Chapter 30
Quadratic Programming for Large-Scale Portfolio Optimization

Michael J. Best
Jivendra K. Kale

QUADRATIC PROGRAMMING (QP) IS THE MOST WIDELY USED METHOD for portfolio construction. It is most effective when the assert return distributions are approximately normal. This method works particularly well for large stock and bond portfolios which can contain several thousand assets and where risk is typically measured relative to a benchmark. The requirements for portfolio constructions can be quite complex. In addition to the usual budget constraint, a portfolio may be required to have upper and lower bounds on all assets. The portfolio may be required to satisfy constraints such as limits on industry holdings, beta, or dividend yield for stock portfolios, and limit on issuers holdings, duration, and convexity for bond portfolios. Accounting for the transactions costs of trading is necessary where these costs increase with the size of the transaction because of the price impact of trades. Incorporation of some, or all of these features, can result in a quadratic programming (QP) problem which is simply too large for practical use with a general purpose quadratic programming method. However, general purpose QP algorithms can be specialized to take account of the particular structure of portfolio problems enabling large problems to be solved in a practical way. Here, we give an overview of three things: (1) formulation of the portfolio optimization problem as a QP, (2) QP solution methods, and (3) specialization of QP algorithms to solve large-scale portfolio optimization problems.

Portfolio optimization is the technique for finding the best portfolio for the investor, given the available set of portfolios and the investor's

0-8493-9834-7/00/$0.00+$.50

tolerance for risk. The investor's risk tolerance defines the trade-off between the expected return on the portfolio and its risk. In his original formulation of the portfolio optimization problem, Markowitz (1952) defined portfolio risk as the variance of the portfolio return. This approach to portfolio selection is known as mean-variance analysis, and it works well when the asset return distributions are approximately normal. Optimization is used widely for the management of investment portfolios throughout the world. Money managers use it for a variety of applications. Among them are (1) asset allocation, for calculating the optimal investment in asset categories such as stocks, bonds and cash, or the optimal investment in different countries; (2) stock portfolio optimization, for calculating the optimal investment in different stocks for a given risk tolerance or maximizing expected portfolio return for a given level of risk, or for tracking a stock index; (3) bond portfolio optimization, for calculating the optimal investment in different bonds for a given risk tolerance, cash flow matching, portfolio immunization, or for tracking a bond index.

Portfolio optimization problems have linear and quadratic terms in the objective function and are subject to a set of linear constraints. The constraints can be on individual asset holdings, asset group holdings such as industry and issuer constraints, portfolio characteristics such as beta, dividend yield, or duration and convexity. In addition, transactions costs must be paid for trades that are necessary to change the existing portfolio to an optimal one. These transactions costs typically vary with the size of the trade. Larger trades have a larger price impact on the market price of a security, resulting in higher transactions costs. The price impact of large trades is particularly important for portfolios of small company stocks, where it can be as high as 1000 basis points (Sinquefield, 1995).

Portfolio optimization problems can be formulated as Quadratic Programming (QP) problems when risk is measured as the variance of return. For a stock or bond portfolio optimization where the number of assets in the universe may be in the thousands, these problems become very large when constraints and variable transaction costs are introduced into the QP formulation. To solve these problems quickly, particularly on a workstation, requires specialized techniques that take advantage of the structure of the problem. The following sections lay out problem formulation, describe the different types of algorithms that may be used to solve the problem, and describe the specialized techniques that can be used to solve the large-scale portfolio optimization problems quickly on a workstation.

PORTFOLIO OPTIMIZATION AS A QP

The Basic Markowitz Model

Consider a universe of n assets. Assume the data

$$\mu = (\mu_1, \ldots, \mu_n)' \quad \text{and} \quad \Sigma = [\sigma_{ij}]$$

are known, where μ_i is the expected return for asset i and σ_{ij} is the covariance of returns for assets i and j. Thus, μ is the vector of expected returns and Σ is the (n, n) variance-covariance matrix of asset returns. Let $x = (x_1, \ldots, x_n)'$ denote the (as yet unknown) holdings vector; i.e., x_i denotes the investment weight for asset i, $i = 1, \ldots, n$. In terms of x, the expected return of the portfolio μ_p and its variance σ_p^2 are

$$\mu_p = \mu' x \quad \text{and} \quad \sigma_p^2 = x' \Sigma x .$$

Markowitz's seminal idea (Markowitz, 1952) is to define a portfolio to be efficient if for some fixed level of portfolio expected return no other portfolio gives a smaller portfolio variance. Thus, an efficient portfolio is the solution to the optimization problem

$$\min \left\{ \frac{1}{2} x' \Sigma x \,\middle|\, \mu' x = \mu_p, l' x = 1 \right\}, \tag{30.1}$$

where l is a vector of ones. The constraint $l'x = 1$, called the budget constraint, requires the investment weights to sum to 1. The set of all efficient portfolios is generated as the expected portfolio return, μ_p, is varied in (30.1).

Equivalently, one can define an efficient portfolio as one for which at some fixed level of portfolio variance no other portfolio gives a larger portfolio expected return. This results in the optimization problem

$$\max \left\{ \mu' x \,\middle|\, x' \Sigma x = \sigma_p^2, l' x = 1 \right\}. \tag{30.2}$$

Again, the efficient portfolios are generated as the variance of portfolio return, σ_p^2, is varied in (30.2).

A mathematically equivalent formulation of (30.1) and (30.2) is

$$\max \left\{ t\mu' x - \frac{1}{2} x' \Sigma x \,\middle|\, l' x = 1 \right\}, \tag{30.3}$$

where t is a non-negative scalar parameter. t can be thought of as a risk-tolerance parameter. When t = 0, indicating a low tolerance for risk, the solution of (30.3) is the minimum variance portfolio. When t is large, indicating a

high tolerance for risk, the solution to (30.3) will emphasize the maximization of the expected portfolio return, μ_p, and put little weight on the minimization of the variance of portfolio return, σ_p^2.

Problems (30.1), (30.2), and (30.3) are equivalent. For our purposes, it is convenient to work with (30.3). If Σ is positive definite (so that it is invertible), the solution for (30.3) may be written in closed form. Indeed, denoting the optimal solution by $x(t)$ to emphasize its dependence on t, it may be shown that

$$x(t) = h_0 + th_1, \tag{30.4}$$

where

$$h_0 = \left(l'\Sigma^{-1}l\right)^{-1}\Sigma^{-1}l, \quad h_1 = \Sigma^{-1}\left(\mu - \left(l'\Sigma^{-1}l\right)^{-1}\left(l'\Sigma^{-1}\mu\right)l\right).$$

Note that from (30.4), $x(t)$ is a linear function of t.

It is important to realize that the simple model (30.3) can be solved using only matrix multiplication and matrix inversion to calculate Σ^{-1}. No QP algorithm is required. The expected return and variance of efficient portfolios may be calculated in terms of t using (30.4). Doing so and eliminating t gives the relationship between σ_p^2 and μ_p for efficient portfolios:

$$\left(\mu_p - \mu' h_0\right)^2 = \mu' h_1\left(\sigma_p^2 - h_0'\Sigma h_0\right). \tag{30.5}$$

This is the well-known efficient frontier, a parabola in mean-variance space.

Short Sales Restrictions

The computational complexity of the optimization problem (30.3) increases dramatically when bounds on asset holdings are added to the problem formulation. Asset bounds may be restrictions on short sales or legal and institutional restrictions on large holdings in individual stocks and bonds. Short sales may be precluded by adding nonnegativity constraints (i.e., $x \geq 0$) to (30.3), to give

$$max\left\{t\mu' x - \frac{1}{2}x'\Sigma x \mid l'x = 1,\ x \geq 0\right\} \tag{30.6}$$

The computational difficult in (30.6) stems from the nonnegativity constraints. It is not at all clear which constraints will be *active*, i.e., which holdings will be zero, in the optimal solution. At first, this may seem like a fairly simple problem, but it is not. Since each holding may be zero or positive, there are two possibilities for each. With n assets, there are 2^n possibilities. For a medium-sized portfolio having $n = 500$ assets, there are $2^{500} \approx$

3.27×10^{150} possibilities. Obviously, enumeration of all possibilities is not practical for other than problems having a very small number of assets.

What is required for the solution of (30.6) is a QP algorithm. An algorithm in this context is a precise set of instructions that will construct a better portfolio from an existing one. The instructions are then repeated with the new portfolio replacing the previous one, until an optimal solution is obtained. Virtually all QP algorithms have the property that the optimal solution will be obtained in a *finite* number of steps. The execution of a single repetition of the instructions is called an *iteration* of the algorithm. The reader may be familiar with the Simplex Method for Linear Programming (LP). QP algorithms will be discussed in more detail in a later section.

The inclusion of nonnegativity constraints (or any *inequality* constraints) in (30.6) changes the character of the optimal solution for (30.6). Rather than being linear for all t, the efficient portfolios are now *piece-wise linear* in t. This means there are a finite set of numbers $0 \le t_1 \le \ldots \le t_s$ such that the efficient portfolios are given by

$$x_i(t) = h_{0i} + th_{1i}$$

for all t with $t_i \le t \le t_{i+1}$, and for $i = 0, \ldots, s-1$. This means that the efficient portfolios are still linear functions of t but the coefficients differ in each interval (t_i, t_{i+1}). The efficient frontier (30.5) now becomes a piece-wise parabola.

The number of intervals, s, depends on the problem data in (30.6), and there is no way of determining it *a priori*. It could be very large. Each interval differs from the previous in that an assets holding is either reduced from a positive value to zero, or some other assets holding moves from zero to a positive value, or both. That is, each interval corresponds to a change from an adjacent interval of at most two asset holdings in the active constraint set, which is the set of constraints where the asset holdings are zero in (30.6).

To solve (30.6) for all t and thus generate all of the parametric intervals, the appropriate computational tool is a *parametric* QP algorithm (the parameter being t).

Fixed Transaction Costs and Asset Bounds

Fixed transactions costs are costs that do not change with transaction size. These costs can be incorporated into the model (30.6). Suppose x_0 is the vector of holdings in the initial portfolio. Let x^+ denote the vector of asset purchases and x^- denote the vector of asset sales. Then the new asset holding vector x satisfies

$$x = x_0 + x^+ - x^-, \quad 0 \le x^- \le d, \quad 0 \le x^+ \le e,$$

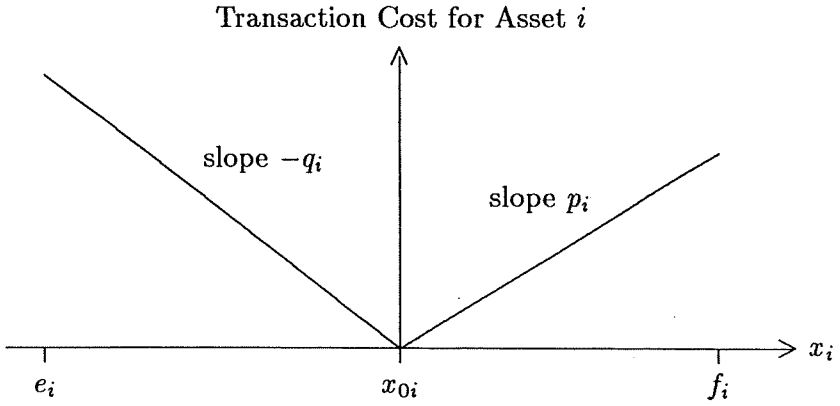

Exhibit 30-1. Fixed transactions costs.

where d and e are vectors of upper bounds on scales and purchases, respectively. The upper bounds on purchases and sales are equivalent to bounds on asset holdings, assuming that the initial holdings satisfy the asset bounds. The new variables x^+ and x^- have to be included in the problem formulation. One way to incorporate fixed transaction costs is to use a linear cost function. For asset i, letting p_i and q_i denote the percentage transactions costs on the buy and sell sides, respectively, the transactions cost represents a reduction in expected return. The total transactions cost for an asset is (see Exhibit 30.1)

$$p_i x_i^+ + q_i x_i^- ,$$

and the transactions cost for the portfolio is

$$p'x^+ + q'x^- .$$

Assembling these features into (30.6), the fixed transaction cost QP formulation becomes

$$\left.\begin{array}{lll} \text{maximize: } t\left(\mu'x - p'x^+ - q'x^-\right) - \frac{1}{2}x'\Sigma x & & \\ \text{subject to:} & l'x & = 1, \\ & x - x^+ + x^- & = x_0 , \\ & 0 \le x^- \le d , & \\ & 0 \le x^+ \le e . & \end{array}\right\} \qquad (30.7)$$

Note that for n assets, the problem formulation (30.7) has $3n$ variables (n each for x, x^+, and x^-), and $5n + 1$ constraints (n equality constraints $x - x^+$

$+ x^- = x_0$, $2n$ lower and upper bounds on x^-, $2n$ lower and upper bounds on x^+ and 1 budget constraint). Incorporation of upper and lower bounds on all assets as well as transaction cost results in a much larger and more complicated QP problem, with three times as many variables as the original QP problem.

Variable Transaction Costs

The fixed transactions costs model assumes that the cost of transacting is the same for all sizes of trades. Allowing transactions costs to vary with the size of the trade to account for the price impact of a trade, is a more accurate representation of securities markets. This approach divides up the range of trading each asset into a number of intervals and assigns a particular cost to each range. Thus, a small trade may have a small transaction cost, whereas a large trade may have a comparatively large transaction cost since the price impact of the trade will be larger. Accounting for this is particularly important when the market for an asset is thin, which is typical for small company stocks and many international assets. Rex Sinquefield (1995) of Dimensional Asset Advisors, who specializes in small stock funds, remarked that the price impact of trades in small stocks can be as high as 1000 basis points. Loeb (1983) provides more evidence of the price impact of trades.

The vector of asset holdings, x, is defined in terms of its deviation from the initial portfolio x_0.

$$x = x_0 + x^+ - x^-,$$

and each of x^+ and x^- is expressed as the sum of holdings in each of k intervals:

$$x^+ = x_1^+ + x_2^+ + \cdots + x_k^+,$$

$$x^- = x_1^- + x_2^- + \cdots + x_k^-,$$

where

$$0 \le x_i^- \le d_i, \quad 0 \le x_i^+ \le e_i, \quad i = 1, \ldots, k.$$

The purchase transaction cost for the first interval x_0 to $x_0 + e_1$ is $p_1' x_1^+$. If trading continues to the second interval $x_0 + e_1$ to $x_0 + e_1 + e_2$, the transaction cost is $p_1' x_1^+ + p_2' x_2^+$. This model of stepped transactions costs corresponds to the execution of limit orders in the specialist's limit order book. Exhibit 30.2 shows transactions costs for $k = 3$ intervals.

In general, the total transaction cost is

$$p_1' x_1^+ + \cdots + p_k' x_k^+ + q_1' x_1^- + \cdots + q_k' x_k^-.$$

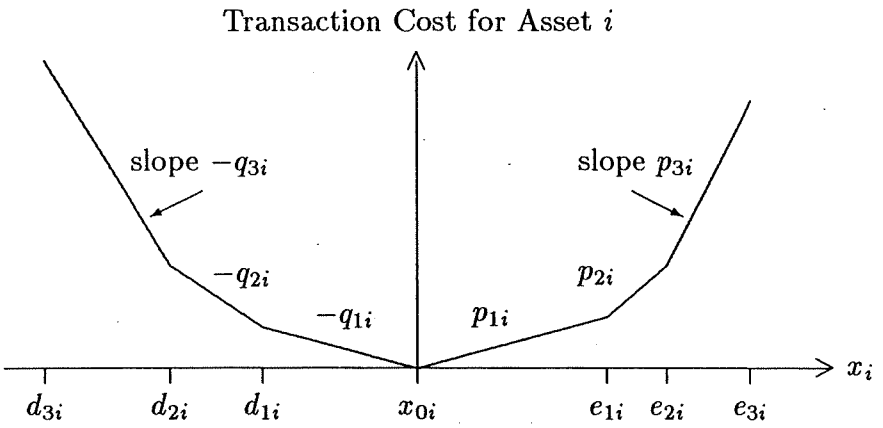

Exhibit 30-2. Variable transactions costs.

Note that this formulation is incremental. It implies an upper bound on total asset holdings of $x_0 + e_1 + \cdots + e_k$ and a lower bound on asset holdings of $x_0 - d_1 - \cdots - d_k$.

It may be important to allow for linear constraints in addition to the budget constraint. Accordingly, we replace $1'x = 1$ with $Ax - b$, where A is an (m, n) matrix. These constraints may include a budget constraint and other constraints, such as group constraints, constraints on beta, or company size. Although the linear constraints have been defined as equality constraints, this formulation allows for inequality constraints, which are converted to equality constraints by the inclusion of slack or surplus variables.

Incorporating these features into (30.7), the variable transactions costs QP formulation of the portfolio optimization problem becomes

$$\left.\begin{aligned} &\text{maximize: } t\left(\mu'x - p_1'x_1^+ - \cdots - p_k'x_k^+ - q_1'x_1^- - \cdots - q_k'x_k^-\right) - \tfrac{1}{2}x'\Sigma x \\ &\text{subject to: } \quad Ax = b, \\ &\qquad\qquad x - x_1^+ - \cdots - x_k^+ + x_1^- + \cdots + x_k^- = x_0, \\ &\qquad\qquad 0 \le x_i^- \le d_i, \quad i = 1,\ldots,k, \\ &\qquad\qquad 0 \le x_i^+ \le e_i, \quad i = 1,\ldots,k. \end{aligned}\right\} \tag{30.8}$$

Note that for n assets, the problem formulation (30.8) now has $(2k + 1)n$ variables and $(4k + 1)n + m$ constraints. For a problem with $n = 2000$ assets, $m = 1$ equality constraint, and $k = 3$ variable transactions costs, (30.8) is a QP with 14,000 variables and 26,001 constraints.

OVERVIEW OF QP ALGORITHMS

In this section we will argue that general purpose or off the shelf QP algorithms are unsuitable for all but very small portfolio optimization problems. This is because general purpose QP algorithms do not take advantage of the algebraic structure inherent in the portfolio optimization problem.

The earliest QP algorithm is that of Beale (1955, 1959). This algorithm has the basic property that it will solve a concave QP in a *finite* number of steps. More recent algorithms also possess this finiteness property but are preferred because they require considerably fewer iterations. Some methods are parametric. These methods introduce a scalar parameter into the QP such that when the parameter has a large value, an optimal solution for the parameterized problem is evident. The parameter is iteratively reduced to zero, whereupon the original QP is solved. Methods of this type are Wolfe (1959), Markowitz (1956), and the Complementary Pivot Method of Lemke and Howsen (1968). Markowitz's method is particularly noteworthy because it is designed specifically for portfolio optimization and generates the entire efficient frontier. In contrast, general purpose QP algorithms solve the problem formulated in (30.8) directly, to get the single optimal portfolio, given the value of the investor's risk tolerance parameter t. For problems with many asset bounds (such as problems with variable transactions costs), the direct method used by the general purpose algorithms can save a tremendous amount of computational time when compared with the parametric approach.

Most recent general purpose QP algorithms solve problems in exactly the same way. Indeed it has been demonstrated in Best (1984) that many QP algorithms will generate the same sequence of points when applied to the same problem and using the identical starting point. These algorithms include those of van de Panne and Whinston (1969), Fletcher (1971), Gill and Murray (1978) and, Best and Ritter (1988). They differ outwardly because of the different methods they use to solve linear equations or because they assume different model problems.

One parametric quadratic programming algorithm developed specifically for portfolio optimization is due to Perold (1984). This method is a generalization of Markowitz's (1956) parametric method which allows the covariance matrix to be semidefinite. In addition, Perold's method uses sparse matrix technology to solve the Karush-Kuhn-Tucker system for each parametric interval. The goal is to reduce the computation time of the algorithm as well as the workspace required by it. Although using sparse matrix factorization techniques is in general a good idea, we argue that it is far more effective to use such techniques *after* full advantage has been taken of the algebraic structure of the problem, as our method does.

We give a more detailed comparison of Perold's method to ours in a later section.

In order to explain some of the basic principles of QP algorithms, consider the model problem:

$$\max\left\{c'x+\frac{1}{2}x'Cx \,\middle|\, a_i'x \le b_i, \quad i=1,\cdots,m\right\}, \tag{30.9}$$

where x is a vector of dimension n. This is a *general* problem, in that no assumptions are made concerning the structure of the m linear constraints. It is assumed that a feasible point x_0 is known. Constraint i is *active* at x_0 if $a_i x_0 = b_i$ and *inactive* otherwise. Let these active constraints be accumulated in a matrix A_0 with corresponding right hand-side b_0. Then $A_0 x_0 = b_0$. Most QP algorithms maximize the objective function for (30.9) in the intersection of the active constraints; i.e., they solve

$$\max\left\{c'x+\frac{1}{2}x'Cx \,\middle|\, A_0 x = b_0\right\}. \tag{30.10}$$

This is equivalent to solving the linear equations

$$\begin{bmatrix} C & A_0' \\ A_0 & 0 \end{bmatrix}\begin{bmatrix} x \\ u \end{bmatrix}=\begin{bmatrix} -c \\ b_0 \end{bmatrix}. \tag{30.11}$$

The system of equations (30.11) is called the *Karush-Kuhn-Tucker* system for the active set. In moving to a better point, a previously inactive constraint may become active. If so, its gradient is added to the rows of A_0 and (30.11) is resolved. The number of active constraints may increase for many iterations until the optimal solution for (30.11) is feasible for (30.9). At this point, the Karush-Kuhn-Tucker multipliers, u are examined. If all are nonnegative, the QP is solved. Otherwise, some active constraint with a negative multiplier is allowed to become inactive. This means that a row of A_0 will be deleted.

In general, the number of rows of A_0 will increase or decrease by 1 at each iteration of a QP algorithm. The dimension of the coefficient matrix of (30.11) will vary between (n, n) and $(2n, 2n)$. Various QP algorithms solve these equations using matrices of different types and sizes. One could, in fact, explicitly compute the inverse of

$$\begin{bmatrix} C & A_0' \\ A_0 & 0 \end{bmatrix}$$

and then the solution of (30.11) is

$$\begin{bmatrix} x \\ u \end{bmatrix}=\begin{bmatrix} C & A_0' \\ A_0 & 0 \end{bmatrix}^{-1}\begin{bmatrix} -c \\ b_0 \end{bmatrix}.$$

Exhibit 30-3. Execution Requirements for a General Purpose QP Algorithm

Portfolio Model	Storage (MBytes)	Operations per Iteration (Millions)	Exec Time (Hrs:Min:Sec)
Restricted short sales	32	4	0:5:4
Fixed transactions Costs andbounds	288	36	0:45:36
Variable transactions Costs and bounds ($k = 3$)	1568	196	4:8:16

The various QP algorithm utilize different linear equation solving methods and matrices to solve (30.10). The sizes of these matrix structures varies between (n, n) and $(2n, 2n)$, depending on the number of active constraints and is a feature of the particular algorithm. The smallest of these is the Best and Ritter method, which requires the use of only an (n, n) matrix, independent of the number of active constraints.

To get a rough feel for the execution times required to solve the QP, suppose we apply this algorithm to a portfolio optimization problem with $n = 2000$ assets. To update a matrix of dimension (n, n) requires approximately n^2 arithmetic operations. Now there is no known way to estimate the total number of iterations required by the QP algorithm: it may be just a few or it may be extremely large. Nonetheless, let us imagine the algorithm takes n iterations. To estimate the total time required by the algorithm, we need to know the speed of the computer we are using. One of the authors who uses a Pentium processor-based workstation running at 60 MHz performed the following experiment. A small program was written to multiply 10^9 pairs of floating point numbers. When run, this required 38 seconds of computing time. Consequently, a single floating point operation required 3.8×10^{-8} seconds. This figure was used to compute the estimated computation time of a general purpose QP algorithm for each of our portfolio models as shown in Exhibit 30.3.

For the Restricted Short Sales model, a storage requirement of 32MB may not be too demanding for a workstation, and the 5 minutes execution time is probably acceptable. However, the computing demands for the Fixed Transactions Costs and Bounds model are excessive, while the Variable Transactions Costs and Bounds model has impractical computational requirements. In the next section we show how an efficient, special purpose QP algorithm can be developed for portfolio optimization problems. One final note: the estimated execution times in Exhibit 30.3 are quite fictitious since they are based on the assumption of n iterations of the QP algorithm. They would be multiplied by 1000 if it turned out that $1000n$ iterations were required.

A SPECIALIZED QP ALGORITHM FOR LARGE-SCALE PORTFOLIO OPTIMIZATION

The specialization of the QP algorithm for large-scale portfolio optimization consists of three innovations: (1) determination of a good starting point for the QP algorithm, (2) efficient solution of the Karush-Kuhn-Tucker system for the active constraints using a small kernel matrix, and (3) handling of all upper and lower bounds as well as break points for variable transactions costs *implicitly* rather than explicitly. Incorporation of these three innovations produces a specialized QP algorithm, which requires a small number of iterations, requires a small workspace, performs a small amount of computational work per iteration, and, consequently, can solve large-scale portfolio optimization problems very quickly on a workstation.

Solving a QP is equivalent to determining the indices of those constraints which are active at the optimal solution. Let us call this the *optimal active set.* Once the optimal active set is known, the optimal solution can be obtained by solving the linear equations (30.11) for the problem formulation (30.10). If we start the QP algorithm with an active set that is close to the optimal active set, then relatively few QP iterations will be required to solve the problem. This is to be compared with a general purpose QP algorithm which would begin with an arbitrary feasible point.

For a portfolio that is optimized regularly, one would expect that the active set for the initial portfolio x_0 would be quite close to the optimal active set, and so x_0 is an excellent candidate with which to initiate the QP algorithm. However, x_0 may not be feasible for (30.8). In this case, we can use a variant of the Upper Bounded Simplex Method (see for example Best and Ritter, 1985) to solve a Phase 1 problem for the constraints

$$Ax = b, \quad d \le x \le e.$$

The modification simply keeps nonbasic variables at their corresponding component of x_0 unless forced to a bound.

The Karush-Kuhn-Tucker system for (30.8) will be extremely large because it has $2(k + 1)n$ variables, the equality constraint relating x to x_0 will always be active, and many variables will be at a bound. If the benefit of moving an asset away from its initial holding is smaller than its transaction cost, then it is implicitly considered to be at a bound. The critical information for the Karush-Kuhn-Tucker system may be obtained by solving a much smaller system than (30.8). The constraints of (30.10) temporarily force the many variables that are at their bounds to remain at their bounds. Let B denote a sub matrix of columns of A corresponding to variables *not* at a bound. B is analogous to the basis matrix for the simplex method of linear programming. In the LP case B is square, whereas here it will generally have more columns than rows. Let x_B denote the sub vector of variables not

at a bound. The components of x_B are called *basic* variables. The remaining variables are called *nonbasic* and are denoted by x_{NB}. By substituting x_{NB} into (30.11) and rearranging terms, we have the linear system

$$\begin{bmatrix} \Sigma_{BB} & B' \\ B & 0 \end{bmatrix} \begin{bmatrix} x_B \\ u \end{bmatrix} = \begin{bmatrix} c_1 \\ b \end{bmatrix}, \tag{30.12}$$

where

Σ_{BB} is the sub matrix of Σ corresponding to x_B,
c_1 is computed from the problem data and x_{NB},
and u are the multipliers associated with the constraints $Ax = b$.

We call (30.12) and its coefficient matrix the *kernel system* and *kernel matrix*, respectively.

Our algorithm is a specialization of the general purpose Algorithm A described in Best (1985). As such, it proceeds first by minimizing the objective function in the intersection of the active constraints. As the current kernel equations are solved, feasibility is accounted for. An asset may be moved to a bound or an asset may be moved to the end of an interval corresponding to the current Variable Transactions Costs. This is treated as a new active constraint, and the corresponding row and column of the kernel matrix are deleted. The inverse of the kernel matrix, or a factorization of it, is then updated. As the algorithm continues, other new constraints become active, and the size of the kernel matrix continues to decrease. Eventually, a quasistationary point will be determined. (See Best [1985] for details.) Note that the kernel matrix can never be smaller than $(2m, 2m)$ since this would mean an extreme point for (30.8) has been obtained, and an extreme point is a quasistationary point.

Once a quasistationary point has been determined, the algorithm tests for optimality. This requires the calculation of the Karush-Kuhn-Tucker multipliers for all of the active constraints. Since (30.8) has $4nk$ inequality constraints, there are potentially this many multipliers to compute. However, each asset holding can only be either in the interior of a Variable Transactions Costs interval or at an end point of it. In the latter case, two multipliers must be computed; one for an increase into the next interval and one for a decrease into the previous interval. Of course, when the asset holding is at either its lower or upper bound, only one multiplier is computed. The various multipliers can be obtained by substituting u from the solution of (30.12) into the dual feasibility conditions for the problem and utilizing complementary slackness. The smallest multiplier associated with an active inequality constraint is then selected. If it is nonnegative, then all optimality conditions for the problem have been satisfied, and an optional solution thus has been obtained. If it is indeed strictly negative, then the

corresponding constraint is dropped from the active set. This means that the data for the new basic variable are added as a new row and column to the coefficient matrix for the kernel problem. The algorithm then proceeds by seeking a new quasistationary point, and the process is repeated. The optimal solution obtained by this method is a true global optimum.

The variance-covariance matrix Σ may not be positive definite. It is possible for it to be positive semidefinite. This would be the case if there were dependencies among the assets, or if one or more risk free assets were included in the model. Some QP algorithms explicitly exclude the semidefinite case. Markowitz's (1956) method is restrictive in this way. Wolfe's (1959) method has two variations, one for the positive definite case (the "short form") and one for the semidefinite case (the "long form"). Our method allows for both the positive definite and positive semidefinite cases in a single unified framework. Positive semidefiniteness, or singularity of Σ, comes into play at just one point in our algorithm. After a quasi-stationary point has been obtained and a previously active constraint is being dropped, then a new variable becomes basic, and its associated data are added as a new row and column to the coefficient matrix of the new kernel problem. If Σ is positive definite, the new coefficient matrix will be non-singular. If Σ is only positive semidefinite, the coefficient matrix may be singular. In this latter case, a vector in the null space of the kernel matrix leads to a new nonbasic variable. The net effect is a switch of nonbasic and basic variables, or a switch of row and column of the kernel matrix. It has been shown in Best (1985) that after the switch, the modified kernel matrix is nonsingular. Note that if Σ were the zero matrix (and thus our model problem were an LP, a perfectly legitimate special case), then the switching would occur at each and every iteration.

We are now in a position to expand the comparison of our method with that of Perold. Suppose both methods are applied to the model problem (30.9) and that the constraints of (30.9) include upper and lower bounds on all assets. Then the coefficient matrix (call it H) of the linear system to be solved at each iteration, namely (30.11), will contain many unit vectors and thus be sparse. Perold's method uses a LU sparse matrix decomposition of H. However, H can be quite large; at least (n, n) and at most $(2n, 2n)$, where n is the number of assets, and there is no guarantee on the size of the factor matrices L and U. By contrast, our method solves the linear system (30.12) at each iteration. Let H_B denote the coefficient matrix for (30.12), i.e., the kernel matrix. The number of rows of the submatrix B is precisely the number of linear equality constraints in the given problem. This number is usually quite small. Indeed, it could represent just the budget constraint. Therefore, the dimension of H_B is close to that of Σ_{BB}, namely, the number of assets, not at a bound, and for many problems this number will be quite small. The situation is similar for variable transactions costs.

If our method were applied to an LP ($\Sigma = 0$), then the kernel matrix would have constant dimension ($2m$, $2m$) at each iteration, and it would be of the form

$$\begin{bmatrix} 0 & B' \\ B & 0 \end{bmatrix},$$

where B is the basis matrix for applying the Upper Bounded Revised Simplex Method [see Best and Ritter (1985)], and the Variable Transactions Costs are handled in a suitable manner.

For Variable Transactions Costs, if an asset holding is at an end point of its interval, the situation is essentially the same as if the holding were at a bound. Our experience has been that only a small proportion of asset holdings tend to differ from their bounds, where transactions costs create implicit bounds. For a 2000 asset problem, there may be 10 percent, or 200 such assets. Usually the number of rows of A is quite small, so let us ignore it for purposes of comparison. The kernel matrix for (30.12) is then at most (200,200). Exhibit 30.4 shows the storage requirements and execution times for the specialized QP algorithm. Comparison of these figures with those of Exhibit 30.4 shows quite dramatically the importance of a specialized QP algorithm for portfolio optimization.

We conclude this section by giving actual execution times for our QP algorithm applied to four problems. Exhibit 30.5 summarizes the results of four portfolio optimizations with Financiometrics, Inc.'s Quadratic Optimization System, which implements the Best-Kale algorithm. The optimization was done on a PC with a 400-Megahertz Pentium II processor. From Exhibit 30.5 the first test problem has 500 assets and a single linear constraint and the number of basic variables (n_B) at each iteration varied between 1 and 6. The dimension of the kernel matrix for (30.12) is thus between (1,1) and (6,6), and the number of arithmetic operations to update the kernel matrix at each iteration varies between 1^2 and 6^2. By contrast, a general purpose QP algorithm would be updating a coefficient matrix like that for (30.11), the dimension of which would vary between (n, n) and ($2n$, $2n$). For the first problem, a general purpose QP algorithm would be updating a (500, 500) to (1000, 1000) matrix at a computational cost of between 500^2 to 1000^2 arithmetic operations per iteration. It is the dramatically smaller number of arithmetic operations per iteration that results in very small execution times for the Best-Kale algorithm.

The situation is similar for the second test problem. This problem has 70 linear equality constraints, and each one of them will have an associated basic variable. For the Best-Kale algorithm, the computational work at each iteration is proportional to between 70^2 and 75^2 operations, whereas a general purpose QP algorithm would require between 570^2 to 1000^2 operations per iteration. The situation is similar for the third and fourth prob-

Exhibit 30-4. Execution Requirements for a Specialized QP Algorithm

Portfolio Model	Storage (MBytes)	Operations per Iteration (Millions)	Exec Time (Hrs:Min:Sec)
Restricted Short Sales	.32	.04	0:0:30
Fixed Transactions Costs and Bounds	.32	.04	0:0:30
Variable Transactions Costs and Bounds	.32	.04	0:0:30

Exhibit 30-5. Execution Requirements for a Specialized QP Algorithm

Problem Number	Number of Linear Constraints	Number of Assets	Computation Time (Seconds)	n_B
1	1	500	2	1 to 6
2	70	500	5	70 to 75
3	1	1000	11	1 to 32
4	70	1000	26	70 to 88

lems and is even more favorable to the Best-Kale algorithm because the number of basic variables remains very small compared to the total number of assets.

CONCLUSION

Specialization of quadratic programming techniques for the purpose of solving large-scale portfolio optimization problems by taking advantage of the algebraic structure of these problems can yield dramatic improvements in computer memory requirements and execution times. The innovations we have introduced can produce improvements of over 1000 times in execution times for the optimization of large portfolios. These innovations make it practical to solve large-scale portfolio optimization problems on a workstation.

Notes

Beale, E.M.L., On minimizing a convex function subject to linear inequalities, *J. R. Stat. Soc.* (B), 17, 173, 1955.

Beale, E.M.L., On quadratic programming, *Nav. Res. Log. Q.*, 6, 227, 1959.

Best, M.J., Equivalence of some quadratic programming algorithms, *Math. Prog.*, 30, 71, 1984.

Best, M.J. and Grauer, R.R., The analytics of sensitivity analysis for mean-variance portfolio problems, *Int. Rev. Fin. Anal.*, 1(1), 17, 1992.

Best, M.J. and Ritter, K., *Linear Programming: Active Set Analysis and Computer Programs,* Prentice-Hall, Englewood Cliffs, NJ 1985.

Best, M.J. and Ritter, K., An effective algorithm for quadratic minimization problems, *Z. Op. Res.*, 32(5), 271, 1988.
Fletcher, R., A general quadratic programming algorithm, *J. Inst. Math. Applic.* 7, 76, 1971.
Gill, P.E. and Murray W., Numerically stable methods for quadratic programming, *Math. Prog.*, 14, 349, 1978.
Lemke, C.E., On complementary pivot theory, in *Mathematics of the Decision Sciences, Part 1*, Dantzig, G.B. and Veinott, A.F., Jr., Eds., American Mathematical Society, 1968, 95-114.
Loeb, T.F., Trading cost: the critical link between investment information and results, *Fin. Anal. J.*, 39(3), 39, 1983.
Markowitz, H., Portfolio selection, *J. Fin.*, 77, March 1952.
Markowitz, H., The optimization of a quadratic function subject to linear constraints, *Nav. Res. Log. Q.*, III, 111, 1956.
van de Panne, C. and Whinston A., The symmetric formulation of the simplex method for quadratic programming, *Econometrica*, 37, 507, 1969.
Perold, A. F., Large-scale portfolio optimization, *Manage. Sci.*, 30(10), 1143, 1984.
Sinquefield, R.A., Value and Size Factors in Balanced Portfolios, Ibbotson Associates' 1995 Institutional Software and Data User Conference.
Wolfe, P., The simplex method for quadratic programming, *Econometrica*, 27, 382, 1959.

Authors' Bios

Michael J. Best *is on the faculty of the Department of Optimization and Combinatorics at the University of Waterloo, and is a principal at Financiometrics Inc. He has a Ph. D. in Operations Research from the University of California, Berkeley, and is a leading authority in the field of quadratic programming. He has written several articles and a book on mathematical programming, and his consulting work includes optimization applications in banking and portfolio management.*

Jivendra K. Kale *is on the faculty of the School of Business at Golden Gate University in San Francisco, and is a Principal at Financiometrics Inc. He has a Ph. D. in Finance from the University of California, Berkeley, and has worked as a senior consultant at BARRA and Gifford Fong Associates. He has written several articles in the areas of market microstructure, equity risk models, and portfolio optimization, and his consulting work includes portfolio optimization and performance attribution for money management applications.*