THE JOURNAL OF FINANCE • VOL. XLIII, NO. 1 • MARCH 1988

A Simple Algorithm for the Portfolio Selection Problem

ALAN L. LEWIS*

ABSTRACT

The author presents a rapidly convergent algorithm to solve the general portfolio problem of maximizing concave utility functions subject to linear constraints. The algorithm is based on an iterative use of the Markowitz critical line method for solving quadratic programs. A simple example, taken from the theory of state-contingent claims, is worked out in detail. For technical convergence results, the reader is referred to the appropriate mathematical programming literature.

THE MARKOWITZ CRITICAL LINE algorithm[1] is well known in finance. It is a complete solution to the problem of one-period portfolio selection when one's preferences depend only upon mean return and variance of return. Financial economists often consider this approach to portfolio selection only of historical interest because of the apparent restriction to quadratic utility. The purpose of this note is to demonstrate that this same algorithm, when used in an iterative fashion, can be used effectively to solve the more general portfolio problem of maximizing any concave, continuously twice-differentiable von Neumann-Morgenstern utility function with linear constraints.

It is not a new result to suggest that nonlinear programming problems can be effectively approached via iterative quadratic programming. Techniques very close to the one suggested here are well known in the mathematical programming literature. (See, for example, Stoer [10], Han [3], and Robinson [9].) However, our approach differs in two ways from this literature: (a) we maximize, at each instance of the quadratic subproblem, over the original vector of portfolio weights—the techniques referenced above maximize over a difference (or step) vector; (b) we propose explicitly using the Markowitz algorithm for the quadratic subproblem. (This algorithm is rarely referenced in the above literature.) The use of this algorithm has particular advantages for the problems of interest to the finance community; besides its wide distribution, promotion, and understanding, it readily handles the important technical case of a positive semidefinite covariance matrix. This case occurs whenever a riskless security is introduced into the investment universe. More important than these two differences, the contribution here is to demonstrate that, once one has assembled the resources to solve the quadratic utility problem, it is a relatively minor step to solve the general one-period[2] utility problem in many cases.

* Analytic Investment Management, Inc., Irvine, California. The author gratefully acknowledges stimulating conversations over the years on portfolio theory with Sheen T. Kassouf.

[1] See Markowitz [6, 7].

[2] See Cox [1] for a method by which to convert the multiperiod dynamic programming problem into a one-period problem.

We explain the algorithm in Section II and provide some simple examples, taken from the theory of state-contingent claims, in Section III. When the universe of securities contains only conventional stocks and bonds, with limited borrowing allowed, Kroll, Levy, and Markowitz [4] suggest that mean-variance analysis is often a sufficient numerical approximation to the more exact "direct utility maximization". However, once the universe of securities contains option-like positions, it is clear that mean-variance approximations (i.e., the conventional mean-variance approach, not the iterative approach introduced here) can fail badly. For example, single-security portfolios of long calls or puts will have a finite mean and variance, but, if the utility function is chosen from the class of isoelastic functions $(1 + R)^a$, where R is the return and a is nonpositive, then expected utility does not exist.

For technical results involving convergence criteria, we will summarize some basic results from the literature and point the reader to the relevant references.

I. The Problem

We use the notation that x is a portfolio (column) vector; i.e., its components are a set of n non-negative[3] weights that sum to one. Besides this constraint, there may be additional linear equality constraints. (Linear inequality constraints may be converted to equality by the addition of suitable "slack" variables to the problem.) Consequently, without loss of generality, we will assume that our problem has m linear equality constraints represented by the system of equations $Ax = b$, where A is an $m \times n$ matrix and b is an m-element column vector.

Subject to the constraints, we want to maximize the expected value of $U(R)$, where U is the investor's utility function and R is the portfolio return; i.e., $R = x^T r$, where x^T denotes the associated row vector and r is the vector of individual security returns. The expectation is taken over some distribution defined on r; the result is some new function we denote $F(x)$. We assume concave $U(R)$, which produces a concave $F(x)$. So we may summarize our problem as

$$\max F(x) \quad \text{subject to} \quad Ax = b, \quad x \geq 0. \tag{1}$$

It is useful to define a Lagrangian expression:

$$L(x, v) = F(x) - v(Ax - b), \tag{2}$$

where v is an m-element row vector. Furthermore, we define an n-element row vector u to be the negative gradient of L with respect to x; that is,

$$u = vA - DF(x), \tag{3}$$

where D is the gradient operator that produces row vectors. Then, the Kuhn-Tucker-Karush (KTK) conditions[4] for problem (1) are that x solves this problem

[3] Even if an investor can engage in short sales, the correct treatment of margin collateral requires inequality constraints on at least some portfolio weights.

[4] The pioneering 1939 contribution of W. Karush is detailed in Kuhn [5].

if and only if

(a) $Ax = b, \quad x \geq 0,$

(b) there exists a vector v such that

(i) $u \geq 0$ and

(ii) $ux = 0.$

Condition (b)(ii) implies that, if $u_j > 0$, then $x_j = 0$ and, conversely, if $x_j > 0$, then $u_j = 0$. These conditions have a simple geometric meaning that is best discussed in the context of the examples. In the next section, we shall show that the proposed algorithm, when it converges, automatically satisfies conditions (a) and (b).

II. The Algorithm

Feasible points are defined to be any points that satisfy the constraints $Ax = b$, $x \geq 0$. This set is a convex set—the line connecting any two feasible points also is feasible. The algorithm begins by selecting any arbitrary starting feasible point where $F(x)$ and its first two derivatives are well defined. (All the partials are finite.) Call this starting point y; make a second-order Taylor series expansion of $F(x)$ about y—the resulting function we denote $f(x; y)$. One has, apart from a constant that cannot alter the maximum,

$$f(x; y) = cx - (\tfrac{1}{2})x^T Bx, \tag{4}$$

where c is an n-element "mean" return vector and B is an $n \times n$ "covariance" matrix. Explicitly, one has

$$B(y) = -D^2F(y) \tag{5}$$

and

$$c(y) = DF(y) + y^T B(y). \tag{6}$$

The notation is that $D^2F(y)$ is the usual Hessian matrix of second partial derivatives of $F(x)$, evaluated at y. The concavity of $U(R)$ assures that B will be at least positive semidefinite, and strictly positive definite unless there is a riskless portfolio with a zero return.

Next, one simply solves, using the critical line algorithm (or another quadratic programming routine, if one prefers), the subproblem:

$$\max f(x) \quad \text{subject to} \quad Ax = b, \quad x \geq 0. \tag{7}$$

There should be no confusion if we also designate by x the maximizing solution to (7). Then, x becomes the new point for the Taylor expansion, and one simply iterates in this manner until a fixed point $x = y$ is reached to some approximation. If the original problem happened to involve either linear or quadratic utility, there would be convergence after one (major) step.

The associated Lagrangian for the quadratic problem is

$$L(x, v; y) = cx - (\tfrac{1}{2})x^T Bx - v(Ax - b), \tag{8}$$

and the negative gradient of the quadratic Lagrangian $u(x; y)$ is

$$u(x; y) = vA - c + x^T B. \tag{9}$$

The critical line algorithm solves (7) by imbedding the problem in a larger one parameterized by multiplying the linear term by a positive scalar. Then, this scalar is reduced sequentially from infinity (where there is a linear programming problem to solve) to one (the desired problem). Along the way, various "critical" values of this scalar are encountered, defined by some component of either x or u becoming zero.

Under the assumptions[5] that (a) the linear programming problem has a unique, nondegenerate solution and (b) a unique component of x or u becomes zero at each critical point, Markowitz proves that the critical line algorithm always converges to the solution of (7) in a finite number of steps. Consequently, at this solution, because the KTK conditions for the quadratic problem will be satisfied, we will have

(a′) $Ax = b, \quad x \geq 0, \quad$ and

(b′) there exists a vector $v(x; y)$ such that

(i) $u(x; y) \geq 0 \quad$ and

(ii) $u(x; y)x = 0.$

Now, assuming that the sequence of quadratic programming problems converges, we will ultimately have a fixed point $x = y$ of this process. Moreover, at this fixed point, we will have $u(x) = u(x; x)$ because, from (6) and (9),

$$u(x; y) = v(x; x)A - DF(x) - x^T B + x^T B = u(x), \tag{10}$$

with $v(x) = v(x; x)$. This shows that the KTK conditions (a) and (b) are satisfied. Hence, a fixed point of the algorithm must be the solution to problem (1). Also, note that the precise form of the "covariance" matrix, given by (5), is not required for this result. Hence, if the covariance matrix is expensive or time consuming to compute, the algorithm discussed here may be modified by substituting simpler expressions for this matrix (at the expense of possibly losing a quadratic convergence property discussed in Section IV).

III. Some Examples

Well-chosen examples in mathematical programming are a tremendous help to the practical problem of implementing an algorithm on the computer. The Markowitz monograph is a good illustration of this; his method is presented on an example problem small enough to be worked out (and verified) by hand calculation if necessary. That is our philosophy here in choosing an example, namely one easily worked out without the need for the computer.

[5] These assumptions are relaxed in Markowitz [6].

We take our illustration from the theory of state-contingent claims, adapted from Cox [1]. Imagine that we have three securities and three "states of the world". The states are generated by a simple binomial model of the price changes of an underlying security (a stock), which in a single time step can either advance in price from S to aS or decline to dS. For the example, we take $a = 3/2$ and $d = 1/2$. The respective probabilities of a single up-and-down move are taken to be $3/5$ and $2/5$.

The three states are the possible stock prices after two time steps, a^2S, adS, and d^2S. The three securities are the so-called "state-contingent claims" associated with these three states. Security 1 has a terminal value (payoff) of one when the stock advances twice and zero otherwise; security 2 has a payoff of one if either an up-down move or down-up move occurs; security 3 has a payoff of one only if the stock declines twice.

Suppose, in such a world, that the riskless-return multiplier is t; that is, riskless investments grow to t times their original value after one time step. Then, one can show that, to prevent arbitrage opportunities in such a world, there are simple formulas for the current prices of the claims. Denote by s_k the current price of security k ($k = 1, 2, 3$). Then, one can show that

$$\text{security 1:} \quad s_1 = p^2/t^2,$$

$$\text{security 2:} \quad s_2 = 2pq/t^2,$$

$$\text{security 3:} \quad s_3 = q^2/t^2,$$

where $p = (t - d)/(a - d)$ and $q = (a - t)/(a - d)$. These formulas are easily generalized to arbitrary price steps and time steps. For our example, we take $t = 1$ (a zero interest rate), so that our three claims have prices: 1/4, 1/2, and 1/4. Note that the riskless portfolio would consist of one share of each of the claims. The return matrix R_{ij} of security i in state j is given in Table I.

The advantage of this model, for illustrative purposes, is that $1 + R_{ij}$ is a diagonal matrix, so that one has

$$F(x) = \sum P_k U(x_k/s_k), \tag{11}$$

where the sum is over all states k and P_k is their probability. The example utility function will be logarithmic utility; i.e., $U(R) = \ln(1 + R)$. Note that $F(x)$ and its derivatives are well defined for all strictly positive x_k.

The constraints will be the standard constraints that the weights x_k be nonnegative and sum to one plus the constraint that the portfolio, no matter what

Table I

State Returns for the Example

		States		
Security	Price	1	2	3
1	1/4	3	−1	−1
2	1/2	−1	1	−1
3	1/4	−1	−1	3
	Probability:	0.36	0.48	0.16

state of the world occurs, have a specified minimum return b, where $-1 \leq b \leq 0$. The additional (nonstandard) constraints we will call the minimum-return constraints.

Clearly, if $b = -1$, the minimum-return constraints add nothing to the standard constraints. In this case, it is easy to show (see below) that the maximizing solution to (1), with $F(x)$ given by (11), occurs at $x = (P_1, P_2, P_3)$; that is: (0.36, 0.48, 0.16). At the other limit, if $b = 0$, there is only one point in the feasible set, namely the riskless portfolio (¼, ½, ¼), and so this is the maximizing solution. So these limiting cases are rather trivial, and the interesting regime is the interior one, $-1 < b < 0$.

It is helpful to see the geometry of the constraints in a plane. Eliminating x_1 via $x_1 = 1 - x_2 - x_3$, the full set of constraints are

$$\text{(a)}\ x_2 \geq 0, \quad x_3 \geq 0, \quad x_2 + x_3 \leq 1,$$

$$\text{(b)}\ x_2 + x_3 \leq (3/4) - (b/4),$$

$$\text{(c)}\ x_2 \geq (1/2) + (b/2),$$

$$\text{(d)}\ x_3 \geq (1/4) + (b/4).$$

Figure 1 shows a graph of the feasible region in the (x_2, x_3) plane for $b = -1$, a value of b in the range $-1 \leq b \leq 0$, and $b = 0$.

The first step in the algorithm is to determine the vector $c(y)$ and the matrix $B(y)$. From (5) and (6), one has

$$c_i(y) = 2P_i/y_i \tag{12}$$

and

$$B_{ij}(y) = \delta_{ij} P_i/(y_i)^2, \tag{13}$$

where δ_{ij} is the usual Kronecker delta symbol, equal to one when the indices are equal and zero otherwise.

For the first example, take $b = -1$, so that the minimum-return constraints are redundant. In this case, it is clear from (13) that, for each quadratic subproblem, the maximizing solution must be strictly interior to the triangle shown in Figure 1 (because if any y_i in (13) is zero, which occurs on the edges of the triangle, the covariance term in the quadratic form is negative infinity). That

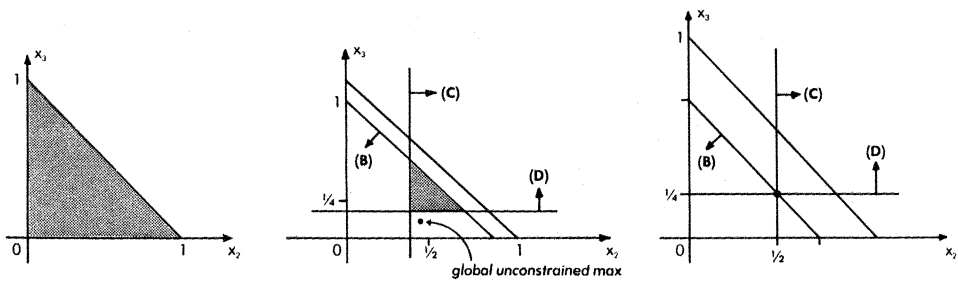

Figure 1. Feasible Regions

is, no constraints are active except that the sum of the weights is one. Consequently, the Lagrangian for the quadratic subproblem, expression (8), can be simplified to read

$$L(x, v_1; y) = 2 \sum (x_i P_i / y_i) - (\tfrac{1}{2}) \sum (x_i^2 P_i / y_i^2) - v_1 \sum x_i. \tag{14}$$

Here, v_1 denotes the only nonzero component of the v vector. The KTK conditions imply that the maximizing solution to (7) can be found by simple differentiation of the Lagrangian expression with respect to x and v_1; the resulting equations are readily solved. The final result is that the maximizing solution x is given by

$$x_i = 2y_i - \frac{(y_i^2/P_i)}{\sum (y_i^2/P_i)}. \tag{15}$$

Of course, the Markowitz critical line algorithm, because it "works", produces this same result. The iteration of (15), starting at (⅓, ⅓, ⅓), generates the sequence of quadratic subproblem solutions shown in Table II.

It is important to stress that, while we are solving these problems by "hand" calculation, the proposed algorithm is that one use the Markowitz critical line procedure to produce the solution at each major step in Table II. As indicated, this procedure involves a number of minor iterations that occur in the procedure. Since this is not a paper on the Markowitz algorithm, only a few brief remarks are necessary in this regard. First of all, since our example problem involves additional linear inequality constraints, one would introduce non-negative slack variables to put the problem in the standard form of (7). In our case, there would be three such slack variables. Next, as already indicated, one works on a parameterized problem produced by multiplying the linear term in (4) by a positive scalar, call it s_E. The procedure is started by solving the linear programming problem associated with the linear term alone. In general, there will be m linear equality constraints for this problem and hence, in linear programming parlance, m variables "in the basis". In our case, there would be, after the addition of slack variables, six securities including slacks and four variables in this basis to start. The three additional slack securities, denoted x_4, x_5, x_6, are defined by

$$x_4 = 3x_1 - x_2 - x_3 - b \geq 0,$$

$$x_5 = -x_1 + x_2 - x_3 - b \geq 0,$$

$$x_6 = -x_1 - x_2 + 3x_3 - b \geq 0.$$

Table II

Quadratic Subproblem Solutions, First Example

Major Step	x_1	x_2	x_3
0	0.333333	0.333333	0.333333
1	0.416667	0.479167	0.104167
2	0.364400	0.493210	0.142390
3	0.360813	0.480827	0.158360
4	0.360005	0.480008	0.159986
5	0.360000	0.480000	0.160000

Table III shows the sequence of critical portfolios that make up the minor iterations of the quadratic subproblem for the major iteration 1 in Table II.

For the second example, we select a value of b corresponding to the middle panel in Figure 1 so that some minimum-return constraints will be active in the solution. As one sees from Figure 1, for b close to 0, the feasible region is a small triangle (within the larger triangle) that does not contain the unconstrained maximum (P_1, P_2, P_3) found above. Also from Figure 1, one can imagine drawing a series of contours of constant utility about this latter point. From this picture, one would then guess (and then verify) that the active constraints will be, for b in a range close to zero, constraint (d) and possibly constraint (c). In terms of our three original variables and our three slack variables introduced above, our current assumption is that, of $(x_1, x_2, \cdots, x_6)$, only x_5 and x_6 are possibly zero, and the rest are all strictly positive.

It is convenient to relabel (and rescale) x_5 and x_6 to $G_2(x)$ and $G_3(x)$, respectively, where we define

$$G_2(x) = x_2 - 2d \tag{16}$$

and

$$G_3(x) = x_3 - d, \tag{17}$$

using $d = (1/4) + (b/4)$. It is also convenient to eliminate x_1 from the problem altogether and work only in the (x_2, x_3) plane. After these relabelings, the KTK conditions read as follows: (x_2, x_3) solves (7) if and only if there exists a vector $(u_2, u_3) > 0$ such that

$$u_2 G_2 + u_3 G_3 \geq 0 \tag{18}$$

and

$$(\partial f/\partial x_i) + u_2(\partial G_2/\partial x_i) + u_3(\partial G_3/\partial x_i) = 0 \quad (i = 2, 3), \tag{19}$$

and, of course, x must be feasible. These conditions reflect some simple geometric facts—namely, that the gradient of the objective function $f(x_2, x_3)$ must lie within the cone formed by the (negative) normals to the active constraints. In our case, if only constraint (d) is active, the KTK conditions are that the gradient of f at the solution must point in the direction of the negative x_3 axis. Clearly, this must be true or else one could improve upon the objective function by moving along the line represented by constraint (d). Similarly, if constraints (c) and (d) are

Table III

Critical Portfolios of the Markowitz Algorithm

	Minor Step 0	Step 1	Step 2	Step 3
s_E	∞	6	1.2273	1
x_1	(out) 0.0	(in) 0.0	(in) 0.454545	(in) 0.416667
x_2	(in) 1.0	(in) 1.0	(in) 0.545455	(in) 0.479167
x_3	(out) 0.0	(out) 0.0	(in) 0.000000	(in) 0.104167
x_4	(in) 0.0	(in) 0.0	(in) 1.818182	(in) 1.666667
x_5	(in) 2.0	(in) 2.0	(in) 1.090909	(in) 0.958333
x_6	(in) 0.0	(in) 0.0	(in) 0.00000	(in) 0.416667

both active (and hence the solution is at the lower left corner of the small feasible triangle), the KTK conditions state that the gradient of f must point somewhere into the lower left quadrant. Again, if this were not true, one could improve upon the objective function by moving along constraint (c) or (d). Finally, if neither constraint is active, the gradient of the objective function must be the zero vector.[6]

Performing the indicated differentiation in (19) gives us

$$P_2y_1^2(2y_2 - x_2) - P_1y_2^2(2y_1 - x_1) + u_2y_1^2y_2^2 = 0 \tag{20}$$

and

$$P_3y_1^2(2y_3 - x_3) - P_1y_3^2(2y_1 - x_1) + u_3y_1^2y_3^2 = 0, \tag{21}$$

where x_1 is to be replaced by $1 - x_2 - x_3$. From the geometry of the figure, there are only two cases to consider.

case (i): constraint (c) is not active; (d) is active.

case (ii): both constraints (c) and (d) are active.

First, take case (i). In this case, we have $x_3 = d$, but $G_2 > 0$. Consequently, $u_2 = 0$, and equations (20) and (21) provide two equations in two unknowns x_2 and u_3—namely,

$$P_2y_1^2(2y_2 - x_2) - P_1y_2^2(2y_1 - (1 - x_2 - d)) = 0 \tag{22}$$

and

$$P_3y_1^2(2y_3 - x_3) - P_1y_3^2(2y_1 - (1 - x_2 - d)) + u_3y_1^2y_3^2 = 0. \tag{23}$$

These equations will be valid only as long as $u_3 \geq 0$ and $G_2 \geq 0$. The maximizing point for the original logarithmic utility will be given by the fixed point $x = y$. This condition produces the solution:

$$x_1 = (1 - d)P_1/(P_1 + P_2), \tag{24}$$

$$x_2 = (1 - d)P_2/(P_1 + P_2), \tag{25}$$

$$x_3 = d, \tag{26}$$

and

$$u_3 = (P_1 + P_2)/(1 - d) - (P_3/d). \tag{27}$$

One can solve for the range where $u_3 \geq 0$, and one finds it to be the interval $-0.36 \leq b \leq 0$. Now, to see the behavior of the algorithm in this case, take $b = -\frac{1}{4}$ ($d = \frac{3}{16}$). The ultimate fixed point, given by equations (24) through (26), is that $x = (0.348214 \cdots, 0.464286 \cdots, 0.1875)$.

The major iterations, given by equations (22) and (23) and starting again at $(\frac{1}{3}, \frac{1}{3}, \frac{1}{3})$, produce the sequence shown in Table IV. Convergence is more rapid this time because we are iterating in a smaller subspace.[7]

[6] For a more detailed geometric discussion of the KTK conditions, see Panik [8].

[7] The reader should not be misled by the examples into believing that each solution of the quadratic subproblem will have the same set of active constraints. However, this set should settle down to a constant set in a neighborhood of the true solution.

Table IV

Quadratic Subproblem Solutions, Second Example

Major Step	x_1	x_2	x_3
0	0.333333	0.333333	0.333333
1	0.369048	0.443452	0.1875
2	0.348015	0.464485	0.1875
3	0.348214	0.464286	0.1875

Finally, take case (ii), where $x_1 = 1 - 3d$, $x_2 = 2d$, and $x_3 = d$. In this case, the only unknowns in equations (20) and (21) are u_2 and u_3. Solving for them yields

$$u_2 = P_1/(1 - 3d) - P_2/2d, \tag{28}$$

$$u_3 = P_1/(1 - 3d) - P_3/d. \tag{29}$$

Case (ii) will hold in the range where $u_2 \geq 0$, which can easily be shown from (28) to be the interval $-1/9 \leq b \leq 0$. In this range, the algorithm converges after two major steps, jumping from (1/3, 1/3, 1/3) to $(1 - 3d, 2d, d)$ and then remaining at the latter point.

IV. Convergence Questions

There are two issues: does the method always converge, and, when it does converge, what is the nature of the convergence? In general, the answer to the first question is no. Sufficient global convergence conditions have been presented by Wolfe [11, 12], and the method presented here does not meet them. However, it could be modified to do so,[8] or one could employ certain known globally convergent algorithms to reach a neighborhood of the fixed point. In practice, the method without modification may be very well behaved for most finance applications. (We have used the method, with no modification, to solve problems with twenty or so constraints of the type in the example and two hundred securities.)

As one sees from the examples, convergence, when it occurs, is very rapid. More specifically, the examples show what is termed "quadratic" convergence, in which the error at each major iteration is proportional to the square of the previous error. To see this in our example problem, let $y_i = P_i + e_i$ and $x_i = P_i + \bar{e}_i$ in equation (15). Then, after some manipulation, one has, after dropping terms of order e^3,

$$\bar{e}_i = P_i \sum (e_j^2/P_j) - e_i^2/P_i. \tag{30}$$

The type of algorithm discussed here, when one is close to the fixed point, is basically no more than a multidimensional version of the well-known Newton's method for finding roots of an equation $g(x) = 0$. In our case, $g(x)$ is the gradient vector of the objective function evaluated in the subspace of the active constraints.

[8] See, for example, Fletcher [2].

Local quadratic convergence has been proven in general for multidimensional Newton's method under certain conditions. "Local" means an iteration sequence starting sufficiently close to the fixed point. The conditions are that (a) $g(x)$ have continuous first derivatives, (b) $Dg(x)$ (the matrix with rows that are each gradients of the components of g) be nonsingular, and (c) there is a bound on the rate of change of $Dg(x)$ near the fixed point. In our case here, since $g(x)$ is the gradient of the objective function, these conditions translate into the requirements that concave $F(x)$ have continuous second derivatives and that there are bounds on the third derivatives of $F(x)$. For a review of these issues and a further guide to the literature, see Stoer [10].

V. Summary

We have presented a method of using iterative quadratic programming to solve a very general form of the portfolio selection problem. When the method converges, it converges very rapidly (quadratically) to the correct solution and could be modified to guarantee global convergence (although this may not be necessary in many applications).

We presented a simple example in which a logarithmic utility function was maximized, subject to a minimum-return constraint. More generally, the method can be used to maximize any concave, continuously twice-differentiable utility function with linear constraints. (The method can be extended to nonlinear constraints also by linearizing them.) Consequently, one could rephrase, if one wanted, all the language of the mean-variance "efficient frontier" of the Markowitz portfolio theory into a discussion of a $(\underline{U}, b)$ efficient frontier, where $\underline{U}$ is the expected utility and b is some additional single parameter.[9] The method here allows one to compute this frontier. This would enable one to conduct a type of appealing "risk-return" discussion in a less theoretically objectionable manner.

[9] There are a multitude of choices for b beyond the minimum-return parameter used in this paper. For example, if one allowed nonlinear constraints, b could represent a constraint on the variance or perhaps a risk measure derived from the utility function itself. Also, b need not even represent a constraint; for example, b could parameterize the utility function itself, such as choosing the utility function from the family $(1 + R)^b$.

REFERENCES

1. J. C. Cox. "Optimal Investment Strategy." Lecture notes, Berkeley Program in Finance Seminar, September 1984.
2. R. Fletcher. "An Algorithm for Solving Linearly Constrained Optimization Problems." *Mathematical Programming* 2 (1972), 133–65.
3. S. P. Han. "Superlinearly Convergent Variable Metric Algorithms for General Nonlinear Programming Problems." *Mathematical Programming* 11 (1976), 263–82.
4. Y. Kroll, H. Levy, and H. Markowitz. "Mean-Variance versus Direct Utility Maximization." *Journal of Finance* 39 (March 1984), 47–61.
5. H. W. Kuhn. "Nonlinear Programming: A Historical View." *SIAM-AMS Proceedings* 9 (1976), 1–26.
6. H. Markowitz. "The Optimization of a Quadratic Function Subject to Linear Constraints." *Naval Research Logistics Quarterly* 3 (1956), 111–33.

7. ———. *Portfolio Selection: Efficient Diversification of Investments*. New Haven: Yale University Press, 1970.
8. M. J. Panik. *Classical Optimization: Foundations and Extensions*, Chapter 12. Amsterdam, Oxford: North-Holland, 1976, 242–82.
9. S. M. Robinson. "Perturbed Kuhn-Tucker Points and Rates of Convergence for a Class of Nonlinear Programming Algorithms." *Mathematical Programming* 7 (1974), 1–16.
10. J. Stoer. "Principles of Sequential Quadratic Programming Methods for Solving Nonlinear Programs." In K. Schittkowski (ed.), *Computational Mathematical Programming* (Nato Advanced Science Institutes Series). Berlin: Springer-Verlag, 1985, 165–207.
11. P. Wolfe. "Convergence Conditions for Ascent Methods." *SIAM Review* 11 (April 1969), 226–35.
12. ———. "Convergence Conditions for Ascent Methods II: Some Corrections." *SIAM Review* 13 (April 1971), 185–88.