# Frontier: A graphical interface for portfolio optimization in a piecewise linear-quadratic risk framework

by D. L. Jensen
A. J. King

***"Frontier" is a pilot graphical user interface for portfolio optimization built for the new IBM workstation, the RISC System/6000™, out of basic X-windows and OSL utilities. The program asks the user to select a piecewise linear-quadratic risk measure, draws a risk/reward efficient frontier, and permits the user to examine the efficient frontier using zoom and histogram display facilities. This paper describes the interfaces and discusses possible extensions.***

Composing a portfolio of financial investments with the optimal characteristics of both risk and expected reward is the central task of modern portfolio theory, as advocated by Markowitz[1] and his successors. Modern portfolio theory guides the investor to quantify measures of risk and expected return for the appropriate set of assets, identify those combinations that are "efficient"—that is, those assets that meet all the requirements of the investor and provide the lowest risk for a desired level of expected return—and then select one portfolio from this "efficient frontier" to match the investor's tolerance for risk.

In principle, computing a minimum risk portfolio for a given level of expected return is a well-defined optimization problem. Over the years a body of literature has developed concerning the practical aspects of estimating asset return distributions from past data, using statistical methods and modeling investor requirements by means of mathematical programming techniques. However, the modeling of appropriate risk measurement criteria is critical to the success of this method.

The risk measure chosen by Markowitz was the variance of portfolio return. He chose the method for reasons of simplicity and computational efficiency. In the variance-as-risk framework, the investor constructs the variance-covariance matrix of asset returns (perhaps from historical data) and applies techniques of parametric quadratic programming to draw a mean/variance efficient frontier. This procedure is simple. It relies only on the estimation of the covariance matrix to specify risk, and leads to relatively low-dimensional quadratic programs for which effective algorithms exist. But variance-as-risk suffers from a serious drawback. It penalizes both high and low portfolio returns and fails to capture an investor's basic

preference for high versus lower portfolio returns.

A more appealing family of risk measures proposed by many investigators is based on the notion of "downside risk." These are constructed out of pieces: a linear or quadratic piece to penalize below-target portfolio returns, and a piece with zero slope to indicate indifference to above-target portfolio returns. Such a risk measure applies a penalty only when returns fall below a certain target. An efficient "frontier" can be produced by computing a minimum risk portfolio for each level of target expected return, just as for the mean/variance efficient frontier.

The chief drawback of this procedure lies in the difficulty of computing the minimum risk portfolio for the piecewise linear-quadratic risk measures used to model downside risk. The authors[2] outlined a method to compute the minimum risk portfolio for downside risk models, which has the same simplicity as the variance-as-risk model but is far more computationally intensive. This procedure takes the same empirical data used to construct the variance-covariance matrix and employs it directly as an empirical (nonparametric) estimate of the distribution of asset returns. For each datum a set of new variables is introduced to represent each piece of the risk objective, and the whole problem is then solved as a large-scale quadratic program.

Justification for this nonparametric downside risk framework comes from two sources. The first is theoretical. In Reference 2 it is shown that the two procedures have similar error distributions. This means that one is not able to control errors more effectively by separately estimating the variance-covariance matrix than by employing the nonparametric procedure. The second is practical. Using ten years of global stock market data, W. V. Harlow[3] tabulates an impressive series of results showing the year-by-year superiority of the nonparametric downside risk framework.

The computational requirements of these piecewise linear-quadratic risk models are greater than those imposed by variance models. Nevertheless, they are well within the capabilities of modern RISC (Reduced Instruction-Set Computer) hardware and advanced optimization software. In Reference 2 we reported reasonable times to draw an efficient frontier for a 1000 asset portfolio with 1000 data points (one million observations in all), on an IBM RISC System/6000* Model 540 using the quadratic programming solver that is standard in the IBM Optimization Subroutine Library (OSL).[4]

This paper describes a pilot user interface for portfolio optimization called "Frontier." Frontier has been successfully demonstrated to audiences in Europe, the United States, and Japan. It is intended to illustrate the capabilities of advanced workstation environments and the usefulness of X-windows and OSL utilities in custom-designing a portfolio manager's interface with state-of-the-art optimization software. The source code for Frontier is available as a sample OSL/X-windows driver.

This paper is divided into five parts. The first part describes the mathematical model that must be passed to OSL; the second discusses the particular application that is available with the sample code for Frontier. In the third and fourth parts the implementation of the Frontier program is described in detail. Finally, since Frontier is just a pilot program and as such is deficient in many respects, in the last section we discuss possible extensions to the program to make it a more complete solution.

## The mathematical model

If all asset returns are known at the outset, i.e., there is no risk, then the portfolio optimization problem is to maximize the total portfolio return subject to a budget constraint. Additional constraints may be added to reflect investment policies, such as distribution requirements. The mathematical statement of the risk-free portfolio optimization problem is

$$\begin{aligned} \text{maximize:} \quad & r^{\top}x = \sum_{j=1}^{n} r_j x_j \\ \text{subject to:} \quad & \ell r \le Ax \le ur \\ & \ell c \le x \le uc \end{aligned} \tag{1}$$

The vector $r$ represents the asset returns for each of the $j = 1, \cdots, n$ securities. The inequalities are vector inequalities, interpreted componentwise. The vectors $\ell r$ and $ur$ are the row upper and lower bounds; $\ell c$ and $uc$ are the column upper and lower bounds. The problem shown in Equation 1 is a *linear program* that can be solved by the Op-

timization Subroutine Library. But if the asset returns are not known, a mathematical model must be formed that accounts for the risk of investing in securities with uncertain returns.

The nonparametric linear-quadratic risk framework presented in Reference 2 is as follows. First, a list is obtained of possible asset return *scenarios* $r_s$ together with a probability weighting $p_s$, for each $s = 1, \cdots, S$. (Such scenarios may come from historical data or simulations.) Next, a *risk measure* is designed from the class of linear-quadratic tracking functions by selecting a *lower slope* $q^-$, an *upper slope* $q^+$, and a *curvature* $e$. This produces a risk function of the form:

$$\rho_{q^-,q^+;e}(t) = \begin{cases} q^- t - \frac{1}{2e}(q^-)^2 & \text{if } t \le \frac{q^-}{e} \\ \frac{1}{2} et^2 & \text{if } \frac{q^-}{e} \le t \le \frac{q^+}{e} \\ q^+ t - \frac{1}{2e}(q^+)^2 & \text{if } \frac{q^+}{e} \le t \end{cases}$$

Examples of risk measures for particular parameter settings follow shortly. Finally, a *target* is selected. In the examples in this paper, the target is the level of *expected return* (other targets are possible, as shown in Reference 5). Putting all this together, a mathematical model is designed that minimizes the expected risk of deviations about a target expected return, subject to the original constraints of the model shown in Equation 1. For convenience we denote by $\bar{r}$ the expected return vector

$$\bar{r} = \sum_{s=1}^{S} p_s r_s$$

and by $X$ the set of vectors satisfying the constraints of Equation 1, namely

$$X = \{x | \ell r \le Ax \le ur,\ \ell c \le x \le uc\}$$

With this notation, the mathematical program to compute the efficient frontier describing the minimum risk for given level $\tau$ of expected return is

$$\begin{aligned} \text{minimize:} \quad & \sum_{s=1}^{S} p_s \rho_{q^-,q^+;e}(r_s^\top x - \bar{r}^\top x) \\ \text{subject to:} \quad & \bar{r}^\top x \ge \tau \\ & x \in X \end{aligned} \tag{2}$$

Examples of risk measures that can be constructed from the piecewise linear-quadratic tracking function are the theoretically significant and intuitively appealing *downside risk* measures: the *lower partial mean* and *lower partial variance*. These risk measures apply a positive penalty to all values coming in below zero, and are zero for all positive values. The lower partial mean is constructed by setting $q^- = 1$, $q^+ = 0$, $e = +\infty$:

$$\rho_{1,0;+\infty}(t) = \begin{cases} |t| & \text{if } t < 0 \\ 0 & \text{otherwise} \end{cases}$$

and the lower partial variance by $q^- = -\infty$, $q^+ = 0$, $e = 1$:

$$\rho_{-\infty,0;1}(t) = \begin{cases} \frac{1}{2} t^2 & \text{if } t < 0 \\ 0 & \text{otherwise} \end{cases}$$

(As far as the program Frontier is concerned, any real number with absolute value in excess of $1.0 \times 10^{31}$ is infinite.)

The data requirements of the mathematical model may be summarized as: the number $n$ of securities in the portfolio universe and the number and type of constraints describing the feasible set $X$, the list of asset return scenarios $r_s$, $s = 1, \cdots, S$, together with their probabilities $p_s$, and the shape parameters $(q^-, q^+, e)$ for the piecewise linear-quadratic risk measure.

## Example applications

The program Frontier has an interactive facility for defining the risk measure. In its present design, however, Frontier must generate the remainder of the required data from files and by simulation. This section describes the data structures and discusses two example applications.

The data read by Frontier are in two parts, (1) the constraints that define the portfolio universe $X$, and (2) the parameters describing the probability distribution of the asset returns. To understand how this information is structured, it helps to follow the simple example called ASSET distributed with Frontier. In this model the investor wishes to choose a portfolio consisting of long positions in each of four assets: Corporate bonds ($C$), Nikkei stock index ($N$), Standard and Poors 500 stock index ($S$), and U.S. Treasury bills ($T$). There are nonnegativity restrictions on each asset holding, plus three other constraints describing budget limitations and diversity requirements:

$$\begin{aligned} x_C + x_N \quad + x_S + x_T &= 1 \qquad \text{(Budget)} \\ x_N - 0.5x_S \quad &\leq 0 \qquad \text{(Diversity 1)} \\ x_C \qquad\quad + x_T &\geq 0.3 \qquad \text{(Diversity 2)} \end{aligned}$$

$$x_C \geq 0,\ x_N \geq 0,\ x_S \geq 0,\ x_T \geq 0$$

(long positions only) (3)

The asset returns ($r$) depend on two factors, World GNP ($s_W$) and U.S. ten-year Treasury rates ($s_U$), plus an independent component for each asset:

$$\begin{aligned} r_C &= 1.05s_U - 0.20s_W + t_C + \bar{r}_C \\ r_N &= -0.10s_U - 10.50s_W + t_N + \bar{r}_N \\ r_S &= -0.20s_U - 8.35s_W + t_S + \bar{r}_S \\ r_T &= 1.00s_U - 0.005s_W + t_T + \bar{r}_T \end{aligned} \qquad (4)$$

The expected return vector is $\bar{r}$. The factors $(s_U, s_W)$ are distributed approximately log-normal in the following manner:

$$\begin{aligned} s_U &= 1 - e^{v_U} \\ s_W &= e^{v_W} - 1.0 \end{aligned}$$

where $(v_U, v_W)$ are jointly normal random variables with joint covariance matrix

$$C = \begin{bmatrix} 0.003 & 0.000005 \\ 0.000005 & 0.00025 \end{bmatrix} \qquad (5)$$

and mean value 0. The independent components are also approximately log-normal:

$$t_j = e^{v_j} - 1.0 \qquad j \in \{C, N, S, T\}$$

where the $v_j$ are independent normal random variables with mean zero and standard deviations.

$$d_C = 0.025,\ d_N = 0.04,\ d_S = 0.04,\ d_T = 0.0 \qquad (6)$$

Finally, we recall that in the minimal risk model (Equation 2), we must have a constraint bounding the expected returns from below:

$$1.07x_C + 1.31x_N + 1.18x_S + 1.03x_T \geq 0 \qquad (7)$$

The data needed to define the model are:

1. The expected return vector $\bar{r}$, as in Equation 7
2. The factor covariance matrix $C$ and standard deviations $d$, as in Equations 5 and 6
3. The factor matrix $F$ as in Equation 4
4. The linear system of constraints and bounds: $\ell r \leq Ax \leq ur$ and $\ell c \leq x \leq uc$, as in Equation 3.

We now perform a simple algebraic manipulation to simplify the data structures (and confuse the innocent). Notice that the return on a portfolio $x = (x_C, x_N, x_S, x_T)$ may be computed as

$$\begin{aligned} (\bar{r} + r)^\top x &= (\bar{r} + Fs + t)^\top x \\ &= \bar{r}^\top x + s^\top F^\top x + t^\top x \end{aligned}$$

We define now two additional "assets" in our portfolio, namely $(y_U, y_W)$, corresponding to "positions" we will take in the factors as a consequence of our positions in the actual assets. These "factor positions" are given by the system of equations

$$\begin{aligned} y_U &= 1.05x_C - 0.10x_N - 0.20x_S + 1.00x_T \\ y_W &= 0.20x_C - 10.50x_N - 8.35x_S + 0.00x_T \end{aligned} \qquad (8)$$

or, in symbols

$$y = F^\top x$$

If we include the system of Equation 8 in our linear system of Equation 3, we may compute the return on a portfolio $x$ as

$$(\bar{r} + r)^\top x = \bar{r}^\top x + s^\top F^\top x + t^\top x = \bar{r}^\top x + s^\top y + t^\top x$$

This allows us to enter the matrix $F$ as part of the constraints of the model, as we see below in Equation 9.

In summary, the data required by Frontier are in two parts. The first describes the set of linear equalities, inequalities and bounds from the systems shown in Equations 7, 8, and 3:

$$\begin{aligned} \bar{r}^{\top}x &\geq 0 \quad \text{(expected returns)} \\ -y + F^{\top}x &= 0 \quad \text{(factor equations)} \\ \ell r \leq Ax &\leq ur \quad \text{(portfolio universe)} \\ \ell c \leq x &\leq uc \end{aligned} \tag{9}$$

The second describes the covariances and standard deviations of the normal random vector $v$ that is used to construct the log-normal random factor $s$ and independent components $t$, namely

$$\begin{bmatrix} C \\ d \end{bmatrix} \tag{10}$$

The system depicted by Equation 9 is passed in to Frontier in standard MPS[4] (Mathematical Programming System) format (see the file ASSET.DTX in Appendix A). The first row is the expected return row, the next bloc of rows are the factor rows, followed by the portfolio constraints. The covariance data (Equation 10) are read in from another file by columns. The first entry of this file is an integer NF stating the number of factors in the model (see the file ASSET.COVDAT in Appendix A).

The problem ASSET is a simple model that is included in the Frontier package merely as an example for the user to explore. Also included is a much larger problem (of a similar type), called EX3B, with a portfolio universe of 1000 assets and a factor model involving 15 factors. Solution times are reported in Reference 2. They range from about one minute for a small sample of 50, to over one hour for a large sample of 1000. It should be remarked that a sample of 1000 corresponds to over three years of daily closing data, and 85 years of monthly closing data! Typical asset allocation problem sizes would involve far fewer stocks, although the constraints might be more complex (and a realistic problem formulation would incorporate minimum purchase conditions, requiring a mixed-integer problem formulation).

## Computing the efficient frontier with OSL

The fundamental computation is to calculate and draw the efficient frontier for the problem shown in Equation 2, whether the probability model is a factor model as described in the previous section, or the probability model is generated from historical data.

Frontier does this by employing OSL subroutines to generate the equivalent quadratic program

$$\text{minimize:} \quad \sum_{s=1}^{S} p_s \left[ \frac{1}{2} e v_s^2 + q^{-} v_s^{-} + q^{+} v_s^{+} \right] \tag{11}$$

$$\text{subject to:} \quad \bar{r}^{\top}x + \hat{R} - R = 0 \tag{12}$$

$$x \in X \tag{13}$$

$$R \geq 0 + \tau \tag{14}$$

$$r_s^{\top}x - R - v_s - v_s^{-} - v_s^{+} = 0 \tag{15}$$

$$\sum_{s=1}^{S} r_s^{\top}x - \hat{R} = 0 \tag{16}$$

$$v_s^{-} \leq 0,\ v_s^{+} \geq 0,\ v_s \text{ free} \qquad (s = 1, \cdots, S) \tag{17}$$

Equations 12 and 16 serve to set the variable $R$ to the true expected value (because $r_s$ is a sample from a random return with mean zero, it may be that $\Sigma_s\, p_s\, r_s$ is not zero). Equation 14 is therefore the lower bound row for the expected returns. Equation 15 allocates the difference $r_s^{\top}\, x - R$ to the three pieces of the piecewise linear quadratic tracking function. The variable $v_s^{-}$ picks up the negative linear part, the variable $v_s^{+}$ the positive linear part, and the variable $v_s$ (as a consequence of the optimization) will be assigned to the part over which the tracking function is quadratic.

Frontier assumes that the first row of the model is the "idealized" expected return row, and that the true expected return is the sum of the idealized and the sample expected return. Frontier builds the model as follows:

1. Use OSL subroutines EKKCOL to add variables $\hat{R}$ and $R$ to the first row (Equation 12).
2. Use OSL subroutine EKKROW to add the expected return bounding row (Equation 14). Initially, the lower bound on this row is set to 0.
3. Call subroutine SAMPLE a total of $S$ times to read in the samples $r_s$ and construct, one row at a time, a matrix block for the Equation 15.
4. Call OSL subroutine EKKDSCB to add the block (Equation 15) to the model. Set the linear objective values $p_s q^-$ and $p_s q^+$ and the bounds for the variables: $v^-$, $v$, and $v^+$.
5. Call OSL subroutine EKKROW to add the sample expected return row in Equation 16.
6. Call OSL subroutine EKKQMDL to add the quadratic penalty (the diagonal matrix with entries $p_s e$) for each quadratic variable $v_s$.

Frontier next sets up the parametric perturbation in preparation for a call to EKKQPAR, the parametric quadratic programming subroutine of OSL that draws efficient frontiers. In the case at hand, the quadratic model is being "perturbed" by adding $\tau$ times the perturbation vector $[0, \cdots, 0, 1, 0, \cdots, 0]$ to the row bounds, where the 1 lies in the position corresponding to Equation 14. EKKQPAR is an efficient parametric programming routine designed for the specific task of computing an efficient frontier with either cost or row-bound perturbations. There are two phases. First, EKKQPAR solves the quadratic program with $\tau$ set to its lower bound—in our case the lower bound is zero, corresponding to the minimum risk problem. Then, repeatedly, at the given complementary basic solution EKKQPAR increases $\tau$ up to the point where one of the variables (primal or dual) would violate its bound if $\tau$ were to increase any further. This variable is then pivoted out of the basis. The procedure repeats until $\tau$ has reached its maximum size, which corresponds to the maximum expected return problem.

If the objective value ROBJVAL were captured and solution NCOLSOL returned by OSL at each $\tau$ visited in this procedure, the user would be able to trace and store the efficient frontier and the efficient portfolios. Fortunately, OSL facilities allow the user to follow the progress of the algorithm and examine OSL variables while the program is executing via a user-exit subroutine EKKITRU. Each time EKKQPAR finds a point $\tau$ on the efficient frontier (after making the pivot), it calls EKKITRU with mode 9. The EKKITRU subroutine tests the mode and writes each $\tau$-value, objective value, and optimal portfolio out to a binary file, ready to be accessed by the graphics subroutines that display the efficient frontier.

### Frontier user interface

Frontier is initialized (in X-windows) by the command string

run *pname* *nsamp*

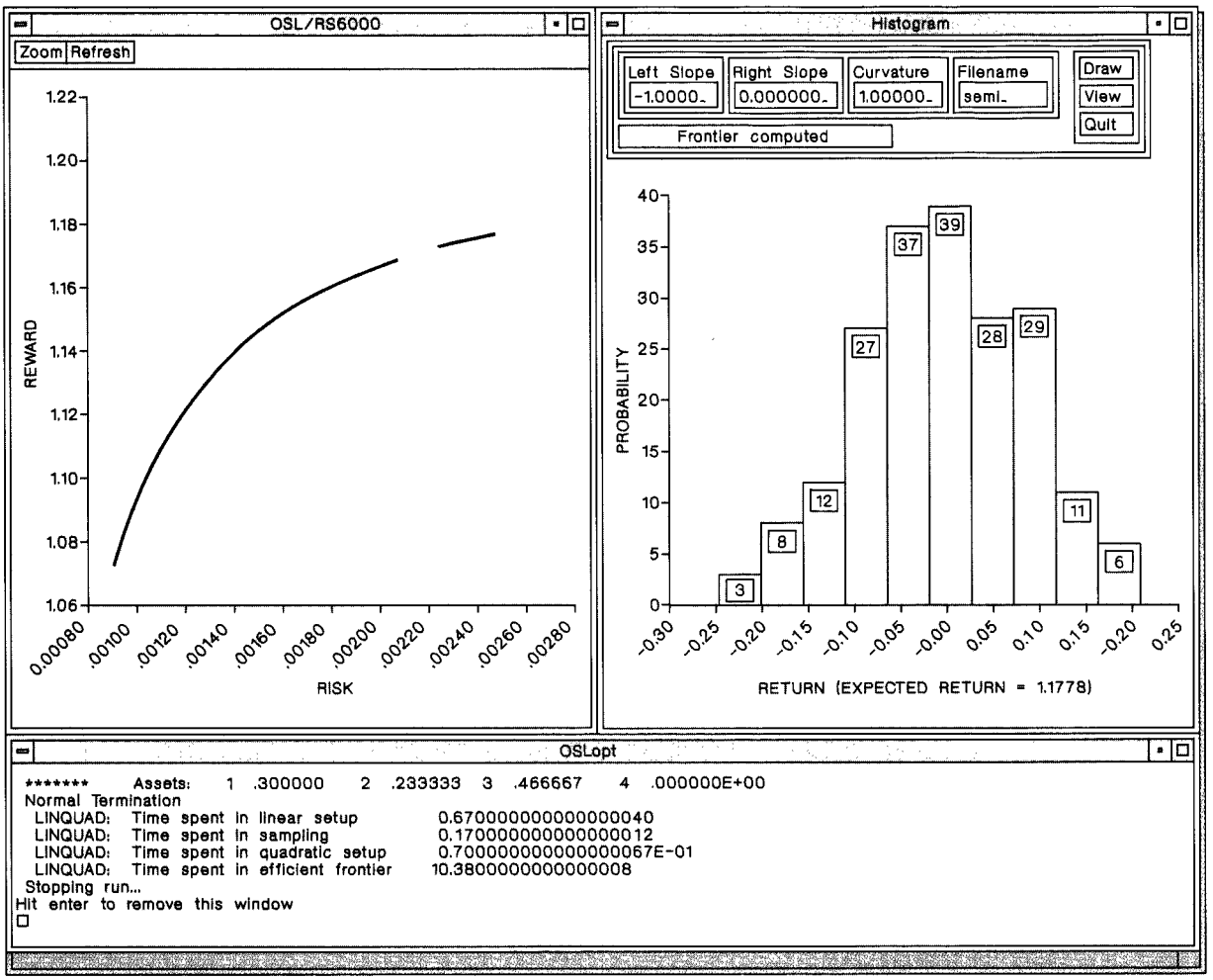
where *pname* is a problem name and *nsamp* is an integer specifying the number of samples to be taken (the size of $S$). The two files *pname*.dtx and *pname*.covdat are found and then the program Frontier is executed, displaying the user interface windows.

The user interface for Frontier is designed to allow the user to specify the tracking function parameters: left slope $q^-$, right slope $q^+$, and curvature $e$. Since the efficient frontier will be written to a file, the user also is asked to provide an identifier for the particular frontier to be drawn. Once these items are keyed into the appropriate boxes, the frontier can be drawn by selecting the appropriate box and clicking a mouse button. If all goes well, the OSL output scrolls by and the efficient frontier grows in leaps and bounds. (See Figure 1.)

The interface is designed so that the user may examine the efficient frontier. Clicking the mouse on the frontier will display a histogram of simulated total returns from the corresponding efficient portfolio, generated by locating the appropriate record in the frontier file, reading in the optimal portfolio $x$, computing the total return $v_s = r_s^\top x$ for each sample $r_s$ of asset returns, and then displaying the distribution of $v_s$, $s = 1, \cdots, S$ as a histogram. (The histogram is centered at 0; to get the true value one must add the expected return values.) Clicking up and down the frontier gives a graphic demonstration of the risk vs reward concept; the histogram changes shape dramatically for different levels of reward.

The user may also "zoom" on the frontier to examine it more closely. By pressing on the zoom button, the mouse pointer is set to outline a zoom rectangle: the first click locates the upper-left corner and the second the lower-right corner. The zoom rectangle magnifies to the whole window;

**Figure 1 Frontier output—the efficient frontier with return histogram for highlighted portfolio (OSL output at bottom)**




the original frontier may be restored by clicking the "refresh" button. Efficient frontiers already drawn may be recalled by entering the frontier name and clicking the "view" button. In this way, efficient frontiers for various settings of risk parameters may be visually compared.

## Conclusions and extensions

Frontier is a pilot program, created to demonstrate the capabilities of OSL and X-windows in a realistic application in financial portfolio management. It is also a state-of-the-art implementation of the piecewise linear-quadratic risk framework—the user can select a risk function, draw a frontier, and examine the results of investing in any one of the efficient portfolios. Drawing the frontier gives a compelling graphical demonstration of the power of OSL and the IBM RISC System/6000. A wealth of output documents the solution of complex problems, and the efficient frontier leaps across the screen.

More capabilities could be built into Frontier and the efficient frontier more quickly drawn. Our suggestions fall into three groups: (1) the graphical user interface, (2) the data interface, and (3) the computational kernel.

Possible extensions to Frontier would allow the user to view the efficient portfolio as a table of entries, edit the portfolio table, and simulate the

result on another histogram window for comparison. For example, the user may wish to delete from the efficient portfolio all those stocks represented at levels below 0.1 percent. Also, the user should be able to edit the list of drawn frontiers and select any one of them for comparative viewing in a separate frontier/histogram window. These are relatively simple tasks that can be implemented with standard X-toolkit (or Motif) text widgets.

The data interface could also be extended to permit a wide variety of user inputs and interfaces. This can be a challenging task. It should not be difficult to integrate the user's database with the Frontier program. But what if the user wants to enter in a constraint *ex post*? One could design an interface such as those in the above paragraph to display and edit a table of column or row bounds (say, only those pertaining to the portfolio universe $X$), but that is about all one can do without a more general modeling language or data interfaces than OSL presently provides. A further, easy-to-implement, extension would be a facility to allow the user to generate an entirely new set of sample returns with which to simulate the total portfolio return and test the robustness of the efficient frontier with respect to different sample sets.

The computational kernel can be modified in a number of ways. As we indicated in Reference 2, by far the greatest time is spent in finding the first point on the efficient frontier. The issue here is to find a good starting basis. One possibility is to find the point at the other (maximum expected return) end of the frontier. This may be faster. Another is to apply some decomposition techniques more suited to the problem structure—such as the finite generation method as reported in King.[6]

A more important innovation in a completely different direction is to allow the user to set minimum purchase levels, a level below which the asset holding is constrained to be zero. This can be accomplished with zero-one variables, as follows. Suppose that one wishes to hold an amount greater than $y$ or none of a certain stock $x_j$. Introduce two variables $y_j$ and $z_j$, and add the following set of constraints

$$x_j = a_j z_j + y_j$$

$$0 \le x_j \le M z_j$$

$$y_j \ge 0, \text{ and } z_j = 0 \text{ or } 1$$

where $M$ is an overestimate of an upper bound for the variable $x_j$. When $x_j$ is not zero, the zero-one variable is forced to take the value one and this in turn forces $x_j$ to be at least as large as $a_j$. This problem now falls in the (largely unexplored) realm of mixed-integer quadratic programming. OSL provides basic facilities for creating branch and bound algorithms for solving this class of problems.

*Trademark or registered trademark of International Business Machines Corporation.

## Appendix A

### Example: ASSET.DTX

```
NAME          ASSETS
ROWS
 N  OBJRW
 E  REWARD
 E  USLONG
 E  WLDGNP
 E  BUDGET
 L  DIVERS1
 G  DIVERS2
COLUMNS
    USLONG    USLONG     -1.00
    WLDGNP    WLDGNP     -1.00
    CORPB     USLONG      1.05  WLDGNP    -.20
    CORPB     BUDGET      1.00  DIVERS2     1.00
    CORPB     REWARD      1.07
    NIKKEI    USLONG      -.10  WLDGNP   10.50
    NIKKEI    BUDGET      1.00  DIVERS1     1.00
    NIKKEI    REWARD      1.31
    SP500     USLONG      -.20  WLDGNP    8.35
    SP500     BUDGET      1.00  DIVERS1     -.50
    SP500     REWARD      1.18
    TBILLS    USLONG      1.00  WLDGNP   0.005
    TBILLS    BUDGET      1.00  DIVERS2     1.00
    TBILLS    REWARD      1.03
RHS
    RHS       USLONG      0.00  WLDGNP    0.00
    RHS       BUDGET      1.00  DIVERS1   0.00
    RHS       DIVERS2     0.30  REWARD    0.00
BOUNDS
 FR BND       USLONG
 FR BND       WLDGNP
 LO BND       SP500       0.0
 LO BND       NIKKEI      0.0
 LO BND       TBILLS      0.0
 LO BND       CORPB       0.0
ENDATA
```

### Example: ASSET.COVDAT

```
2
0.003
0.000005
0.000005
0.00025
0.004
0.004
0.001
0.0025
```

## Cited references


1. H. M. Markowitz, *Portfolio Selection = Efficient Diversification of Investments*, John Wiley & Sons, Inc., New York (1959).
2. A. J. King and D. L. Jensen, *Linear-Quadratic Efficient Frontiers for Portfolio Optimization*, Research Report RC-16524, IBM Thomas J. Watson Research Center, Yorktown Heights, NY 10598 (1991).
3. W. V. Harlow, *Asset Allocation in a Downside Risk Framework*, Salomon Brothers, New York (1991).
4. *Optimization Subroutine Library Guide and Reference*, SC23-0519-2, IBM Corporation (1991); available through IBM branch offices.
5. R. S. Dembo and A. J. King, *Tracking Models and the Optimal Regret Distribution in Portfolio Optimization*, Research Report, IBM Thomas J. Watson Research Center, Yorktown Heights, NY 10598 (1991).
6. A. J. King, "An Implementation of the Lagrangian Finite Generation Method," in *Numerical Technique for Stochastic Optimization*, Yu. Ermoliev and R. J-B Wets, Editors, Springer-Verlag, New York (1988).



*Accepted for publication September 19, 1991.*



**David L. Jensen** *IBM Research Division, Thomas J. Watson Research Center, P.O. Box 218, Yorktown Heights, New York 10598.* Dr. Jensen received his Ph.D. in operations research from Cornell University in 1985. He joined IBM's Thomas J. Watson Research Center in 1987. His interests are in network flows, combinatorial optimization, interior point algorithms, and applications of mathematical programming.

**Alan J. King** *IBM Research Division, Thomas J. Watson Research Center, P.O. Box 218, Yorktown Heights, New York 10598.* Dr. King earned his Ph.D. at the University of Washington, Seattle, in 1986, and joined IBM's Thomas J. Watson Research Center in 1988. His interests are in stochastic programming and its application to decision-making under uncertainty for engineering and economic systems.



Reprint Order No. G321-5461.