MSCI Barra

Barra Optimizer

Practical Convex Quadratic Programming

version 1.0b

CLIENT SERVICE ASSISTANCE IS AVAILABLE 24 HOURS A DAY

Americas 1.888.588.4567 (toll free)	Chicago + 1.312.706.4999	Milan + 39.027.633.5429	Sydney + 61.2.9220.9333
Amsterdam + 31.20.462.1382	Dubai +971.4.683.4861	Montreal + 1.514.847.7506	Tokyo + 813.5424.5470
Atlanta + 1.404.949.4529	Frankfurt + 49.69.2166.5325	New York + 1.212.762.5790	Toronto + 1.416.943.8390
Boston + 1.617.856.8716	Geneva + 41.22.817.9800	Paris 0800.91.59.17 (toll free)	Zurich + 41.44.220.9300
Cape Town + 27.21.683.3245	Hong Kong + 852.2848.7333	San Francisco + 1.415.576.2323	
China Netcom 10800.852.1032 (toll free)	London + 44.20.7618.2222	São Paulo + 5511.3048.6080	
China Telecom 10800.152.1032 (toll free)	Madrid + 34.91.700.7275	Singapore + 65.6834.6777	

clientservice@mscibarra.com

www.mscibarra.com

Practical
Convex Quadratic
Programming:

Barra Optimizer for Portfolio Optimization

Version 1.6

June 10, 2004

Scott Liu[♠]

Barra, Optimization Research

♠ Special thanks go to Mike Levinson and Manjula Chandramouli for their tremendous editorial assistance, and to Leo Kopman, Rong Xu, Wei Yuan, as well as many others for their valuable comments and suggestions.

TABLE OF CONTENTS

1. INTRODUCTION

Barra's products are used world-wide by leaders throughout the Investment industry. Barra's Optimizer is at the core of our product offerings and is designed to facilitate the portfolio construction process. It incorporates state-of-the-art optimization technology and all of the special structures of Barra's multiple factor risk models. Barra's Optimizer is relatively easy to use and is designed to provide accurate, fast, and theoretically robust solutions to a wide array of portfolio optimization problems. It can solve generic convex quadratic programming problems with which portfolio managers typically are faced when performing portfolio optimization as part of their regular portfolio construction process. More importantly, it offers innovative, high quality heuristics to solve very complex portfolio construction problems. These include but are not limited to optimization with paring constraints, hedge optimization, downside risk models, after-tax optimization, and Parametric Optimization.

Modern portfolio theory subscribes to the portfolio optimization approach of maximizing *risk-adjusted return* subject to certain constraints. This theory is an outgrowth of the work of Harry M. Markowitz, a 1990 winner of the Nobel Prize in Economic Sciences. Markowitz had a groundbreaking approach to the use of statistical decision theory in portfolio construction. He proposed what is now considered to be a classical mean-variance optimization problem, which Barra has adapted to a convex quadratic programming problem framework. This paper only will describe methodologies used for generic *Convex Quadratic Programming* problems (*CQP*). It will not provide details of the more complex algorithms used in Barra's Optimizer such as those with special types of constraints including paring, hedge optimization, and return and risk target problems. The prerequisites for this paper are basic knowledge of optimization and computational mathematics.

The remainder of the paper is organized as follows. Section 1.1 will provide a general description of the portfolio optimization problem. Section 2 will present common characteristics shared by most Barra risk models, salient features of the Barra Optimizer, and the mathematical formulation of Barra Portfolio Optimization problems. Section 3 is devoted to the complexity of portfolio optimization problems. As such, it will provide a classification of these optimization problems. Section 4 presents a general Convex Quadratic Programming problem and describes those conditions that an optimal portfolio should satisfy: the *optimality conditions*. Section 5 will explain active set methods, simplicial decomposition methods, and interior point methods for general convex quadratic programming problems. The contents for this section may be too technical for certain readers. Section 6 will elaborate on some implementation details for taking advantage of Barra risk models when utilizing methods described in Section 5 to solve portfolio optimization problems.

1.1. A General Portfolio Optimization Problem

Since the pioneering work on Modern Portfolio Theory by Markowitz (1959), increasing numbers of portfolio managers are using mean-variance optimization as a tool with which to exercise significant control over their portfolio construction and selection processes. For example, a portfolio manager might use optimization as a method by which to incorporate a minimum forecast rate of return while minimizing the risk of major losses on their investments. Such a manager also might use optimization to maximize the expected return of their portfolio while limiting tracking error to within a certain range. A portfolio manager must decide upon the percentage of total available capital to invest in each of various assets within their investable universe. To achieve optimal performance, portfolio managers must not only consider the trade-off between risk and return, they also should consider the interaction of factors such as risk and return with transaction costs, asset constraints (e.g., long only or long and short), and tax implications. Most importantly, Barra's Optimization tools are very useful to portfolio managers regardless of the level of complexity of their optimization problems.

The essential elements of most portfolio optimization problems can be represented by one of the following categories:

Minimum Risk:

Minimize Risk + Transaction Costs
subject to: Return >= a given level.
Any other constraints required by the user.

Maximum Return:

Maximize Return –Transaction Costs
subject to: Risk <= a given level.
Any other constraints required by the user.

Maximum Sharpe Ratio:

Maximize Sharpe Ratio
subject to: Any constraints required by the user.

A general form of the portfolio optimization problem entails the maximization of risk-adjusted return.

Maximum Risk-Adjusted Return:

Maximize Utility = Return – λ*Risk –Transaction Costs,
where λ is a risk aversion.
subject to: Any constraints required by the user.

Transaction costs typically are piecewise linear functions. Constraints are any additional conditions that might be imposed either by portfolio managers, investors, or as a result of regulations. A variety of constraints might be imposed such as on holdings (e.g., a limit of 5% to any sector or region), on turnover (e.g., total turnover should not exceed 50%), or on number of assets held (e.g., the optimal portfolio cannot hold more than 50 assets).

2. Barra Portfolio Optimization Model and Optimizer

In this section, we will present a high-level description of Barra's risk models as well as Barra's approach to portfolio optimization. We will then provide a mathematical formulation of Barra's standard portfolio optimization problem. Although Barra's optimizer can handle problems that utilize alternative risk models, we only will describe a portfolio optimization problem using Barra's risk models in this section.

2.1. Barra Risk Model

Modern Portfolio Theory defines a portfolio's risk as the variance of its return. Barra's approach to modeling risk is consistent with Modern Portfolio Theory, but represents an additional refinement. This is evident in Barra's definition of portfolio risk as a *decomposition* of the variance of its return. Using Barra's multiple factor model, the return (r) of a portfolio can be decomposed into both *common-factor* return (Xf) and asset-*specific* return (u) components as: $r = Xf + u$.

Barra's multiple factor modeling approach entails the creation of a *factor covariance matrix*. Essentially, this is a short-term risk forecast that describes the trade-offs of each *common factor* within the model. Barra's approach also requires the periodic calculation of *exposures* of each asset to these common factors. Finally, Barra provides a forecast of the level of each asset's *specific risk*, a measure that is independent of an asset's exposure to any model common factor. Consequently, a covariance matrix for all assets in Barra's model provides a short-term risk forecast which can be used in gauging the contribution of each asset to a prospective portfolio's overall risk to facilitate the portfolio construction process. This *covariance matrix* is defined as:

$$XFX^T + D \tag{2.1}$$

Assume that there are n assets from which to select an optimal portfolio, or *optimization universe*. Also assume that our multiple factor model consists of k common factors where,

X	=	$n \times k$ matrix of asset exposures to the factors,
F	=	$k \times k$ *positive semi-definite* factor covariance matrix, and
D	=	$n \times n$ *positive semi-definite* covariance matrix representing a forecast of asset specific risk.

In the previous section, we mentioned that portfolio risk level is a critical element within the portfolio optimization problem. The typical portfolio manager might take one of two risk perspectives during the portfolio optimization process. That of the *total risk* of the portfolio, in which only the portfolio holdings are considered, and the benchmark holdings are treated as irrelevant for optimization purposes. The other is *active risk*, or *tracking error*, in which the difference between the portfolio holdings and those of the benchmark is given primary consideration in the optimization problem. To accommodate both the total risk and active risk (tracking error) perspectives, the risk term in the utility function of a portfolio optimization problem has two modes:

Total Risk: $h^T (\lambda_F XFX^T + \lambda_D D) h$ (2.2.1)

Active Risk: $(h - h_B)^T (\lambda_F XFX^T + \lambda_D D)(h - h_B)$ (2.2.2)

where,

λ_F = common factor risk aversion parameter,

λ_D = specific risk aversion parameter,

h = $n \times 1$ vector of managed portfolio's holdings, and

h_B = $n \times 1$ vector of normal (benchmark) portfolio's holdings

Please note that Barra's definitions of both total risk and active risk incorporate the use of both a common factor and a specific *risk aversion parameter*. These risk aversion parameters are scalar variables whose value is determined by the portfolio manager at the outset. The introduction of risk aversion parameters into Barra's portfolio optimization models allows the portfolio manager to incorporate a numeric representation of personal risk preferences into the portfolio optimization process. It also provides the opportunity for them to quantify their relative aversion to common factor risk vis-à-vis specific risk. Consequently, these risk aversion parameters are important tools that are available to assist the portfolio manager in the construction of an optimal portfolio that is consistent with their goals.

2.2. Barra Portfolio Optimization Models

2.2.1 Standard Portfolio Optimization

By introducing a risk aversion parameter, we can combine maximum return, minimum risk, and maximum Sharpe ratio problems into a maximum risk-adjusted return problem. Typical standard portfolio optimization problems have linear or convex quadratic utility functions with only linear or piecewise linear constraints (including bounds). At Barra, we formulate and solve a more general form of the standard portfolio optimization problem in the following fashion:

Maximize Utility function
subject to: Turnover Constraint,
Transaction Cost Limit Constraint,
General Piecewise Linear Constraints,
Holding Constraints,
General Linear Constraints,
Factor Constraints,
Bounds on Assets.

The utility function in Barra's standard portfolio optimization problem has the following terms:

(1) Linear coefficients (e.g., alphas)

(2) Two types of portfolio risk terms (defined in (2.2.1), (2.2.2))

(3) Separable, convex, piecewise linear terms with multiple breakpoints (this enables Barra's optimizer to incorporate simple buy and sell transaction costs as well as complex transaction cost structures from the Barra Market Impact Model);

(4) Symmetric or asymmetric, quadratic or linear only, penalties on any deviations from the target values of constraint slacks and factors.

The Barra Optimizer will utilize standard portfolio optimization techniques to arrive at an optimal portfolio. A standard portfolio optimization problem can be solved using any of the methods described in Section 5.

2.2.2. Other Special Features in Barra Optimization Models

Barra's optimizer is designed to solve the standard portfolio optimization problems described earlier. However, it also can be used to solve much more complicated and practical problems, such as:

- After-tax optimization
- Hedge optimization
- Risk target or return targets
- Risk or return-constrained problems
- Efficient frontier (parametric optimization)
- Optimal round-lotting
- Paring problems, including the setting of a limit to the maximum number of assets, the minimum threshold holding level, the minimum transaction threshold level, and the maximum number of trades.
- Soft constraints
- Downside risk models

For detailed definitions of these problems, please see *Barra Optimizer User's Manual.* Most of these special types of problems cannot be solved using only methods described in Section 5. Optimization problems requiring the use of risk targets (or risk-constraints), for example, frequently must be solved using more complicated techniques. Other problems, such as those that require paring, are solved using specially designed heuristics. As a result, the final solution provided by the Barra Optimizer is not necessarily optimal.

2.3. Mathematical Formulation of a Standard Portfolio Optimization Problem

A general, mathematical representation of Barra's standard portfolio optimization problem can be presented as follows:

$$\underset{h,w,s}{\text{Maximize}} \qquad \alpha^T h - 100\ Risk(h) - \sum_{i=1}^{n} TC_i(h_i) - Ps(s) - Pw(w)$$

subject to:

$$\sum_{i=1,\, i\neq cash}^{n} \left| h_{0i} - h_i \right| \le 2T \qquad \text{(Turnover Constraint)} \tag{2.3}$$

$$\sum_{i=1}^{n} TC_i(h_i) \le TCB \qquad \text{(Transaction Cost Limit Constraint)} \tag{2.4}$$

$$l_a \le Ah \le u_a \qquad \text{(General Linear Constraints)} \tag{2.5}$$

$$l_f \le X'h \le u_f \qquad \text{(Factor Slack Constraints)} \tag{2.6}$$

$$l_h \le h \le u_h \qquad \text{(Bound Constraints)} \tag{2.7}$$

where,

n	=	number of assets (regular assets + composites)
m	=	number of general constraints
k	=	number of common factors
h, h_0	=	$n\times 1$ vector of portfolio holdings and initial portfolio holdings
s	=	$m\times 1$ vector of slackness of constraints
w	=	$k\times 1$ vector of slackness for factor constraints
α	=	$n\times 1$ vector of asset alphas or excess alphas
$Risk(h)$	=	risk term (variance) in utility function defined by (2.2)
$TC_i(h_i)$	=	piecewise linear transaction cost function for asset i, $i = 1, 2, \ldots, n$
$Ps(s)$	=	quadratic penalty function for constraint slack s
$Pw(w)$	=	quadratic penalty function for factor slack w
A	=	$m\times n$ constraint matrix for continuous general constraints (including the holding constraint)
T	=	maximum allowable turnover
TCB	=	upper bound on transaction cost limit constraint
l_h, u_h	=	lower and upper bounds on assets
l_a, u_a	=	lower and upper bounds on constraints
l_f, u_f	=	lower and upper bounds on portfolio factor exposures

Typical piecewise linear functions appearing in a standard portfolio optimization problem are transaction cost terms within the utility expression, as well as turnover and transaction cost limit constraints. A simple transaction cost function might incorporate the use of one buy and one sell rate for each asset. More complex transaction cost functions, such as those encountered in Barra's Market Impact Model, have multiple buy and sell rates for each asset. Barra's optimizer also supports the use of quadratic transaction cost functions.

3. Classification of Portfolio Optimization Problems

The complexity of portfolio optimization problems depends upon both the risk model and type of constraints utilized. For example, problems using paring constraints will be much more difficult to solve than standard portfolio optimization problems. In this section, we discuss the complexity of portfolio optimization problems by viewing them within the framework of general optimization problems. We will first define general optimization problems. Then, we will describe the classification of optimization problems into various categories. Almost all portfolio optimization problems belong to these categories. Finally, we will demonstrate that a standard portfolio optimization problem is actually a convex quadratic programming problem.

3.1. Optimization Problem Definition

In a typical optimization problem, our goal is either to *maximize* or to *minimize* the value of a function (e.g., maximize profit or minimize risk) under a set of conditions imposed according to the nature of the problem to be addressed. The quantity that we wish to maximize or minimize is known as the value of the *objective function*. The additional conditions imposed upon this optimization problem are known as *constraints*. Mathematically, an optimization problem can be written as:

$$
\text{(NLP)} \quad \begin{array}{ll} \text{Minimize} & f(x) \\ \text{subject to:} & h_i(x) = 0, i = 1, 2, ..., m \\ & g_j(x) \geq 0, \; j = 1, 2, ..., r, \end{array}
$$

where x represents the decision variables, $f(x)$ is the objective function, which might represent a utility function, and $h_i(x) = 0$ and $g_j(x) \geq 0$ represent constraints, which might reflect financial, marketing, or many other considerations.

3.2. Classification of Optimization Problems

The methods used to solve the general form of the optimization problem typically are very complex and require considerable computational effort. We can classify optimization problems according to the nature of their objective function and associated constraints into the following types: linear programming, quadratic programming, nonlinear programming, and integer programming. A brief description of each of these optimizations problem types is as follows:

Linear Programming (LP) Problems

The optimization problem is referred to as a *linear programming* (LP) problem if the objective function and all constraints are either linear or piecewise linear functions. Risk-indifferent portfolio strategies might result from the application of linear programming to the portfolio optimization procedure. An example of this is an LP that maximizes return subject to a linear limitation of the assets held. Other examples of LP problems are fixed-income portfolio immunization and horizon matching.

Quadratic Programming (QP) Problems

A *quadratic programming* (QP) problem is one in which the objective function is a quadratic function and all constraint functions are linear or piecewise linear. The QP problem is referred to as a *convex quadratic programming* problem, or CQP, if the objective function also is either convex for a minimization problem, or concave for a maximization problem. The CQP problem is a generic form of the Markowitz portfolio optimization problem.

Nonlinear Programming (NLP) Problems

Nonlinear programming problems (NLP) are general constrained optimization problems in which one or more functions are nonlinear. Risk target, risk constrained, and risk budgeting problems are examples of NLPs because the risk constraints in each case are quadratic or nonlinear functions. A quadratic programming problem is a special type of nonlinear programming problem.

Integer Programming (IP) Problems

We assume that the decision variables are continuous in each of the three problem types summarized above. The problem would be referred to as an *(Mixed) Integer Programming* (IP) problem if one or more decision variables were to assume only integer values. For example, in round-lotting, all decision variables must be multiples of a given lot size. Paring problems also can be formulated as integer programs since they limit the number of assets in the optimal portfolio.

3.3. Convex Quadratic Programming and Portfolio Optimization

If one were to incorporate the optimization problem classifications introduced within the previous section into our prior discourse, one could reach the following conclusions. In general, the variance term that one would hope to *minimize* in the Markowitz mean-variance portfolio optimization problem is a *convex quadratic function* and can be represented as positive semi-definite covariance matrix. In addition, since one hopes to *maximize* the value of the utility function, it can be represented as a *concave quadratic function*. Finally, if all constraints are either linear or piecewise linear functions, one can conclude that the maximum risk-adjusted return portfolio optimization problem is a *convex quadratic programming* problem. Therefore, one can use any *convex quadratic programming* methodology to solve the portfolio optimization problem.

We will now apply these conclusions to a review of Barra's standard portfolio optimization problem. When reviewing our earlier discussion on Barra risk models (Section 2.1), we can see that the risk term in the utility function is a *convex quadratic function*. Since Barra's standard portfolio optimization problems only allow the imposition of quadratic or linear penalties and they are to be *minimized*, the penalty terms also are *convex quadratic* functions. In addition, the corresponding utility function whose value is to be *maximized* is a *concave quadratic* function. As we alluded to in our earlier discussions, all general constraints and factor constraints (Sections 2.2-2.7) to be utilized in Barra's standard portfolio optimization problem are either linear or piecewise linear functions. Consequently, the definition provided in Section 3.2 would allow us to conclude that the Barra standard portfolio optimization problem is a *convex quadratic programming* problem.

The next two sections focus entirely on convex quadratic programming problems. Although the ideas and methodologies to be described therein may be applied or extended to more complicated portfolio optimization problems, detailed methods are beyond the scope of this paper.

.

4. OPTIMALITY CONDITIONS

We stated the rationale in the previous section that led us to conclude the Barra standard portfolio optimization problem is a *convex quadratic programming* (CQP) problem. In this section, we focus our discussion on *optimality conditions*. These are mathematical conditions that must be satisfied by an optimal solution to a generic CQP. We also define the *dual* approach of a convex quadratic programming problem.

4.1. A General Convex Quadratic Programming Problem

In order to simplify our discussion, we only will present the optimality conditions for a convex quadratic programming problem in the following form:

$$\text{(CQP)} \quad \begin{array}{ll} \underset{h}{\text{Minimize}} & c^T h + 0.5 * h^T Q h \\ \text{subject to:} & Ah \le b, \end{array} \tag{4.1}$$

where c, h are $n \times 1$ vectors, Q is an $n \times n$ positive semi-definite matrix, A is an $m \times n$ matrix, and b is an $m \times 1$ vector. Note that if $Q = 0$, the above equation reduces to a linear programming problem.

Please note that more general convex quadratic programming problems can be transformed readily into the form given by (4.1). For example,

(1) Maximization problem: A maximization problem can be transformed into a minimization problem represented by a general form of a CQP simply by reversing the signs as follows: Maximize f(x) = -Minimize {-f(x)}.

(2) Equality constraints: A general equality linear constraint can be transformed into two linear inequality constraints.

(3) Asset bound constraints: $l_h \le h \le u_h$ can be treated as two sets of general inequality constraints. However, for efficiency, we treat asset bounds $l_h \le h \le u_h$ as special bounding constraints. We will not discuss the details here.

(4) Piecewise linear function: In theory, we can transform a piecewise linear function into a linear function by adding new variables. However, doing so would cause the requisite number of variables within the problem to be doubled or tripled. At Barra, we find it much more computationally efficient to treat piecewise linear functions, such as transaction costs, implicitly. We will not discuss the details here.

4.2. Lagrangian Function and Dual Problems

In this section, we demonstrate the use of a Lagrangian function to facilitate the transformation of a given *convex quadratic programming* (CQP) problem into a *dual problem*. To achieve this result expeditiously, we re-state the optimality conditions for a convex quadratic programming problem defined by (4.1) in terms of the following Lagrangian function:

$$L(h,\pi) = c^T h + 0.5 h^T Q h + \pi^T (Ah - b).$$

After including the associated constraints, we know that the gradient of the Lagrangian function at an optimal point should be zero. The result is as follows:

$$\nabla_{\mathrm{h}} L(h,\pi) = c + Qh + A^T \pi = 0.$$

We then introduce the following additional constraint in order to penalize infeasible portfolios: $\pi \ge 0$. The following *quadratic programming* problem results from the application of the concept of Lagrangian duality:

Dual Problem

$$\text{(QD)} \quad \begin{array}{ll} \text{Maximize} & b^T\pi - 0.5 * h^T Qh \\ \text{subject to:} & Qh + c + A^T\pi = 0, \\ & \pi \geq 0. \end{array}$$

We refer to the *convex quadratic programming* problem (CQP) as a *primal problem*. Also, the quadratic programming problem (QD) is a *dual problem* of (CQP). Please note that a dual problem of a quadratic programming problem also is a quadratic programming problem.

4.3. Karush-Kuhn-Tucker (KKT) Conditions

KKT conditions are the only *necessary* first-order conditions that must be satisfied in order for a solution to a quadratic programming problem to be optimal. We can formally state the KKT conditions as follows:

Theorem 1: (Karush-Kuhn-Tucker Conditions)

For a solution h* to be an optimal solution to a convex quadratic programming problem, h* should fulfill the following three conditions:

(I) Primal Feasibility--All primal constraints defined by the original optimization problem should be satisfied;

(II) Dual Feasibility--All conditions for the implied dual variables should be satisfied; and

(III) Complementary slackness conditions.

Mathematically, we can write the KKT conditions as follows:

h* is an optimal solution to (CQP), as defined above, and (h*, π) is an optimal solution to (QD) if and only if the following conditions hold:

$$(I) \quad Ah \leq b;$$
$$(II) \quad Qh + c + A^T\pi = 0, \pi \geq 0;$$
$$(III) \quad \pi^T(Ah - b) = 0;$$

We refer to a solution that satisfies KKT conditions as a **KKT point**. In general, KKT conditions only are *necessary* but not always sufficient conditions for a solution to be optimal. In other words, a KKT point is not necessarily a *global* optimal solution. In fact, it is true in general that a KKT point is not necessarily even a *local* optimal solution. However, since a standard portfolio optimization problem is a convex quadratic programming problem, a KKT solution also is an optimal solution. Furthermore, all local optimal solutions also are global optimal solutions. In theory, a convex quadratic programming problem will have a unique global optimal solution if it has positive definite Hessian. In other words, Q is strictly positive definite. *Otherwise, it may have multiple solutions with the same optimal objective function value.* In practice, however, since we must use tolerance for a stopping rule, there may exist many portfolios with nearly the same optimal utility but slightly different weights, even if Q is strictly positive definite.

You may recall that one conclusion reached in the previous section was that Barra's standard portfolio optimization problem is a *convex quadratic programming* problem. As may be concluded from the section 2.1 discussion, Barra's standard optimization problem has a positive semi-definite Hessian (i.e., Q). Consequently, Barra's standard optimization problem sometimes may provide multiple solutions having either the same optimal objective function value or similar forecast risk levels.

In general, a method for solving a convex quadratic programming problem will begin with a solution that satisfies two of the KKT conditions and will use an iterative procedure to find a solution that also satisfies the remaining condition. Those methods applicable to convex quadratic programming can be classified as either a *primal method*, a *dual method*, or a *primal-dual method* depending upon which of the two conditions the iterative solution satisfies.

Primal methods begin with a feasible solution that satisfies conditions (I) and (III) and iteratively improve upon it within the *primal* feasible region. The *primal* algorithms terminate when they encounter a solution that also satisfies condition (II). In contrast, *dual methods* begin with a solution that satisfies conditions (II) and (III), and continues to improve upon it within the *dual* feasible region. The *dual* algorithms terminate when they encounter a solution that also satisfies condition (I). As the name indicates, *primal-dual methods* begin with a solution that satisfies both *primal and dual* feasibility and continues to reduce either the error in satisfying the condition (III), or the complementary slackness condition. Most of the algorithms utilized by the Barra Optimizer are *primal* algorithms. Consequently, if one were able to cease the Barra Optimizer's iterative solution process prematurely, the most recently calculated solution would be a feasible portfolio at the very least, if not an optimal one.

5. Methods for Convex Quadratic Programming

In this section, we will describe several efficient methods for solving a convex quadratic programming problem. In order to maintain the generality of our discussion, we only will describe those algorithms that can be used for solving convex quadratic programming problems in the form given by (4.1).

5.1. CQP with Equality Constraints Only

If a quadratic programming problem only has equality constraints, it becomes a very easy problem. In fact, if the Hessian Q also is positive definite, we can obtain a closed form solution for an *equality-constrained quadratic programming* problem (EQP).

Consider the following CQP with only equality constraints:

$$\begin{array}{lll} & \text{Minimize} & c^T h + 0.5 h^T Q h \\ \text{(EQP)} & \text{subject to:} & Ah = b. \end{array} \qquad (5.1)$$

In this case, it is clear that a unique solution exists if the matrix A is of full rank and the matrix Q is positive definite on the subspace R = {h: $Ah=0$}. Consistent with the *optimality conditions* enumerated in Section 4, in order for a solution to be optimal for EQP, it should satisfy:

$$\begin{array}{l} Qh + c \ + A^T \pi = 0 \\ Ah \qquad\qquad = b. \end{array}$$

This linear equation system can be rewritten in matrix form as follows:

$$\begin{bmatrix} Q & A^T \\ A & 0 \end{bmatrix} \begin{pmatrix} h \\ \pi \end{pmatrix} = \begin{pmatrix} -c \\ b \end{pmatrix}. \qquad (5.2)$$

There are several methods for solving this system of linear equations. We will provide an elaboration in the next section.

5.2. Active Set Methods

A general convex quadratic programming problem may have a variety of inequality constraints, including a limit on total turnover and asset bounds. One efficient method for solving a CQP with inequality constraints is the *active set method*. We will describe a generic active set method in this section. We also will discuss the most popular active set methods, the *range space* and the *null space* methods.

Active Set: If a constraint is binding for a given solution h_k, we refer to this constraint an *active constraint relative to* h_k. The active set for a given solution is the set of all active constraints. Mathematically, an active set I_k of h_k for CQP (4.1) is defined as

$$I_k = \{i \mid A_i h_k = b_i, 1 \le i \le m\}.$$

Please note that if there are equality constraints, the active set always will incorporate these equality constraints, and possibly some of the inequality constraints.

An active set method algorithm begins with a given feasible solution and attempts to solve the corresponding equality-constrained quadratic programming problem for the given active set. Assume this new solution $h_{k+1} = h_k + d_k$ is an optimal solution for the corresponding EQP. Then the *searching direction*, d_k, must solve the following EQP:

$$
\begin{array}{lll}
 & \text{Minimize} & g_k^T d_k + 0.5 d_k^T Q d_k \\
(\text{EQP})_\text{k} & \text{subject to:} & A_k d_k = 0,
\end{array}
\qquad (5.3)
$$

where $g_k = Qh_k + c$ is the gradient for objective function at h_k.

The discussion in Section 5.1 indicates that a solution to $(\text{EQP})_k$ can be determined by solving a linear equation system similar to (5.2) with any efficient method. If d_k is non-zero and $h_{k+1} = h_k + d_k$ is feasible for all constraints, then $h_{k+1} = h_k + d_k$ becomes the new feasible solution. However, if $h_{k+1} = h_k + d_k$ is not feasible, a line search of the form $h_{k+1} = h_k + \tau_k d_k$ is made. Concurrently, the step length τ_k is selected so as to be as large as possible while maintaining feasibility. In general, the step length is obtained by using the following *minimum ratio test* (MRT):

$$
\tau_k = \min_{i \notin I_k} \{1, (b_i - A_i h_{ki}) / A_i d_{ki} \ \mid \text{for all } A_i d_{ki} > 0\}. \qquad (5.4)
$$

A new inequality constraint is binding (i.e., it *becomes active*) at the new solution $h_{k+1} = h_k + \tau_k d_k$. Subsequently, this constraint is added to the active set I_{k+1}. This solution process may continue to proceed in this fashion, while iteratively adding additional constraints to the active set. Eventually, since there only are a finite number of constraints, the process will terminate with a solution that is a minimum over the current active set. Upon achieving such a solution, it must be the case that $d_k = 0$. Subsequently, the corresponding dual solution, π_k, is examined at the current solution h_k. This examination process is referred to as *pricing*. If the dual solution π_k is feasible (i.e. π_k >=0), then the current solution h_k is an optimal solution to the CQP. In this instance, it follows that (h_k, π_k) is an optimal solution to the dual problem (QD). However, if at least one component of π_k is negative, then one or more of the constraints corresponding to the negative components of π_k will be dropped from the current active set and an additional iteration of the active set method will be performed.

5.2.1. A General Active Set method

Below is a summary of the steps which comprise the active set method.

Active Set Method:

Step 0: (*Initialization*) Obtain an initial feasible solution h_0. Form the active set I_0. Set k=0.

Step 1: (*Obtain a direction*) Solve the equality-constrained quadratic programming problem as defined by (5.3). In other words, solve the following linear equation system for the direction d_k and the dual vector π_k.

$$\begin{bmatrix} Q & A_k^T \\ A_k & 0 \end{bmatrix} \begin{pmatrix} d_k \\ \pi_k \end{pmatrix} = \begin{pmatrix} -g_k \\ 0 \end{pmatrix} \tag{5.5}$$

If d_k=0, go to **Step 3**.

Step 2: (*Line search for step length*) Set $h_{k+1} = h_k + \tau_k d_k$, where step length τ_k is given by the minimum ratio test (MRT) defined in (5.4). If $\tau_k \geq 1$, go to **Step 3**.

Since $\tau_k < 1$, add the binding consraint i_0 into current active set I_k, *i.e.* $I_{k+1} = I_k \cup \{i_0\}$, where i_0 is the index such that $\tau_k = (b_{i_0} - A_{i_0} h_{ki_0}) / A_{i_0} d_{ki_0}$.

Set k = k+1, go to **Step 1**.

Step 3: (*Pricing*) Determine whether the dual vector π_k satisfies $\pi_{ki} \geq 0$ for all $i \in I_k$

If so, the current solution h_k is optimal. **STOP**.

Otherwise, let $\pi_{kq} = \min\{\pi_{ki}, i \in I_k\}$.

Drop the active constraint q from the current active set I_k to form $I_{k+1} = I_k - \{q\}$. Set k = k +1, go to **Step 1**.

5.2.2. Range-Space and Null-Space Active Set methods

For all iterations of the active set method described above, it is necessary to solve a linear system of equations (5.5) to obtain the searching direction and dual variables. There are many efficient approaches to the solution of (5.5) (or (5.3)) which will result from the use of a variety of active set methods. Two of the most popular active set methods are as follows:

Range-Space and Null-Space Active Set Methods

Let A be an $m \times n$ ($m < n$) full rank matrix. There is an $n \times n$ orthonormal matrix (Y, Z) such that

$$A(Y \quad Z) = (L \quad 0), \tag{5.6}$$

where L is a $m \times m$ non-singular lower-triangular matrix, Y is an $n \times m$ matrix, and Z is an $n \times (n\text{-}m)$ matrix. In other words, we have

$$AY = L, \quad \text{and} \quad AZ = 0. \tag{5.7}$$

So Y forms a basis for the *range-space* of A while Z forms a basis for *null-space* of A.

Null-Space Active Set Method:

Since we assume that the rows of A are linearly independent, it follows that the rows of sub-matrix A_k also must be linearly independent. Let Z_k denote a matrix whose columns form a basis for the set of vectors orthogonal to the rows of A_k. Any direction d_k that satisfies the linear constraints in (5.5) also should satisfy the equation

$$d_k = Z_k d_{zk} \tag{5.8}$$

for some $t \times 1$ vector d_{zk}, where t is number of rows of A_k. Substituting (5.8) into (5.5) and solving the systems yields the following:

$$d_k = -Z_k (Z_k^T Q Z_k)^{-1} Z_k^T g_k \tag{5.9.1}$$
$$\pi_k = -(A_k A_k^T)^{-1} A_k (Q d_k + g_k). \tag{5.9.2}$$

When we use formula (5.9) to solve the linear equation system (5.5) for direction and dual variables, the active set method is referred to as the *null-space active set* method.

Range-Space Active Set Method

When Q is positive definite and the rows of A are linearly independent, we can obtain the inverse matrix of the left hand side of (5.5) as follows:

$$\begin{bmatrix} Q & A_k^T \\ A_k & 0 \end{bmatrix}^{-1} = \begin{bmatrix} Q^{-1} - Q^{-1} A_k^T (A_k Q^{-1} A_k^T)^{-1} A_k Q^{-1} & Q^{-1} A_k^T (A_k Q^{-1} A_k^T)^{-1} \\ (A_k Q^{-1} A_k^T)^{-1} A_k Q^{-1} & -(A_k Q^{-1} A_k^T)^{-1} \end{bmatrix}$$

Consequently, we can obtain the solution to (5.5) as:

$$d_k = Q^{-1}(A_k^T(A_kQ^{-1}A_k^T)^{-1}A_kQ^{-1}g_k - g_k) \tag{5.10.1}$$
$$\pi_k = -(A_kQ^{-1}A_k^T)^{-1}A_kQ^{-1}g_k. \tag{5.10.2}$$

Let $H_k = Q^{-1}A_k^T(A_kQ^{-1}A_k^T)^{-1}A_kQ^{-1} - Q^{-1}$. Then we have

$$d_k = H_k g_k \tag{5.11.1}$$
$$\pi_k = -(A_kQ^{-1}A_k^T)^{-1}A_kQ^{-1}g_k. \tag{5.11.2}$$

Next, we obtain $A_k = L_kY_k^T$ using both the definition of Y and (5.7). For practical computation purposes, we can substitute it into H_k and obtain the following:

$$H_k = Q^{-1}Y_k(Y_k^TQ^{-1}Y_k)^{-1}Y_k^TQ^{-1} - Q^{-1}. \tag{5.12}$$

Since Y_k is a sub-matrix of an orthogonal matrix, the condition number of $Y_k^TQ^{-1}Y_k$ is no worse than that of *Q*. The computation of (5.12) may be implemented by storing the Cholesky factors of *Q*, and updating L_k, Y_k and the Cholesky factors of $Y_k^TQ^{-1}Y_k$ as constraints are added to and deleted from the active set.

5.2.3. Comparison of Range-Space and Null-Space Active Methods

The computer storage requirement for both the range-space and the null-space methods will depend upon the number of active constraints, *t*. If *t* is very small relative to the number of assets, *n*, then the range-space method will require less storage because the size of $Y_k^TQ^{-1}Y_k$ is smaller than that of $Z_k^TQZ_k$. For a typical portfolio optimization problem, there are relatively few constraints compared to the number of assets. For example, we may want to select a portfolio from a large universe of 500 assets while utilizing several factor and other constraints, for a total of 10 active constraints. In this instance, the size of $Y_k^TQ^{-1}Y_k$ would be 10-by-10, while the size of $Z_k^TQZ_k$ would be 490-by-490. In addition, the range-space active set method can take advantage of the special structure of a covariance matrix. As discussed in Section 2, the covariance matrix of Barra's risk model does indeed have a special structure. Consequently, the range-space active set method generally is preferable for use in solving portfolio optimization problems.

5.3. Simplicial Decomposition Method

In this section, we will describe a *simplicial decomposition method* for solving a CQP. Essentially, this entails solving a sequence of linear programming problems and small-size quadratic programming programs. Once again, we only consider a convex quadratic programming problem in the following form:

$$\text{(CQP)} \qquad \begin{array}{ll} \text{Minimize} & f(h) = c^T h + 0.5 * h^T Q h \\ \text{subject to:} & Ah \leq b. \end{array}$$

5.3.1 Searching Direction and Vertex of Feasible Region

Assume that we have arrived at a feasible solution h_k, and that we want to obtain a new feasible solution with a lower objective function value. Let h_{k+1} represent this improved feasible soluton. Then it must be the case that

$$g_k^T (h_{k+1} - h_k) \leq 0 \text{ and } Ah_{k+1} \leq b, \tag{5.13}$$

where $g_k = Qh_k + c$ is the gradient of the objective function at h_k. Consequently, a soluton to the following linear programming problem will yield this improved feasible solution:

Subproblem:

$$\text{(SDLP)} \qquad \begin{array}{ll} \text{Minimize} & \mathrm{g}_\mathrm{k}^\mathrm{T} h \\ \text{subject to:} & Ah \leq b. \end{array}$$

Let h^* be the optimal solution to SDLP. Then h^* also is a feasible solution to the original problem CQP as well as a vertex of the feasible region. If

$$(h^* - h_k)^T g_k \geq 0, \tag{5.14}$$

then the implication is that moving along the feasible region in any direction away from h_k will increase the objective function value. Therefore, h_k must be an optimal solution to CQP.

5.3.2 The Improved Feasible Solution

If we assume the feasible region is bounded, then linear programming theory indicates that one of the vertices of this feasible region will be the optimal solution. Although the optimal solution for a quadratic programming problem is not necessarily a vertex of the feasible region, it can be represented as the convex combination of these vertices. Therefore, after we obtain a new vertex by solving SDLP, we can solve the following QP to arrive at a superior feasible solution for CQP:

Master Problem:

$$\text{(SDQP)} \quad \begin{array}{ll} \text{Minimize} & f_z(z) = f(W^k z) \\ \text{subject to:} & e^T z \;=\; 1,\; z \geq 0, \end{array}$$

where $W^k = (v^1, v^2, ..., v^k)$, $f(W^k z) = c^T W^k z + 0.5 * z^T W^{kT} Q W^k z$ and v^i, i=1,…,k are vertices of the feasible region.

After solving SDQP, we obtain a new feasible solution $h_{k+1} = W^k z^*$. We then can solve the SDLP to obtain the new vertex. The process continues until we arrive at an optimal solution.

SD Algorithm for CQP:

Below is a summary of the steps which comprise the SD algorithm.

Step 0: Find a feasible vertex v^0. Set k = 0, $z^0 = 1$ and

$$W^0 = (v^0).$$

Step 1: Solve the subproblem SDLP to obtain a new vertex v^{k+1}.

If $\nabla f(W^k z^k)(v^{k+1} - h_k) \;\geq\; 0$, then output $h_k = W^k z^k$ and stop.
Else set $W^{k+1} = [W^k, v^{k+1}]$.

Step 2. Solve the master problem SDQP.

Let z^{k+1} be the optimal solution of SDQP. Set k=k+1.
Drop from W^k any vertex v^k whose z_i^k is zero. Goto Step 1.

5.4. Interior Point Methods

As we discussed in the previous section, an optimal solution to a linear programming problem will be situated at one of the vertices of a bounded feasible region. Consequently, most of the methods developed to solve linear programming problems begin with a basic feasible solution that also lies at such a vertex. Subsequently, these algorithms pivot along the boundary of the feasible region to other vertices until an optimal solution is reached. Although an optimal solution to a quadratic programming problem is not necessarily a vertex, most of the methods utilized to solve convex quadratic programming problems also begin from such a vertex and move along this boundary, or stop at the interior of the boundary. For example, using the active set method, if $h_{k+1} = h_k + d_k$ is feasible and the corresponding $\pi_k \geq 0$, we will obtain an optimal solution that is not a vertex.

Since the seminal paper by Karmarkar (1984), a new class of methods known as *Interior Point Methods* for solving linear programming and convex quadratic programming problems has been developed. As the name implies, these methods start from an interior feasible solution and take appropriate steps along descent directions until an optimal solution is found. There is a variety of interior point methods that were designed to solve linear programming and quadratic programming problems. Examples include projective scaling algorithms, affine scaling algorithms, barrier function algorithms, analytic center algorithms, and potential reduction algorithms. We will describe a *potential reduction algorithm* in this sub section.

For the simplicity of presentation, we consider the CQP and its dual problem QD in the following form:

$$(CQP) \quad \begin{aligned} &\text{Minimize} && c^T h + 0.5 * h^T Q h \\ &\text{subject to:} && Ah = b \\ & && h \geq 0. \end{aligned} \qquad\qquad (\text{QD}) \quad \begin{aligned} &\text{Maximize} && b^T y - 0.5 * h^T Q h \\ &\text{subject to:} && s = Qh + c - A^T \mathrm{y} \geq 0. \end{aligned}$$

The KKT conditions corresponding to this form of CQP are as follows.

(I) $Ah = b, h \geq 0$ (primal feasibility)

(II) $s = Qh + c - A^T y \geq 0$ (dual feasibility)

(III) $h^T s = 0$ (complementary slackness)

Interior point methods begin at an interior primal and dual feasible solution $(h^0, s^0) > 0$ and move inside the feasible region until arriving at a point (h^k, s^k) such that

$$h^k s^k \leq \varepsilon \tag{5.15}$$

for a small number ε. At each iteration, the potential reduction method uses the following formula to obtain a search direction.

$$\begin{aligned} d_h &= (G^{-1} - G^{-1}A^T(AG^{-1}A^T)^{-1}AG^{-1})H_k^{-1}w \\ d_y &= -(AG^{-1}A^T)^{-1}AG^{-1}H_k^{-1}w \\ d_s &= Gd_h - A^T d_y \end{aligned} \tag{5.16}$$

where $H_k = diag\{h_{k1}, h_{k2}, ..., h_{kn}\}, S_k = diag\{s_{k1}, s_{k2}, ..., s_{kn}\}$,

$G = Q + H_k^{-1}S_k, w = e - \frac{\rho}{h^{kT}s^k} H_k S_k e,$ e is a vector of all ones,

$\rho = n + \sigma\sqrt{n},$ and σ>0 is a small number.

The line search for step length is not based upon feasibility, since the solution is always selected from within the interior of the feasible region. Instead, we utilize line search to make certain to obtain a reduction in the value of following potential function:

$$\Psi(h,s) = \rho \ln(h^T s) - \sum_{i=1}^{n} \ln(h_i s_i).$$

Given a direction (d_h, d_y, d_s), line search will be used to obtain the step length δ such that new solution $(h(\delta), y(\delta), s(\delta)) = (h^k + \delta d_h, y^k + \delta d_y, s^k + \delta d_s)$ satisfies

$$1^0.\ \psi(h(\delta), s(\delta)) - \psi(h^k, s^k) \le \lambda\delta(\partial_h \psi(h^k, s^k) + \partial_s \psi(h^k, s^k))$$
$$2^0.\ \psi(h(\delta), s(\delta)) - \psi(h^k, s^k) \le -\varepsilon,$$

where 0 < λ<1 and ε>0 are constant.

Next, we depict the following result:

Theorem 2: Assume that the feasible region is bounded**,** if $h^0 > 0, s^0 > 0$ are primal and dual feasible, $\psi(h^0, s^0) = O(\sqrt{n}L)$ and

$$\psi(h^{k+1}, s^{k+1}) \le \psi(h^k, s^k) - \beta, \text{ for all k}$$

and β is a strictly positive constant number, then after at most $K = O(\sqrt{n}L)$ iterations $h^{KT}s^K \le 2^{-L}$, where L is a large positive number, *O(.)* is a polynomial function, i.e. if *f=O(g)*, then *f* is a polynomial function of *g*.

Potential Reduction Algorithm:

Below is a summary of the steps which comprise the potential reduction algorithm.

Step 0. Obtain h^0, y^0, s^0 satisfying $Ah^0 = b, h^0 > 0, s^0 = Qh^0 + c - A^T y^0, y^0 > 0.$
Set k=0, $\varepsilon > 0, \lambda > 0, \sigma > 0, \eta > 0.$

Step 1. If $h^{kT}s^k < \eta,$ then stop.

Step 2. Compute direction $p = (d_h, d_y, d_s)$ using (5.16).

Step 3. Set $h^{k+1} = h^k + \delta d_h, y^{k+1} = y^k + \delta d_y, s^{k+1} = s^k + \delta d_s$, where δ is determined by a line search along with direction p subject to condition 1^0 and 2^0.
Set k = k+1 and go to Step 1

Since interior point methods maintain a current solution that is strictly positive, the optimal portfolio may have many tiny nonzero weights. Consequently, one may want to eliminate these tiny nonzero portfolio holdings to obtain a basic feasible solution, or for practical reasons. One way to achieve this result is to switch to the use of active set methods at the conclusion of the application of the interior point method. After several iterations of the active set method, the resulting portfolio obtained will have either the same or a similar utility function value without tiny holdings.

6. Practical Implementation Considerations

In this section, we discuss details that are important to the implementation of an efficient CQP algorithm. Essentially, an effectual algorithm takes advantage of the special structure of each give problem, state-of-the-art methods, and a variety of numerical stability technologies.

6.1. Finding an Initial Feasible Solution

The ultimate goal of most individuals interested in solving portfolio optimization problems is to rebalance an initial feasible portfolio in an unbiased and systematic manner according to a predefined set of decision rules. Consequently, nearly all of the methods designed to solve convex quadratic programming problems begin with a solution that is feasible with respect to either primal or dual constraints. However, the current initial portfolio sometimes may not be feasible due to constraints that were added or changed subsequent to its construction. In this event, we need to obtain an initial feasible portfolio before proceeding any further.

Since all of the constraints to be considered are linear or piecewise linear, we can utilize a linear programming technique to find an initial feasible solution. One such well-known technique is known as *Phase I*. In accordance with Phase I, we solve the following linear programming problem:

$$\begin{aligned} &\text{Minimize } \; F(h) = \text{maximize}\{0, \sum_{i=1}^{m} (A_i h - b_i)\} \\ &\text{subject to:} \qquad Ah \le b. \end{aligned}$$

The value of the objective function associated with the Phase I problem will be zero for any feasible solution. Also, the objective function value will be positive if a solution violates any constraint. Consequently, the optimization process will reduce the value of the objective function to zero if a feasible solution exists. If we obtain a positive optimal objective function value after solving this Phase I problem, we then can claim that the original problem was infeasible.

6.2. Tolerances for Optimality and Feasibility

While using an algorithm described in Section 5, termination criteria used to decide upon how to proceed sometimes must be tested upon reaching the end of a step. Due to the possibility of computational rounding error, tolerances must be incorporated into this testing. The following tolerance tests should be utilized at a minimum:

Test to determine whether the new direction is zero: i.e. $d_k = 0$. The value of the *new direction* must be within the 1st tolerance value

$$\|d_k\| \le \varepsilon_1$$

where ε_1 is a small positive number, say 1.0e-8 for example.

Feasibility test: When $d_k \neq 0$, we may need to perform a MRT to ascertain the feasibility of the new solution $h_{k+1} = h_k + \tau_k d_k$. The 2nd tolerance value ε_2 is applied as

$$\underset{i \notin I_k}{\text{minimize}}\{1, (b_i - A_i h_{ki}) / A_i d_{ki} \ \ | \text{ for all } A_i d_{ki} \ge \varepsilon_2\}$$

where ε_2 is a small positive number, say 1.0e^{-9}. If there are many instances where $0 \le A_i d_k < \varepsilon_2$, it is possible that rounding errors may accumulate as the algorithm runs through a multitude of steps. Infeasibility will then result from a failure of the feasibility test as the cumulative rounding error becomes too large. Consequently, there is a need to balance the stability of the algorithm to control for the possibility of infeasibility due to rounding errors. One example of where this problem might appear is when an algorithm requires the division by a number that is extremely close to zero.

Dual feasibility test: When $d_k = 0$, we need to test for $\pi_k \ge 0$. In this case, we apply the 3rd tolerance ε_3

$$\|\pi_k\| / (\|g_k\| + 1) \le \varepsilon_3$$

where ε_3 is a small positive number, say 1.0e-7.

6.3. Special Structure of Hessian for Barra Risk Model

The Barra Optimizer has been designed to take advantage of the special structure of risk models of the form:

$$XFX' + D$$

where X, F and D are as defined in Section 2.1.

For most portfolio optimization cases that incorporate the use of these risk models, D is a diagonal or block diagonal matrix. Also, the dimensions of the $n \times n$ covariance matrix $XFX'+D$ usually are much larger than that of the $k \times k$ factor covariance F. For example, there may be hundreds or thousands of assets in the selectable universe (e.g., n=1000), while there are only k=68 factors in Barra's USE3 model. By introducing factor variables, we can rewrite the portfolio optimization problem to incorporate the Barra risk model as follows:

$$\begin{array}{lll} & \text{Minimize} & f(h,s,w) = c^T h + \lambda_D h^T D h + s^T P_S s + \lambda_F w^T F w + TC(h,s,w) \\ \text{(BarraQP)} & \text{subject to:} & Ah - s = 0 \\ & & X^T h - w = 0 \\ & & l_h \le h \le u_h, l_s \le s \le u_s, l_w \le w \le u_w. \end{array}$$

For this problem, the Hessian matrix is

$$Q = \begin{bmatrix} D & & \\ & P_s & \\ & & \mathrm{F} \end{bmatrix}.$$

Instead of obtaining and updating the LU factorizations of the $n \times n$ matrix Q, we only will need to obtain and update LU factorizations for the much smaller matrix F. Note that both D and P_s are diagonal or block diagonal matrices. Also, it is much easier to solve the linear system of equations (5.5) for this new Hessian matrix.

6.4. Asset Bound Constraints

In the convex quadratic programming problem given by (4.1), we assume that all linear constraints are in the general form of $Ah \le b$. In theory, one might incorporate the following asset bound constraints into this problem:

$$l_h \le h \le u_h \quad \text{as} \quad h \le u_h \text{ and } -h \le -l_h$$

and then directly apply methods to the assumed form of CQP. However, the efficiency of the optimization algorithms can be enhanced by applying such asset bound constraints more thoughtfully. When a holding variable is either at its upper bound or lower bound, or the corresponding constraint is binding, we know that the direction for this asset in the EQP should be zero. Consequently, this binding constraint becomes redundant and can be excluded from the active working set. This will result in a size reduction of EQP and also will facilitate its' solution.

6.5. Handling Piecewise Linear Functions

A given optimization problem incorporates the use of *piecewise linear functions* if it contains either a transaction cost term in the objective function, any turnover limit, or a transaction cost constraint. One method by which these piecewise linear functions can be converted into linear functions is through the introduction of new variables. For example, if there were only one buy and one sell rate for asset *j*, the transaction cost term for asset *j* would be

$$TC(h_j) = tc_{buy}\max(0, h_j - h_{j0}) + tc_{sell}\max(0, h_{j0} - h_j),$$

where tc_{buy} and tc_{sell} are the buy and sell costs per unit, respectively. By introducing buy and sell variables, h^+ and h^-, the equation can rewritten as

$$TC(h_j) = tc_{buy}h_j^+ + tc_{sell}h_j^- \text{ and } h_j = h_{j0} + h_j^+ - h_j^-.$$

Unfortunately, this modification will double the size of the problem for a simple case involving only buy and sell transaction costs. Alternatively, for the Barra Market Impact Model, the size of problem may be increased by a factor of three or five relative to the original size. At Barra, we do not split the variables used to transform piecewise linear terms into pure linear terms. Instead, we deal with the piecewise linear terms directly and modify the methods so that we can treat these terms more efficiently. The details are beyond the scope of this paper.

References

Bertsekas Dimitri P. (1999), *Nonlinear Programming 2nd ed.* Athena Scientific

Fletcher, R. (2000), *Practical Methods of Optimization*, (John Wiley & Sons, New York; 2nd edition.)

Gill, P.E., W. Murray and M.H. Wright (1981), *Practical Optimization,* (Academic Press, London-New York)

Goldfarb, D. and S. Liu (1993), "A Primal-Dual Potential Reduction Algorithm Convex Quadratic Programming." *Mathematical Programming 61*, p161-170.

Karmarkar, N. (1984), "A New Polynomial Time Algorithm for Linear Programming." *Combinatorica* Vol 4, p373-395

Markowitz, H. M.(1959), *Portfolio Selection: Efficient Diversification of Investments,* John Wiley & Sons, New York

Nemhauser, G.L., A.H.G. Rinnooy Kan and M.J. Todd (1989), *Handbooks in Operations Research and Management Science, Vol 1: Optimization*, (Elsevier Science Publishers B. V., North-Holland, London-New York.)

Nemhauser, G. L. and L.A. Wolsey (1988), *Integer and Combinatorial Optimization*, (John Willey & Sons, New York.)

Nesterov Y. and A. Nemirovskii (1994), *Interior-Point Polynomial Algorithms in Convex Programming*. SIAM studies in applied mathematics. Vol 13

Stefek D. S. Liu and R. Xu (2003), *Barra Optimizer User's Manual. V4.5.1*

Von Hohenbalken, B. (1977), "Simplicial Decomposition in Nonlinear Programming." *Mathematical Programming*, Vol. 13, pp49-68

Wolsey, L. A. (1998), *Integer Programming*, 1st edition, (Wiley-Interscience)