GE Asset Management, Genworth Financial, and GE Insurance Use a Sequential-Linear-Programming Algorithm to Optimize Portfolios

Kete Charles Chalermkraivuth, Srinivas Bollapragada, Michael C. Clark, John Deaton, Lynn Kiaer, John P. Murdzek, Walter Neeves, Bernhard J. Scholz, David Toledano
General Electric Global Research Center, 1 Research Circle, Schenectady, New York 12309
(kete.chalermkraivuth@research.ge.com, bollapragada@research.ge.com, clark@crd.ge.com, john.deaton@genworth.com, kiaer@crd.ge.com, john.murdzek@genworth.com, walter.neeves@genworth.com, scholz@research.ge.com, toledano@research.ge.com)

GE Asset Management Incorporated (GEAM), a wholly owned subsidiary of General Electric Company (GE), manages investment portfolios on behalf of various GE units and over 200 unaffiliated clients worldwide, including Genworth Financial (Genworth) and GE Insurance (GEI) portfolios worth billions of dollars. GEAM invests portfolios of assets—derived from cash flows for various insurance, reinsurance, and financial products—primarily in corporate and government bonds in accordance with risk and regulatory constraints. In asset-liability management (ALM) applications, portfolio managers try to maximize return or minimize risk and match the characteristics of asset portfolios with corresponding liabilities. While risk is widely represented by variance or volatility, it is usually a nonlinear measure; ALM portfolio managers traditionally need to use linear-risk sensitivities for computational tractability. We developed a novel, sequential-linear-programming algorithm that handles the nonlinearity iteratively but efficiently. Patented and implemented on a limited basis since 2003, GE used it to optimize more than 30 portfolios valued at over $30 billion. It is now in broader use at GEAM, GEI, and Genworth. Hypothetically, based on $100 billion of assets, the present value of potential benefits could approximate $75 million over five years.

Key words: finance: portfolio; programming: linear, sequential; finance: asset liability management.

Genworth Financial (Genworth) and GE Insurance (GEI), insurance units of General Electric Company (GE), own several insurance businesses that offer a range of insurance, reinsurance, and financial products for consumer and commercial customers. (GE transferred most of the businesses of GE Financial and GE Mortgage Insurance to a new company called Genworth Financial. Genworth’s initial public offering took place May 25, 2004.) The assets supporting GE’s insurance businesses come mainly from insurance premiums and contract payments. GE Asset Management (GEAM), a portfolio manager of Genworth and GEI, manages the portfolios of assets produced by various insurance businesses.

Portfolio optimization is a key component in portfolio management. The objective is to identify risk/return trade-offs of the portfolio by maximizing the return or minimizing the risk. In the insurance industry, portfolio managers must choose the assets within a portfolio so their characteristics match those of the liabilities to protect the value of surplus over changes in underlying interest rates.

Traditionally, in asset-liability management (ALM), portfolio managers manage their portfolios using an immunization strategy, which controls the mismatches between asset and liability risk sensitivities using duration and convexity measures. Portfolio managers solve linear programs to determine asset allocations for the portfolio. However, they may not minimize the total risk; for a given level of risk, a higher return may exist, and the solution may be suboptimal. For portfolios of assets in the tens of billions
of dollars, even a small increase in return for a given level of risk would add several tens of millions of dollars to the bottom line.

In 2002, GE funded a project to develop algorithms and tools to quantify the diverse risks of asset portfolios backing insurance liabilities and to identify optimal risk/return trade-offs in the asset-allocation strategy for their insurance portfolios. A multidisciplinary team from GE Global Research Center worked closely with GEAM, Genworth, and GE to understand their requirements. This team included portfolio managers, actuaries, financial analysts, statisticians, information technologists, and operations researchers. They determined that solving the ALM portfolio management problem to optimality would result in a material difference in financial performance and in a better understanding of the trade-off between risk and return. They also developed an efficient approach to a class of ALM problems that was previously viewed as computationally intractable.

The team formulated the ALM portfolio optimization model using the variance of economic surplus as a measure of total risk. The economic surplus is the difference between the market values of assets and liabilities. This model is similar to the well-known Markowitz (1952) model but with added constraints to match the duration and convexity of the assets and liabilities within an acceptable tolerance. The surplus variance is calculated based on a multifactor risk framework rather than a cross product of variances of individual asset returns and their correlations.

Most GEAM portfolios consist of a large number of securities. Solving this optimization model using commercially available solvers is not practical. We therefore developed a novel algorithm based on sequential linear programming (SLP) (Bazaraa et al. 1993) that quickly generates the risk/return efficient frontier.

Our patented algorithm constructs the efficient frontier between risk and return, starting with the portfolio that generates the highest possible return without regard to the risk level. We obtain this portfolio by eliminating the nonlinear risk constraint from the ALM model and solving the resulting linear program (LP). In the next step, we generate a portfolio with a marginally smaller return and a marginally smaller risk value. We do this by approximating the risk function at the previously obtained solution. We then add a linear constraint to the LP relaxation, which restricts this linear approximation to the risk function to be below a new target risk value. The solution to the resulting LP generates the next portfolio along the efficient frontier. We repeat this process to approximate the entire efficient frontier one step at a time. The SLP algorithm is extremely fast, because it requires solving just one LP for every point on the efficient frontier.

We implemented the algorithm in the MATLAB software environment. We coded the entire SLP algorithm in the MATLAB programming language, using the MATLAB linear-program solver to solve the LP relaxation at each step in the SLP algorithm. The algorithm generates the efficient frontier for portfolios consisting of several thousand assets within minutes. The portfolio managers and analysts at GEAM, Genworth, and GEI use the algorithm from a Web-based user interface.

**Portfolio Optimization Models**

Based on the modern portfolio theory proposed by Markowitz (1952), the goal of portfolio optimization is to manage risk through diversification and obtain an optimal risk-return trade-off. Risk measures play a crucial role in portfolio optimization. To characterize the investor’s risk objectives and capture the potential risk-return trade-offs, portfolio managers use risk measures to quantify various aspects of portfolio risk. Markowitz’s mean-variance portfolio optimization model uses variance (or standard deviation) of returns as a measure of risk. In this framework, the portfolio variance is a cross product of variances of individual asset returns and their correlations.

For ALM applications, we use surplus variance as a measure of risk. We compute portfolio variance using an analytical method based on a multifactor risk framework (Fong and Vasicek 1997, Hull 2000). In this framework, we can characterize the value of a security as a function of multiple underlying risk factors and can approximate the change in the value of a security with the changes in the risk-factor values and risk sensitivities to these risk factors. We can derive the portfolio variance equation analytically from the underlying value-change function (Appendix).
In ALM applications, the portfolios have assets and liabilities that are affected by the changes in common risk factors. Because a majority of the assets are fixed-income securities, the dominant risk factors are interest rates. In ALM applications, in addition to maximizing return or minimizing risk, portfolio managers are constrained to match the characteristics of asset portfolios with those of the corresponding liabilities to preserve portfolio surpluses caused by interest rate changes. Therefore, the formulation of the ALM portfolio optimization problem has linear constraints that match the asset-liability characteristics in addition to those in the traditional Markowitz model. We use the following ALM portfolio optimization formulation (PF1):

Maximize

\[
\text{portfolio expected return}
\]

subject to

\[
\text{surplus variance} \leq \text{target}_1,
\]

\[
\text{duration mismatch} \leq \text{target}_2,
\]

\[
\text{convexity mismatch} \leq \text{target}_3,
\]

and

\[
\text{linear portfolio investment constraints}.
\]

The duration and convexity mismatches are the absolute values of the differences between the effective durations and convexities of the assets and liabilities in the portfolio, respectively (Appendix). Although they are nonlinear (because of the absolute value function), the constraints can easily be made linear by replacing each of them with two new constraints that ensure that the actual value of the mismatch is less than the target mismatch and that it is greater than the negative of the target mismatch, respectively. The other portfolio investment constraints include asset-sourcing constraints that impose a maximum limit on each asset class or security, overall portfolio credit quality, and other linear constraints.

The multifactor-based ALM portfolio-optimization model is similar to Markowitz’s portfolio-optimization model, but it has additional constraints to match the durations and convexities of the assets and liabilities. In addition, the surplus variance is calculated based on a multifactor framework and not on a cross product of variances of individual asset returns and their correlations. Because the risk measure is in a quadratic form, one needs to use quadratic or nonlinear-programming solvers to solve the formulation. Most real-world ALM problems consist of thousands of securities to choose from for investment. In this situation, solving an ALM optimization problem is beyond the computational limits of quadratic and nonlinear-programming solvers.

Because of this computational challenge, ALM portfolio managers have generally used LP approximations. A typical LP model (PF2) based on risk sensitivity matching is as follows:

Maximize

\[
\text{portfolio expected return}
\]

subject to

\[
\text{duration mismatch} \leq \text{target}_1,
\]

\[
\text{convexity mismatch} \leq \text{target}_2,
\]

\[
\text{first key-rate duration mismatch} \leq \text{target}_3,
\]

\[
\text{second key-rate duration mismatch} \leq \text{target}_4,
\]

\[
\text{kth key-rate duration mismatch} \leq \text{target}_{k+2},
\]

and

\[
\text{linear portfolio investment constraints}.
\]

Here, the objective and constraints are all linear functions; thus, we can solve the problem formulation efficiently with commercial LP solvers. However, these traditional models have several shortcomings. First, we cannot determine a meaningful efficient frontier because we have no one aggregate measure of risk. The portfolio risk is characterized by multiple risk sensitivities: duration, convexity, and key-rate durations. Portfolio managers have difficulties making decisions on risk/return trade-offs. Second, these risk measures capture only the severity aspect of the portfolio risk but not the frequency aspect. Therefore, they do not fully characterize the portfolio risk. Finally, setting the right mismatch targets is not a trivial task. In particular, it is difficult to adjust the partial duration targeted mismatches. Because the surplus volatility is not visible to the LP solvers, we must adjust the key-rate duration constraints to maintain the overall portfolio risk, measured by the surplus variance in a postoptimization step. An experienced portfolio manager must do the adjusting using a trial-and-error process; different experts will obtain different final portfolios.

We developed a novel SLP algorithm for solving high-dimensional portfolio-optimization problems
using the better formulation (PF1). It approximates efficient frontiers for large portfolios in minutes. Although we implemented the algorithm for ALM portfolios, it can be used to solve any portfolio-optimization problem with linear return and convex risk objectives.

The Sequential-Linear-Programming Algorithm

Our novel and patented algorithm approximates the efficient frontier between risk and return with a sequence of portfolios, starting with a portfolio that yields the highest possible return without reference to its risk. We obtain this portfolio by eliminating the nonlinear risk constraint from the model (PF1) and solving the resulting LP relaxation (Figure 1).

In the next step, we generate a portfolio with a marginally smaller risk value and a marginally smaller return by approximating the risk function with the linear function tangent to the risk contour passing through the previously generated portfolio (Figure 2). We then add a linear constraint to the LP, which restricts this linear risk function to be below a new target risk value that is marginally smaller than the risk of the previously generated portfolio (Figure 3). The solution to the resulting LP generates the next portfolio in the efficient frontier. We repeat this process to approximate the efficient frontier one step at a time (Figure 4).

Figure 1: To illustrate our sequential-linear-programming (SLP) algorithm, we show the feasible region of a two-asset portfolio-optimization problem with three linear constraints. For this problem, the SLP algorithm begins by generating the portfolio \( w_0 \) (point 1) that has the highest return by eliminating the risk constraint and solving the resulting linear program.

Figure 2: After obtaining the initial portfolio (point 1) with the highest return and risk at each subsequent step in the SLP algorithm, we approximate the risk function with a linear function. This linear function is the tangent to the risk contour passing through the previously generated portfolio on the efficient frontier.

Figure 3: In the SLP algorithm, once we obtain a linear approximation to the risk function at the previously generated portfolio, we add a new linear constraint to the LP, which restricts this linear risk function to be below a new target risk value. The new risk value is marginally smaller than the risk value of the previous portfolio. The dashed line is the tangent to the risk function, and the solid line parallel to the dashed line is the newly added risk constraint. Point 2 is the solution to the resulting LP, which is the new point on the efficient frontier.
The SLP Algorithm

Step 0. Eliminate the risk constraint in (PF1) and solve the resulting LP to generate portfolio \( P_0 \), which has the highest possible return, \( r_{P,0} \).

Let \( i = 1 \). Choose a suitably small value for \( \varepsilon_i \).

Step i. Evaluate the risk \( \sigma_{P,i-1} \) of portfolio \( P_{i-1} \). Compute the linear function tangent to the risk contour at \( P_{i-1} \). Let \( t_k \) denote the coefficient corresponding to the \( k \)th security. Add a new linear constraint, \( \sum_k t_k w_k \leq \sigma_{P,i-1} - \varepsilon_i \).

Solve the resulting LP to obtain the next portfolio on the efficient frontier.

If \( \sigma_{P,i} \) < lower bound on risk or if \( r_{P,i} \) < lower bound on return, stop. Otherwise, increment \( i \) and execute the next step.

Naturally, we want the sequence of portfolios obtained using the SLP algorithm to closely approximate the true efficient frontier, which depends on the value of \( \varepsilon \) being sufficiently small. For infinitesimally small values of \( \varepsilon \), the algorithm is guaranteed to converge to the true efficient frontier if the risk function is convex. In most portfolio-optimization problems, including the one we consider here, the risk function is quadratic and is convex relative to the feasible region. The algorithm will succeed even when the risk function is not convex, provided it is locally convex in the region of interest.

Computational Study

We first compared the computational efficiency of our algorithm and a third-party quadratic-programming solver by solving the problem formulation (PF1) multiple times for a varying number of assets. We constructed the test portfolios by varying the number of assets (mostly bonds) from 40 to 160. We optimized each of the portfolios using the quadratic-programming (QP) solver provided in the MATLAB optimization toolbox. We performed our analysis on a 1 GHz server with 4 GB RAM in the Windows NT environment. Run time increases exponentially with the increase in number of assets in the portfolio (Figure 5). Using a curve-fitting function in MATLAB, we obtained a transfer function that approximates the run time of the QP solver for a given number of assets. Upon extrapolation, we found that it would take 37 hours to generate a single portfolio on the efficient frontier using the QP solver for an ALM problem with 1,000 assets. If the number of assets were increased to 1,500, the run time would increase to approximately 133 hours. Approximating an efficient frontier for an ALM portfolio requires solving multiple optimization problems to obtain a number of points to determine the shape of the efficient frontier. Therefore, the total run time for creating an efficient frontier would be a large multiple of the run time for solving a single QP problem.

Naturally, we want the sequence of portfolios obtained using the SLP algorithm to closely approximate the true efficient frontier, which depends on the value of \( \varepsilon \) being sufficiently small. For infinitesimally small values of \( \varepsilon \), the algorithm is guaranteed to converge to the true efficient frontier if the risk function is convex. In most portfolio-optimization problems, including the one we consider here, the risk function is quadratic and is convex relative to the feasible region. The algorithm will succeed even when the risk function is not convex, provided it is locally convex in the region of interest.
Next, we constructed a test portfolio consisting of 1,500 assets. We optimized the portfolio to approximate the efficient frontier with the SLP algorithm written in MATLAB’s scripting language (Figure 6). We set the step size for the decrease in risk at each step at not more than 10 percent of the distance between the maximum and the minimum risk levels of interest. The resulting efficient frontier consists of 12 points (each point corresponding to an efficient portfolio). The run time for generating the 12 points on the efficient frontier was 226 seconds, a huge reduction from the run time using a QP solver. The run time increases linearly with the increase in the number of points on the efficient frontier.

We then compared the quality of the solutions from the SLP algorithm and the QP solver. We used a portfolio consisting of 30 assets to enable the QP solver to produce the efficient frontier fairly quickly. We compared the efficient frontiers obtained by the SLP algorithm with the true efficient frontier obtained by the QP solver. We found that the SLP with coarse step size (10 percent) provides a reasonable approximation of the efficient frontier (Figure 7). With a finer step size (one percent), the SLP’s efficient frontier very closely approximates the true efficient frontier (Figure 8).

These comparisons with the QP solver show that the SLP approach allows us to optimize much larger portfolios with a high degree of accuracy. Perhaps more important, however, is the question of whether the SLP approach also provides a significant improvement, in solution quality and run time, over the traditional LP approach.

In the second test, we compared the performance of the ALM problem formulation (PF1) based on portfolio surplus variance, solved using the SLP algorithm, with the performance of the ALM problem formulation (PF2) based on risk sensitivities, solved using an
LP solver. We conducted this test using a portfolio consisting of 1,500 bonds. The portfolio risk is measured by the volatility of the economic surplus. In addition to the risk-sensitivity constraints, the model included 65 linear investment constraints for credit quality and asset sourcing.

First, we optimized the portfolio using the traditional approach (PF2). We solved the problem using an LP solver. Because the economic surplus volatility is not visible, we have to adjust the key-rate duration constraints iteratively to maintain the overall portfolio risk. (In practice, an experienced portfolio manager, using a trial-and-error process, must perform these adjustments; different experts obtain different final portfolios.) After several iterations, we were able to move the portfolio in risk/return space from the baseline to one solution and finally to another (Figure 9). This manual iterative process could take several hours to several days, and one is forced to stop the process at some point without knowing whether the final solution is on the efficient frontier.

Next, we optimized the portfolio using the formulation (PF1) and the SLP algorithm, which was solved in minutes. The SLP algorithm was able to obtain several portfolios, which represent an increase in return, a decrease in risk, or both (Figure 9). Our studies showed that the SLP algorithm made major improvements in solutions obtained and computational time.

### System Implementation and Usage

We implemented the SLP algorithm in a portfolio-optimization tool developed as part of a larger initiative aimed at quantifying the diverse risks of insurance assets and identifying optimal risk/return trade-offs in asset-allocation strategy. In addition to bringing the analytic power of the SLP algorithm to analysts’ desktops, this tool also provides a test bed for developing and deploying new optimization techniques without disrupting the end-user’s environment.

To provide deployment flexibility, we implemented the portfolio optimizer tool as a Web application (Figure 10). Users do not need to install software on their desktops; they can interact with the application over the intranet via a standard Web browser. To insure design flexibility and development speed, we followed an n-tier application architecture model (Table 1). In an n-tier model, the functionality of the application is logically separated into distinct sets of components, or tiers. This separation provides for a modular and maintainable design. Another advantage is that one can physically separate tiers to take advantage of performance improvements via distributed processing. As problem size and usage scale up, one can break up tiers further and spread the load across multiple servers.

The lowest tier is the optimization engine. We use MATLAB from The Mathworks as the engine upon which to implement the SLP algorithm. We run the MATLAB engine on a centralized application server, with access to the engine provided by the Web application.

The Web application processes user inputs and requests, displays results, and manages the state information for the user’s session. We wrote the Web application using the Java 2 Platform, Enterprise Edition (J2EE). We built the application on top of an architectural framework for Web-based applications developed at GE Global Research. This framework implements the model-view-controller (MVC) pattern (Figure 11). The MVC pattern enforces a separation of presentation (view), application flow (controller),

---

**Figure 9:** Solving an ALM portfolio-optimization problem using the traditional approach is an iterative and trial-and-error process. By adjusting the key-rate duration constraints, we move the portfolio gradually toward the efficient frontier. After several iterations, we move the portfolio in risk/return space from the baseline to LP Solution #1 and finally to LP Solution #2. This manual iterative process could take several hours to several days. The SLP algorithm can approximate the efficient frontier in one run.
and underlying business logic (model). Built into the framework are utilities providing common services, such as logging, data access, and client communications. This framework allows us to focus on developing application-specific functionality.

Furthermore, the framework provides us with a means to implement business logic into reusable components, Java objects called command beans. We weave these command beans together via an XML-based language into scripts called interactions. An interaction defines an HTTP request we process from a Web browser and the response sent back to it. Furthermore, it defines the business logic components (for example, the command beans) that are used to process the request and to determine the order in which they are called. We use interactions to define at a high level how the application processes and responds to any HTTP queries; the framework handles the underlying implementation. Each interaction also defines a view. Views are typically implemented as Java server pages (JSPs), and for example, they construct the response messages (usually in HTML format) that are streamed back to a client (a Web browser). Like command beans, views can be used in more than one interaction. Taken together, these views make up
the application’s presentation tier. Using the framework, we can separate and develop the control flow, business logic, and user interface modules in parallel. We can also rapidly script new command beans and views to increase functionality so we can accelerate the development, testing, and deployment of new features.

The Web application invokes MATLAB via a third-party freeware library called JMatLink, passing to it pointers to the information on the optimization problem setup contained in a persistent data store. We use an Oracle database as our data store. The database acts as a tier between the Web application and the optimization engine. The data storage tier reduces the communication traffic between the Web application and the optimization engine tiers and improves performance. The data storage tier also provides encapsulation and flexibility: we can minimize the impact of changes to one tier on the other tiers. Development can proceed more or less independently within each tier.

Using the Web tool, a portfolio manager can browse through existing optimization results or set up and run new optimizations. To set up an optimization, a user must first define a portfolio and then load data on the portfolio’s current holdings, projections of the assets available for purchase in the market, a snapshot of the portfolio’s liabilities, and a covariance matrix describing how interest rates move over time. The user then enters a number of constraints that correspond to strategic investment guidelines and policies set by the business. Examples of such constraints are minimum and maximum holdings for individual assets, allocations into various asset classes or groups, credit quality, interest-rate risk, capital gains, effective duration and convexity, and partial durations. The user then chooses one return measure and one risk measure on which to optimize. All user-entered portfolio data are stored in the database.

Once the setup is complete, the user runs the optimization. The Web application invokes MATLAB, which runs the SLP algorithm as a separate process. When the algorithm finishes the optimization, it writes the results to the database. The results include asset buy and sell recommendations and statistics on overall yield, partial durations, credit quality, and interest-rate risk for every portfolio on the efficient frontier.

We provide a variety of reports to help users to explore the results. These reports are tools for performing trade-off analyses of the various sample portfolios returned by the SLP algorithm. Perhaps most important is a report allowing the user to view the efficient frontier for an optimization. We also provided users with means to visualize different portfolio metrics in addition to those selected as optimization objectives. They can then analyze sample portfolios in a number of risk/return spaces. Users can select any of the sample optimized portfolios to get more information—the most important source of information is a report that specifies what assets to buy or sell.

We developed the application at GE Global Research in Schenectady, New York during late 2002 and early 2003. After an extensive trial period and production use on GE Global Research servers, we transferred the application to GE Asset Management in early 2004. The system is currently running in production at GE Asset Management and also at Genworth.

**Benefits**

GEAM and Genworth have been using the SLP algorithm since April 2003, via our Web-based portfolio-optimization tool. Between early 2003 and April 2004, GEAM and Genworth used the algorithm to rebalance more than 30 portfolios, totaling over $30 billion. In addition, they used the tool to develop strategies and benchmarks for new cash investment for multiple product lines.

In examining existing portfolios, GEAM and Genworth quantified the potential for risk reduction or yield improvement to be gained from selling a portion of the currently held assets and replacing them with securities available in the market. The algorithm has proven to be well suited to such large-scale asset-liability optimizations involving thousands of securities. Such optimizations would otherwise be computationally infeasible. Using the tool, portfolio managers can be proactive, rather than strictly reactive; they can run an optimization multiple times as market conditions change, rather than only once after it becomes clear that a change is needed.

Based on the portfolios we analyzed during 2003 and the first half of 2004, GEAM and Genworth generated two basis points of annual benefits on average. The benefits are after-tax and based on incremental
returns attributable to our technology over and above any benefits we might have gotten using previous technology. GEAM and Genworth combined have in excess of $120 billion of fixed-income assets under management (AUM). For $1 billion AUM, one basis point is equivalent to a present value of $380,000 over five years, assuming a 10 percent discount rate. Hypothetically, based on $100 billion of assets, the present value of potential benefits could approximate $75 million over five years.

GEAM and Genworth have also used the tool to obtain a benchmark for new cash investment; for example, investing insurance premiums. In this application, analysts run the SLP algorithm regularly, using current market data and liability characteristics, to produce an unbiased assessment of the risk/return trade-offs for current investment.

Developing optimum risk/return positions (benchmarks) for new investment is difficult, because the benchmark must reflect asset mixes that match the specific characteristics of the corresponding liabilities (such as timing of expected payouts) while maximizing yield. External benchmarks, such as the Lehman indices, are inappropriate because they do not necessarily resemble the characteristics of any particular insurance product. The SLP algorithm has been highly successful at optimizing assets available in the market versus liability durations to develop the portfolio efficient frontier. Through April 2004, GEAM has used the SLP algorithm to develop initial investment strategies for a number of products, as well as to set an investment benchmark for new investment totaling over $1 billion annually.

GEAM, Genworth, and GEI are using the SLP algorithm to manage life insurance portfolios, property and casualty insurance portfolios, and mortgage insurance portfolios. Portfolio managers can perform portfolio analytics much faster than they could, and can therefore better respond to changing market conditions. Given the computational efficiency of the SLP algorithm, they can also analyze portfolios more frequently. However, they decide on trades with the trade-off on transaction costs in mind. Another potential extension of the SLP algorithm is portfolio optimization on total return. This analysis is typically limited to a small number of asset classes (30 to 50) due to the computational complexity of optimizing on the nonlinear total risk/return metrics. The SLP algorithm could allow the number of asset classes under consideration by a portfolio manager in a total-return optimization to be expanded much farther than the capabilities of commercially available solvers.

Appendix

Notation

We use the following notation for portfolio-optimization models:

- Portfolio expected return: \( r_p = \sum_{i=1}^{n} w_i r_i \)
- Expected return of \( i \)th security: \( r_i \)
- Weight of \( i \)th security in the portfolio: \( w_i \)
- Surplus variance: \( \sigma_p^2 = \sum_{k=1}^{m} \sum_{i=1}^{n} \delta_{p,k} \delta_{p,i} \sigma_{k,i} \)
- Portfolio key-rate duration mismatch for the \( k \)th key rate, defined as \( \delta_{p,k} = \sum_{i=1}^{n} w_i \delta_{p,k} - \delta_{p,k}^i \)
- \( \delta_{p,k}^i \) th asset, defined as \( \delta_{p,k}^i = 1/V^A \cdot \partial V^A / \partial F_k \)
- \( \delta_{k}^i \) th asset key-rate duration of the liability, defined as \( \delta_{k}^i = 1/V^L \cdot \partial V^L / \partial F_k \)
- \( V^A \) value of the \( i \)th asset.
- \( V^L \) value of the liability.
- \( F_k \) value of the \( k \)th risk factor.
- Duration mismatch: \( |\text{asset duration} - \text{liability duration}| \)
- Convexity mismatch: \( |\text{asset convexity} - \text{liability convexity}| \)
- Duration: \( (V^- - V^+) / 2V^0 (\Delta y) \)
- Convexity: \( (V^+ + V^- - 2V^0) / 2V^0 (\Delta y)^2 \)
- \( V^+ \) value of a security given an upward parallel shift to the yield curve.
- \( V^- \) value of a security given a downward parallel shift to the yield curve.
- \( V^0 \) initial value of a security.
- \( \Delta y \) a parallel shift in the yield curve.

Computing Variance Using the Multifactor Risk Framework

The value of a financial security can be characterized as a function of multiple risk factors

\[ V = f(F_1, F_2, \ldots, F_m), \]

where

- \( V \) value of a financial security.
- \( F_k \) th risk factor value.
- \( m \) number of risk factors.

Using a Taylor series expansion up to the first order, we can estimate the relative change in the security...
value from the following expression:

\[
\frac{\Delta V}{V} \approx \sum_{k=1}^{m} \left( \frac{1}{V} \cdot \frac{\partial V}{\partial F_k} \right) \Delta F_k, \quad \text{where}
\]

\[\Delta V / V = \text{relative change in the security value,} \]

\[\Delta F_k = \text{change in the } k\text{th risk factor,} \]

\[1/V \cdot \partial V/\partial F_k = \text{first-order risk sensitivity to the} \]

\[k\text{th risk factor, also known as delta.} \]

We can easily extend the framework using a Taylor series expansion up to the second-order term to estimate for change in the security value. The approach is also known as a delta-gamma approximation.

For a fixed-income security, the first-order risk sensitivity to change in key nodes on the interest-rate curve is known as key-rate duration. We define this term as follows:

\[\delta_k = \frac{1}{V} \cdot \frac{\partial V}{\partial F_k}, \quad \text{where} \]

\[\delta_k = \text{key-rate duration for the } k\text{th rate on the interest} \]

\[\text{rate curve.} \]

The relative change in a portfolio value due to change in multiple risk factors can be expressed in the following form:

\[
\frac{\Delta V_p}{V_p} = \sum_{i=1}^{n} w_i \Delta V_i = \sum_{i=1}^{n} \left( w_i \sum_{k=1}^{m} \delta_k \Delta F_k \right), \quad \text{where}
\]

\[w_i = \text{weight of } i\text{th security in the portfolio defined by} \]

\[w_i = V_i / V_p. \]

Assuming that all risk factors have zero mean change, the variance of the relative change in the portfolio value due to change in multiple risk factors can be estimated by

\[\sigma_p^2 = \sum_{k=1}^{m} \sum_{l=1}^{m} \delta_{p,k} \delta_{p,l} \sigma_{kl}, \quad \text{where} \]

\[\delta_{p,k} = \text{first-order portfolio risk sensitivity to the} \]

\[k\text{th risk factor defined by} \]

\[\delta_{p,k} = \sum_{i=1}^{n} w_i \delta_{i,k}. \]

\[\sigma_{kl} = \text{covariance between the } k\text{th and } l\text{th risk factors} \]

\[\text{defined as} \]

\[\sigma_{kl} = \text{cov} (\Delta F_k, \Delta F_l). \]

References