Contents lists available at ScienceDirect

European Journal of Operational Research

journal homepage: www.elsevier.com/locate/ejor

Decision Support

Large-scale MV efficient frontier computation via a procedure of parametric quadratic programming

Markus Hirschberger[a], Yue Qi[b], Ralph E. Steuer[c,*]

[a] *Department of Mathematics, University of Eichstätt-Ingolstadt, Eichstätt, Germany*
[b] *Department of Financial Management, College of Business, Nankai University, Tianjin, China*
[c] *Department of Banking and Finance, University of Georgia, Athens, GA 30602-6253, USA*

A R T I C L E I N F O

Article history:
Received 15 April 2009
Accepted 11 November 2009
Available online 19 January 2010

Keywords:
Bi-criterion
Portfolio selection
Parametric quadratic programming
Efficient frontiers
Dense covariance matrices
Large-scale
Hyperbolic segments

A B S T R A C T

Despite the volume of research conducted on efficient frontiers, in many cases it is still not the easiest thing to compute a mean–variance (MV) efficient frontier even when all constraints are linear. This is particularly true of large-scale problems having dense covariance matrices and hence they are the focus in this paper. Because standard approaches for constructing an efficient frontier one point at a time tend to bog down on dense covariance matrix problems with many more than about 500 securities, we propose as an alternative a procedure of parametric quadratic programming for more effective usage on large-scale applications. With the proposed procedure we demonstrate through computational results on problems in the 1000–3000 security range that the efficient frontiers of dense covariance matrix problems in this range are now not only solvable, but can actually be computed in quite reasonable time.

1. Introduction

When one hears reference to "mean–variance efficient frontiers," one almost certainly thinks of portfolio selection in finance. But mean–variance efficient frontiers need not necessarily be confined to finance. In most any problem with a stochastic linear objective, it is conceivable that a mean–variance efficient frontier could arise (Caballero et al., 2001). And in multiple criteria optimization, the nondominated set of a problem with one linear and one convex quadratic objective is in principle a mean–variance efficient frontier. It is just that we do not see many of these other types of applications. Consequently, with portfolio selection widely known, the paper is presented in this context for greatest familiarity.

Mean–variance efficient frontiers were introduced over fifty years ago by Roy (1952) and Markowitz (1952) in their work on portfolio selection. In the thirty years following, many papers were written on efficient frontier topics. But after about a fifteen-year period of somewhat diminished production, there has been a pick-up of interest in portfolio selection since about 2000. For instance, among others, there have been the papers by Chang et al. (2000), Ballestero and Plà-Santamaría (2003), Ruszczyński and Vanderbei (2003), Hallerbach et al. (2005), Fang and Wang (2005), Lin and Liu (2008), Branke et al. (2009) and Zhang et al. (2009) dealing with all manner of ideas from the use of fuzzy techniques and stochastic dominance concepts, to alternative risk measures, and to the application of evolutionary algorithms for dealing with non-smooth conditions.

Despite the volume of research conducted on efficient frontiers, in many cases it is still not the easiest thing to compute a mean–variance efficient frontier even when all constraints are linear. For short, let us call mean–variance problems with all linear constraints Markowitz problems.[1] The difficulty just mentioned is particularly true with large-scale Markowitz problems having dense covariance matrices. In fact, the LP-solvable models developed in Konno and Yamazaki (1991) and Mansini et al. (2003) have in part been in response to this. However, these models involve compromises. To address the challenges of mean–variance efficient frontiers without having to make compromises, and because the desire to be able to solve Markowitz problems with over 2000 securities is on the rise at large financial services organizations, we have this paper.

Currently, computing the efficient frontier of a large-scale problem (defined here to be between 1000 and 3000 securities) with a dense covariance is a task that can take up to hours with standard approaches. Of course problem size and how the efficient frontier is to be rendered play a role, but the primary culprit is the density

* Corresponding author.
E-mail address: rsteuer@uga.edu (R.E. Steuer).

[1] Because of the extent to which Markowitz has studied such problems.

of the covariance matrix. It is to be pointed out that portfolio problems in their natural state most often have dense covariance matrices. For instance, when running with a historical covariance matrix, using an index model for covariance coefficient estimation, or applying average correlation coefficient techniques as described in Elton et al. (2007), the resulting formulations will typically possess 100% dense covariance matrices. Although in the case of an index model the covariance matrix can be diagonalized to allow for faster solutions, the solutions are on the efficient frontier of the diagonalized problem which can essentially always be expected to be different from the efficient frontier of the problem in its natural state. Thus, to meet the task of being able to compute the true efficient frontier of a large-scale Markowitz problem with a dense covariance matrix, we have the algorithm of this paper along with computational results to place it in the frame of what can now be accomplished in this formerly almost inaccessible area of portfolio selection problem sizes.

Basically, there are two ways to solve for the efficient frontier of a Markowitz problem. One is to "construct" the efficient frontier *discretely*. This, to our knowledge, is the only way in which efficient frontiers are constructed in practice today. This approach is customarily carried out in an e-constraint[2] fashion by repetitively minimizing variance subject to, say, anywhere from 20 to 200 different lower bounds on expected return. We use the word "construct" in connection with this approach because only a dotted representation of the efficient frontier results.

An appeal of this approach is that it only requires an ordinary quadratic programming code to implement. A negative of the approach is that, because of the time to conduct a quadratic optimization, it is easy for run times to become enormous as problem size increases. For instance, to compute just a single e-constraint point on the efficient frontier of a 500-security problem with a dense covariance matrix, LINGO's quadratic solver takes roughly 35 seconds and Matlab's takes roughly 50 seconds.[3] With this implying 17.5 and 25 minutes, respectively, for just a 30-dot efficient frontier representation, we can see the computational difficulties lying before us with dense covariance matrix problems even before entering the 1000–3000 security range of this paper.

The other way is to "compute" the efficient frontier *parametrically*. This is accomplished by modeling the problem with a parameter and then smoothly varying the parameter to trace out the full equation structure of the complete efficient frontier. That is, the equations of all curves (hyperbolas, and typically many) that contribute segments to the efficient frontier are derived for a precise mathematical specification of the frontier. We use the word "compute" in connection with parametric procedures because the entire efficient frontier can be solved for in closed-form in this way. However, to our knowledge, it is not known that efficient frontiers are computed anywhere in practice in this way.

Even though Markowitz proposed a parametric procedure for tracing out the efficient frontier in the form of his critical line method in (1956), the critical line method is not seen utilized in practice (Michaud, 1989). While it might take a separate study to enumerate a full list of reasons why, one would almost certainly involve the position that the critical line method is difficult for many to understand and teach, and lengthy in its description. For instance, even in Markowitz's most recent book with Todd (2000), chapters are involved in its explanation. Also, because of CPU-time and memory requirements, it was difficult for the method to be applicable until about the early 1980 s. Furthermore, the whole idea of parametric quadratic programming, of which the critical line method is a variant, was not given a boost by Karmarkar (1984) as that paper created great interest in procedures (such as interior point methods) of polynomial-time complexity, of which parametric procedures for efficient frontier computation are not. Of course, complexity issues do not always correlate well with performance and interior point algorithms can't do all that a parametric procedure can do, but irrelevant or not, all of this was enough so that except for work by Markowitz and a few others such as Guddat (1976), Best (1996), Korhonen and Yu (1997) and Stein et al. (2007), little else has appeared in the literature on the topic of parametric procedures that can be adapted to efficient frontier computation.

With this as background, the procedure of this paper is a contribution to the literature in the following ways. As an alternative to the critical line method, the procedure pursues a more simplex-based strategy, utilizing better-known components from operations research, for greater transparency and understandability. Even when specified in detail as in this paper to allow programming by a reader, it is not of great length to fully describe. Also, with the procedure computational results are presented. In fact, just these results make the paper unique due to the lack of studies on the computation of the efficient frontiers of problems anywhere near in size to the problems addressed in this paper. The results demonstrate the performance of the procedure on the most difficult of Markowitz problems, those with dense covariance matrices in the large-scale range of 1000–3000 securities. Whereas on these problems discrete approaches with standard solvers can often take up to prohibitive time, the parametric procedure of this paper is able to compute the efficient frontier in at most minutes as shown later in the paper. While it might be noted that there have been two other studies on efficient frontier computation by Pardalos et al. (1994) and Jobst et al. (2001), they are not competitors in that they are mostly concerned with specialized formulations in which 225 securities is a large problem.

In summary, this paper not only presents a parametric procedure for the computation of the full efficient frontiers of dense covariance matrix problems in the range of 1000–3000 securities, but does so along with computational experiments conducted across a variety of test problems with different characteristics to document that the efficient frontiers of such problems can now be computed in times that would have to be considered reasonable, if not very reasonable, by almost any standard.

Also, because parametric procedures for computing efficient frontiers are not contained in any commercial packages of which we are aware, the structure of the particular code that we prepared for carrying out the experiments of this paper may be of interest to potential programmers. Called CIOS, the code reads from flat text files, possesses a random problem generator for research experimentation, is entirely self contained, and is written in Java for easy movement among our machines. Furthermore, because of a growing interest in portfolio selection with objectives beyond risk and return as expressed in papers, for instance, by Arenas Parra et al. (2001), Lo et al. (2003), Bana e Costa and Soares (2004), Ben Abdelaziz et al. (2007) and Ehrgott et al. (2009), the algorithm and code of this paper are purposely designed to permit its current *uni*-parametric capability to be generalized (not part of this paper) to a *multi*-parametric capability (to address problems with one quadratic and two or more linear objectives) in the future.

The paper is organized as follows. Section 2 overviews Markowitz portfolio selection and demonstrates the piecewise hyperbolic structure of the efficient frontier. Section 3 begins the detailed specification of the procedure by describing the reduced Karush–Kuhn–Tucker setup employed and the way in which bounds on the variables are handled. Section 4 describes how the procedure's Phase II obtains the topmost point of the efficient frontier. Then Section 5 describes how the procedure's Phase III parameterizes down the

[2] A technique from multiple criteria optimization that computes efficient points by converting all objectives to constraints except one (see Miettinen, 1999).

[3] All CPU times are from a Dell 2.66 GHz Core 2 Duo desktop at the University of Georgia.

efficient frontier to enumerate all of its hyperbolic segments. After Section 6 comments on the random problem generator built into CIOS, Section 7 reports on the results of computational experiments conducted on large-scale problems. Section 8 closes the paper with some concluding remarks.

2. Markowitz problem background

With Markowitz portfolio selection attempting to simultaneously minimize variance and maximize expected return, the problem in bi-criterion format is

$$\min \ \mathbf{x}^T\boldsymbol{\Sigma}\,\mathbf{x} = \sigma_p^2 \quad \text{variance} \tag{1.1}$$

$$\max \boldsymbol{\mu}^T\mathbf{x} = \mu_p \quad \text{expected return} \tag{1.2}$$

$$s.t. \quad \mathbf{1}^T\mathbf{x} = 1, \tag{1.3}$$

$$\mathbf{A}\mathbf{x} \lesseqqgtr \mathbf{b}, \tag{1.4}$$

$$\boldsymbol{\ell} \leqslant \mathbf{x} \leqslant \boldsymbol{\omega}, \tag{1.5}$$

in which

(a) $\mathbf{x} = (x_1, \ldots, x_n)$, n is the number of securities eligible for use in a portfolio, and x_i is the proportion of the capital available to be invested in security i

(b) $\boldsymbol{\Sigma}$ is the covariance matrix of the problem and $\boldsymbol{\mu}$ is the expected return vector

$$\boldsymbol{\Sigma} = \begin{bmatrix} \sigma_{11} & \sigma_{12} & \cdots & \sigma_{1n} \\ \sigma_{21} & \sigma_{22} & & \\ \vdots & & & \vdots \\ \sigma_{n1} & & \cdots & \sigma_{nn} \end{bmatrix} \quad \boldsymbol{\mu} = \begin{bmatrix} \mu_1 \\ \mu_2 \\ \vdots \\ \mu_n \end{bmatrix}$$

(c) an $\mathbf{x}$-vector is a *portfolio* if it satisfies (1.3), and is a *feasible* portfolio if it also satisfies (1.4) and (1.5).

As a result of its multiobjective nature, (1.1)–(1.5) has two feasible regions, $S \subset \mathbb{R}^n$ in decision space and $Z \subset \mathbb{R}^2$ in criterion space. S is the set of all feasible portfolios, and Z is the set of all image vectors, in terms of standard deviation and expected return,[4] of the portfolios in S. A feasible portfolio has a point on the efficient frontier if and only if there is no other feasible portfolio that has a higher expected return without a higher variance, or a lower variance without a lower expected return. Or in more mathematical terms, $\mathbf{x}^* \in S$ is an *efficient* portfolio if and only if there does not exist another $\mathbf{x} \in S$ such that $\mu_p(\mathbf{x}) > \mu_p(\mathbf{x}^*)$ and $\sigma_p(\mathbf{x}) \leqslant \sigma_p(\mathbf{x}^*)$, or $\sigma_p(\mathbf{x}) < \sigma_p(\mathbf{x}^*)$ and $\mu_p(\mathbf{x}) \geqslant \mu_p(\mathbf{x}^*)$. Because of the noncomparability of such portfolios, the solution set of (1.1)–(1.5) is then the set of all efficient portfolios $\mathbf{x}^* \in S$ along with the set of all of their image vectors $(\sigma_p(\mathbf{x}^*), \mu_p(\mathbf{x}^*)) \in Z$, which otherwise is known in portfolio selection as the efficient frontier.

As is known, the efficient frontier of a Markowitz problem is piecewise hyperbolic. That is, it consists of a connected series of segments from different hyperbolas. This follows from the fact that the image set of any straight line in S is hyperbolic. In this way, the inverse image sets of the different hyperbolic segments in Z form a piecewise linear path in S.

To illustrate the piecewise hyperbolic structure of an efficient frontier, let us consider two 30-security Markowitz problems whose efficient frontiers are in Figs. 1 and 2. Shown in the figures are the hyperbolas that supply the segments out of which the efficient frontiers consist. For instance, the topmost two dots on each frontier define the portion of the most nested hyperbola that forms the topmost hyperbolic segment of the efficient frontier. The reason the first efficient frontier has fewer hyperbolic segments is because it comes from a problem whose $\boldsymbol{\Sigma}$ is fully dense, whereas the second comes from a problem whose $\boldsymbol{\Sigma}$ is diagonal. It is typical for the number of efficient frontier hyperbolic segments to increase as covariance matrix density decreases.

[4] While theory and computation are carried out in terms of variance, efficient frontiers are typically shown to users with standard deviation on the horizontal axis.

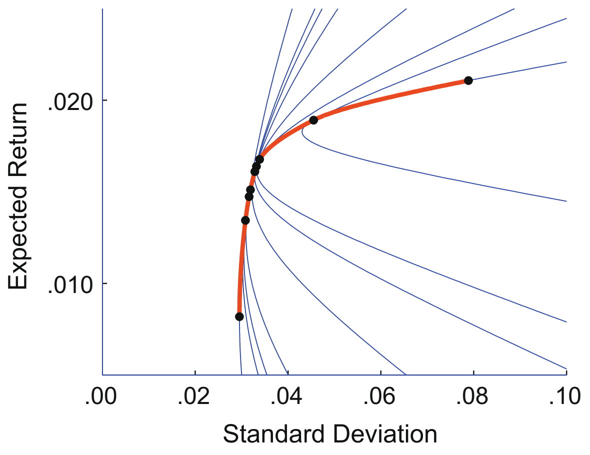

Fig. 1. Efficient frontier of a 30-security 100% dense covariance matrix problem and the 8 hyperbolas that contribute segments to the efficient frontier.

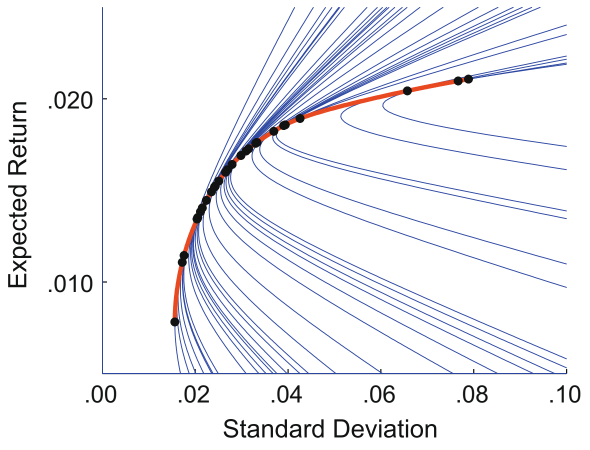

Fig. 2. Efficient frontier of a 30-security diagonal covariance matrix problem and the 29 hyperbolas that contribute segments to the efficient frontier.

To provide a parametric specification of the efficient frontier, information as in Tables 1 and 2, whose contents are from the efficient frontier of Fig. 1, must be generated. With the rows indexed from the top of the frontier down, Table 1 provides information about the efficient frontier's hyperbolic segments, and Table 2 specifies the endpoints of the linear line segments in S pertaining to them.

Consider hyperbolic segment h and the entries in that row of Table 1. The a_0, a_1, a_2 specify the hyperbola that supplies the hyperbolic segment. They do so in such a way that the standard deviation of a point on the hyperbolic segment can be expressed as a

Table 1
Coefficients that enable the hyperbolas of the segments that make up the efficient frontier of Fig. 1 to be expressed as a function of expected return along with the intervals of expected return over which the segments operate.

Hyperbolic segment	a_0	a_1	a_2	μ^{upper}	μ^{lower}
1	.00190	−.20628	5.62990	.02107	.01891
2	.00035	−.04194	1.29597	.01891	.01676
⋮	⋮			⋮	
8	.00106	−.04636	2.82223	.01344	.00818

Table 2
Compositions of the portfolios that define the endpoints of the straight lines in S that correspond to the hyperbolic segments of the efficient frontier in Fig. 1.

Endpoint portfolio	x_1	x_2	x_3	x_4	x_5	x_6	$\cdots$
$\mathbf{x}^1$	.0	.0	.0	.0	1.0000	.0	$\cdots$
$\mathbf{x}^2$	.0	.0	.0	.51201	.48799	.0	
$\mathbf{x}^3$	.0	.0	.0	.35396	.31795	.0	
$\vdots$	$\vdots$						
$\mathbf{x}^9$	.21886	.0	.0	.00611	.12361	.0	

function of its expected return as in (2). The μ^{upper}, μ^{lower} specify the range of expected returns over which the hyperbolic segment operates. As for the linear line segment in S that corresponds to the hyperbolic segment, it is specified in Table 2 by the straight line whose endpoints are $\mathbf{x}^h$ and $\mathbf{x}^{h+1}$.

To illustrate use of the tables, suppose we would like to know more about the point on the efficient frontier whose expected return $\mu^* = .018$. This places the point (σ^*, μ^*) on the 2nd hyperbolic segment. To calculate the standard deviation σ^*, we use the a_0, a_1, a_2 of the hyperbolic segment as follows:

$$\begin{aligned}\sigma^* &= \sqrt{a_0 + a_1\mu^* + a_2(\mu^*)^2} \qquad (2)\\ &= \sqrt{.00035 - .04194(.018) + 1.29597(.018)^2} = .03913.\end{aligned}$$

To calculate the composition of the portfolio $\mathbf{x}^*$ corresponding to efficient frontier point (σ^*, μ^*), we use the μ^{upper}, μ^{lower} values of the hyperbolic segment as follows:

$$\begin{aligned}\mathbf{x}^* &= \frac{\mu^* - \mu^{lower}}{\mu^{upper} - \mu^{lower}}\mathbf{x}^2 + \frac{\mu^{upper} - \mu^*}{\mu^{upper} - \mu^{lower}}\mathbf{x}^3 \qquad (3)\\ &= \frac{.018 - .01676}{.01891 - .01676}\mathbf{x}^2 + \frac{.01891 - .018}{.01891 - .01676}\mathbf{x}^3\\ &= (.0, .0, .0, .44511, .41600, .0, \text{etc.}).\end{aligned}$$

Instead of being of sizes 8×5 and 9×30, the tables for the efficient frontier of Fig. 2 if displayed would be of sizes 29×5 and 30×30, respectively. Thus, because of the work involved, we can expect the parametric specification of an efficient frontier with more hyperbolic segments to take longer to compute than an efficient frontier with fewer hyperbolic segments.

3. Standardized format and Karush–Kuhn–Tucker set up

To prepare (1.1)–(1.5) for the computation of its efficient frontier, we now do three things. The first, because (1.1)–(1.5) is convex when $\mathbf{\Sigma}$ is a covariance matrix, is to re-formulate (1.1)–(1.5) as

$$\begin{aligned}&\max\{-\mathbf{x}^T\mathbf{\Sigma}\mathbf{x} + \lambda\, \boldsymbol{\mu}^T\mathbf{x}\} \quad \lambda \geqslant 0 && (3.1)\\ s.t.\quad &\mathbf{Hx} = \mathbf{e}, && (3.2)\\ &\mathbf{Gx} \leqslant \mathbf{g}, && (3.3)\\ &\mathbf{x} \leqslant \boldsymbol{\omega}, && (3.4)\\ &\mathbf{x} \geqslant \ell, && (3.5)\end{aligned}$$

where constraints (1.3)–(1.5) are re-written in matrix form as (3.2)–(3.5). That is, all left-hand side coefficients of the equality constraints in (1.3) and (1.4) form $\mathbf{H}$ and all left-hand side coefficients of the inequality constraints in (1.4) after they have been rearranged if necessary form $\mathbf{G}$. Then, by observing that if (3.1)–(3.5) can be solved for all $\lambda \geqslant 0$, a set of $\mathbf{x}$-vectors will be obtained whose set of (σ_p, μ_p) image vectors is precisely the efficient frontier.

A second thing, to lighten the computer load when $\ell \neq \mathbf{0}$, is to translate the axis system to the point $\ell \in \mathbb{R}^n$. This saves the simplex-based procedure a row for each x_i lower bound that is nonzero. This is done by replacing $\mathbf{x}$ with $\mathbf{x} + \ell$. In terms of the new $\mathbf{x}$-vector, we have

$$\begin{aligned}&\max\{-\mathbf{x}^T\mathbf{\Sigma}\mathbf{x} - 2\ell^T\mathbf{\Sigma}\mathbf{x} - \ell^T\mathbf{\Sigma}\ell + \lambda\boldsymbol{\mu}^T\mathbf{x} + \lambda\boldsymbol{\mu}^T\ell\} \quad \lambda \geqslant 0\\ s.t.\quad &\mathbf{Hx} = \mathbf{e} - \mathbf{H}\ell,\\ &\mathbf{Gx} \leqslant \mathbf{g} - \mathbf{G}\ell,\\ &\mathbf{x} \leqslant \boldsymbol{\omega} - \ell,\\ &\mathbf{x} \geqslant \ell - \ell.\end{aligned}$$

After dropping the $\ell^T\mathbf{\Sigma}\ell$ and $\lambda\boldsymbol{\mu}^T\ell$ constant terms from the objective function and defining $\mathbf{d} \geqslant \mathbf{0}$ (which may require changes in sign to some of the rows of $\mathbf{H}$), $\mathbf{b}$, and $\boldsymbol{\beta}$ for the new right-hand sides, we have the problem to be solved in parametric quadratic programming *standardized format*

$$\begin{aligned}&\max\{-\mathbf{x}^T\mathbf{\Sigma}\mathbf{x} + \lambda\, \boldsymbol{\mu}^T\mathbf{x} - 2\, \ell^T\mathbf{\Sigma}\mathbf{x}\} \quad \lambda \geqslant 0 && (4.1)\\ s.t.\quad &\mathbf{Hx} = \mathbf{d}, && (4.2)\\ &\mathbf{Gx} \leqslant \mathbf{b}, && (4.3)\\ &\mathbf{x} \leqslant \boldsymbol{\beta}, && (4.4)\\ &\mathbf{x} \geqslant \mathbf{0}. && (4.5)\end{aligned}$$

The third thing, to facilitate the transition to the Karush–Kuhn–Tucker Conditions, is to write (4.1)–(4.5) in the style of Winston (2004) as

$$\begin{aligned}&\max\{f(\mathbf{x})\} && (5)\\ s.t.\quad &h_i(\mathbf{x}) = d_i \quad i = 1, \ldots, l,\\ &g_j(\mathbf{x}) \leqslant b_j \quad j = 1, \ldots, m,\\ &\mathbf{x} \leqslant \boldsymbol{\beta},\\ &\mathbf{x} \geqslant \mathbf{0},\end{aligned}$$

and then note that if $f : \mathbb{R}^n \to \mathbb{R}$ is concave and all constraints are convex, $\mathbf{x} \in \mathbb{R}^n$ solves (5) if and only if there exist vectors $\mathbf{v} \in \mathbb{R}^l$, $\mathbf{u}^s \in \mathbb{R}^m$, $\mathbf{u}^\beta \in \mathbb{R}^n$ and $\mathbf{u}^x \in \mathbb{R}^n$ such that $\mathbf{x}$ satisfies the Karush–Kuhn–Tucker Conditions

$$\begin{aligned}&\frac{\partial f(\mathbf{x})}{\partial x_j} - \sum_{i=1}^{l} v_i \frac{\partial h_i(\mathbf{x})}{\partial x_j} - \sum_{i=1}^{m} u_i^s \frac{\partial g_i(\mathbf{x})}{\partial x_j} - u_j^\beta + u_j^x = 0 \quad j = 1, \ldots, n,\\ &h_i(\mathbf{x}) = d_i \quad i = 1, \ldots, l,\\ &g_j(\mathbf{x}) \leqslant b_j \quad j = 1, \ldots, m,\\ &\mathbf{x} \leqslant \boldsymbol{\beta},\\ &\mathbf{x} \geqslant \mathbf{0},\\ &\mathbf{u}^s \geqslant \mathbf{0},\\ &\mathbf{u}^\beta \geqslant \mathbf{0},\\ &\mathbf{u}^x \geqslant \mathbf{0},\\ &(b_j - g_j(\mathbf{x}))u_j^s = 0 \quad j = 1, \ldots, m,\\ &(\beta_j - x_j)u_j^\beta = 0 \quad j = 1, \ldots, n,\\ &x_j u_j^x = 0 \quad j = 1, \ldots, n,\\ &\mathbf{v} \text{ unrestricted.}\end{aligned}$$

Inserting (4.1)–(4.5) into the above Karush–Kuhn–Tucker Conditions, we have in matrix format the following system:

$$\begin{aligned}&2\mathbf{\Sigma}\mathbf{x} - \lambda\boldsymbol{\mu} + 2\mathbf{\Sigma}\ell + \mathbf{H}^T\mathbf{v} + \mathbf{G}^T\mathbf{u}^s + \mathbf{I}_n\mathbf{u}^\beta - \mathbf{I}_n\mathbf{u}^x = \mathbf{0}, && (6.1)\\ &\mathbf{Hx} = \mathbf{d}, && (6.2)\\ &\mathbf{Gx} \leqslant \mathbf{b}, && (6.3)\\ &\mathbf{I}_n\mathbf{x} \leqslant \boldsymbol{\beta}, && (6.4)\\ &\mathbf{x} \geqslant \mathbf{0}, \quad \mathbf{u}^s \geqslant \mathbf{0}, \quad \mathbf{u}^\beta \geqslant \mathbf{0}, \quad \mathbf{u}^x \geqslant \mathbf{0}, && (6.5)\\ &(b_j - g_j(\mathbf{x}))u_j^s = 0, \quad j = 1, \ldots, m \qquad (\beta_j - x_j)u_j^\beta = 0,\\ &\qquad j = 1, \ldots, n, && (6.6)\\ &x_i u_i^x = 0, \quad i = 1, \ldots, n, && (6.7)\\ &\mathbf{v} \text{ unrestricted.} && (6.8)\end{aligned}$$

Moving the two constant terms to the right in (6.1), adding slack variables $\mathbf{s} \in \mathbb{R}^m$ to (6.3), and replacing the $(b_j - g_j(\mathbf{x}))$ with their s_j variables in (6.6), we obtain the *full* Karush–Kuhn–Tucker system

$$2\mathbf{\Sigma x} + \mathbf{H}^T\mathbf{v} + \mathbf{G}^T\mathbf{u}^s + \mathbf{I}_n\mathbf{u}^\beta - \mathbf{I}_n\mathbf{u}^x = -2\mathbf{\Sigma}\ell + \lambda\boldsymbol{\mu}, \tag{7.1}$$

$$\mathbf{Hx} = \mathbf{d}, \tag{7.2}$$

$$\mathbf{Gx} + \mathbf{I}_m\mathbf{s} = \mathbf{b}, \tag{7.3}$$

$$\mathbf{I}_n\mathbf{x} \leqslant \boldsymbol{\beta}, \tag{7.4}$$

$$\mathbf{x} \geqslant \mathbf{0}, \quad \mathbf{u}^s \geqslant \mathbf{0}, \quad \mathbf{u}^\beta \geqslant \mathbf{0}, \quad \mathbf{u}^x \geqslant \mathbf{0}, \quad \mathbf{s} \geqslant \mathbf{0}, \tag{7.5}$$

$$s_j u_j^s = 0, \quad j = 1, \ldots, m \qquad (\beta_j - x_j)u_j^\beta = 0, \quad j = 1, \ldots, n, \tag{7.6}$$

$$x_i u_i^x = 0, \quad i = 1, \ldots, n, \tag{7.7}$$

$$\mathbf{v} \text{ unrestricted}. \tag{7.8}$$

A problem with the full Karush–Kuhn–Tucker system is that it is too big. Just (7.1)–(7.4) alone is $(2n + m + l) \times (3n + l + 2m)$. Fortunately, it is possible to reduce the system by modeling the upper bound constraints $\mathbf{x} \leqslant \boldsymbol{\beta}$ implicitly. This enables the inequalities of (7.4) to be eliminated.

Following known procedures, the upper bound constraints $\mathbf{x} \leqslant \boldsymbol{\beta}$ can be modeled by replacing $\mathbf{x}$ by $\bar{\mathbf{x}} \in \mathbb{R}^n$ where it is understood that we *substitute* any x_i in $\bar{\mathbf{x}}$ by $\beta_i - x_i^*$ whenever x_i hits its β_i upper bound, and that we *re-substitute* any $\beta_i - x_i^*$ in $\bar{\mathbf{x}}$ by x_i whenever x_i^* hits its β_i upper bound.

In addition, we can combine $\mathbf{u}^\beta$ and $\mathbf{u}^x$ into one vector $\bar{\mathbf{u}} \in \mathbb{R}^n$. This is possible because when x_i is in $\bar{\mathbf{x}}, u_i^x$ is unnecessary; and when $\beta_i - x_i^*$ is in $\bar{\mathbf{x}}$, u_i^β is unnecessary. Let

$$I^* = \{i | i\text{th component of } \bar{\mathbf{x}} \text{ is currently substituted by } \beta_i - x_i^*\}. \tag{8}$$

Then $\mathbf{I}_n\mathbf{u}^\beta - \mathbf{I}_n\mathbf{u}^x$ in (7.1) can be replaced by $\mathbf{D}\bar{\mathbf{u}}$ where $\mathbf{D}$ is a diagonal matrix such that if $i \in I^*$, $d_{ii} = 1$, otherwise $d_{ii} = -1$. This enables us to form the *reduced* Karush–Kuhn–Tucker system

$$2\mathbf{\Sigma}\bar{\mathbf{x}} + \mathbf{H}^T\mathbf{v} + \mathbf{G}^T\mathbf{u}^s + \mathbf{D}\bar{\mathbf{u}} = -2\mathbf{\Sigma}\ell + \lambda\boldsymbol{\mu}, \tag{9.1}$$

$$\mathbf{H}\bar{\mathbf{x}} = \mathbf{d}, \tag{9.2}$$

$$\mathbf{G}\bar{\mathbf{x}} + \mathbf{I}_m\mathbf{s} = \mathbf{b}, \tag{9.3}$$

$$\bar{\mathbf{x}} \geqslant \mathbf{0}, \quad \mathbf{u}^s \geqslant \mathbf{0}, \quad \bar{\mathbf{u}} \geqslant \mathbf{0}, \quad \mathbf{s} \geqslant \mathbf{0}, \tag{9.4}$$

$$s_j u_j^s = 0, \quad j = 1, \ldots, m \qquad \bar{x}_i\bar{u}_i = 0, \quad i = 1, \ldots, n, \tag{9.5}$$

$$\mathbf{v} \text{ unrestricted}. \tag{9.6}$$

The significance of this system is that Eaves (1971) has proved that (9.1)–(9.6) is solvable if and only if there exists a basis of (9.1)–(9.3) that solves the system. Since the reduced Karush–Kuhn–Tucker system is solvable for all $\lambda \geqslant 0$ by virtue of the boundedness imposed by (4.4) and (4.5), the task is now to find all such bases for $\lambda \geqslant 0$.

4. Locating maximum expected return point

In computing an efficient frontier, three phases are distinguished. Phase I, like in linear programming, is simply to find a feasible vertex, but here, it need only be a vertex of the region of solutions defined by the constraints (9.2) and (9.3). In Phase II, we maximize the linear portion of the objective in (3.1) for the ultimate purpose of obtaining a maximizing solution that satisfies the reduced Karush–Kuhn–Tucker system (9.1)–(9.6) for some $\lambda \geqslant 0$.

4.1. Phase I

To find a vertex of the region defined by the constraints (9.2) and (9.3), we solve the linear program

$$\min \left\{ \sum_{i=1}^m a_i^1 + \sum_{j=1}^l a_j^2 \right\} \tag{10}$$

$$\begin{aligned} s.t. \quad & \mathbf{H}\bar{\mathbf{x}} + \mathbf{I}_l\mathbf{a}^2 = \mathbf{d}, \\ & \mathbf{G}\bar{\mathbf{x}} + \mathbf{I}_m\mathbf{s} - \mathbf{I}_m\mathbf{a}^1 = \mathbf{b}, \\ & \bar{\mathbf{x}} \geqslant \mathbf{0}, \quad \mathbf{s} \geqslant \mathbf{0}, \quad \mathbf{a}^1 \geqslant \mathbf{0}, \quad \mathbf{a}^2 \geqslant \mathbf{0}, \end{aligned}$$

in which $\mathbf{a}^1 \in \mathbb{R}^m$ and $\mathbf{a}^2 \in \mathbb{R}^l$ are vectors of artificial variables. The starting basic variables are a_j^2, $j = 1, \ldots, l$, along with s_i for each $b_i \geqslant 0$ and a_i^1 for each $b_i < 0$, $i = 1, \ldots, m$. Unless the optimal objective function value of (10) is non-zero, in which case we stop because no feasible solution of (4.1)–(4.5) exists, we pass the optimal $(\bar{\mathbf{x}}, \mathbf{s})$ solution of Phase I to Phase II (making sure, in cases of degeneracy, that no a_i^1 or a_j^2 are left remaining in the basis).

4.2. Phase II

After receiving an $(\bar{\mathbf{x}}, \mathbf{s})$ from Phase I, the purpose of Phase II is to proceed from this point to the maximum expected return point of the efficient frontier by solving the linear program

$$\max\{\boldsymbol{\mu}^T\bar{\mathbf{x}}\} \tag{11.1}$$

$$s.t. \quad \mathbf{H}\bar{\mathbf{x}} \ = \mathbf{d}, \tag{11.2}$$

$$\mathbf{G}\bar{\mathbf{x}} + \mathbf{I}_m\mathbf{s} = \mathbf{b}, \tag{11.3}$$

$$\bar{\mathbf{x}} \geqslant \mathbf{0}, \quad \mathbf{s} \geqslant \mathbf{0}, \tag{11.4}$$

$$\mathbf{v} \quad \text{unrestricted}. \tag{11.5}$$

If (11.1)–(11.5) yields a unique maximizing solution, we are ready to move on to Phase III. If there are alternative maxima, then it must be assured that the point passed on to Phase III is one of smallest standard deviation. When such a point is needed, one can always be obtained by temporarily perturbing $\boldsymbol{\mu}$ as in Bank et al. (1983).

5. Parameterizing down the frontier

Phase III operates on the reduced Karush–Kuhn–Tucker system by maintaining feasibility as λ is parameterized down from ∞ to 0, thus completing the task of finding $\mathbf{x}$-vector solutions of (4.1)–(4.5) for all $\lambda \geqslant 0$. This is done by employing a modification of Guddat's (1974) multi-parametric Phase III. To describe the method as applied to the reduced Karush–Kuhn–Tucker system, let

$$\mathbf{A} = \begin{bmatrix} 2\mathbf{\Sigma} & \mathbf{H}^T & \mathbf{G}^T & \mathbf{I}_n & \mathbf{0} \\ \mathbf{H} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{G} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{I}_m \end{bmatrix} \quad \mathbf{b}^1 = \begin{bmatrix} -2\mathbf{\Sigma}\ell \\ \mathbf{d} \\ \mathbf{b} \end{bmatrix} \quad \mathbf{b}^2 = \begin{bmatrix} \boldsymbol{\mu} \\ \mathbf{0} \\ \mathbf{0} \end{bmatrix}$$

and $\hat{\mathbf{x}} = (\bar{\mathbf{x}}, \mathbf{v}, \mathbf{u}^s, \bar{\mathbf{u}}, \mathbf{s})$. The system has to be modified in each step to account for the currently substituted variables. After recalling index set I^* from (8), for all $i \in I^*$ multiply column i and column $n + m + l + i$ of $\mathbf{A}$ by -1 and subtract column i of $\mathbf{A}$ times ω_i from $\mathbf{b}^1$.

Denoting by $\mathbf{B}$ the columns of $\mathbf{A}$ that comprise a basis, the (9.1)–(9.3) portion of the reduced Karush–Kuhn–Tucker system can be rewritten as

$$\mathbf{B}\hat{\mathbf{x}}_B = \mathbf{b}^1 + \lambda\mathbf{b}^2,$$

where $\hat{\mathbf{x}}_B = (\bar{\mathbf{x}}_B, \mathbf{v}, \mathbf{u}_B^s, \bar{\mathbf{u}}_B, \mathbf{s}_B)$. Since a basis of the $(n + l + m) \times (2n + l + 2m)$ reduced Karush–Kuhn–Tucker system contains $(n + l + m)$ variables, we observe two things about Phase III bases. By virtue of the complementary slackness conditions of (9.5), one is that all $\mathbf{v}$-variables are always in a basis. The other is that all nonbasic variables always have their complementary variables in the basis. This means that

(a) whenever we pivot in Phase III, whatever variable leaves, in comes its complementary slackness counterpart;

(b) the initial solution for Phase III is constructed as follows: whenever a variable $\bar{\mathbf{x}}$ or $\mathbf{s}$ was basic in Phase II, it will be basic in Phase III. Whenever the variable was nonbasic in Phase II, its complementary slackness counterpart $\bar{\mathbf{u}}$ or $\mathbf{u}^s$ will be basic in Phase III. Moreover, index set I^* is also taken as is from the solution of Phase II.

5.1. For computing the linear line segments

Let $\mathbf{B}_1$ be the basis of the initial solution for Phase III, and let J_{B_1} be its basic index set. Since $\mathbf{B}_1$ is invertible,

$$\hat{\mathbf{x}}_{B_1} = \mathbf{B}_1^{-1}\mathbf{b}^1 + \lambda\mathbf{B}_1^{-1}\mathbf{b}^2.$$

Now, delete the components corresponding to the $\mathbf{v}$-variables in the right-hand side vectors $\mathbf{B}_1^{-1}\mathbf{b}^1$ and $\mathbf{B}_1^{-1}\mathbf{b}^2$ and denote the resulting right-hand side vectors $\mathbf{r}^1 \in \mathbb{R}^{n+m}$ and $\mathbf{r}^2 \in \mathbb{R}^{n+m}$, respectively. Then use the $\mathbf{r}$-vectors to define vectors $\xi^1 \in \mathbb{R}^n$ and $\mathbf{\Delta}^1 \in \mathbb{R}^n$ where

$$\xi_i^1 = \begin{cases} \ell_i & i \notin J_{B_1} \text{ and } i \notin I^*, \\ \omega_i & i \notin J_{B_1} \text{ and } i \in I^*, \\ \ell_i + r_{j_i}^1 & i \in J_{B_1} \text{ and } i \notin I^*, \\ \omega_i - r_{j_i}^1 & i \in J_{B_1} \text{ and } i \in I^*, \end{cases} \qquad \Delta_i^1 = \begin{cases} 0 & i \notin J_{B_1} \text{ and } i \notin I^*, \\ 0 & i \notin J_{B_1} \text{ and } i \in I^*, \\ r_{j_i}^2 & i \in J_{B_1} \text{ and } i \notin I^*, \\ -r_{j_i}^2 & i \in J_{B_1} \text{ and } i \in I^*, \end{cases}$$

where j_i is the index of $\bar{x}_i$ in $r_j^1,\ i \in J_{B_1}$. Then

$$\mathbf{x} = \xi^1 + \lambda\mathbf{\Delta}^1$$

solves (3.1) for all $\lambda \geqslant 0$ as long as all components of $\bar{\mathbf{x}}_{B_1}$ stay within their upper bounds and

$$(\bar{\mathbf{x}}_{B_1}, \mathbf{u}_{B_1}^s, \bar{\mathbf{u}}_{B_1}, \mathbf{s}_{B_1}) = \mathbf{r}^1 + \lambda\mathbf{r}^2 \geqslant \mathbf{0}.$$

In other words, λ may be increased up to

$$\lambda_{max} = \min_{j=1,\dots,n+m} \begin{cases} \frac{-r_j^1}{r_j^2} & r_j^2 < 0 \\ \frac{\beta_{i_j} - r_j^1}{r_j^2} & r_j^2 > 0 \text{ and } i_j \in \{1,\dots,n\} \end{cases}$$

or decreased down to

$$\lambda_{min} = \max_{j=1,\dots,n+m} \begin{cases} \frac{-r_j^1}{r_j^2} & r_j^2 > 0 \\ \frac{\beta_{i_j} - r_j^1}{r_j^2} & r_j^2 < 0 \text{ and } i_j \in \{1,\dots,n\} \end{cases}$$

at which point λ hits a binding constraint. This means that either

(a) a basic variable $\bar{x}_i,\ u_i^s,\ \bar{u}_i$ or s_i is driven to 0, in which case we remove the variable and enter its counterpart $\bar{u}_i,\ s_i,\ \bar{x}_i$ or u_i^s to obtain $\mathbf{B}_2$, or
(b) a basic variable $\bar{x}_j$ hits its upper bound, in which case, after substituting for $\bar{x}_j$ and $\bar{u}_j$ (possibly affecting the contents of I^*), we exchange the variables to obtain $\mathbf{B}_2$.

In the initial solution, we will have $\lambda_{max} = \infty$ since the solution of Phase II already maximizes the linear objective. We set $\lambda^2 := \lambda_{min}$ and $\lambda^1 := \lambda_{max}$. Repeating in this way, we compute a sequence of linear line segments

$$\xi^h + \lambda\mathbf{\Delta}^h \quad \lambda \in [\lambda^{h+1}, \lambda^h]$$

with $\lambda^{h+1} < \lambda^h$ until λ^{h+1} becomes zero (or negative, at which point we stop and set $\lambda^{h+1} = 0$). While these are the efficient line segments that "zig-zag" through S, their outcome vectors form the hyperbolic segments comprising the efficient frontier of Z.

5.2. Iterative procedure

Letting the linear line segment endpoint portfolios of S be denoted $\mathbf{x}^h$ and their images in Z be denoted (σ^h, μ^h), the Phase III procedure of the paper is as follows.

```
STEP 1. Let $\mathbf{B}_1$ be from Phase II, $\lambda^1 = \infty$, and $h = 1$.
STEP 2. From the $\mathbf{r}^1$ and $\mathbf{r}^2$ associated with $\mathbf{B}_h^{-1}\mathbf{b}^1$ and $\mathbf{B}_h^{-1}\mathbf{b}^2$, construct
            $\xi^h$ and $\mathbf{\Delta}^h$
STEP 3. If $h > 1$, go to Step 4. Otherwise, do
            $\mathbf{x}^1 = \xi^1$
            $\mu^1 = \boldsymbol{\mu}^T\mathbf{x}^1$
            write to a file $\mathbf{x}^1$
STEP 4. Compute $\lambda^{h+1}$. If $\lambda^{h+1} \leqslant 0$, replace by $\lambda^{h+1} = 0$.
STEP 5. Do
            $\mathbf{x}^{h+1} = \xi^h + \lambda^{h+1}\mathbf{\Delta}^h$
            $\mu^{h+1} = \boldsymbol{\mu}^T\mathbf{x}^{h+1}$
            if $\boldsymbol{\mu}^T\mathbf{\Delta}^h > 0$, compute $a_0^h,\ a_1^h,\ a_2^h$
            write to a file $h,\ \mu^h,\ \mu^{h+1},\ \lambda^h,\ \lambda^{h+1},\ a_0^h,\ a_1^h,\ a_2^h$ and $\mathbf{x}^{h+1}$
            as long as $\lambda^{h+1} > 0$, pivot to $\mathbf{B}_{h+1}$
STEP 6. If $\lambda^{h+1} = 0$, stop. Else, $h = h + 1$ and go to Step 2.
```

We now explain the $a_0^h,\ a_1^h,\ a_2^h$. They are used to compute the σ of the (σ, μ) that lie along the hth hyperbolic segment of the efficient frontier, which corresponds to the hth linear line segment whose endpoints are given by $\mathbf{x}^h$ and $\mathbf{x}^{h+1}$. Thus for $\lambda \in [\lambda^{h+1}, \lambda^h]$,

$$\mu = \boldsymbol{\mu}^T\xi^h + \lambda\boldsymbol{\mu}^T\Delta^h, \tag{12}$$

$$\sigma = \sqrt{(\xi^h)^T\mathbf{\Sigma}\xi^h + \lambda 2(\xi^h)^T\mathbf{\Sigma}\mathbf{\Delta}^h + \lambda^2(\mathbf{\Delta}^h)^T\mathbf{\Sigma}\mathbf{\Delta}^h}. \tag{13}$$

If $\boldsymbol{\mu}^T\mathbf{\Delta}^h = 0$, the hyperbolic segment is just a single point, and for such h, we do not compute $a_0^h,\ a_1^h$ and a_2^h . Otherwise, from (12) we obtain

$$\lambda = \frac{\mu - \boldsymbol{\mu}^T\xi^h}{\boldsymbol{\mu}^T\mathbf{\Delta}^h}.$$

Inserting λ into (13) and doing a little algebra, we obtain σ as a function of μ as follows:

$$\sigma = \sqrt{a_0^h + a_1^h\mu + a_2^h(\mu)^2} \quad \mu \in [\mu^{h+1}, \mu^h], \tag{14}$$

where

$$a_0^h = (\xi^h)^T\mathbf{\Sigma}\xi^h - \frac{2\boldsymbol{\mu}^T\xi^h}{\boldsymbol{\mu}^T\mathbf{\Delta}^h}(\xi^h)^T\mathbf{\Sigma}\mathbf{\Delta}^h + \frac{(\boldsymbol{\mu}^T\xi^h)^2}{(\boldsymbol{\mu}^T\mathbf{\Delta}^h)^2}(\mathbf{\Delta}^h)^T\mathbf{\Sigma}\mathbf{\Delta}^h,$$

$$a_1^h = \frac{2}{\boldsymbol{\mu}^T\mathbf{\Delta}^h}(\xi^h)^T\mathbf{\Sigma}\mathbf{\Delta}^h - \frac{2\boldsymbol{\mu}^T\xi^h}{(\boldsymbol{\mu}^T\mathbf{\Delta}^h)^2}(\mathbf{\Delta}^h)^T\mathbf{\Sigma}\Delta^h,$$

$$a_2^h = \frac{1}{(\boldsymbol{\mu}^T\mathbf{\Delta}^h)^2}(\mathbf{\Delta}^h)^T\mathbf{\Sigma}\mathbf{\Delta}^h.$$

In this way, hyperbolic segment h can be plotted as a function of the μ-values, $\mu \in [\mu^{h+1}, \mu^h]$, assumed over the segment.

Some comments are in order for when $\mathbf{\Sigma}$ is non-invertible. In this case, there can be degeneracy, but this is not the normal type of degeneracy from linear programming. Unfortunately, none of the more elegant anti-cycling devices from linear programming such as the lexicographic positive ordering process (as in Hadley (1962)) is applicable. Therefore, we simply keep track of all bases already visited in a binary tree to prevent any from being revisited.

6. Random problem generator

When an individual covariance matrix is needed for testing, it is often derived from historical data. However, if one were to need many covariance matrices of different sizes, ranks, and densities, where would all the universes of historical data come from? There is also the issue of a covariance matrix's diagonal and off-diagonal elements being distributed as in portfolio selection. The difficulty in obtaining varieties of "realistic" covariance matrices, we believe, goes a long way in explaining why there is so little reported in the way of computational experience on efficient frontiers in the literature.

So that CIOS can be used for testing, it is equipped with a built-in random problem generator. The random problem generator is patterned after Hirschberger et al. (2007). The following is a subset of the random problem generator's settings:

n	number of securities
r	rank of the covariance matrix
d	non-zero density of the covariance matrix
μ_v	mean of the distribution from which the diagonal covariance matrix elements are drawn
σ_v	standard deviation of the distribution from which the diagonal covariance matrix elements are drawn
μ_c	mean of the distribution from which the off-diagonal covariance matrix elements are drawn
σ_c	standard deviation of the distribution from which the off-diagonal covariance matrix elements are drawn
μ_e	mean of the distribution from which the security expected returns are drawn
σ_e	standard deviation of the distribution from which the security expected returns are drawn

Settings r, d allow the rank and density of the covariance matrix to be pre-chosen. Settings μ_v, σ_v and μ_c, σ_c allow for the separate controlling of the distributional characteristics of the diagonal and off-diagonal elements of the covariance matrix. Also, there are other settings such as for lower and upper bounds on the variables.

Chopra and Ziemba (1993) have pointed out the importance of expected returns in portfolio selection. While not implying that the random problem generator possesses any clairvoyance in this regard, μ_e, σ_e facilitate, at least on a first-order approximation basis, the generation of realistic test problems.

7. Computational results

To test the procedure of this paper over problems of differing sizes, covariance matrix densities, covariance matrix ranks, and covariance matrix element distributional characteristics, test problems were randomly generated using settings from the following intervals:

$$\begin{array}{lll} n \in [1000, 3000] & r \in [24, n] & d \in [\text{diagonal}, 100\%], \\ \mu_v \in [.010, .025] & \mu_c \in [.000, .025] & \mu_e \in [.06, .14], \\ \sigma_v \in [.0010, .0025] & \sigma_c \in [.0000, .0025] & \sigma_e \in [.02, .10], \end{array}$$

with **A** vacuous in (1.4), and all lower and upper bounds 0.00 and 0.04 in (1.5).

About run times and some other quantities, we have Table 3. It contains the results of experiments with three batches of large Markowitz problems (20 each) with 100% dense covariance matrices. To lend diversity to each batch, the problems were generated with ranks randomly chosen from the integers 24 to n. For example, in the table, 6.77 seconds is the average CIOS run time to compute all information about the hyperbolic segments that make up the efficient frontier of problems with 1000 securities.[5]

Table 3
A comparison of the results of three experiments of 20 problems each involving the computation of the entire efficient frontiers of 100% dense covariance matrix problems with ranks randomly selected from the integers in the interval $[24, n]$.

n	Percent down frontier	Cumulative hyperbolic segments	Securities in portfolios	Whole frontier CIOS time (seconds)	
				Ave	Stdev
1000	0	0.0	25.0	–	–
	25	48.8	65.3	–	–
	50	95.4	78.2	–	–
	75	145.6	86.2	–	–
	100	237.6	89.9	6.77	1.11
2000	0	0.0	25.0	–	–
	25	70.8	87.2	–	–
	50	139.7	114.8	–	–
	75	215.4	128.4	–	–
	100	335.2	128.7	37.37	14.20
3000	0	0.0	25.0	–	–
	25	93.0	111.0	–	–
	50	186.4	147.7	–	–
	75	283.6	171.0	–	–
	100	418.1	178.8	100.16	39.05

Utilizing the term "down the frontier" in the sense of expected return, column 3 shows the accumulation of hyperbolic segments as we move down the frontier. For instance, in problems with 2000 securities, 139.7 is the number of hyperbolic segments on average that have already been computed to get to 50% down the frontier. By comparing the numbers in the three parts of column 3, we can appreciate why run times increase with problem size. Not only does it take longer to compute a given hyperbolic segment the bigger the problem, but there are more hyperbolic segments to compute.

Column 4 is interesting because it shows the relatively modest number of securities at a positive level in portfolios along the efficient frontiers of problems with dense covariance matrices. For instance, in problems with 3000 securities, 171.0 is the number on average at 75% down the frontier.

Pursuing this issue a little further, we have the results of an experiment with 1500-security problems (36 in total) in Table 4. Holding covariance matrix rank constant at 1500, covariance matrix density is varied. What is seen is how the numbers of securities in the portfolios along the efficient frontier increase as density decreases.

Covariance matrix rank is also a factor. Consider the correlation matrix of Table 5 taken from the results of the 20 problems of the 2000-security experiment of Table 3. In the first column we see how rank affects the total number of hyperbolic segments, the number of securities at a positive level in portfolios (taken at 75% down the frontier), and CIOS run time to compute the whole efficient frontier. Thus, the greater the rank, the greater the number of hyperbolic segments, and hence the more work (run time) required to solve the problem.

8. Concluding remarks

Suppose a client says: "I have no interest in the equations that specify the efficient frontier. I am perfectly happy with dotted effi-

[5] To validate the accuracy of CIOS, the code was tested against Matlab and LINGO at numerous points along the efficient frontiers of 500-security problems and the results among the three solvers matched to six places to the right of the decimal point in all cases.

Table 4
Holding rank constant, this table shows how the number of securities in portfolios along the efficient frontier increases as covariance matrix density decreases.

n	Percent down frontier	Number of securities at a positive level					
		Covariance matrix density					
		100%	80%	60%	40%	20%	Diagonal
1500	0	25.0	25.0	25.0	25.0	25.0	25.0
	25	104.5	82.7	109.0	142.2	184.2	252.3
	50	143.8	156.7	245.8	348.3	476.2	746.7
	75	167.8	224.3	386.8	584.5	831.5	1387.5
	100	178.3	329.7	507.3	714.2	978.0	1500.0

Table 5
Correlation matrix resulting from the 2000-security experiment of Table 3.

	Rank	Segments	Securities	CIOS time
Rank	1			
Segments	0.695	1		
Securities	0.779	0.846	1	
CIOS time	0.705	0.990	0.796	1

cient frontiers and I do not wish to change." It would still be best to proceed parametrically as in this paper. This is because with knowledge of the equations, dots in virtually any pattern and number can be placed on the hyperbolic segments of the efficient frontier with only a small amount of post-processing effort. Although the integrations to compute the arc length of the efficient frontier can not be carried out in closed-form, they can be carried out numerically, thus enabling one to very accurately measure any point's arc length from either end of the curve. For example, with a routine written in something like Matlab (which is what we have used), the necessary calculations can be carried out in just a matter of a second or two. This can give a client great flexibility as one can experiment very inexpensively with different patterns and different numbers of points to find the most aesthetically pleasing representation for presentation without having to re-run the problem each time.

Although other experiments could have been conducted in Section 7, they will be left for a later time, but Tables 3–5 present the picture.

References

Arenas Parra, M., Bilbao Terol, A., Rodríguez Uría, M.V., 2001. A fuzzy goal programming approach to portfolio selection. European Journal of Operational Research 133 (2), 287–297.

Ballestero, E., Plà-Santamaría, D., 2003. Portfolio selection on the Madrid exchange: A compromise programming model. International Transactions in Operational Research 10 (1), 33–51.

Bana e Costa, C.A., Soares, J.O., 2004. A multicriteria model for portfolio management. European Journal of Finance 10 (3), 198–211.

Bank, B., Guddat, J., Klatte, D., Kummer, B., Tammer, K., 1983. Nonlinear Parametric Optimization. Birkhäuser Verlag, Basel.

Ben Abdelaziz, F., Aouni, B., El-Fayedh, R., 2007. Multi-objective stochastic programming for portfolio selection. European Journal of Operational Research 177 (3), 1811–1823.

Best, M.J., 1996. An algorithm for the solution of the parametric quadratic programming problem. In: Riedmüller, B., Fischer, H., Schäffler, S. (Eds.), Applied Mathematics and Parallel Computing: Festschrift for Klaus Ritter. Physica-Verlag, pp. 57–76.

Branke, J., Scheckenbach, B., Stein, M., Deb, K., Schmeck, H., 2009. Portfolio optimization and an envelope-based multi-objective evolutionary algorithm. European Journal of Operational Research 199 (3), 684–693.

Caballero, R., Cerdá, E., Muñoz, M.M., Rey, L., Stancu-Minasian, I.M., 2001. Efficient solution concepts and their relations in stochastic multiobjective programming. Journal of Optimization Theory and Applications 110 (1), 53–74.

Chang, T.-J., Meade, N., Beasley, J.E., Sharaiha, Y.M., 2000. Heuristics for cardinally constrained portfolio optimisation. Computers and Operations Research 27 (13), 1271–1302.

Chopra, V.K., Ziemba, W.T., 1993. The effect of errors in means, variances, and covariances on optimal portfolio choice. Journal of Portfolio Management, 6–11.

Eaves, B.C., 1971. On quadratic programming. Management Science 17 (11), 698–711.

Ehrgott, M., Waters, C., Kasimbeyli, R., Ustem, O., 2009. Multiobjective programming and multiattribute utility functions in portfolio optimization. INFOR, vol. 47, No. 1.

Elton, E.J., Gruber, M.J., Brown, S.J., Goetzmann, W.N., 2007. Modern Portfolio Theory and Investment Analysis, seventh ed. John Wiley, New York.

Fang, Y., Wang, S., 2005. Fuzzy Portfolio Optimization: Theory and Methods. Publisher of High Level Education, Beijing, China.

Guddat, J., 1974. Stability Investigations in Quadratic Parametric Programming. Doctoral Dissertation, Humboldt University, Berlin (in German).

Guddat, J., 1976. Stability in convex quadratic programming. Mathematische Operationsforschung und Statistik 7 (2), 223–245.

Hadley, G., 1962. Linear Programming. Addison-Wesley, Reading, Massachusetts.

Hallerbach, W.G., Hundack, C., Pouchkarev, I., Spronk, J., 2005. Market dynamics from the portfolio opportunity perspective: The DAX case. Zeitschrift für Betriebswirtschaft 75 (7–8), 739–764.

Hirschberger, M., Qi, Y., Steuer, R.E., 2007. Randomly generating portfolio-selection covariance matrices with specified distributional characteristics. European Journal of Operational Research 177 (3), 1610–1625.

Jobst, N.B., Horniman, M.D., Lucas, C.A., Mitra, G., 2001. Computational aspects of alternative portfolio selection models in the presence of discrete asset choice constraints. Quantitative Finance 1, 1–13.

Karmarkar, N., 1984. A new polynomial-time algorithm for linear programming. Combinatorica 4 (4), 373–395.

Konno, H., Yamazaki, H., 1991. Mean-absolute deviation portfolio optimization model and its application to Tokyo stock market. Management Science 37 (5), 519–531.

Korhonen, P., Yu, G.-Y., 1997. A reference direction approach to multiple objective quadratic-linear programming. European Journal of Operational Research 102 (3), 601–610.

Lin, C.-C., Liu, Y.-T., 2008. Genetic algorithms for portfolio selection problems with minimum transaction lots. European Journal of Operational Research 185 (1), 393–404.

Lo, A.W., Petrov, C., Wierzbicki, M., 2003. It's 11pm – Do you know where your liquidity is? The mean–variance–liquidity frontier. Journal of Investment Management 1 (1), 55–93.

Mansini, R., Ogryczak, W., Speranza, M.G., 2003. On LP solvable models for portfolio selection. Informatica 14 (1), 37–62.

Markowitz, H.M., 1952. Portfolio selection. Journal of Finance 7 (1), 77–91.

Markowitz, H.M., 1956. The optimization of a quadratic function subject to linear constraints. Naval Research Logistics Quarterly 3, 111–133.

Markowitz, H.M., Todd, G.P., 2000. Mean–Variance Analysis in Portfolio Choice and Capital Markets. Frank J. Fabozzi Associates, New Hope, Pennsylvania.

Michaud, R.O., 1989. The Markowitz optimization enigma: Is 'optimized' optimal? Financial Analysts Journal 45, 31–42.

Miettinen, K.M., 1999. Nonlinear Multiobjective Optimization. Kluwer, Boston.

Pardalos, P.M., Sandström, M., Zopounidis, C., 1994. On the use of optimization models for portfolio selection: A review and some computational results. Computational Economics 7 (4), 227–244.

Roy, A.D., 1952. Safety first and the holding of assets. Econometrica 20 (3), 431–449.

Ruszczyński, A., Vanderbei, R.J., 2003. Frontiers of stochastically nondominated portfolios. Econometrica 71 (4), 1287–1297.

Stein, M., Branke, J., Schmeck, H., 2007. Efficient implementation of an active set algorithm for large-scale portfolio selection. Computers and Operations Research 35 (12), 3945–3961.

Winston, W.L., 2004. Operations Research: Applications and Algorithms, fourth ed. Brooks Cole.

Zhang, W.-G., Zhang, X.-L., Xiao, W.-L., 2009. Portffolio selection under possibilistic mean–variance utility and a SMO algorithm. European Journal of Operational Research 197 (2), 693–700.