, then the complementary slackness condition $\nu^*(C_{k:}x - d_k) = 0$ may not be satisfied since the full primal step was not taken. Thus the dual variable of some constraint previously in the working set $\mathcal{A}$ reached zero and was dropped from the appropriate working set. Since the working sets are of finite cardinality, only a finite number of subiterations can be taken before a degenerate working set is encountered or the iteration is complete.

If a degenerate working set is not encountered, then eventually a full primal step must be taken and $C_{k:}x^* \geq d_k$ will be satisfied at equality. Since a full primal step was taken, $\|ZC_{k:}^T\|_2 > 0$, and we know the rows of $C_{\mathcal{A}:}$ and the vector $C_{k:}$ are linearly independent. The complementary slackness condition of the new active constraint is satisfied so the updated $(x^*, \mathcal{A})$ is an S-list. □

Theorem 6.3.8 Let $(x^*, \mathcal{A})$ be an S-list for (6.7) and $C_{k:}x \geq d_k$ be a constraint of (6.7) such that $C_{k:}x^* < d_k$. If (6.7) is not infeasible, then, provided a degenerate working set is not encountered, after every subiteration the objective function value does not decrease and after the final subiteration, the objective function value strictly increases.

Proof: At the beginning of a subiteration, we have primal variables x^*, dual variables ν^* and τ^*, and the working set $\mathcal{A}$ such that the dual

feasibility constraint

$$-\mu + \frac{\theta}{\sqrt{x^{*T} V x^*}} V x^* - C_{\mathcal{A}:}^T \tau_{\mathcal{A}}^* - \nu^* C_{k:}^T = \mathbf{0}$$

is satisfied. Let Z represent the appropriate nullspace matrix of $C_{\mathcal{A}:}$; then $ZC_{\mathcal{A}:}^T = \mathbf{0}$. Let $\Delta x(\varpi)$ be the primal step direction calculated during the subiteration. Since $\theta^2 > (\nu^* C_{k:}^T + \mu)^T Z (\nu^* C_{k:}^T + \mu)$ we see that differentiating $\hat{f}(\bar{x}(\varpi))$ with respect to ϖ gives

$$\begin{aligned}
\hat{f}'(\bar{x}(\varpi)) &= \hat{f}'(x^* + \Delta x(\varpi)) \\
&= -qC_{k:}Z\mu \left(\theta^2 - \left((\nu^* + \varpi)C_{k:}^T + \mu\right)^T Z \left((\nu^* + \varpi)C_{k:}^T + \mu\right)\right) \\
&\quad + q\left((\nu^* + \varpi)C_{k:}ZC_{k:}^T + C_{k:}Z\mu\right)\left(\theta^2 - (\nu^* + \varpi)C_{k:}Z\mu - \mu^T Z\mu\right) \\
&= q(\nu^* + \varpi)\left((C_{k:}Z\mu)^2 + \theta^2 C_{k:}ZC_{k:}^T - C_{k:}ZC_{k:}^T\mu^T Z\mu\right) \\
&\geq q(\nu^* + \varpi)\left((C_{k:}Z\mu)^2 + C_{k:}ZC_{k:}^T(\nu^{*2}C_{k:}ZC_{k:}^T + 2\nu^* C_{k:}Z\mu)\right) \\
&= q(\nu^* + \varpi)\left((C_{k:}Z\mu) + \nu^* C_{k:}ZC_{k:}^T\right)^2 \\
&\geq 0,
\end{aligned}$$

where

$$q = \frac{\sqrt{\bar{x}(\varpi)^T V \bar{x}(\varpi)}}{\theta\left(\theta^2 - \left((\nu^* + \varpi)C_{k:}^T + \mu\right)^T Z \left((\nu^* + \varpi)C_{k:}^T + \mu\right)\right)} > 0.$$

The scalar q is strictly positive since $\|d_{\mathcal{A}}\|_2 > 0$ implies that $\|\bar{x}(\varpi)\|_V > 0$. Thus $\hat{f}(x^* + \Delta x(\varpi))$ is an increasing function on the interval $(0, \bar{\varpi})$. Since $\|ZC_{k:}^T\|_2 > 0$, $\varpi > 0$, and $q > 0$ in the final subiteration, $\hat{f}'(\bar{x}(\varpi))$ is strictly positive in the final iteration. □

The algorithms for updating the Cholesky factorization of $C_{\mathcal{A}:}V^{-1}C_{\mathcal{A}:}^T$ are the same as in Chapter 5.

6.3.3 Degenerate Case

If, after dropping an index from the working set, the new working set $\mathcal{A}$ is either empty or $d_{\mathcal{A}} = \mathbf{0}$, the algorithm as written above will terminate before the optimal solution is determined. After an index is dropped, we have primal variables x^* and dual variables $\tau^*_{\mathcal{A}}$ and $\nu^* > 0$. Since we assume that this is the first degenerate subiteration of this iteration, we can assume that d_i, where i is the index that was dropped from $\mathcal{A}$, is nonzero, and thus $x^* \neq \mathbf{0}$. Moreover, the variables x^*, $\tau^*_{\mathcal{A}}$, and ν^* are such that

$$\begin{aligned} -\mu + \tfrac{\theta}{\sqrt{x^{*T} V x^*}} V x^* - C^T_{\mathcal{A}:} \tau^*_{\mathcal{A}} &= \nu^* C^T_{k:} \\ C_{\mathcal{A}:} x^* &= \mathbf{0} \\ C_{k:} x^* &< d_k. \end{aligned}$$

This primal iterate evaluates to

$$\begin{aligned} \hat{f}(x^*) &= -\mu^T x^* + \theta \|x^*\|_V \\ &= \tau^{*T}_{\mathcal{A}} C_{\mathcal{A}:} x^* + \nu^* C_{k:} x^* \\ &= \nu^* C_{k:} x^* \\ &< \nu^* d_k. \end{aligned}$$

Assuming that $\theta^2 > \mu^T V^{-1} \mu$, we know that $\hat{f}(x^*) > 0$. Since $\hat{f}(x^*) = \nu^* C_{k:} x^*$, we know that $C_{k:} x^* > 0$. Setting the primal step to be $\Delta x = x^*$, the dual step to be $\Delta \tau_{\mathcal{A}} = \mathbf{0}$, and the step size to be $\varpi = (d_k - C_{k:} x^*)/C_{k:} x^*$, we see that the dual feasibility and complementary slackness conditions are satisfied, since $C_{k:}(x^* + \varpi \Delta x) = d_k$. Moreover, since the step size ϖ is strictly positive,

$$\begin{aligned} \hat{f}(x^* + \varpi \Delta x) &= \hat{f}((1 + \varpi) x^*) \\ &= (1 + \varpi) \hat{f}(x^*) \\ &> \hat{f}(x^*). \end{aligned}$$

Clearly, we satisfy the conditions of a new iterate, since $d_k > 0$ and $(x^* + \varpi \Delta x, \mathcal{A} \cup \{k\})$ is an S-list. Based on the theorems from §6.2, we know that $\theta^2 > \mu^T \bar{Z} \mu$, where $\bar{Z}$ is the nullspace matrix for $C_{\bar{\mathcal{A}}:}$ and $\bar{\mathcal{A}} = \mathcal{A} \cup \{k\}$.

6.3.4 Assumption that $\theta^2 > \mu^T V^{-1} \mu$

In §6.3.2 and §6.3.3, we made the assumption that $\theta^2 > \mu^T V^{-1} \mu$. Although this is not a good assumption in general, we use this assumption in two places. The first time is in the process of determining a starting S-list. We conjecture that if $x^* = \mathbf{0}$ is not the optimal solution to (6.7), it would not be difficult to determine an S-list $(x^*, \mathcal{A})$ such that x^* is the optimal solution to $\mathcal{D}(\mathcal{A})$. The only other place we use the assumption that $\theta^2 > \mu^T V^{-1} \mu$ is in the degenerate case to ensure that the function evaluation of the iterates are nonnegative.

Chapter 7

Computational Results

7.1 The Problems

In this chapter we report computational results on two problems. The first problem, called *prob.52*, has 11 rows and 62 columns; the variables associated with the columns are partitioned into a set of 52 asset variables and a set of 10 penalty variables. The second problem, called *prob.500*, has 11 rows and 510 columns; the variables associated with the columns are partitioned into a set of 500 asset variables and a set of 10 penalty variables. The computational tests for *prob.52* were run on a 200MHz Sun Ultra-2 with 262M of memory. The computational tests for *prob.500* were run on a 143MHz Sun Ultra-1 with 131M of memory. In both cases, the code was compiled using `cc` and the `-xO4` optimizer flag. Furthermore, wherever CPLEX [8] is used, it is version 4.0.9 for the Ultra. For all problems, the process time is measured in seconds. All tests are run for the quadratic objective function unless otherwise stated.

In each section, computational tests are performed for various values of κ, where κ is the factor that limits diversification, *i.e.*, $\sum_{j \in \mathcal{N}} y_j \leq \kappa$. The optimal solution is denoted (x^*, y^*).

7.2 Implicit Branch-and-Bound

The first set of computational results compare the standard branch-and-bound algorithm to the implicit branch-and-bound algorithm. The code used to generate these results uses the CPLEX QP solver to solve the quadratic programming problems at the nodes with no warm start. The only difference between the two codes is that the

binary variables are included explicitly in the standard branch-and-bound code and the binary variables are removed in the implicit branch-and-bound code as discussed in Chapter 3. Tables 7.1 and 7.2 give the timings for various values of κ for *prob.52* and *prob.500*, respectively.

prob.52			Standard B&B		Implicit B&B	
κ	$f(x^*, y^*)$	Nodes†	Time	Time/Node	Time	Time/Node
31	-4.915392e-02	28825	946.49	3.31e-02	198.13	6.87e-03
34	-6.743432e-02	8928	275.07	3.08e-02	56.25	6.31e-03
37	-7.965479e-02	1832	56.41	3.08e-02	10.77	5.88e-03
40	-8.658508e-02	407	11.69	2.87e-02	1.98	4.86e-03
43	-8.951690e-02	256	7.12	2.78e-02	1.17	4.57e-03
46	-9.157293e-02	103	3.08	2.99e-02	0.42	4.08e-03

† Node ≡ subproblem in search tree

Table 7.1 Standard Branch-and-Bound vs. Implicit Branch-and-Bound

prob.500			Standard B&B		Implicit B&B	
κ	$f(x^*, y^*)$	Nodes†	Time	Time/Node	Time	Time/Node
127	-4.131053e-01	3635	1792.61	4.93e-01	168.05	4.62e-02
130	-4.132057e-01	852	436.49	5.12e-01	38.44	4.51e-02
133	-4.132662e-01	301	157.89	5.25e-01	13.35	4.44e-02
136	-4.132888e-01	282	148.13	5.25e-01	12.45	4.41e-02

† Node ≡ subproblem in search tree

Table 7.2 Standard Branch-and-Bound vs. Implicit Branch-and-Bound

As expected, the implicit branch-and-bound algorithm spends much less time per node than does the standard branch-and-bound algorithm. This is because a quadratic programming subproblem in the implicit branch-and-bound tree has many fewer variables and constraints than does the corresponding subproblem in the standard branch-and-bound tree.

7.3 Branching Variable Selection Strategy

The computational tests in this section use our implicit branch-and-bound algorithm and our extended Goldfarb-Idnani algorithm to solve the quadratic programming subproblems at the nodes of the search tree.

The first branching variable selection strategy is based on the desire to find a good upper bound and then to prune quickly any nodes remaining in the branch-and-bound tree. We expect that the continuous variables that are nonzero for the optimal solution will contain a subset of the continuous variables that are nonzero in the optimal solution of the implicit relaxation of the root node. The first strategy is to branch recursively on the continuous variable with the smallest absolute value and then to work on the down-branch until a feasible solution for the mixed-integer quadratic programming problem is found. Since the goal is no longer to find a good upper bound, we continue to branch through the tree, but now branch on the continuous variable with the largest absolute value. This strategy is denoted "Change Criteria".

Unfortunately, this strategy could create a much larger tree than necessary, since we have made many choices based on the fact that an upper bound was not known. To rid ourselves of these poor choices, we instead throw away any of the tree that has been created and start the branch-and-bound tree again. This leads to the second strategy which is to branch recursively on the continuous variable with the smallest absolute value and to work on the down-branch until a feasible solution for the mixed-integer quadratic programming problem is found. We then restart the tree from the root, this time recursively branching on the continuous variable with the largest absolute value. This branching variable selection strategy is denoted "Restart Tree". Although this strategy could be interpreted as an upper bound heuristic, we prefer to think of it as a strategy for choosing good branching variables based on the current value of the

upper bound. Tables 7.3 and 7.4 give the timings for various values of κ for *prob.52* and *prob.500*, respectively.

prob.52		Change Criteria		Restart Tree	
κ	$f(x^*, y^*)$	Nodes†	Time	Nodes†	Time
25	2.801919e-02	850757	495.13	345857	248.55
28	-2.044631e-02	312219	171.92	224030	143.08
29	-3.095907e-02	236789	144.10	69701	42.68
31	-4.915392e-02	106527	60.58	28827	18.83
34	-6.743432e-02	32115	18.20	8930	5.32
37	-7.965479e-02	7059	3.76	1835	0.96
40	-8.658508e-02	1993	1.00	410	0.20
43	-8.951690e-02	1083	0.57	259	0.14
46	-9.157293e-02	541	0.29	106	0.08

† Node ≡ subproblem in search tree

Table 7.3 Branching Variable Selection Strategy

prob.500		Change Criteria		Restart Tree	
κ	$f(x^*, y^*)$	Nodes†	Time	Nodes†	Time
127	-4.131053e-01	822783	2489.36	3995	20.89
130	-4.132057e-01	213167	642.25	1212	6.58
133	-4.132662e-01	103431	324.00	661	3.95
136	-4.132888e-01	99009	304.04	642	3.74

† Node ≡ subproblem in search tree

Table 7.4 Branching Variable Selection Strategy

As expected, the strategy that restarts the tree after finding an upper bound performs much better than the strategy that changes the branching variable selection criteria without restarting the tree.

We implemented both a strong branching routine and a one-sided strong branching routine that examines only the possible down-branches. For both of these codes,

however, the tree size rarely was smaller than if we had used the "Restart Tree", strategy and the time spent per node increased dramatically.

7.4 Upper Bound Heuristics

We have built other techniques for determining a good upper bound into the branching variable selection strategy that restarts the tree. The first technique, discussed in §4.2, involves linearizing the objective function and solving the resulting mixed-integer linear programming problem (MILP) using the CPLEX [8] mixed-integer linear programming solver. The set of continuous variables that are nonzero in the optimal solution to the implicit relaxation of the root node of the branch-and-bound tree is augmented by the set of continuous variables that are nonzero in the optimal solution to the MILP; the strategy is to perform depth-first search (DFS) on this reduced set until a feasible solution is found. As in the second branching variable selection strategy discussed in §7.3 above, the tree is restarted after a feasible solution is encountered in the tree.

The second upper bound heuristic we have implemented is the same as the first, but this heuristic does not stop immediately upon encountering a feasible solution in the tree. Instead, it continues through the tree for an extra predetermined number of nodes in hopes of finding a better upper bound. Once the number of extra nodes allotted have been examined, the tree is restarted as before. Because the optimal solution may be the first feasible solution encountered, this technique may force the heuristic to enumerate more nodes than necessary. However, the upper bound determined by this "Subtree" approach is at least as good as determined by the "Reduced DFS" approach.

Comparing Tables 7.3 and 7.5 we see that for *prob.52* calling CPLEX on the MILP and doing depth-first search on the reduced set of nodes until an upper bound

is encountered is not a large improvement over solely doing depth-first search until an upper bound is encountered. However, in comparing Tables 7.4 and 7.6 we see that for *prob.500*, doing depth-first search on the reduced set works much better. This is because in *prob.52* there are 48 of the 52 assets nonzero at the root node, while in *prob.500* there are 139 of the 500 assets nonzero at the root node, so working on the reduced set makes a difference for *prob.500*. By examining Tables 7.5 and 7.6, we see that continuing the depth-first search though a small subtree before restarting after encountering the first upper bound is also a good idea in practice. Tables 7.7 and 7.8 give the optimality gap

$$\frac{|f(x^{IP}, y^{IP}) - f(x^*, y^*)|}{|f(x^*, y^*)|} \tag{7.1}$$

for each of the two upper bound heuristics, where (x^*, y^*) is the optimal solution to the problem and (x^{IP}, y^{IP}) is the upper bound determined by the heuristic.

prob.52		Reduced DFS		"Subtree"	
κ	$f(x^*, y^*)$	Nodes†	Time	Nodes†	Time
19	2.220023e-01	499564	405.87	148830	126.30
22	9.498409e-02	925197	702.95	617431	477.89
23	7.080893e-02	953382	718.72	354545	278.01
25	2.801920e-02	345854	247.40	340935	251.67
28	-2.044631e-02	224027	147.30	93734	62.92
31	-4.915392e-02	28824	18.26	28861	18.39
34	-6.743432e-02	8927	5.30	7750	4.71
37	-7.965479e-02	1832	1.03	1551	0.90
40	-8.658508e-02	407	0.20	426	0.25
43	-8.951690e-02	256	0.18	231	0.17
46	-9.157293e-02	103	0.10	110	0.10

† Node ≡ subproblem in search tree

Table 7.5 Upper Bound Heuristics: Node Count

prob.500		Reduced DFS		"Subtree"	
κ	$f(x^*, y^*)$	Nodes†	Time	Nodes†	Time
118	-4.124560e-01	764854	4988.49	511015	2993.57
119	-4.125602e-01	418799	2403.87	286972	1816.63
121	-4.127616e-01	102495	558.08	102578	623.10
122	-4.128442e-01	80246	419.16	41481	215.51
124	-4.129730e-01	14114	70.25	14191	69.46
127	-4.131053e-01	3657	19.02	3500	18.08
130	-4.132057e-01	874	5.52	809	5.08
133	-4.132662e-01	323	3.13	382	3.31
136	-4.132888e-01	304	3.05	357	3.22

† Node $\equiv$ subproblem in search tree

Table 7.6 Upper Bound Heuristics: Node Count

prob.52		Reduced DFS		"Subtree"	
κ	$f(x^*, y^*)$	$f(x^{IP}, y^{IP})$†	Gap††	$f(x^{IP}, y^{IP})$†	Gap††
19	2.220023e-01	2.579059e-01	1.62e-01	2.237245e-01	7.76e-03
21	1.289750e-01	1.461169e-01	1.31e-01	1.354677e-01	5.03e-02
22	9.498409e-02	1.198860e-01	2.62e-01	1.092015e-01	1.50e-01
23	7.080893e-02	8.601420e-02	2.15e-01	7.123676e-02	6.04e-03
25	2.801919e-02	2.953775e-02	5.42e-02	2.935869e-02	4.78e-02
28	-2.044631e-02	-1.280505e-02	3.74e-01	-2.044631e-02	0.00
31	-4.915392e-02	-4.915392e-02	0.00	-4.915392e-02	0.00
34	-6.743432e-02	-6.689961e-02	7.93e-03	-6.743432e-02	0.00
37	-7.965479e-02	-7.912163e-02	6.69e-03	-7.965479e-02	0.00
40	-8.645956e-02	-8.658508e-02	1.45e-03	-8.645956e-02	0.00
43	-8.951690e-02	-8.931161e-02	2.29e-03	-8.951080e-02	6.81e-05
46	-9.157293e-02	-9.157293e-02	0.00	-9.157293e-02	0.00

† Upper bound determined by heuristic

†† Gap is relative difference between upper bound heuristic objective function value and optimal solution; see (7.1).

Table 7.7 Upper Bound Heuristics: Optimality Gap

prob.500		Reduced DFS		"Subtree"	
κ	$f(x^*, y^*)$	$f(x^{IP}, y^{IP})^{\dagger}$	Gap††	$f(x^{IP}, y^{IP})^{\dagger}$	Gap††
118	-4.124560e-01	-4.124007e-01	1.34-e04	-4.124383e-01	4.29e-05
119	-4.125602e-01	-4.125217e-01	9.33e-05	-4.125543e-01	1.43e-05
121	-4.127616e-01	-4.127410e-01	4.99e-05	-4.127410e-01	4.99e-05
122	-4.128442e-01	-4.128008e-01	1.05e-04	-4.128442e-01	0.00
124	-4.129730e-01	-4.129730e-01	0.00	-4.129730e-01	0.00
127	-4.131053e-01	-4.131016e-01	8.96e-06	-4.131053e-01	0.00
129	-4.131739e-01	-4.131665e-01	1.79e-05	-4.131739e-01	0.00
130	-4.132057e-01	-4.131991e-01	1.60e-05	-4.132057e-01	0.00
133	-4.132662e-01	-4.132662e-01	0.00	-4.132662e-01	0.00
136	-4.132888e-01	-4.132888e-01	0.00	-4.132888e-01	0.00

† Upper bound determined by heuristic

†† Gap is relative difference between upper bound heuristic objective function value and optimal solution; see (7.1).

Table 7.8 Upper Bound Heuristics: Optimality Gap

7.5 Node Selection Strategy

If the upper bound heuristic does not return the optimal solution, the optimal solution may be "pushed" to be one of the last ones evaluated when the tree is restarted and we work on the down-branch first. Choosing the up-branch instead of the down-branch to work on first may alleviate this effect. If the upper bound determined by the upper bound heuristic is the optimal solution, exactly the same number of nodes will be enumerated by choosing the up-branch first as by choosing the down-branch first. If the upper bound found by the upper bound heuristic is not the optimal solution, fewer nodes may be enumerated. From Tables 7.9 and 7.10 we see that while the number of nodes is either the same or fewer in most cases, the time per node is increased. As conjectured in §4.4, the list of open nodes becomes much larger when we examine the

up-branch first. Since we update the approximating constraints at the up-branches and store the updated approximating constraint of the parent for the down-branches, the set of inequalities that are valid for at least one node on the list becomes very large. This slows the code tremendously as the set of stored inequalities grows.

prob.52		Down-Branch First		Up-Branch First	
κ	$f(x^*, y^*)$	Nodes†	Time	Nodes†	Time
17	3.943248e-01	236885	222.51	180109	186.21
18	2.948757e-01	223117	182.01	140347	126.63
19	2.220023e-01	148830	126.30	139528	122.31
21	1.289750e-01	324343	250.27	222837	182.49
22	9.498409e-02	617431	477.89	251697	214.15
25	2.801920e-02	340935	251.67	304945	255.37
28	-2.044631e-02	93734	62.92	93734	70.18
31	-4.915392e-02	28861	18.39	28861	21.59
34	-6.743432e-02	7750	4.71	7750	5.56
37	-7.965479e-02	1551	0.90	1551	1.03
40	-8.658508e-02	426	0.25	408	0.31
43	-8.951690e-02	231	0.17	231	0.18
46	-9.157293e-02	110	0.10	110	0.12

† Node $\equiv$ subproblem in search tree

Table 7.9 Node Selection Strategy

7.6 Valid Inequalities

7.6.1 Symmetric Dominance Inequalities

The code found 32 symmetric dominance cuts for *prob.52* and 15335 for *prob.500*. The computational tests in this section use the implicit branch-and-bound method and the extended Goldfarb-Idnani method to solve the quadratic subproblems at the

prob.500		Down-Branch First		Up-Branch First	
κ	$f(x^*, y^*)$	Nodes†	Time	Nodes†	Time
121	-4.127616e-01	102578	623.10	76754	3292.12
122	-4.128442e-01	41481	215.51	41481	1913.82
123	-4.129134e-01	23996	120.65	23996	1031.08
124	-4.129730e-01	14191	69.46	14191	618.26
125	-4.130203e-01	9266	45.64	9188	399.08
126	-4.130662e-01	5709	28.33	5575	258.63
127	-4.131053e-01	3500	18.08	3500	148.42
130	-4.132057e-01	809	5.08	809	28.68
133	-4.132662e-01	382	3.31	382	10.42
136	-4.132888e-01	357	3.22	357	8.79

† Node ≡ subproblem in search tree

Table 7.10 Node Selection Strategy

nodes. For both problems, we use the "Reduced DFS" upper bound heuristic and work on the down-branch first.

7.6.2 Other Cuts

Although we have implemented the knapsack cuts, the disjunctive cuts, and the cuts discussed in §3.5, the code found none of these cuts.

7.7 Extended Goldfarb-Idnani Algorithm

In this section, we demonstrate the extent to which using a dual quadratic programming method, which allows a "warm start" of the children nodes, is preferable to discarding the information and solving the children nodes from scratch. Although not reported, the number of nodes evaluated for each problem is slightly different for each algorithm due to a slight difference in the implementations. By examining Tables 7.13 and 7.14, it is clear that the solution time per node is greatly reduced when we

prob.52		Without SD Cuts		With SD Cuts	
κ	$f(x^*, y^*)$	Nodes†	Time	Nodes†	Time
27	-8.005632e-03	1681490	1045.73	280922	194.48
28	-2.044631e-02	1255707	755.09	224027	147.30
31	-4.915392e-02	120900	69.24	28824	18.26
34	-6.743432e-02	28361	15.15	8927	5.30
37	-7.965479e-02	4316	2.13	1832	1.03
40	-8.658508e-02	647	0.34	407	0.20
43	-8.951690e-02	304	0.18	256	0.18
46	-9.157293e-02	103	0.08	103	0.10

† Node $\equiv$ subproblem in search tree

Table 7.11 Symmetric Dominance Cuts

prob.500		Without SD Cuts		With SD Cuts	
κ	$f(x^*, y^*)$	Nodes†	Time	Nodes†	Time
118	-4.124560e-01	764854	4988.49	28420	1058.99
119	-4.125602e-01	418799	2403.87	19925	752.73
120	-4.126638e-01	209414	1175.01	13086	498.46
121	-4.127616e-01	102495	558.08	8227	319.87
124	-4.129730e-01	14114	70.25	2582	101.82
127	-4.131053e-01	3657	19.02	1267	51.99
130	-4.132057e-01	874	5.52	552	24.48
133	-4.132662e-01	323	3.13	291	14.25
136	-4.132888e-01	304	3.05	274	13.73

† Node $\equiv$ subproblem in search tree

Table 7.12 Symmetric Dominance Cuts

use a warm start algorithm such as our quadratic programming problem solver versus using a "cold start" procedure, such as the CPLEX quadratic programming problem solver.

prob.52		CPLEX QP		Restart QP	
κ	$f(x^*, y^*)$	Time	Time/Node†	Time	Time/Node†
25	2.801919e-02	2480.88	7.17e-03	247.40	7.15e-04
28	-2.044631e-02	1592.85	7.11e-03	147.30	6.58e-04
31	-4.915392e-02	198.13	6.88e-03	18.26	6.34e-04
34	-6.743432e-02	56.25	6.30e-03	5.30	5.93e-04
37	-7.965479e-02	10.77	5.88e-03	1.03	5.62e-04
40	-8.658508e-02	1.98	4.86e-03	0.20	4.91e-04
43	-8.951690e-02	1.17	4.57e-03	0.18	7.03e-04
46	-9.157293e-02	0.42	4.08e-03	0.10	9.71e-04

† Node ≡ subproblem in search tree

Table 7.13 Cold Start QP vs. Warm Start QP

prob.500		CPLEX QP		Restart QP	
κ	$f(x^*, y^*)$	Time	Time/Node†	Time	Time/Node†
126	-4.130662e-01	260.50	4.64e-02	29.55	5.24e-03
127	-4.131053e-01	168.05	4.62e-02	19.02	5.20e-03
130	-4.132057e-01	38.44	4.51e-02	5.52	6.32e-03
133	-4.132662e-01	13.35	4.44e-02	3.13	9.69e-03
136	-4.132888e-01	12.45	4.41e-02	3.05	1.00e-02

† Node ≡ subproblem in search tree

Table 7.14 Cold Start QP vs. Warm Start QP

Table 7.15 shows the results from a MATLAB [40] implementation of our extended Goldfarb-Idnani algorithm. The code that gives these results is a truncated tree; it only processes the down-branches until there are no more than κ assets that

are nonzero. The Goldfarb-Idnani code treats violated bounds as full inequality constraints, while the extended Goldfarb-Idnani (extended G-I) algorithm exploits the structure of the simple bounds as discussed in Chapter 5. In both cases a Cholesky factorization of the appropriate matrix is updated. It is clear that treating the simple bounds as full inequality constraints is much less efficient than our method which exploits their structure.

prob.52	Goldfarb-Idnani		Extended G-I	
κ	Time	Flops	Time	Flops
25	5.6333	1544653	3.8333	252591
26	5.5667	1463405	3.6667	245916
27	5.0833	1383955	3.5833	239101
30	3.4667	1015330	2.5667	200054
33	2.9167	888358	2.1833	188669
36	2.2000	712667	1.7000	169586
39	1.6500	565275	1.3000	153979
40	1.4667	525623	1.2333	149949
43	1.0833	408245	0.9167	137703
46	0.6833	293108	0.5833	125223

Table 7.15 Simple Bounds as Explicit Constraints

7.8 Graphical Presentation of Results

In this section, we include a few graphical representations of our results from solving problems *prob.52* and *prob.500*.

In Figures 7.1 and 7.2, we plot each value of κ against the time in seconds for the code to complete. For *prob.52*, it is clear from looking at Figure 7.1 the "Subtree" and "Up-First" heuristics with symmetric dominance cuts take the least amount of time to complete. Although if our code were implemented differently this would also be true of *prob.500*, we see that that the down-first "Reduced DFS" approach with symmetric dominance cuts works best for *prob.500*. Although not included in the results, the down-first "Subtree" approach with symmetric dominance cuts performs slightly better in practice.

In Figures 7.3 and 7.4, we see similar results as seen in Figures 7.1 and 7.2, respectively. One observation that can be made by comparing the two is that for *prob.500* the time per node increases when symmetric dominance cuts are added. This is because there are 15335 of these cuts, and a large amount of time is spent processing these cuts at each node. A better implementation may alleviate this reason for an increase in time per node.

In Figures 7.5 and 7.6 we compare the time per node for three different algorithms. It is clear from examining these figures that the implicit branch-and-bound approach is more efficient than the standard branch-and-bound approach. Furthermore, the warm restart of our extended Goldfarb-Idnani is an even better method than a cold start method which uses no previous information.

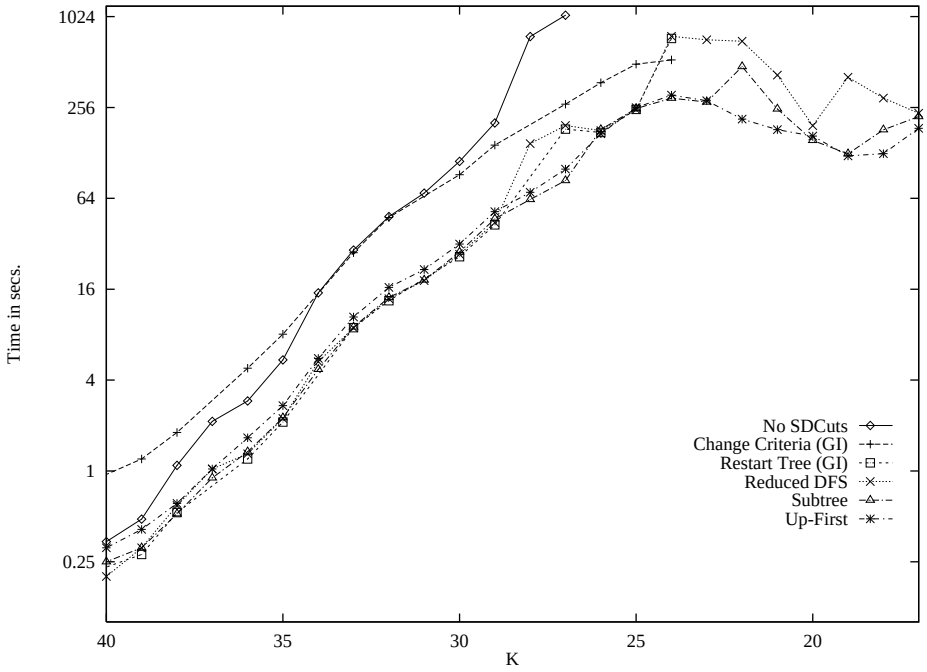

Figure 7.1 Timings for Implicit B&B for *prob.52*

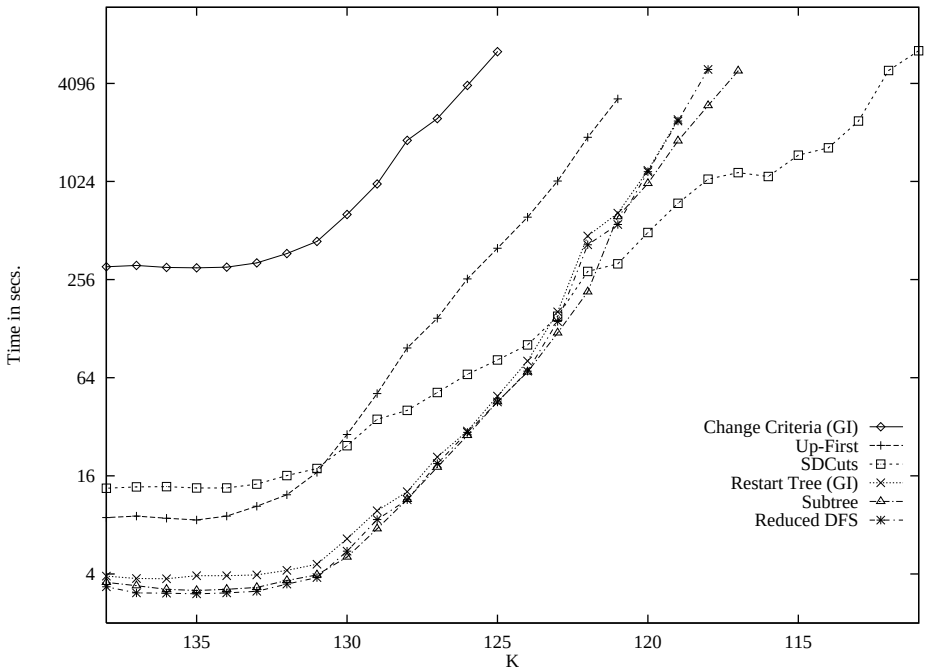

Figure 7.2 Timings for Implicit B&B for *prob.500*

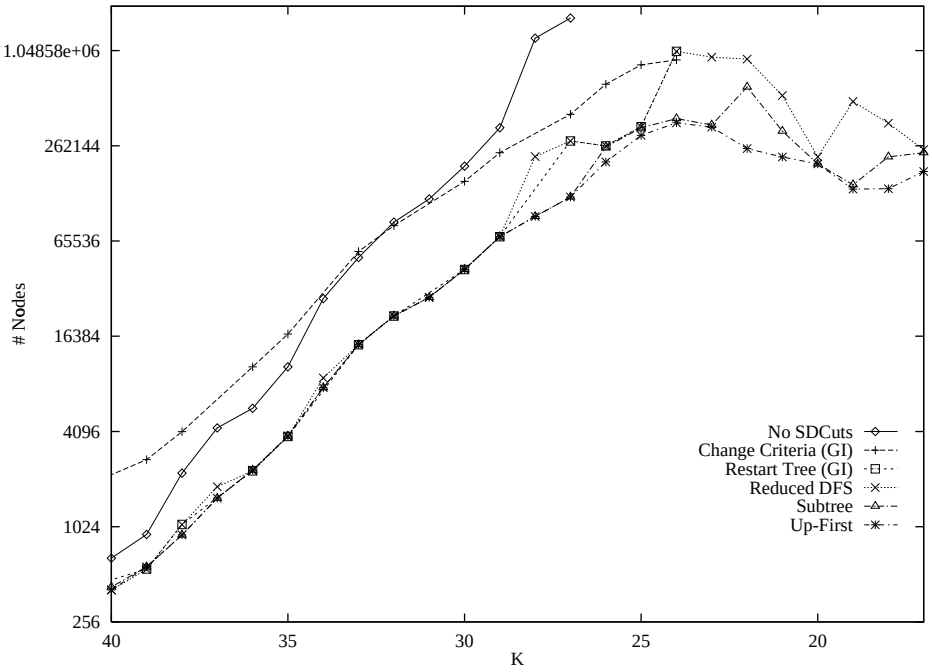

Figure 7.3 Number of Nodes for Implicit B&B for *prob.52*

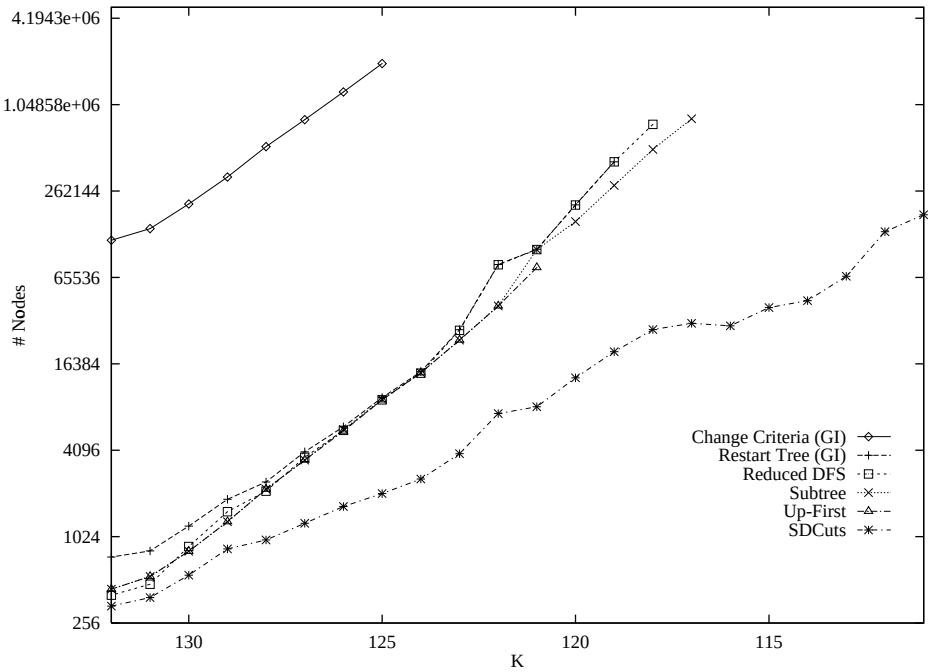

Figure 7.4 Number of Nodes for Implicit B&B for *prob.500*

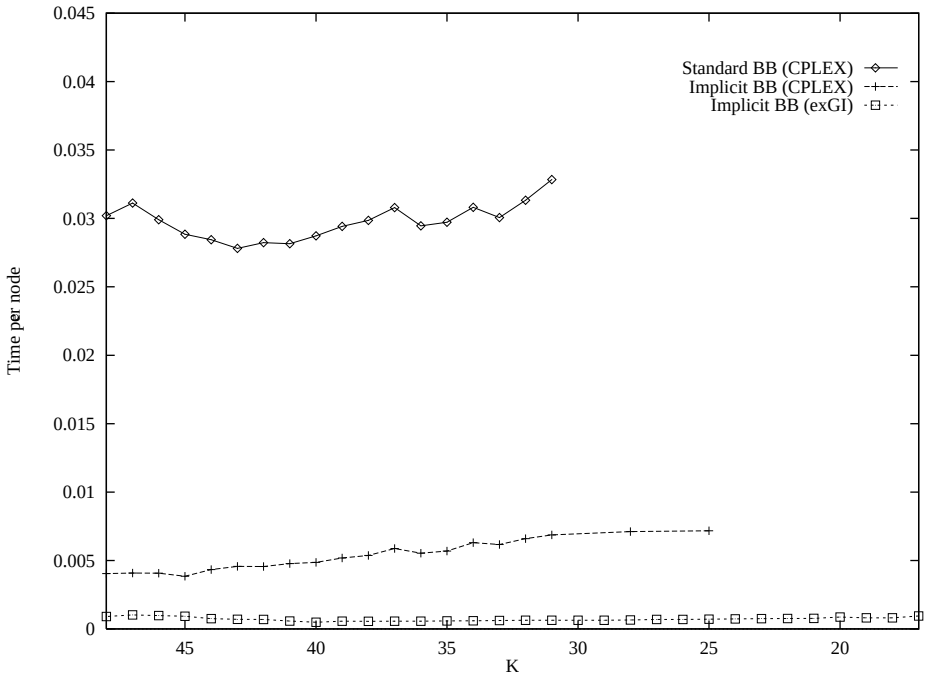

Figure 7.5 Time per Node for *prob.52*

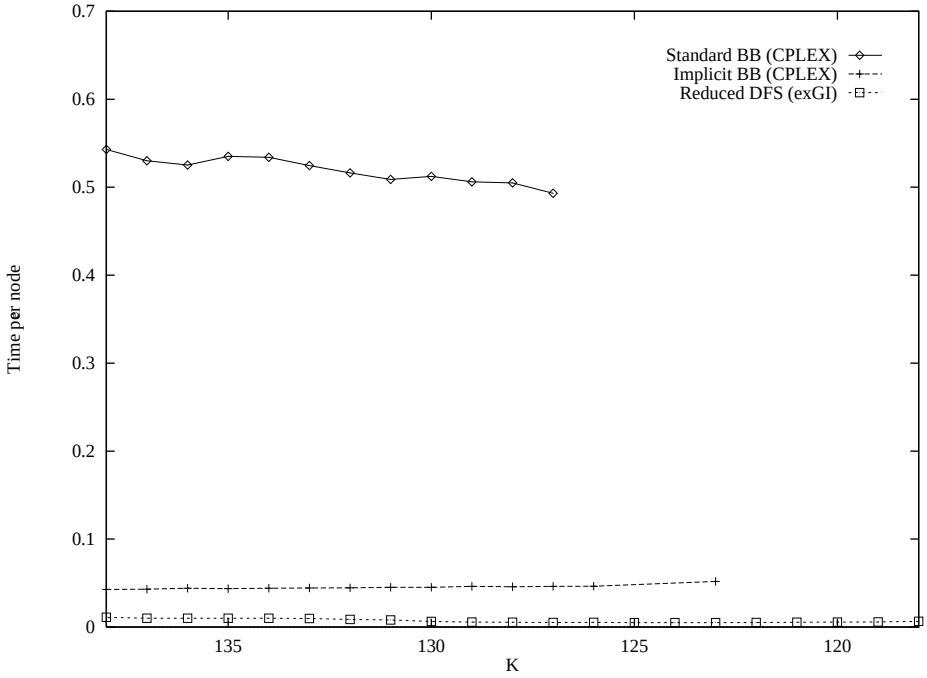

Figure 7.6 Time per Node for *prob.500*

7.9 Larger Problems

For *prob.500*, the size of the branch-and-bound tree becomes much larger when we examine smaller values of κ. Table 7.16 show results for smaller values of κ than we have shown in the previous sections. For this problem, we set the maximum number of nodes to be 1.5 million; $f(x^{IP}, y^{IP})$ is the objective function value of the best upper bound found by the code and $f(x^L, y^L)$ is the value of the lower bound at the time the code completes. This code uses the "Subtree" upper bound heuristic, the "Down-First" node selection criteria and symmetric dominance cuts.

prob.500				
κ	$f(x^{IP}, y^{IP})$	$f(x^L, y^L)$	Nodes†	Time
60	-1.022380e-01	-2.450006e-01	1500000	6.449166e+04
70	-3.039158e-01	-4.033516e-01	1500000	6.264430e+04
80	-3.837499e-01	-4.132943e-01	1500000	5.741596e+04
100	-4.068966e-01	-4.132943e-01	1500000	5.360170e+04
110	-4.111856e-01	-4.111856e-01	268647	9.804920e+03

† Node ≡ subproblem in search tree

Table 7.16 Upper and Lower Bounds for *prob.500*

Table 7.17 shows results for a larger problem. This problem has 1194 columns and 97 rows, with a universe of 1140 assets. Here $f(x^{IP}, y^{IP})$ is the objective function value of the best upper bound found by the code and $f(x^L, y^L)$ is the value of the lower bound at the time the code completes. Since there are no valid symmetric dominance cuts for this problem, this code is run with the "Subtree" upper bound heuristic and the "Down-First" node selection criteria.

prob.1140				
κ	$f(x^{IP}, y^{IP})$	$f(x^L, y^L)$	Nodes†	Time
1133	8.608667e-07	8.608667e-07	210538	2.0289e+04
1135	8.157303e-07	8.157303e-07	21068	2.0433e+03
1137	7.830330e-07	7.830330e-07	2404	2.3295e+02
1138	7.673880e-07	7.673880e-07	2285	2.1510e+02
1139	7.561090e-07	7.561090e-07	3	1.5230e+01

† Node ≡ subproblem in search tree

Table 7.17 Upper and Lower Bounds for *prob.1140*

7.10 Effect of Changing θ

It is an interesting experiment to examine the effects of changing θ. It appears that the smaller the value of θ is, the fewer nodes that need to be enumerated in order to determine the optimal solution.

prob.52			
θ	$f(x^*, y^*)$	Nodes†	Time
1.40	1.721255e-02	507413	309.89
1.20	-1.565444e-02	113450	73.04
1.00	-4.915392e-02	28824	20.61
0.80	-8.376025e-02	5373	3.71
0.60	-1.212010e-01	870	0.70

† Node ≡ subproblem in search tree

Table 7.18 Different Values of θ

7.11 Confidence Region Objective

Tables 7.19 and 7.20 show the results from a MATLAB [40] implementation of the dual algorithm for solving the nonlinear programming problem resulting from using

Baumol's objective function, $\hat{f}(x) = -\mu^T x + \theta\sqrt{x^T V x}$. The code that gives these results is a truncated tree. The results in Table 7.19 are from a code that processes the down-branches until κ assets have been branched down. The results in Table 7.20 are from a code that processes the up-branches until κ assets have been branched up. The 2-norm of the gradient of the Lagrangian function at the solution is denoted $\|\nabla_x \mathcal{L}\|_2$; in theory, the value of $\|\nabla_x \mathcal{L}\|_2$ should be zero at the optimal solution. The number of iterations of the dual algorithm is also reported.

prob.52		Down-Branches		
κ	$\|\nabla_x \mathcal{L}\|_2$	Iterations	Time	Flops
27	8.428e-12	50	9.45	3968702
30	2.382e-11	14	3.87	1332384
33	2.299e-15	0	1.85	500945

Table 7.19 Confidence Interval Objective

prob.52		Up-Branches		
κ	$\|\nabla_x \mathcal{L}\|_2$	Iterations	Time	Flops
24	6.675e-15	36	6.10	3412672
27	6.216e-15	29	4.68	2712113
30	4.113e-15	22	3.52	2058257
33	3.874e-15	19	2.90	1770012

Table 7.20 Confidence Interval Objective

Chapter 8

Conclusions and Future Work

In this thesis we have made both theoretical and computational advances in the solution of limited diversification portfolio selection problems. We have applied many of the techniques used in the solution of mixed-integer linear programming problems to enhance our ability to solve the mixed-integer nonlinear programming problems arising from the field of portfolio selection. Some of these techniques include procedures for determining feasible points, for deriving cutting planes and for choosing the next subproblem on which to work. Moreover, we have shown that, similar to mixed-integer linear programming, the relaxations of the subproblems in the branch-and-bound tree can be solved efficiently using dual algorithms. We have provided dual algorithms for both objective functions examined in this thesis; these dual algorithms can be implemented to exploit the structure of the simple bounds on the variables.

The work on this problem is by no means done. More procedures for determining valid inequalities and feasible solutions can be developed. Improvements in either of these areas would result in a branch-and-bound tree with fewer nodes. Moreover, if the implementation of the valid inequalities, including the symmetric dominance cuts and the updated approximating constraints, is improved, the time spent solving each subproblem could be decreased enough that the node selection strategy of choosing the up-branch first is better that the node selection strategy of choosing the down-branch first.

It may be possible to extend the dual algorithms to the case where the variance-covariance matrix V is positive semi-definite, but not positive-definite. Although we

have made much progress on developing a dual nonlinear programming algorithm for solving models with the confidence interval objective function, future work on this problem could include proving that Algorithm 6.3.1 does not fail in the degenerate case when $\theta^2 \geq \sqrt{\mu^T V^{-1} \mu}$, so long as the feasible region is bounded.

Appendix A

Derivation of Step Size for Dual NLP Algorithm

For this appendix, we assume that we are at the beginning of a subiteration in Algorithm 6.3.1. This means that we have primal variables x^*, a working set $\emptyset \neq \mathcal{A} \subseteq \mathcal{M}_2$, an index $k \in \mathcal{M}_2 \backslash \mathcal{A}$, dual variables $\tau^*_{\mathcal{A}}$, and a $\nu^* \in \mathbb{R}_+$ such that

$$-\mu + \frac{\theta}{\sqrt{x^{*T} V x^*}} V x^* - C^T_{\mathcal{A}:} \tau^*_{\mathcal{A}} - \nu^* C_{k:} = \mathbf{0},$$

$\theta^2 > (\nu^* C^T_{k:} + \mu)^T Z (\nu^* C^T_{k:} + \mu)$, $C_{\mathcal{A}:}$ has full row rank, $C_{\mathcal{A}:} x^* = d_{\mathcal{A}}$, $C_{k:} x^* < d_k$, and $\|d_{\mathcal{A}}\|_2 > 0$, where

$$Z = V^{-1} - V^{-1} C^T_{\mathcal{A}:} (C_{\mathcal{A}:} V^{-1} C^T_{\mathcal{A}:})^{-1} C_{\mathcal{A}:} V^{-1}.$$

In the first section of this appendix, we examine the case that $\|Z C^T_{k:}\|_2 > 0$. For this case, we derive the full primal step size, ϖ_1, and show that it is strictly positive. Moreover, we prove that for all $0 < \varpi \leq \varpi_1$, and

$$\theta^2 > \left((\nu^* + \varpi) C^T_{k:} + \mu\right)^T Z \left((\nu^* + \varpi) C^T_{k:} + \mu\right).$$

We also show how to determine the maximum dual step size ϖ_2 such that $\bar{\tau}_{\mathcal{A}}(\varpi_2) \geq 0$. In the second section, we examine the case that $Z C^T_{k:} = \mathbf{0}$. Given the dual step size ϖ_2, we prove that for all $0 < \varpi \leq \varpi_2$, and

$$\theta^2 > \left((\nu^* + \varpi) C^T_{k:} + \mu\right)^T Z \left((\nu^* + \varpi) C^T_{k:} + \mu\right).$$

A.1 Case 1: $\|Z C^T_{k:}\|_2 > 0$

Given the above conditions, we need to determine a step size ϖ_1, primal variables $\bar{x}$ and dual variables $\bar{\tau}_{\mathcal{A}}$ that satisfy the system of nonlinear equations

$$-\mu + \frac{\theta}{\sqrt{\bar{x}^T V \bar{x}}} V \bar{x} - C^T_{\mathcal{A}:} \bar{\tau}_{\mathcal{A}} \;=\; (\nu^* + \varpi_1) C^T_{k:}$$

$$\begin{aligned} C_{\mathcal{A}:}\bar{x} &= d_{\mathcal{A}} \\ C_{k:}\bar{x} &= d_k. \end{aligned}$$

Setting $\bar{\mathcal{A}} = \mathcal{A} \cup \{k\}$, this is equivalent to determining the optimal solution to the equality constrained problem

$$\begin{aligned} \text{minimize} \quad & -\mu^T x + \theta\sqrt{x^T V x} \\ \text{subject to} \quad & C_{\bar{\mathcal{A}}:}x = d_{\bar{\mathcal{A}}}. \end{aligned} \tag{A.1}$$

Since $\|ZC_{k:}^T\|_2 > 0$, we know that $C_{\bar{\mathcal{A}}:}$ has full row rank. By Theorem 6.2.4, we know that (A.1) has a unique solution if and only if $\theta^2 > \mu^T \bar{Z}\mu$, where

$$\bar{Z} = V^{-1} - V^{-1}C_{\bar{\mathcal{A}}:}^T(C_{\bar{\mathcal{A}}:}V^{-1}C_{\bar{\mathcal{A}}:}^T)^{-1}C_{\bar{\mathcal{A}}:}V^{-1}.$$

Theorem A.1.1 The nullspace matrix

$$\bar{Z} = V^{-1} - V^{-1}C_{\bar{\mathcal{A}}:}^T(C_{\bar{\mathcal{A}}:}V^{-1}C_{\bar{\mathcal{A}}:}^T)^{-1}C_{\bar{\mathcal{A}}:}V^{-1}$$

of $C_{\bar{\mathcal{A}}:}$ is such that $\theta^2 > \mu^T \bar{Z}\mu$.

Proof: Since

$$\begin{aligned} &(C_{\bar{\mathcal{A}}:}V^{-1}C_{\bar{\mathcal{A}}:}^T)^{-1} \\ &\quad = (C_{k:}ZC_{k:}^T)^{-1}\begin{bmatrix} C_{k:}ZC_{k:}^T(C_{\mathcal{A}:}V^{-1}C_{\mathcal{A}:}^T)^{-1} + NC_{k:}^TC_{k:}N^T & -NC_{k:}^T \\ -C_{k:}N^T & 1 \end{bmatrix}, \end{aligned}$$

where $N = (C_{\mathcal{A}:}V^{-1}C_{\mathcal{A}:}^T)^{-1}C_{\mathcal{A}:}V^{-1}$, it is straightforward to show that

$$\bar{Z} = Z - \frac{ZC_{k:}^TC_{k:}Z}{C_{k:}ZC_{k:}^T}. \tag{A.2}$$

Because $\bar{Z}$ is a nullspace matrix for $C_{\bar{\mathcal{A}}:}$, we know that $\bar{Z}C_{k:}^T = \mathbf{0}$. Thus,

$$\begin{aligned} \mu^T\bar{Z}\mu &= (\nu^*C_{k:}^T + \mu)^T\bar{Z}(\nu^*C_{k:}^T + \mu) \\ &= (\nu^*C_{k:}^T + \mu)^TZ(\nu^*C_{k:}^T + \mu) - \left((\nu^*C_{k:}^T + \mu)ZC_{k:}^T\right)^2/(C_{k:}ZC_{k:}^T) \\ &\leq (\nu^*C_{k:}^T + \mu)^TZ(\nu^*C_{k:}^T + \mu) \\ &< \theta^2, \end{aligned}$$

since $\theta^2 > (\nu^* C_{k:}^T + \mu)^T Z (\nu^* C_{k:}^T + \mu)$ and $\bar{Z}$ is a positive semi-definite matrix.□

Theorem 6.2.5 from §6.2 gives us that the solution to (A.1) is

$$\begin{aligned} \bar{x} &= s\bar{Z}\mu + V^{-1}C_{\bar{\mathcal{A}}:}^T (C_{\bar{\mathcal{A}}:} V^{-1} C_{\bar{\mathcal{A}}:}^T)^{-1} d_{\bar{\mathcal{A}}} \\ \bar{\tau}_{\bar{\mathcal{A}}} &= -(C_{\bar{\mathcal{A}}:} V^{-1} C_{\bar{\mathcal{A}}:}^T)^{-1} C_{\bar{\mathcal{A}}:} V^{-1}\mu + \frac{1}{s}(C_{\bar{\mathcal{A}}:} V^{-1} C_{\bar{\mathcal{A}}:}^T)^{-1} d_{\bar{\mathcal{A}}}, \end{aligned}$$

where

$$\begin{aligned} s &= \left(\frac{d_{\bar{\mathcal{A}}}^T (C_{\bar{\mathcal{A}}:} V^{-1} C_{\bar{\mathcal{A}}:}^T)^{-1} d_{\bar{\mathcal{A}}}}{\theta^2 - \mu^T \bar{Z}\mu} \right)^{\frac{1}{2}} \\ &= \left(\frac{d_{\mathcal{A}}^T (C_{\mathcal{A}:} V^{-1} C_{\mathcal{A}:}^T)^{-1} d_{\mathcal{A}} + (C_{k:} N^T d_{\mathcal{A}} - d_k)^2 / (C_{k:} Z C_{k:}^T)}{\theta^2 - \mu^T Z\mu + (C_{k:} Z\mu)^2 / (C_{k:} Z C_{k:}^T)} \right)^{\frac{1}{2}}. \end{aligned}$$

The dual variable associated with the constraint $C_{k:}x = d_k$ is

$$\bar{\nu} \equiv \bar{\tau}_k = -\frac{C_{k:} Z\mu}{C_{k:} Z C_{k:}^T} - \frac{1}{s}\left(\frac{C_{k:} N^T d_{\mathcal{A}} - d_k}{C_{k:} Z C_{k:}^T} \right).$$

Thus $\varpi_1 = \bar{\nu} - \nu^*$ is the full primal step size; we show in Theorem A.1.4 that $C_{k:}(x + \Delta x(\varpi_1)) = d_k$. In order to prove that $\varpi_1 > 0$, we first need to prove the following theorem.

Theorem A.1.2 Let

$$g(\varpi) = \theta^2 - ((\nu^* + \varpi) C_{k:}^T + \mu)^T Z ((\nu^* + \varpi) C_{k:}^T + \mu).$$

Then the function $g(\varpi) > 0$ for all ϖ between 0 and ϖ_1.

Proof: We first show that $g(0) > 0$ and $g(\varpi_1) > 0$, and then show that $g(\varpi)$ is a strictly concave function.

Since $g(0) = \theta^2 - (\nu^* C_{k:}^T + \mu)^T Z (\nu^* C_{k:}^T + \mu)$ and $\theta^2 > (\nu^* C_{k:}^T + \mu)^T Z (\nu^* C_{k:}^T + \mu)$, we know $g(0) > 0$.

Since $g(\varpi_1) = \theta^2 - \mu^T Z\mu + \frac{(C_{k:}Z\mu)^2}{C_{k:}ZC_{k:}^T} - \frac{(C_{k:}N^T d_{\mathcal{A}} - d_k)^2}{s^2 C_{k:}ZC_{k:}^T}$, it is easy to show that

$$g(\varpi_1) = \frac{d_{\mathcal{A}}^T (C_{\mathcal{A}:}V^{-1}C_{\mathcal{A}:}^T)^{-1} d_{\mathcal{A}}}{s^2}.$$

Since $\|d_{\mathcal{A}}\|_2 > 0$, $g(\varpi_1) > 0$.

The function $g(\varpi)$ is strictly concave since

$$g'(\varpi) = -2C_{k:}ZC_{k:}^T.$$

Thus $g(\varpi) > 0$ for all ϖ between 0 and ϖ_1.□

To prove that the step size $\varpi_1 > 0$, we first show that the function $c(\varpi) = C_{k:}(x^* + \Delta x(\varpi))$ is a strictly increasing function on the interval between 0 and ϖ_1, without specifying an ordering of 0 and ϖ, and then show that $c(0) < c(\varpi_1)$.

Theorem A.1.3 Let $c(\varpi) = C_{k:}(x^* + \Delta x(\varpi))$. Then $c(\varpi)$ is a strictly increasing function on the interval between 0 and ϖ_1.

Proof:

$$\begin{aligned} c(\varpi) &= C_{k:}x^* + \frac{\sqrt{\bar{x}^T V \bar{x}}}{\theta}\left((\nu^* + \varpi)C_{k:}ZC_{k:}^T + C_{k:}Z\mu\right) - \\ &\quad \frac{\sqrt{x^{*T}Vx^*}}{\theta}(\nu^* C_{k:}ZC_{k:}^T + C_{k:}Z\mu) \end{aligned}$$

The first derivative of $\sqrt{\bar{x}^T V \bar{x}}$ with respect to ϖ is

$$\frac{\sqrt{\bar{x}^T V \bar{x}}\left((\nu^* + \varpi)C_{k:}ZC_{k:}^T + C_{:k}Z\mu\right)}{g(\varpi)}.$$

Thus

$$\begin{aligned} c'(\varpi) &= \frac{\sqrt{\bar{x}^T V \bar{x}}}{\theta} \cdot \frac{\left((\nu^* + \varpi)C_{k:}ZC_{k:}^T + C_{:k}Z\mu\right)^2}{g(\varpi)} \\ &\quad + \frac{\sqrt{\bar{x}^T V \bar{x}}}{\theta} C_{k:}ZC_{k:}^T \\ &= \frac{\sqrt{\bar{x}^T V \bar{x}}}{\theta}\left(C_{k:}ZC_{k:}^T + \frac{\left((\nu^* + \varpi)C_{k:}ZC_{k:}^T + C_{:k}Z\mu\right)^2}{g(\varpi)}\right). \end{aligned}$$

Since $g(\varpi) > 0$ for all ϖ between 0 and ϖ_1, $c(\varpi)$ is strictly increasing between 0 and ϖ_1. □

Theorem A.1.4 Let $c(\varpi) = C_{k:}(x^* + \Delta x(\varpi))$. Then $c(0) < c(\varpi_1)$.

Proof: Since $\bar{x}(\varpi) = x^*$ at $\varpi = 0$, $c(0) < d_k$.

At ϖ_1, we see that

$$\begin{aligned}
c(\varpi_1) &= C_{k:}(x^* + \Delta x(\varpi_1)) \\
&= C_{k:}x^* + \frac{\sqrt{\bar{x}^T V \bar{x}}}{\theta}\left((\nu^* + \varpi_1)C_{k:}ZC_{k:}^T + C_{k:}Z\mu\right) \\
&\quad - \frac{\sqrt{x^{*T}Vx^*}}{\theta}(\nu^* C_{k:}ZC_{k:}^T + C_{k:}Z\mu) \\
&= C_{k:}x^* - \frac{\sqrt{x^{*T}Vx^*}}{\theta}(\nu^* C_{k:}ZC_{k:}^T + C_{k:}Z\mu) \\
&\quad + \frac{\sqrt{x^{*T}Vx^*}}{\theta}\frac{\sqrt{g(0)}}{\sqrt{g(\varpi_1)}}\left((\nu^* + \varpi_1)C_{k:}ZC_{k:}^T + C_{k:}Z\mu\right) \\
&= C_{k:}N^T d_{\mathcal{A}} + sC_{k:}Z\mu \\
&\quad + s\left(\nu^* - \frac{C_{k:}Z\mu}{C_{k:}ZC_{k:}^T} - \frac{1}{s}\left(\frac{C_{k:}N^T d_{\mathcal{A}} - d_k}{C_{k:}ZC_{k:}^T}\right) - \nu^*\right)C_{k:}ZC_{k:}^T \\
&= C_{k:}N^T d_{\mathcal{A}} - C_{k:}N^T d_{\mathcal{A}} + d_k \\
&= d_k.
\end{aligned}$$

Thus $c(0) < c(\varpi_1)$.□

Corollary A.1.1 The scalar ϖ_1 is strictly positive.

Proof: Since $c(\varpi)$ is a strictly increasing function on the interval between 0 and ϖ_1 and $c(0) < c(\varpi_1)$, we know that $\varpi_1 > 0$.□

Thus we have determined the full primal step, $\Delta x(\varpi_1)$. We now need to determine a ϖ_2 in the interval $[0, \varpi_1]$ such that $\tau_{\mathcal{A}}^* - \Delta\tau_{\mathcal{A}}(\varpi_2) \geq \mathbf{0}$ and either $\varpi_2 = \varpi_1$ or there exists a $j \in \mathcal{A}$ such that $\tau_j^* - \Delta\tau_j(\varpi_2) = 0$. We first show that the function $\bar{\tau}_j(\varpi)$ as a function of ϖ is either convex or concave.

Theorem A.1.5 The dual variable $\bar{\tau}_j(\varpi) = \tau_j^* - \Delta\tau_j(\varpi)$, $j \in \mathcal{A}$, as a function of ϖ, is either convex or concave.

Proof: Since

$$\bar{\tau}_j(\varpi) = \tau_j^* - \left(1 - \frac{\sqrt{x^{*T}Vx^*}}{\sqrt{\bar{x}^T V \bar{x}}}\right)\left(N_{j:}(\nu^* C_{k:}^T + \mu) + \tau_j^*\right) - \varpi N_{j:} C_{k:}^T$$

differentiating $\bar{\tau}_j(\varpi)$ with respect to ϖ gives

$$\bar{\tau}_j'(\varpi) = -N_{j:}C_{k:}^T - \frac{\alpha\left((\nu^* + \varpi)C_{k:}ZC_{k:}^T + C_{k:}Z\mu\right)}{g(\varpi)} \cdot \left(N_{j:}(\nu^* C_{k:}^T + \mu) + \tau_j^*\right),$$

where $\alpha = (x^{*T}Vx^*)^{1/2} \cdot (\bar{x}^T V \bar{x})^{-1/2}$. Differentiating $\bar{\tau}_j(\varpi)$ again with respect to ϖ gives

$$\bar{\tau}_j''(\varpi) = -\bar{\alpha}\left(N_{j:}(\nu^* C_{k:}^T + \mu) + \tau_j^*\right),$$

where

$$\bar{\alpha} = (x^{*T}Vx^*)^{1/2} \cdot (\bar{x}^T V \bar{x})^{-1/2} \cdot \left(\frac{\left((\nu^* + \varpi)C_{k:}ZC_{k:}^T + C_{k:}Z\mu\right)^2}{g(\varpi)^2} + \frac{C_{k:}ZC_{k:}^T}{g(\varpi)}\right).$$

Since $\bar{\alpha} \geq 0$, $\bar{\tau}_j(\varpi)$ is either convex or concave.□

This means that $\bar{\tau}_j(\varpi)$ changes from being positive to being negative at most once as ϖ increases from 0 to ϖ_1.

If $\bar{\tau}_j(\varpi)$ is strictly concave, then it evaluates to zero no more than once on the interval $[0, \varpi_1]$. If $\bar{\tau}_j(\varpi)$ is strictly convex and $\bar{\tau}_j'(0) \geq 0$, then $\bar{\tau}_j(\varpi)$ is nonnegative on the interval $[0, \varpi_1]$. If $\bar{\tau}_j(\varpi)$ is strictly convex and $\bar{\tau}_j'(0) < 0$, then $\bar{\tau}_j(\varpi)$ evaluates to zero at most twice on the interval $[0, \varpi_1]$. In each case for which we know that $\bar{\tau}_j(\varpi)$ evaluates to zero at least once on the interval $[0, \varpi_1]$, we can use a line search technique to find the first such zero.

Letting $\bar{\varpi}$ be the minimum value of ϖ_1 and ϖ_2, if $\bar{\varpi} = \varpi_1$, then a full primal step is taken and the set of working constraints is updated to $\bar{\mathcal{A}} = \mathcal{A} \cup \{k\}$. In this case we need to show that $\theta^2 > \mu^T \bar{Z} \mu$, where $\bar{Z}$ is the nullspace matrix for $C_{\bar{\mathcal{A}}:}$. If, on the other hand, a partial step is taken such that the set of working constraints is updated to $\bar{\mathcal{A}} = \mathcal{A} \backslash \{i\}$, then we need to show that $\theta^2 > \left((\nu^* + \bar{\varpi}) C_{k:}^T + \mu\right)^T \bar{Z} \left((\nu^* + \bar{\varpi}) C_{k:}^T + \mu\right)$, where $\bar{Z}$ is the nullspace matrix for $C_{\bar{\mathcal{A}}:}$. Moreover, if $\bar{\mathcal{A}} = \emptyset$ or $d_{\bar{\mathcal{A}}} = \mathbf{0}$, we are now in the degenerate case as discussed in §6.3.3. Since that special case is treated in a different manner, here we assume that we are not in a degenerate case.

Theorem A.1.6 If a full primal step is taken ($\bar{\varpi} = \varpi_1$), then $\theta^2 > \mu^T \bar{Z} \mu$.

Proof: Since a full primal step was taken, we know $\|ZC_{k:}\|_2 > 0$, so from Theorem A.1.1, we see that

$$\bar{Z} = Z - \frac{ZC_{k:}^T C_{k:} Z}{C_{k:} Z C_{k:}^T}. \tag{A.3}$$

Because $\bar{Z}$ is a nullspace matrix for $C_{\bar{\mathcal{A}}:}$, we have that $\bar{Z} C_{k:}^T = \mathbf{0}$. Thus,

$$\begin{aligned}
\mu^T \bar{Z} \mu &= (\nu^* C_{k:}^T + \mu)^T \bar{Z} (\nu^* C_{k:}^T + \mu) \\
&= (\nu^* C_{k:}^T + \mu)^T Z (\nu^* C_{k:}^T + \mu) - \left((\nu^* C_{k:}^T + \mu) Z C_{k:}^T\right)^2 / (C_{k:} Z C_{k:}^T) \\
&\leq (\nu^* C_{k:}^T + \mu)^T Z (\nu^* C_{k:}^T + \mu) \\
&< \theta^2,
\end{aligned}$$

since $\theta^2 > (\nu^* C_{k:}^T + \mu)^T Z (\nu^* C_{k:}^T + \mu)$ and $\bar{Z}$ is a positive semi-definite matrix.□

Theorem A.1.7 If a partial primal step is taken such that $i \in \mathcal{M}_2$ is dropped from the working set, then $\bar{\mathcal{A}} = \mathcal{A} \backslash \{i\}$. If $\bar{A} \neq \emptyset$ and $\|d_{\bar{\mathcal{A}}}\|_2 > 0$

$$\theta^2 > \left((\nu^* + \bar{\varpi}) C_{k:}^T + \mu\right)^T \bar{Z} \left((\nu^* + \bar{\varpi}) C_{k:}^T + \mu\right),$$

where $\bar{Z}$ is the nullspace matrix for $C_{\bar{\mathcal{A}}:}$.

Proof: The updated primal and dual variables satisfy the system of nonlinear equations

$$\begin{aligned} -\mu + \frac{\theta}{\sqrt{\bar{x}(\bar{\varpi})^T V \bar{x}(\bar{\varpi})}} V\bar{x}(\bar{\varpi}) - C_{\bar{\mathcal{A}}:}^T \bar{\tau}_{\bar{\mathcal{A}}}(\bar{\varpi}) &= (\nu^* + \bar{\varpi}) C_{k:}^T \\ C_{\bar{\mathcal{A}}:} x^* &= d_{\bar{\mathcal{A}}} \\ C_{k:} x^* &= \bar{d}_k, \end{aligned}$$

where $\bar{d}_k = C_{k:}x^*$. Let $\hat{\mathcal{A}} = \bar{\mathcal{A}} \cup \{k\}$,

$$C_{\hat{\mathcal{A}}} = \begin{bmatrix} C_{\bar{\mathcal{A}}:} \\ C_{k:} \end{bmatrix} \text{ and } d_{\bar{\mathcal{A}}} = \begin{bmatrix} d_{\bar{\mathcal{A}}} \\ \bar{d}_k \end{bmatrix}.$$

The matrix $C_{\hat{\mathcal{A}}:}$ has full row rank since $\|ZC_{k:}\|_2 > 0$.

This implies that $\bar{x}(\bar{\varpi})$ is an optimal solution to the problem

$$\begin{aligned} \text{minimize} \quad & -\mu^T x + \theta\sqrt{x^T V x} \\ \text{subject to} \quad & C_{\bar{\mathcal{A}}:} x = d_{\bar{\mathcal{A}}} \\ & C_{k:} x = \bar{d}_k. \end{aligned}$$

Then, since $\|d_{\bar{\mathcal{A}}}\|_2 > 0$, we know that $\theta^2 > \mu^T \bar{Z} \mu$ and, by Theorem 6.2.5,

$$\begin{aligned} \bar{x} &= \hat{s}\hat{Z}\mu + \hat{N} d_{\hat{\mathcal{A}}} \\ \bar{\tau}_{\bar{\mathcal{A}}} &= -\hat{N}\mu + \frac{1}{\hat{s}} (C_{\hat{\mathcal{A}}:} V^{-1} C_{\hat{\mathcal{A}}:}^T)^{-1} d_{\hat{\mathcal{A}}}, \end{aligned}$$

where

$$\begin{aligned} \hat{N} &= (C_{\hat{\mathcal{A}}:} V^{-1} C_{\hat{\mathcal{A}}:}^T)^{-1} C_{\hat{\mathcal{A}}:} V^{-1}, \\ \hat{Z} &= V^{-1} - V^{-1} C_{\hat{\mathcal{A}}:}^T \hat{N}, \text{ and} \\ \hat{s} &= \left(\frac{d_{\hat{\mathcal{A}}}^T (C_{\hat{\mathcal{A}}:} V^{-1} C_{\hat{\mathcal{A}}:}^T)^{-1} d_{\hat{\mathcal{A}}}}{\theta^2 - \mu^T \hat{Z} \mu} \right)^{\frac{1}{2}} \end{aligned}$$

$$= \left(\frac{d_{\bar{\mathcal{A}}}^T (C_{\bar{\mathcal{A}}:} V^{-1} C_{\bar{\mathcal{A}}:}^T)^{-1} d_{\bar{\mathcal{A}}} + (C_{k:} \bar{N}^T d_{\bar{\mathcal{A}}} - \hat{d}_k)^2 / (C_{k:} \bar{Z} C_{k:}^T)}{\theta^2 - \mu^T \bar{Z} \mu + (C_{k:} \bar{Z} \mu)^2 / (C_{k:} \bar{Z} C_{k:}^T)} \right)^{\frac{1}{2}}.$$

As in the proof for Theorem A.1.2, since

$$g(\bar{\varpi}) = \theta^2 - \mu^T \bar{Z} \mu + \frac{(C_{k:} \bar{Z} \mu)^2}{C_{k:} \bar{Z} C_{k:}^T} - \frac{(C_{k:} N^T d_{\bar{\mathcal{A}}} - \hat{d}_k)^2}{\hat{s}^2 C_{k:} \bar{Z} C_{k:}^T},$$

it is easy to show that

$$g(\bar{\varpi}) = \frac{d_{\bar{\mathcal{A}}}^T (C_{\bar{\mathcal{A}}:} V^{-1} C_{\bar{\mathcal{A}}:}^T)^{-1} d_{\bar{\mathcal{A}}}}{\hat{s}^2}.$$

Since $\|d_{\bar{\mathcal{A}}}\|_2 > 0$, $g(\bar{\varpi}) > 0$.□

A.2 Case 2: $\|ZC_{k:}^T\|_2 = 0$

We now discuss the case that $ZC_{k:}^T = \mathbf{0}$. Since in this case a partial step is taken such that $\Delta x(\bar{\varpi}) = \mathbf{0}$, we only need to show that $g(\bar{\varpi}) > 0$ and

$$\theta^2 > \left((\nu^* + \bar{\varpi}) C_{k:}^T + \mu \right)^T \bar{Z} \left((\nu^* + \bar{\varpi}) C_{k:}^T + \mu \right),$$

where $\bar{Z}$ is the nullspace matrix for $\bar{\mathcal{A}} = \mathcal{A} \backslash \{i\}$ and index $i \in \mathcal{M}_2$ was the index dropped from the working set. If $\bar{\mathcal{A}} = \emptyset$ or $d_{\bar{\mathcal{A}}} = \mathbf{0}$, we are now in the degenerate case as discussed in §6.3.3. Since that special case is treated in a different manner, here we assume that we are not in a degenerate case.

Theorem A.2.1 If $ZC_{k:}^T = \mathbf{0}$, then for any $\varpi \in \mathbb{R}$,

$$\theta^2 > \left((\nu^* + \varpi) C_{k:}^T + \mu \right)^T Z \left((\nu^* + \varpi) C_{k:}^T + \mu \right),$$

where Z is the nullspace matrix for $C_{\mathcal{A}:}$.

Proof: Since $g(0) = \theta^2 - (\nu^* C_{k:}^T + \mu)^T Z(\nu^* C_{k:}^T + \mu) > 0$ and $ZC_{k:} = \mathbf{0}$, we know that

$$\begin{aligned} g(\bar{\varpi}) &= \theta^2 - \left((\nu^* + \bar{\varpi})C_{k:}^T + \mu\right)^T Z \left((\nu^* + \bar{\varpi})C_{k:}^T + \mu)\right) \\ &= g(0) \\ &> 0.\Box \end{aligned}$$

We present the next result as a corollary to Theorem A.1.7.

Corollary A.2.1 If a partial step is taken such that $i \in \mathcal{M}_2$ is dropped from the working set, then $\bar{\mathcal{A}} = \mathcal{A}\backslash\{i\}$. If $\bar{A} \neq \emptyset$ and $\|d_{\bar{\mathcal{A}}}\|_2 > 0$

$$\theta^2 > \left((\nu^* + \bar{\varpi})C_{k:}^T + \mu\right)^T \bar{Z} \left((\nu^* + \bar{\varpi})C_{k:}^T + \mu\right),$$

where $\bar{Z}$ is the nullspace matrix for $C_{\bar{\mathcal{A}}:}$.

Proof: The proof of Theorem A.1.7 is applicable in this case as long as we show that the matrix $C_{\hat{\mathcal{A}}:}$ has full row rank, where $\hat{\mathcal{A}} = \bar{\mathcal{A}} \cup \{k\}$,

$$C_{\hat{\mathcal{A}}} = \begin{bmatrix} C_{\bar{\mathcal{A}}:} \\ C_{k:} \end{bmatrix} \text{ and } d_{\bar{\mathcal{A}}} = \begin{bmatrix} d_{\bar{\mathcal{A}}} \\ \bar{d}_k \end{bmatrix}.$$

We know that $C_{\bar{\mathcal{A}}:}$ has full row rank, so we just need to show that $\|\bar{Z}C_{k:}^T\|_2 > 0$.

$$\begin{aligned} 0 &= C_{k:} Z C_{k:}^T \\ &= C_{k:} \bar{Z} C_{k:}^T - (C_{i:} \bar{Z} C_{k:}^T)^2 / (C_{i:} \bar{Z} C_{i:}^T) \\ &\leq C_{k:} \bar{Z} C_{k:}^T, \end{aligned}$$

with strict inequality occurring if and only if $C_{i:}\bar{Z}C_{k:}^T \neq 0$. Because $i \in \mathcal{A}$ was the index that was dropped, we know that $N_{i:}C_{k:}^T > 0$. But since $N_{i:} = \bar{Z}C_{i:}^T$, we get that $C_{i:}\bar{Z}C_{k:}^T > 0.\Box$

Bibliography

[1] Egon Balas. Disjunctive programming: Cutting planes from logical conditions. In O. L. Magnasarian et. al., editor, *Nonlinear Programming 2*, pages 279–312. Academic Press, New York, 1975.

[2] William J. Baumol. An expected gain-confidence limit criterion for portfolio selection. *Management Science*, 10(1):174–182, 1963.

[3] Daniel Bienstock. Computational study of a family of mixed-integer quadratic programming problems. *Mathematical Programming*, 74(2):121–140, August 1996.

[4] B. Blog, G. van der Hoek, A. H. G. Rinnooy Kan, and G. T. Timmer. The optimal selection of small portfolios. *Management Science*, 29(7):792–798, 1983.

[5] William Breen and James Savage. Portfolio distributions and tests of security selection models. *Journal of Finance*, 23(5):805–819, 1968.

[6] Kalman J. Cohen and Jerry A. Pogue. An empirical evaluation of alternative portfolio-selection models. *Journal of Business*, 40(2):166–193, 1967.

[7] Mary W. Cooper and Keyvan Farhangian. An integer programming algorithm for portfolio selection with fixed charges. *Naval Research Logistics Quarterly*, 29(1):147–150, 1982.

[8] *Using the CPLEX Callable Library.* CPLEX Optimization, Inc., 1995.

[9] J. W. Daniel, W. B. Gragg, L. Kaufman, and G. W. Stewart. Reorthogonalization and stable algorithms for updating the Gram-Schmidt QR factorization. *Mathematics of Computation*, 30(136):772–795, October 1976.

[10] Marco A. Duran and Ignacio E. Grossman. An outer-approximation algorithm for a class of mixed-integer nonlinear programming problems. *Mathematical Programming*, 36:307–339, 1986.

[11] Edwin J. Elton, Martin J. Gruber, and Manfred W. Padberg. Simple criteria for optimal portfolio selection with upper bounds. *Operations Research*, 25(6):952–967, November-December 1977.

[12] Bruce Faaland. An integer programming algorithm for portfolio selection. *Management Science*, 20(10):1376–1384, 1974.

[13] Eugene F. Fama. The behavior of stock-market prices. *Journal of Business*, 38(1):34–105, 1965.

[14] Donald Eugene Farrar. *The Investment Decision Under Uncertainty.* Prentiss-Hall, Inc., Englewood Cliffs, N.J., 1st edition, 1962.

[15] William Feller. *An Introduction to Probability Theory and Its Applications.* John Wiley and Sons, New York, 2nd edition, 1957.

[16] Anthony V. Fiacco and Garth P. McCormick. *Nonlinear Programming: Sequential Unconstrained Minimization Techniques.* John Wiley and Sons, New York, 1968.

[17] Lawrence Fisher. Some new stock-market indexes. *Journal of Business*, 39(1):191–225, 1966.

[18] Roger Fletcher and Sven Leyffer. Solving mixed integer nonlinear programs by outer approximation. *Mathematical Programming*, 66(3):327–349, 1994.

[19] Irwin Friend and Douglas Vickers. Portfolio selection and investment performance. *Journal of Finance*, 20(3):391–415, September 1965.

[20] P. E. Gill, G. H. Golub, W. Murray, and M. A. Saunders. Methods for modifying matrix factorizations. *Mathematics of Computation*, 28(126):505–535, April 1974.

[21] Philip E. Gill, Walter Murray, and Margaret H. Wright. *Practical Optimization.* Academic Press, New York, 1981.

[22] D. Goldfarb and A. Idnani. A numerically stable dual method for solving strictly convex quadratic programs. *Mathematical Programming*, 27:1–33, 1983.

[23] Gene H. Golub and Charles F. Van Loan. *Matrix Computations.* The Johns Hopkins University Press, Baltimore, 2nd edition, 1989.

[24] Moshe Hagigi and Brian Kluger. Safety first: An alternative performance measure. *Journal of Portfolio Management*, 13(4):34–40, 1987.

[25] Giora Hanoch and Haim Levy. Efficient portfolio selection with quadratic and cubic utility. *Journal of Business*, 43(2):181–189, 1970.

[26] W. V. Harlow. Asset allocation in a downside-risk framework. *Financial Analysts Journal*, 47(5):28–40, 1991.

[27] W. V. Harlow and Ramesh K. S. Rao. Asset pricing in a generalized mean-lower partial moment framework: Theory and evidence. *Journal of Financial and Quantitative Analysis*, 24(3):285–311, September 1989.

[28] Nancy Jacob. A limited-diversification portfolio selection model for the small investor. *Journal of Finance*, 29(3):847–856, 1974.

[29] Robert H. Jeffrey. A new paradigm for portfolio risk. *Journal of Portfolio Management*, 11(1):33–40, 1984.

[30] A. H. Land and A. G. Doig. An automatic method for solving discrete programming problems. *Econometrica*, 28(3):497–520, July 1960.

[31] Rafael Lazimy. Mixed-integer quadratic programming. *Mathematical Programming*, 22:332–349, 1982.

[32] Rafael Lazimy. Improved algorithm for mixed-integer quadratic programs and a computational study. *Mathematical Programming*, 32:100–113, 1985.

[33] Jan R. Magnus and Heinz Neudecker. *Matrix Differential Calculus with Applications in Statistics and Economics*. John Wiley and Sons, New York, 1st edition, 1988.

[34] James C. T. Mao. Essentials of portfolio diversification strategy. *Journal of Finance*, 25(5):1109–1121, December 1970.

[35] James C. T. Mao and Carl Erik Särndal. A decision theory approach to portfolio selection. *Management Science*, 12(8):B323–B333, April 1966.

[36] Harry Markowitz. Portfolio selection. *Journal of Finance*, 7:77–91, 1952.

[37] Harry Markowitz. The optimization of a quadratic function subject to linear constraints. *Naval Research Logistics Quarterly*, 3:111–133, 1956.

[38] Harry Markowitz. *Portfolio Selection: Efficient Diversification of Investments*, volume 16 of *Cowles Foundation for Research in Economics at Yale University*. John Wiley and Sons, New York, 1st edition, 1959.

[39] Harry Markowitz. *Mean-Variance Analysis in Portfolio Choice and Capital Markets*. Basil Blackwell, New York, 1st edition, 1987.

[40] *MATLAB User's Guide*. The MathWorks, Inc., 1991.

[41] Stephen G. Nash and Ariela Sofer. *Linear and Nonlinear Programming*. McGraw-Hill, New York, 1996.

[42] Jong-Shi Pang. A new and efficient algorithm for a class of portfolio selection problems. *Operations Research*, 28(3):754–767, May-June 1980.

[43] Nittin R. Patel and Marti G. Subrahmanyam. A simple algorithm for optimal portfolio selection with fixed transaction costs. *Management Science*, 28(3):303–314, 1982.

[44] Andre F. Perold. Large-scale portfolio optimization. *Management Science*, 30(10):1143–1160, 1984.

[45] G. A. Pogue. An extension of the Markowitz portfolio selection model to include variable transactions' costs, short sales, leverage polices and taxes. *Journal of Finance*, 25(5):1005–1027, 1970.

[46] Kelly Price, Barbara Price, and Timothy J. Nantell. Variance and lower partial moment measures of systematic risk: Some analytical and empirical results. *Journal of Finance*, 37(3):843–855, June 1982.

[47] Richard Roll. A mean/variance analysis of tracking error. *Journal of Portfolio Management*, 18(3):13–22, 1992.

[48] A. D. Roy. Safety first and the holding of assets. *Econometrica*, 20(3):431–449, 1952.

[49] Markus Rudolf. *Algorithms for Portfolio Optimization and Portfolio Insurance.* Verlag Paul Haupt, Bern, Switzerland, 1st edition, 1994.

[50] Alexander Schrijver. *Theory of Linear and Integer Programming.* John Wiley and Sons, New York, 1986.

[51] William F. Sharpe. A simplified model for portfolio analysis. *Management Science*, 9(2):277–293, 1963.

[52] William F. Sharpe. Mutual fund performance. *Journal of Business*, 39(1):119–138, 1966.

[53] William F. Sharpe. A linear programming algorithm for mutual fund portfolio selection. *Management Science*, 13(7):499–510, March 1967.

[54] Karl Shell. Selected elementary topics in the theory of economic decision making under uncertainty. In Giorgio P. Szegö and Karl Shell, editors, *Mathematical Methods in Investment and Finance.* America Elsevier Publishing Company, New York, 1972.

[55] Frank A. Sortino and Robert van der Meer. Downside risk. *Journal of Portfolio Management*, 17(4):27–32, 1991.

[56] G. W. Stewart. On the stability of sequential updates and downdates. Technical Report TR-94-30, Department of Computer Science, University of Maryland, College Park, MD, 1994.

[57] S. C. Tsiang. The rationale of the mean-standard deviation analysis, skewness preference, and the demand for money. *American Economic Review*, 62(1):354–371, 1972.