Optimal Asset Allocation for Retirement Planning
MS&E 348 Optimization Under Uncertainty with Applications in Finance
Group Project -- Spring 2001

Submitted by:
Robert Feinglass
Holly Lutze
Charles Ng

March 20, 2001
**Introduction**

In this section, we will begin by thoroughly describing our problem, including assumptions and parameters. Then, we will discuss some basic intuition for the results we hope to achieve.

**Asset Allocation Problem**

An individual is planning to retire in fifteen years and would like to have a certain amount of money, her "goal", with which to do so. In our problem, the retirement goal is assumed to be $1,000,000. The individual has an initial wealth level to invest in four different asset types: stocks, corporate bonds, government bonds, and cash. Assuming that the individual has an opportunity to cash out and reinvest at the beginning of each year, what is the optimal asset allocation policy for her to follow over the next fifteen years until her retirement?

The returns of each asset are random, and the multivariate normal distribution for the joint returns has the following mean vector and covariance matrix.

\[
\text{Mean} = \begin{pmatrix}
\text{Stocks} \\
\text{CorpBonds} \\
\text{GvmtBonds} \\
\text{Cash}
\end{pmatrix} = \begin{pmatrix}
10.5 \\
8.5 \\
7 \\
4
\end{pmatrix}
\]

\[
\text{Covariance Matrix} = \begin{pmatrix}
240.25 & 72.68 & 41.43 & 0 \\
72.68 & 141.61 & 94.25 & 0 \\
41.43 & 94.25 & 67.24 & 0 \\
0 & 0 & 0 & 0.001
\end{pmatrix}
\]

We assume that the returns in each period are stochastically independent and that no transaction fees are applied.

We consider initial wealth levels ranging from $10,000 to $1,505,000 in intervals of $5,000. In this way, we consider a range of investors from the poor end of the scale to the rich. We hope to determine the optimal asset allocation policy at each discrete wealth level and at each time period.

**Basic Intuition**

In modeling this problem as a finite-horizon stochastic dynamic program, we are allowing for the possibility that a myopic asset allocation policy is not optimal. Indeed, one can anticipate the investor behaving differently when she is fifteen years away from her retirement than when she only has one year left in which to invest. In particular, we would expect the investor to be more conservative as she approaches her retirement period, because large losses late in the game cannot be easily recovered.
One can also anticipate the investor to behave differently with different wealth levels. At the lowest wealth levels, the investor must be less conservative if she is to have any success in reaching her goal. At moderate wealth levels, we expect the investor to be more conservative, wishing both to promote steady growth of the value of her assets and to preserve this value when close to her retirement goal. At the highest wealth levels, the investor will likely be less conservative again, because the goal is already achieved and additional value can be pursued without high risk of falling below the goal.

Report Outline

This report describes our initial study into this asset allocation problem. Further sections of the report address the following:

- mathematical definition for the model used
- methods employed to convert the mathematical model into a GAMS program
- insights gained from the policy output of the stochastic dynamic program
- comparison of out-of-sample performance of SDP Optimal Policy with two myopic policies
- description of issues and concerns encountered within the project
- suggestions for future extensions of this project

Relevant GAMS programs are listed in the appendices.

Mathematical Formulation

This section details the mathematical formulation of the asset allocation problem as a stochastic dynamic program. We begin with problem definition, including variables, constraints, objective function, and system dynamics. Next, we determine the value of a stochastic solution, as well as an upper bound on policy performance for each wealth level. Last, we discuss desired properties for the final utility function and specify those selected for evaluation.

Problem Definition

The following are the variables used in our model:

\[ j = \text{asset} \in \{\text{Stocks, CorpBonds, GvmtBonds, Cash}\} \]
\[ t = \text{number of time periods to go until retirement} \]
\[ x_t = \text{vector of asset allocations with \( t \) time periods to go} \]
\[ R_t = \text{vector of random returns with \( t \) periods to go} \]
\[ w_t = \text{wealth at beginning of period with \( t \) periods to go} \]

With an appropriate utility function applied to the wealth level at the time of retirement, the optimal policy consists of 15 vectors of optimal asset allocations. Since the optimal asset
allocation $x_t$ in a particular period $t$ is a function of the beginning wealth level $w_t$, it is important to recognize the following system dynamics.

$$w_{t-1} = w_t \ast (R^T_t \ast x_t)$$

This in turn leads to the dynamic programming recursion, where $V_t(w_t)$ indicates the maximum expected utility function value at retirement given the investor has wealth $w_t$ at the beginning of period $t$,

$$V_t(w_t) = \max_{x_{t:}, x_t \geq 0} E[V_t(w_t \ast (R^T_t \ast x_t))]$$

Equivalently, we can represent the problem as follows:

$$\max_{x_{1:n}, x_1} E(U(w_0)) = \max_{x_{1:n}, x_1} E[\max_{x_1} E[\max_{x_1} E[U(w_0)]]]$$

subject to $\sum_j x^j_t = 1 \ \forall t$ and $x^j_t \geq 0 \ \forall j, t$

**Assessment of Stochastic Dynamic Programming Approach**

Modeling the problem as a stochastic dynamic program certainly adds complexity above the deterministic model, which chooses an optimal policy under the assumption that each asset will realize its mean return at every period. The value of the stochastic solution (VSS) is a simple way of measuring the value of this added complexity. If we let $z$ indicate the value of the optimal solution to the model specified above and $\widehat{z}_d$ indicate the value of the solution to the deterministic problem, then the value of the stochastic solution is simply

$$z - \widehat{z}_d$$

However, this value can only be determined after a stochastic dynamic program has been developed and solved. If one wishes to justify in advance the effort involved in developing and solving a stochastic dynamic program, one can calculate an upper bound on the VSS. If we let $z_d$ be the expected utility of the final wealth under the optimal deterministic policy, then

$$VSS \leq \widehat{z}_d - z_d.$$
Equality is attained in the case that \( \hat{z}_d \) equals \( z \); then, exercising the deterministic policy in a deterministic environment supplies us with as much expected utility as does exercising the stochastic policy in the stochastic environment.

Furthermore, once we have a solution to our stochastic dynamic program, we can judge the merits of our optimal policy by assessing the expected value of perfect information (EVPI), the price the investor would be willing to pay for the privilege of perfect foresight; that is, to know the future asset returns between now and the time of her retirement. If we let \( z_{ws} \) be this optimal "wait-and-see" value, the expected utility of final wealth given that the investor knows the future with certainty, then the expected value of perfect information is

\[
z_{ws} - z.
\]

Using similar reasoning as above, the difference \( z_{ws} - z_d \) supplies an upper bound on the EVPI, with equality in the case that we can do no better than what the deterministic policy yields in a stochastic environment.

The following inequalities will help the reader to visualize the relationship between these binding quantities:

\[
z_d \leq z \leq \hat{z}_d \leq z_{ws}
\]

- The stochastic solution \( z \) can do no worse than \( z_d \), because \( z \) is attained from a policy which hedges against the uncertainty in the returns, while \( z_d \) is not.
- The stochastic solution \( z \) can do no better than \( \hat{z}_d \) as a consequence of Jensen’s inequality.
- The deterministic solution \( \hat{z}_d \) can do no better than the "wait-and-see" solution \( z_{ws} \) in the limiting case of enumerating an arbitrarily large number of sample paths, because the average of the returns falling below the stock mean of 10.5% is closer to 10.5% than the average of the returns falling above that value.

In order to evaluate the merits of our study of the asset allocation problem as a stochastic dynamic program, we have calculated the VSS and EVPI over a range of wealth levels. For \( \hat{z}_d \), the obvious deterministic solution is to invest in stocks each period (since they have the highest expected return), for a total fifteen year return of

\[
1.105^{15} \times (\text{initial wealth level}).
\]

We then determine \( z_d \) to be the mean return after fifteen periods, using the deterministic all-stock policy. The GAMS code used to solve for \( z_d \) is placed in Appendix A. Furthermore, we can find \( z_{ws} \) as the mean return after fifteen periods if one has perfect knowledge of the future. We approximate this quantity by simulating 1000 instances of random returns, and letting our total return equal the product of the maximum in each period among the four random asset
returns. The GAMS code used to solve for \( z_{us} \) is placed in Appendix B. From this information, prior to solving the stochastic dynamic program, we can be certain that the VSS and EVPI are each bounded above by \( \hat{z}_d - z_d \), and by \( z_{us} - z_d \), respectively. Finally, \( z \) is the mean return after fifteen periods if one uses the optimal policy found through stochastic dynamic programming. All averages are taken over 1,000 simulated random returns. Table 2 indicates the relevant values for our analysis over a range of initial wealth levels.

<table>
<thead>
<tr>
<th>Initial Wealth</th>
<th>( z_d )</th>
<th>( z_{us} )</th>
<th>Upper Bound ( z )</th>
<th>VSS</th>
<th>VSS % of Upper Bound</th>
<th>EVPI</th>
<th>EVPI % of Upper Bound</th>
</tr>
</thead>
<tbody>
<tr>
<td>$50,000</td>
<td>$223,565</td>
<td>$16,934</td>
<td>$225,399</td>
<td>-13.96%</td>
<td>$331,294</td>
<td>94.50%</td>
<td></td>
</tr>
<tr>
<td>$100,000</td>
<td>$447,130</td>
<td>$651,792</td>
<td>$665,065</td>
<td>8.28%</td>
<td>$584,590</td>
<td>89.69%</td>
<td></td>
</tr>
<tr>
<td>$150,000</td>
<td>$670,696</td>
<td>$961,499</td>
<td>$1,026,091</td>
<td>27.97%</td>
<td>$873,554</td>
<td>90.85%</td>
<td></td>
</tr>
<tr>
<td>$200,000</td>
<td>$894,261</td>
<td>$1,240,787</td>
<td>$1,321,154</td>
<td>8.28%</td>
<td>$1,120,194</td>
<td>90.28%</td>
<td></td>
</tr>
<tr>
<td>$250,000</td>
<td>$1,117,826</td>
<td>$1,501,065</td>
<td>$1,551,083</td>
<td>39.55%</td>
<td>$1,328,026</td>
<td>88.48%</td>
<td></td>
</tr>
<tr>
<td>$300,000</td>
<td>$1,341,391</td>
<td>$1,700,531</td>
<td>$1,716,003</td>
<td>39.55%</td>
<td>$1,474,148</td>
<td>86.69%</td>
<td></td>
</tr>
<tr>
<td>$350,000</td>
<td>$1,564,956</td>
<td>$1,986,260</td>
<td>$1,996,083</td>
<td>41.19%</td>
<td>$1,723,303</td>
<td>86.24%</td>
<td></td>
</tr>
<tr>
<td>$400,000</td>
<td>$1,788,521</td>
<td>$2,274,359</td>
<td>$2,280,083</td>
<td>37.14%</td>
<td>$1,773,115</td>
<td>85.95%</td>
<td></td>
</tr>
<tr>
<td>$450,000</td>
<td>$2,012,087</td>
<td>$2,550,369</td>
<td>$2,556,083</td>
<td>37.14%</td>
<td>$1,710,009</td>
<td>85.36%</td>
<td></td>
</tr>
<tr>
<td>$500,000</td>
<td>$2,235,652</td>
<td>$2,836,649</td>
<td>$2,836,649</td>
<td>37.14%</td>
<td>$2,100,990</td>
<td>85.95%</td>
<td></td>
</tr>
<tr>
<td>$550,000</td>
<td>$2,459,217</td>
<td>$3,128,211</td>
<td>$3,128,211</td>
<td>37.14%</td>
<td>$2,737,841</td>
<td>85.87%</td>
<td></td>
</tr>
<tr>
<td>$600,000</td>
<td>$2,682,782</td>
<td>$3,421,851</td>
<td>$3,421,851</td>
<td>37.14%</td>
<td>$2,552,214</td>
<td>77.06%</td>
<td></td>
</tr>
</tbody>
</table>

As one can see, the value of our stochastic solution above the deterministic solution is only about 8% of its upper bound at $100,000 and increases to over 41% of its upper bound in the high end of the wealth range shown. The fact that our EVPI is on the order of 85-90% of its upper bound indicates to us that our stochastic dynamic programming results yield only a moderate improvement upon the deterministic solution, while we are far from the results gained through perfect foresight.

**Properties of Utility Function**

The above problem definition does not indicate the utility function being used because we feel this subject deserves special attention. An appropriate utility function must first be concave and nondecreasing, which indicates decreasing, non-negative marginal returns. Additionally, we are interested in utility functions with a minimum second derivative point corresponding to our final goal of $1,000,000. Such utility functions would primarily seek to meet the goal with some urgency and to maximize the final wealth as a secondary motivation.

Several functional forms were considered as candidates. The first was a shifted logarithmic function, but we found that the corresponding optimal policy did not vary with the wealth level. Hence, we judged this utility function to represent insufficiently the interests of the investor. A second candidate, a hyperbolic curve \((a \times w - \frac{b}{w} + c)\), appeared to offer theoretical promise. This type of function would be asymptotically bounded by a risk-neutral line for wealth levels above the point of minimum second derivative, a property which seems sensible in the context of our problem. However, adjusting parameters in order to shift the curve to properly align the minimum second derivative point proved to be excessively difficult. A third candidate, the power function \((a + b \times w^c)\), was also soon abandoned for the final candidate, the exponential...
function \((a + b \cdot c^w)\). We were able to successfully adjust the parameters of this function to form a utility curve with the desired properties.

The parameters selected for the utility function in our model are as follows:

\[
\begin{align*}
    a &= 5 \\
    b &= -50 \\
    c &= 0.8
\end{align*}
\]

**GAMS Program Implementation**

At this point, we shall describe how the implementation of this project is done in GAMS. The actual code for the GAMS program is given in Appendix C.

At a certain wealth level with one period to go, 1000 random samples are taken from the joint distribution of returns. We then solve a deterministic nonlinear program that finds the optimal policy \(x_1\) that maximizes average utility at retirement over the 1000 scenarios. By doing this for a large number of wealth levels, 300 in our case, we get a rough approximation for the utility function of wealth with one year to go till retirement. With the 300 points obtained for our utility for \(t=1\), we then attempt to approximate our new utility function. After we obtain the approximations for our utility function at \(t = 1\), we take a step back and then calculate maximum average utility and optimal policy for each wealth level for \(t = 2, 3 \ldots 15\). Three main approaches for approximating the utility at each period were tested, viz., piecewise linear, restricted best-fit, and general best-fit approximation.

The piecewise linear approach assumes a utility function that is known at the 300 wealth levels, and is linear for everything in between. We also extrapolate to get a function that is defined over all wealth levels \(w > 0\). The limitations of this method of approximation have to do with the forced linearity of the function. First, linearity is assumed beyond the range of wealth levels considered, and, second, the linearity, and thus risk neutrality, is a function of the number of wealth levels. Hence, we would like to divided the range of initial wealth considered into as many wealth levels as possible, in order to capture the nonlinearity of the utility function. However, as discussed under "Issues and Concerns," this motivation to increase the number of wealth levels is contradicted by a computational requirements issue that forces us to actually reduce the number of wealth levels considered when adopting the piecewise linear approach.

The restricted best fit approach is named for the fact that we assume, when using it, that all utility functions will be of the same functional form as the utility function used at \(t = 0\), viz., \(a + b \cdot c^w\). At each period, we are simply finding the parameter values \((a, b, c)\) that best fit our 300 data points according to a least-squares approximation.

In order to address the concern that we may be losing information regarding the change in utility function over time, we then used a general best fit approach, which allows for the possibility of other terms entering the utility functions. In our general best fit program, we allowed for constant, linear, power, exponential and logarithmic terms that were a function of the wealth level divided by $100K. The allowed functional form can be written as
\[ a + b \cdot w + c \cdot w^d + e \cdot f^{gw} + h \cdot \log(iw) + j \cdot w^k + l \cdot w^m + n \cdot \omega^{pw}, \]

and the entering coefficients are shown in Table 2.

Table 2. General Best Fit Parameters for Each Time Period

<table>
<thead>
<tr>
<th>Years to Go</th>
<th>a</th>
<th>b</th>
<th>c</th>
<th>d</th>
<th>e</th>
<th>f</th>
<th>g</th>
<th>h</th>
<th>l</th>
<th>j</th>
<th>k</th>
<th>l</th>
<th>m</th>
<th>n</th>
<th>o</th>
<th>p</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>5.00</td>
<td>0.00</td>
<td>0</td>
<td>0</td>
<td>-50.00</td>
<td>0.80</td>
<td>1.00</td>
<td>0.00</td>
<td>1.00</td>
<td>0.00</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0.00</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>4.57</td>
<td>0.03</td>
<td>0</td>
<td>0</td>
<td>-48.02</td>
<td>0.78</td>
<td>1.00</td>
<td>0.07</td>
<td>1.00</td>
<td>-0.42</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>-0.43</td>
<td>0</td>
<td>0.00</td>
</tr>
<tr>
<td>2</td>
<td>3.99</td>
<td>0.06</td>
<td>0</td>
<td>0</td>
<td>-48.07</td>
<td>0.76</td>
<td>1.01</td>
<td>0.16</td>
<td>0.94</td>
<td>-1.00</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>-0.43</td>
<td>0</td>
<td>0.01</td>
</tr>
<tr>
<td>3</td>
<td>3.66</td>
<td>0.09</td>
<td>0</td>
<td>0</td>
<td>-47.21</td>
<td>0.74</td>
<td>1.01</td>
<td>0.27</td>
<td>0.89</td>
<td>-1.29</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>-0.43</td>
<td>0</td>
<td>0.01</td>
</tr>
<tr>
<td>4</td>
<td>3.44</td>
<td>0.11</td>
<td>0</td>
<td>0</td>
<td>-46.48</td>
<td>0.72</td>
<td>1.02</td>
<td>0.37</td>
<td>0.82</td>
<td>-1.50</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>-0.43</td>
<td>0</td>
<td>0.01</td>
</tr>
<tr>
<td>5</td>
<td>3.23</td>
<td>0.13</td>
<td>0</td>
<td>0</td>
<td>-45.65</td>
<td>0.70</td>
<td>1.02</td>
<td>0.51</td>
<td>0.71</td>
<td>-1.70</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>-0.43</td>
<td>0</td>
<td>0.01</td>
</tr>
<tr>
<td>6</td>
<td>3.10</td>
<td>0.15</td>
<td>0</td>
<td>0</td>
<td>-44.84</td>
<td>0.68</td>
<td>1.03</td>
<td>0.66</td>
<td>0.59</td>
<td>-1.83</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>-0.43</td>
<td>0</td>
<td>0.01</td>
</tr>
<tr>
<td>7</td>
<td>3.01</td>
<td>0.17</td>
<td>0</td>
<td>0</td>
<td>-43.95</td>
<td>0.66</td>
<td>1.03</td>
<td>0.82</td>
<td>0.47</td>
<td>-1.91</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>-0.43</td>
<td>0</td>
<td>0.01</td>
</tr>
<tr>
<td>8</td>
<td>2.97</td>
<td>0.18</td>
<td>0</td>
<td>0</td>
<td>-43.06</td>
<td>0.63</td>
<td>1.04</td>
<td>1.00</td>
<td>0.38</td>
<td>-1.95</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>-0.43</td>
<td>0</td>
<td>0.01</td>
</tr>
<tr>
<td>9</td>
<td>2.96</td>
<td>0.20</td>
<td>0</td>
<td>0</td>
<td>-42.19</td>
<td>0.61</td>
<td>1.04</td>
<td>1.19</td>
<td>0.34</td>
<td>-1.96</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>-0.43</td>
<td>0</td>
<td>0.01</td>
</tr>
<tr>
<td>10</td>
<td>2.95</td>
<td>0.21</td>
<td>0</td>
<td>0</td>
<td>-41.37</td>
<td>0.58</td>
<td>1.05</td>
<td>1.37</td>
<td>0.32</td>
<td>-1.96</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>-0.43</td>
<td>0</td>
<td>0.01</td>
</tr>
<tr>
<td>11</td>
<td>2.95</td>
<td>0.22</td>
<td>0</td>
<td>0</td>
<td>-40.46</td>
<td>0.56</td>
<td>1.06</td>
<td>1.57</td>
<td>0.30</td>
<td>-1.96</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>-0.43</td>
<td>0</td>
<td>0.01</td>
</tr>
<tr>
<td>12</td>
<td>2.94</td>
<td>0.24</td>
<td>0</td>
<td>0</td>
<td>-39.49</td>
<td>0.53</td>
<td>1.07</td>
<td>1.79</td>
<td>0.30</td>
<td>-1.96</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>-0.43</td>
<td>0</td>
<td>0.01</td>
</tr>
<tr>
<td>13</td>
<td>2.94</td>
<td>0.25</td>
<td>0</td>
<td>0</td>
<td>-38.48</td>
<td>0.50</td>
<td>1.08</td>
<td>2.03</td>
<td>0.31</td>
<td>-1.96</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>-0.43</td>
<td>0</td>
<td>0.01</td>
</tr>
<tr>
<td>14</td>
<td>2.95</td>
<td>0.26</td>
<td>0</td>
<td>0</td>
<td>-37.76</td>
<td>0.47</td>
<td>1.09</td>
<td>2.20</td>
<td>0.32</td>
<td>-1.96</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>-0.43</td>
<td>0</td>
<td>0.01</td>
</tr>
<tr>
<td>15</td>
<td>2.95</td>
<td>0.27</td>
<td>0</td>
<td>0</td>
<td>-36.90</td>
<td>0.45</td>
<td>1.09</td>
<td>2.41</td>
<td>0.33</td>
<td>-1.96</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>-0.43</td>
<td>0</td>
<td>0.01</td>
</tr>
</tbody>
</table>

As one can see, both log and power terms eventually enter the utility function with some certainty. One application of the general best fit approximation resulted in $R^2 \geq 0.999989...$ for all time periods.

**Properties of Optimal Policy**

Using the above three approximation methods, we were able to derive three different optimal policies, which are represented graphically in this section for $t \in \{1, 5, 10, 15\}$.

First, we present the policy corresponding to the general best fit approximation, which we will henceforth call our SDP (Stochastic Dynamic Programming) Optimal Policy. This policy is displayed in Figure 1.
Figure 1 indicates the percentage of investment principle to be invested in each asset with 1 period until retirement, 5 periods until retirement, and 10 and 15 years until retirement. The red region corresponds to stocks, the yellow to corporate bonds, and the blue to government bonds. The $x$-axis spans wealth levels from zero to $1,000,000$.

Beginning with 15 time periods until retirement, one can see that the smallest wealth levels must play all stocks in order to have any hope of achieving the retirement goal. The range of wealth levels over which this is true increases as one gets closer to retirement, because one is running out of time. The largest wealth levels begin by being more conservative, with a significant portion of the initial wealth being invested in government bonds. This behavior may be explained by the fact that taking excessive risks early on may cause accumulated losses that prevent one from reaching the retirement goal. As retirement approaches, however, the higher wealth levels can not accumulate losses as readily, so one is willing to take more risk. Notice that with one period until retirement, no government bonds are purchased.

Figure 2 represents the optimal policy under a restricted best fit approximation method, with the same graphical depiction as used for the general best fit.
Figure 2. Optimal Policy for Restricted Best Fit Approximation Method

We first note that the optimal policy with one year remaining until retirement is identical to that for the general best fit, since both use our initially defined exponential utility function. However, the behavior at wealth levels close to the goal for the general best fit policy now takes place at wealth levels close to $500,000, and the highest wealth levels tend to be more risk neutral when there are many years until retirement. Essentially, this behavior suggests that individuals at the highest wealth levels are willing to take the risk associated with an all stock policy when they have a large number of years left to recover any losses. As retirement approaches, then, these individuals become more conservative, trying to maintain the wealth already accumulated.

To explain the differences between the restricted best fit and general best fit policies, it is useful to plot the approximated utility functions over time. In Figure 3, we can compare the approximated utility functions, with the red line corresponding to $t = 0$, the gold to $t = 5$, the green to $t = 10$, and the blue to $t = 15$. At first glance the plots look very similar, but the reason for the discrepancy for the two policies may revealed in the wealth level at which the approximated utility function for each time period begins to approach linearity. For the restricted best fit, since all utilities are forced to take the form of our
original utility, which is bounded above by 5, the wealth level at which the expected utility approaches linearity decreases as the number of years until retirement increases. On the other hand, the general best fit approximations do not exhibit more linear behavior until much higher wealth levels, indicating that the wealth level at which an individual becomes risk neutral is higher.

Figure 3. Best Fit Utility Function Approximations, $T = 0, 5, 10, 15$

Figure 4. Optimal Policy for Piecewise Linear Approximation Method
The third policy, displayed in Figure 4, corresponds to a piecewise linear approximation. Due to the linearity of the approximation, the policy becomes considerably more risk neutral, and, hence, less conservative. The policy remains the same, of course, with one period until retirement. As the number of years until retirement increases, the policy compares to the restricted best fit in the same way that the restricted best fit compares to the general best fit. In addition, we see an overall increase in the willingness to take risks by the fact that no government bonds are ever purchased.

**Performance of SDP Optimal Policy**

Using the policy found using a general best fit utility approximation method as our SDP Optimal Policy, in the next section we compare the out-of-sample performance of this policy to two different myopic policies. The first myopic policy, called the myopic mean-variance policy is found by solving the following:

\[
\begin{align*}
\text{maximize} & \quad \sum_j \bar{R}_j x_j - \frac{1}{2} \sum_i \sum_j x_i x_j \sigma_{ij} \\
\text{subject to} & \quad \sum_j x_j = 1 \\
& \quad x_j \geq 0 \quad \forall j \\
\end{align*}
\]

where \(\sigma_{ij} = \text{covariance}(i, j)\)

The optimal myopic mean-variance policy is to allocate one's assets according to

- \(Stocks = 1.0849\%\)
- \(GvmtBonds = 1.5768\%\)
- \(Cash = 97.3384\%\)

The second myopic policy, called the myopic utility policy, simply maximizes in each period the expected value of the exponential utility function defined for \(t = 0\). Hence, the policy chooses the same allocation in every period that the SDP Optimal Policy selects with one period until retirement. The GAMS program that compares the performance of the SDP Optimal Policy with these two myopic policies for a particular wealth level can be found in Appendix D. At each wealth level we find the average utility at retirement over 1000 sample return paths, which are different from the sample used to derive the SDP Optimal Policy.

For wealth levels ranging from $100,000 to $600,000, in increments of $50,000, our SDP Optimal Policy appears to outperform both myopic policies, but in order to test the significance of this result we conducted simple hypothesis tests. These tests determine the degree of confidence with which we can say that the final utility under the SDP Optimal Policy is higher than the final utility under the two myopic policies. The hypothesis test results are displayed in Figure 5.
\( H_0: \ E[\text{SDP Utility} - \text{Myopic Utility}] \leq 0 \)

\( H_a: \ E[\text{SDP Utility} - \text{Myopic Utility}] > 0 \)

\[ \alpha = 0.010 \]

<table>
<thead>
<tr>
<th>Initial Wealth</th>
<th>SDP Optimal Policy</th>
<th>SDP Optimal Policy</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Myopic Utility Policy</td>
<td>Myopic Utility Policy</td>
</tr>
<tr>
<td>$100,000</td>
<td>-1.285019</td>
<td>40.859236</td>
</tr>
<tr>
<td>$150,000</td>
<td>0.2737</td>
<td>46.2974</td>
</tr>
<tr>
<td>$200,000</td>
<td>2.3667</td>
<td>51.3879</td>
</tr>
<tr>
<td>$250,000</td>
<td>3.2295</td>
<td>55.2267</td>
</tr>
<tr>
<td>$300,000</td>
<td>2.9307</td>
<td>58.1704</td>
</tr>
<tr>
<td>$350,000</td>
<td>4.0506</td>
<td>60.1901</td>
</tr>
<tr>
<td>$400,000</td>
<td>4.1133</td>
<td>61.8536</td>
</tr>
<tr>
<td>$450,000</td>
<td>4.2793</td>
<td>64.9700</td>
</tr>
<tr>
<td>$500,000</td>
<td>4.0586</td>
<td>66.3748</td>
</tr>
<tr>
<td>$550,000</td>
<td>4.1885</td>
<td>67.7799</td>
</tr>
<tr>
<td>$600,000</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Figure 5. Hypothesis Tests for SDP Optimal Policy Outperforming Myopic Policies

Regardless of the wealth level, the SDP Optimal Policy clearly outperforms the myopic mean-variance policy, and, for wealth levels of at least $200,000, the SDP Optimal Policy outperforms the myopic utility policy with greater than 99% confidence. One may question how much better than the myopic utility policy the SDP Optimal Policy actually performs. One means of addressing this question is shown in Table 3.

<table>
<thead>
<tr>
<th>Wealth</th>
<th>SDP Optimal Policy</th>
<th>Myopic Utility Policy</th>
<th>% Increase in Equivalent Wealth</th>
</tr>
</thead>
<tbody>
<tr>
<td>$150,000</td>
<td>-9.8547</td>
<td>$543,912</td>
<td>0.08%</td>
</tr>
<tr>
<td>$200,000</td>
<td>-5.5846</td>
<td>$695,794</td>
<td>0.96%</td>
</tr>
<tr>
<td>$250,000</td>
<td>-2.6625</td>
<td>$840,573</td>
<td>1.56%</td>
</tr>
<tr>
<td>$300,000</td>
<td>-0.6111</td>
<td>$980,209</td>
<td>1.94%</td>
</tr>
<tr>
<td>$350,000</td>
<td>0.8422</td>
<td>$1,114,547</td>
<td>2.15%</td>
</tr>
<tr>
<td>$400,000</td>
<td>1.8894</td>
<td>$1,244,577</td>
<td>2.17%</td>
</tr>
<tr>
<td>$450,000</td>
<td>2.6042</td>
<td>$1,361,589</td>
<td>2.19%</td>
</tr>
<tr>
<td>$500,000</td>
<td>3.2359</td>
<td>$1,498,747</td>
<td>2.88%</td>
</tr>
<tr>
<td>$550,000</td>
<td>3.6634</td>
<td>$1,623,133</td>
<td>3.41%</td>
</tr>
<tr>
<td>$600,000</td>
<td>3.9798</td>
<td>$1,744,171</td>
<td>3.66%</td>
</tr>
</tbody>
</table>

Table 3. Comparing SDP Optimal Policy & Myopic Utility Policy

Here we convert the final expected utility value from each policy to the equivalent wealth which obtains this utility value. One can then see from the last column that the SDP Optimal Policy achieves a 1-2% increase in equivalent wealth over the myopic utility policy.
**Issues and Concerns**

Over the course of this project, we encountered a number of challenges, some of which are worth mentioning, and we do so in the section.

**Running Time**

In each time period the piecewise linear and best fit approximation methods require us to solve either a linear program or a nonlinear program for each of 1000 sample return paths. Since the GAMS model loops over all time periods and all wealth levels, it requires $15 \times 300 = 4500$ iterations to complete a run. With the best fit version of the program, we use 1000 sample return paths and 300 wealth levels. This amounts to solving nonlinear programs with 4 rows and 1000 columns, which is not difficult, but because of the large number of iterations, the entire program takes 2 to 3 hours to run.

With the piecewise linear version of the model, we use 300 simulations and 150 wealth levels. Because a separate variable is needed for every piecewise section in every scenario, this results in a linear program in the order of 300 rows and 90,000 columns. Since we are solving this linear program over $15 \times 150 = 2250$ iterations, the entire procedure requires 2 days to run using CONOPT2 as our lp solver.

**Quality of Utility Approximations**

When we approximate the expected utility functions by fitting a curve, for both the restricted and general cases, we are running a non-convex nonlinear program. Therefore, we cannot be certain that the program is giving us a global optimum instead of local optimum. In fact, by finding the utility points and fitting the utility curve manually, using Excel Solver, we manage to obtain much better fits.

As for the piecewise linear approach, we are greatly limited in the number of wealth levels we can consider because this not only affects our number of iterations, but also the size of our linear programs as well. The effectiveness of the piecewise linear approximation depends greatly on the number of wealth levels we use. Using the same parameters as those used for the best fit version, we anticipate the program requiring approximately 5 days to complete a run. Since we were forced to limit the size of the problem, we consequently increased the undesirable effects of linearity mentioned earlier.

**Selected Initial Utility Function**

One may note that, while we approximated the utility curves for wealth levels ranging from $10,000$ to $1,505,000$, we only evaluated the optimal policy for wealth levels up to $600,000$. There are two related reasons for this.
(1) Regardless of the returns over the fifteen year period, one may easily achieve returns amounting to 80% of the original investment principle. Hence, any wealth level larger than $556,000 will achieve the retirement goal even if 100% is held in cash for all fifteen periods.

(2) Because the function asymptotes at 5, our initial utility function may limit the accuracy of our policy for the highest wealth levels, since we effectively hit a ceiling as we approach retirement. Therefore, perhaps a different functional form or a different selection of parameters would improve the robustness of our stochastic dynamic program for selecting an optimal policy for higher wealth levels.

**Future Work**

Throughout the project we have made note of certain simplifying assumptions, and how they have influenced our data and programming methodology. To expand on our results and to gain a more complete understanding of the mathematical framework of this stochastic dynamic programming problem, we might reject one or more of those initial assumptions, or modify them in important ways.

A particularly restrictive assumption lies in designating the final wealth goal to be $1,000,000. The optimal policies that we generated prescribed that individuals with low initial wealth invest in all stocks, simply because the arbitrary goal is otherwise unattainable. A better approach might be to consider instead a continuous, monotonically nondecreasing function of final wealth goals. Such an approach has intuitive appeal; in practice, we would not expect investors with low initial wealth to seek a final goal of $1,000,000. Likewise, we expect wealthy investors to be more ambitious in their long-term financial goals. Instead of an all-stock policy, we expect that lower, more reasonable retirement goals would result in a more diversified portfolio, thereby yielding more insight into the precise relationship among initial wealth, time until retirement, and optimal investment strategy.

We might also allow an individual to contribute either a fixed or monotonically increasing amount of her earnings at the beginning of each year to her investment principle. One might expect this assumption to result in more wealth allocated to risky assets, as the promise of yearly income would dampen the blow of poor asset returns.

Another factor in presenting an accurate and practical formulation of this problem may be to consider how risky assets behave in the real world. Our limiting the problem to considering only the four asset classes may be too restrictive. How might optimal policies distribute wealth among different available stocks, with various mean returns and variances, beyond merely allocating to stocks in general? We might wish to allow options, futures, or other financial instruments to enter our allocation problem.

Furthermore, assuming the asset returns in each period are independent may significantly hinder the power of our stochastic solution. The relationship between past performance and future returns of assets is an important consideration for investors, and incorporating this causality into
our program may provide added value. With returns that are correlated over time, we may anticipate expected utilities that achieve stronger improvements over various myopic strategies.

We also wish to explore the optimal policies resulting from the selection of different initial utility functions. The exponential curve used in this project presented the constraint of a horizontal asymptote, and, as mentioned previously, this feature may affect the reliability of model results for the highest wealth levels. Future work might investigate a variety of curve forms, as well as a variety of parameter values.
Appendix A: zd.gms

*------------------------------------------------
*This program evaluates the optimal deterministic
*policy in a stochastic environment. The policy
*of choosing all stocks is fixed, and 1000 random
*returns are generated to determine how well the
*all-stock policy does
*------------------------------------------------

$offsymlist;
$offsymxref;
$offuellist;
$offlisting;

*-----------------------------
*File to which data is written:
*-----------------------------

FILE out /zd.tab/;
put out;
out.pw = 255;

*----------
*Declarations:
*----------

set j assets /Stocks, CorpBonds, GvmtBonds, Cash/;
set t periods /1*15/;
set i wealthlev /1*12/;
scalar wealth /.50000/;
parameter rbar(j) means /Stocks = 10.5, CorpBonds = 8.5, GvmtBonds = 7, Cash = 4/;
scalar runs /1000/;
alias (j,jj);
set l /1*1000/;

*------
*Table:
*------

  chol(j,jj)  cholesky factorization of covariance matrix
              Stocks  CorpBonds  GvmtBonds  Cash
              Stocks   15.500   4.689    2.673   0.000
              CorpBonds  0.000  10.937    7.472   0.000
              GvmtBonds  0.000   0.000    2.063   0.000
              Cash      0.000   0.000    0.000   0.100

variable normr(j,t);
variable return(t);
variable capital;
variable utility;
variable z;

z.l = 0;
Over varying wealth levels, the all-stock policy is tested over 1000 random returns. The average of the resulting utilities is the pertinent value of zd.

```
loop (i,
    put //;
    put "Wealth level is ";
    put wealth;
    put //;

    loop (l,
        normr.l(j,t) = normal(0,1);
        return.l(t) = 1+(rbar(' Stocks') + sum(jj,chol(' Stocks',jj))*normr)
        capital.l = prod(t, return.l(t)) * wealth;
        utility.l = 5 - 50 * .8 ** (capital.l);
        z.l = z.l + utility.l;
    );

    put "Average Utility is ="
    z.l = z.l / runs;
    put z.l;
    put //;

    wealth = wealth + .5;
);
```
Appendix B: evpi.gms

*------------------------------------------------------
*This program evaluates the "wait-and-see" solution.
*It samples 1000 sets of 15 random return vectors and
*chooses an optimal policy assuming one has perfect
*forsight.
*------------------------------------------------------

$offsymlist;
$offsymxref;
$offuellist;
$offlisting;

*-----------------------------------
*File to which the data is written:
*-----------------------------------

FILE out /output3.tab/;

put out;
out.pw = 255;

*-----------
*Declarations:
*-----------

set i wealthlev /1*12/;
set j assets /Stocks, CorpBonds, GvmtBonds, Cash/;
set t periods /1*15/;
scalar wealth /.50000/;
parameter rbar(j) means /Stocks = 10.5, CorpBonds = 8.5, GvmtBonds = 7, Cash = 4/;
scalar runs /1000/;
alias (j,jj);
set l /1*1000/;

*-------
*Table:
*-------

chol(j,jj) cholesky factorization of covariance matrix

<table>
<thead>
<tr>
<th></th>
<th>Stocks</th>
<th>CorpBonds</th>
<th>GvmtBonds</th>
<th>Cash</th>
</tr>
</thead>
<tbody>
<tr>
<td>Stocks</td>
<td>15.500</td>
<td>4.689</td>
<td>2.673</td>
<td>0.000</td>
</tr>
<tr>
<td>CorpBonds</td>
<td>0.000</td>
<td>10.937</td>
<td>7.472</td>
<td>0.000</td>
</tr>
<tr>
<td>GvmtBonds</td>
<td>0.000</td>
<td>0.000</td>
<td>2.063</td>
<td>0.000</td>
</tr>
<tr>
<td>Cash</td>
<td>0.000</td>
<td>0.000</td>
<td>0.000</td>
<td>0.100</td>
</tr>
</tbody>
</table>

variable normr(j,t);
variable return(j,t);
variable maxret(t);
variable capital
variable utility;
variable z;

scalar count /0/;
z.l = 0;
*We produce a "wait-and-see" optimal solution over varying wealth levels \( i \):*

```plaintext
loop (i,
    put //;
    put "Wealth level is ";
    put wealth;
    put //;
)
```

*Sample of 1000 random asset returns; choose the maximum in each period;
*Multiply all of them to find the total return for 15 periods:

```plaintext
loop (l,
    normr.l(j,t) = normal(0,1);
    return.l(j,t) = 1+(rbar(j) + sum(jj,chol(j,jj)*normr.l(jj,t)))/100;
    maxret.l(t) = max(return.l(' Stocks',t), return.l(' CorpBonds',t), re
    capital.l = prod(t, maxret.l(t)) * wealth;
    utility.l = 5 - 50 * .8 ** (capital.l);
    z.l = z.l + utility.l;
    put "Average Utility is = ";
    z.l = z.l / runs;
    put z.l;
    put //;
    wealth = wealth + .5;
)
```
Appendix C: sdp.gms

$onempty;
$offlisting;
$offsymxref;
$offuellist;

option iterlim = 100000;
option limcol = 0;
option limrow = 0;
option solprint = off;
option reslim = 100000;
option decimals = 6;

* This program starts of with an exponential function but does the best fit on -
a + b*w + c*w^d + e*f^g + h*log(i*w) + j*w^k + l*w^m + n*o^p*w

* This program will give the following output -
* OptPol1.tab - The optimal policy by year and wealthlevels
* p0.tab - p15.tab - The parameters for the utility function at every time period
* sdppolicy.tab - The optimal policy in a GAMS readable form
* Utility.tab - The utilities used to fit the curves at every time period

*----------
set
*----------
j             assets          /Stocks, CorpBonds, GvmtBonds, Cash/
t             time period             /0*15/
twz(t)        time periods without 0
m             simulation runs         /1*1000/
level         wealth levels           /1*300/;

  twz(t) = yes$(ord(t) > 1);
  alias (j,jj);

*---------

  table
*---------

   chol(j,jj)  cholesky factorization of covariance matrix
       Stocks   CorpBonds   GvmtBonds   Cash
    Stocks   15.500      4.689      2.673      0.000
    CorpBonds 0.000     10.937     7.472      0.000
    GvmtBonds 0.000      0.000     2.063      0.000
     Cash    0.000      0.000      0.000     0.100

*---------

  parameter
*---------

  policy(j, t, level)  The policy in percentages at period t and wealth
  util(level)      The utility at a certain level of wealth
* This utility parameter stores the utility points after each iteration
* It's used mostly for debugging
utility( t, level )
    ahist( t )
    bhist( t )
    chist( t )
    dhist( t )
    ehist( t )
    fhist( t )
    ghist( t )
    hhist( t )
    ihist(t)
    jhist(t)
    khist(t)
    lhist(t)
    mhist(t)
    nhist(t)
    ohist(t)
    phist(t)

goal Amount we would like to achieve /10/
wealthstep Difference between each wealth level
r2 R squared value of the best fit
rbar(j) mean returns
    /Stocks = 10.5, CorpBonds = 8.5, GvmtBonds = 7, Cash = 4 /
rstd(j) standard deviations of returns
    /Stocks = 15.5, CorpBonds = 11.9, GvmtBonds = 8.2, Cash = 0.1 /
wealthlev( level ) The amount of wealth at each level

* This is not a function of t, because we will update this vector every time we solve for a t.
initwealth initial wealth at time 0
nrml(m,j) sample out of a normal distribution for run m asset j
return(m,j) random return on simulation run m of asset j
wealth Amount of wealth we are considering
alpha a parameter of utility function > 0
beta b parameter of utility function > 0
cee c parameter of utility function < 1 and > 0
deef
eeff
ggeee
haitch
eye
jay
kay
elle
em
en
oh
pee;

* Include file that initializes the wealthlev variable.
$include C:\MY DOCUMENTS\SCHOOL\MSE348\wealthlevels.inc
* Run simulation and calculate the returns for every simulation run and asset
nrml(m,j) = normal(0,1);
return(m,j) = 1+(rbar(j) + sum(jj,chol(j,jj)*nrml(m,jj))/100;
Initialize the amount of wealth we are considering
this will change in the loop.

wealth = wealthlev("1");

Initialize our utility function to be exponential

alpha = 5;
beta = 0;
cee = 0;
dee = 0;
dee = -50;
eff = 0.8;
gee = 1;
haitch = 0;
eye = 1;
jay = 0;
kay = 0;
elle = 0;
em = 0;
en = 0;
oh = 0;
pee = 0;

ahist("0") = alpha;
bhist("0") = beta;
chist("0") = cee;
dhist("0") = dee;
ehist("0") = eee;
fhist("0") = eff;
ghist("0") = gee;
hhist("0") = haitch;
ihist("0") = eye;
jhist("0") = jay;
khist("0") = kay;
lhist("0") = elle;
mhist("0") = em;
nhist("0") = en;
ohist("0") = oh;
phist("0") = pee;

* positive variables

X(j) percentage of wealth going into asset j
NEXTW(m) Total wealth next period for each simulation;

* variable

AUTIL average utility
MUTIL(m) Utility of nextw(m) of scenario m;

* equations

* These are the equations for solving the main nonlinear program
totalone The sum of the X's should add up to 1
nextwealth(m) Wealth at end of this period for simulation m
calcutil Calc. average utility level of all scenarios m;
totalone.. sum(j, X(j)) =e= 1;
nextwealth(m).. NEXTW(m) =e= wealth * sum(j, X(j)*return(m, j));

* Average utility the sum of all the utilities divided by the total number
calcul.. AUTIL =e= (1/card(m))* (sum(m, alpha + beta*NEXTW(m) + cee*NEXTW(m)**dee + eee*(eff**gee*NEXTW(m)) + haitch*log(eye*NEXTW(m)) + jay*NEXTW(m)**kay + elle*NEXTW(m)**em + en*(oh**(pee*NEXTW(m))));

* These are the variables and equations for the best fit problem
* ----- positive variable
* ------ B C G H I L P;
*-------variable
*------- A alpha D E F JAE K MM N O TOTERR sum of all squared error terms;
*------- equations
*-------

* These are the equations for solving the best fit
sqrsum.. TOTERR =e= sum(level, sqrt( util(level) ) - (A + B*wealthlev(level) + C*wealthlev(level)**D + E*(F**G*wealthlev(level)) + H*log( I*wealthlev(level)) + JAE*wealthlev(level)**K + L*wealthlev(level)**MM + N*( O**( P*wealthlev(level))));

* Set bounds on some of the variables
* Initialize X so we won't start with log(0) in the equations
D.up = 1;
K.up = 1;
MM.up = 1;
X.l(j) = 0.25;
NEXTW.l(m) = 10000;

* Now we will run an iteration over all possible time periods
model mainmod / totalone, nextwealth, calcult/;
model bestfit / sqrsum/;
loop( twz,

    loop( level,
        solve mainmod maximizing AUTIL using nlp;
        display X.l;
    *
        * Set values for policy
        policy( j, twz, level ) = X.l(j);
        util( level ) = AUTIL.l;
        utility( twz, level ) = AUTIL.l;
    *
        * Update wealth
        wealth = wealthlev( level + 1 );
        );
    *
        * Reinitialize wealth to first level for the first loop
        wealth = wealthlev( "1" );
    *
        * Do a best fit for the parameters alpha and beta for the next utility function
        A.l = alpha;
        B.l = beta;
        C.l = cee;
        D.l = dee;
        E.l = eee;
        F.l = eff;
        G.l = gee;
        H.l = haitch;
        I.l = eye;
        JAE.l = jay;
        K.l = kay;
        L.l = elle;
        MM.l = em;
        N.l = en;
        O.l = oh;
        P.l = pee;
        solve bestfit minimizing TOTERR using nlp ;
    *
        * Update alphas and betas
        alpha = A.l;
        beta = B.l;
        cee= C.l;
        dee= D.l;
        eee = E.l;
        eff = F.l;
        gee = G.l;
        haitch = H.l;
        eye = I.l;
        jay = JAE.l;
        kay = K.l;
        elle = L.l;
        em = MM.l;
        en = N.l;
        oh = O.l;
        pee = P.l;
        ahist( twz ) = alpha;
        bhist( twz ) = beta;
        chist( twz ) = cee;
        dhist( twz ) = dee;
        ehist( twz ) = eee;
    )
)
fhist(twz) = eff;
ghist(twz) = gee;
hhist(twz) = haitch;
ihat(twz) = eye;
jhist(twz) = jay;
khist(twz) = kay;
lhist(twz) = elle;
mhist(twz) = em;
nhist(twz) = en;
ohist(twz) = oh;
phist(twz) = pee;

* Update returns
nrml(m,j) = normal(0,1);
return(m,j) = 1+(rbar(j) + sum(jj,chol(j,jj)*nrml(m,jj))/100);

FILE out /OptPol1.tab/;

put out;
out.pw=255;

PUT //;
loop(t,
put "Year " t.tl;
put /
put " Wealth Stocks CBonds";
put " GBonds Cash";
put /
loop(level,
  put wealthlev(level)," policy("Stocks",t,level)," ";
  put policy("CorpBonds",t,level)," policy("GvmtBonds",t,level);
  put policy("Cash",t,level);
  put /
);

file p0 / Params0.tab/;
file p1 / Params1.tab/;
file p2 / Params2.tab/;
file p3 / Params3.tab/;
file p4 / Params4.tab/;
file p5 / Params5.tab/;
file p6 / Params6.tab/;
file p7 / Params7.tab/;
file p8 / Params8.tab/;
file p9 / Params9.tab/;
file p10 / Params10.tab/;
file p11 / Params11.tab/;
file p12 / Params12.tab/;
file p13 / Params13.tab/;
file p14 / Params14.tab/;
file p15 / Params15.tab/;

C-6
loop( t,
    if( ord(t) = 1,
        put p0;
        put "Parameters for T = 0";
    elseif ( ord(t) = 2),
        put p1;
        put "Parameters for T = 1";
    elseif ( ord(t) = 3),
        put p2;
        put "Parameters for T = 2";
    elseif ( ord(t) = 4),
        put p3;
        put "Parameters for T = 3";
    elseif ( ord(t) = 5),
        put p4;
        put "Parameters for T = 4";
    elseif ( ord(t) = 6),
        put p5;
        put "Parameters for T = 5";
    elseif ( ord(t) = 7),
        put p6;
        put "Parameters for T = 6";
    elseif ( ord(t) = 8),
        put p7;
        put "Parameters for T = 7";
    elseif ( ord(t) = 9),
        put p8;
        put "Parameters for T = 8";
    elseif ( ord(t) = 10),
        put p9;
        put "Parameters for T = 9";
    elseif ( ord(t) = 11),
        put p10;
        put "Parameters for T = 10";
    elseif ( ord(t) = 12),
        put p11;
        put "Parameters for T = 11";
    elseif ( ord(t) = 13),
        put p12;
        put "Parameters for T = 12";
    elseif ( ord(t) = 14),
        put p13;
        put "Parameters for T = 13";
    elseif ( ord(t) = 15),
        put p14;
        put "Parameters for T = 14";
    elseif ( ord(t) = 16),
        put p15;
        put "Parameters for T = 15";
    );
    put //;
    put "alpha =" ahist(t):0,";"/;
    put "beta =" bhist(t):0,";"/;
    put "cee =" chist(t):0,";"/;
    put "dee =" dhist(t):0,";"/;
    put "eee =" ehist(t):0,";"/;

C-7
put "eff =" fhist(t):0;
put "gee =" ghist(t):0;
put "haitch =" hhist(t):0;
put "eye =" ihist(t):0;
put "jay =" jhist(t):0;
put "kay =" khist(t):0;
put "elle =" lhist(t):0;
put "em =" mhist(t):0;
put "en =" nhist(t):0;
put "oh =" ohist(t):0;
put "pee =" phist(t):0;
);

FILE out2 /sdppolicy.tab/;
put out2;
out.pw=255;

* This loop will output the policies in the form of a GAMS readable file
loop( t,
    loop( level,
        loop( j,
            put "policy('", j.tl:0,"',", t.tl:0,"',", level.tl:0") = "
            put policy(j,t,level):0;
            put /
        );
    );
);

FILE out4 /Utility.tab/;
put out4;
out.pw=255;
loop( t,
    loop( level,
        put "utility('", t.tl:0,"',", level.tl:0") = "
        put utility( t, level ):0;
        put /
    );
);

C-8
Appendix C: Params0.tab

*Parameters for T = 0

alpha = 5.00;
beta = 0.00;
cee = 0.00;
dee = 0.00;
dee = -50.00;
eff = 0.80;
gee = 1.00;
haitech = 0.00;
eye = 1.00;
jay = 0.00;
kay = 0.00;
elle = 0.00;
em = 0.00;
en = 0.00;
oh = 0.00;
pee = 0.00;
Appendix D: evaluate.gms

$offlisting;
$offuellist;
$offsymxref;
$offsymlist;

* This is a new eval program. This program compares the
* out-of-sample performance of our policy to two myopic policies.
* The first myopic policy uses the output of mean-variance analysis.
* The second myopic policy maximizes the expected value of the same
* utility function used to develop our policies

*--------
set
*--------
t               number of periods until retirement /0*15/
i               wealth levels /1*300/
itemp(i)        temporary subset of i
j               assets /Stocks,CorpBonds,GvmtBonds,Cash/;
alias(j,jj);
alias(i,ii);

*--------
table
*--------
chol(j,jj)    cholesky factorization of covariance matrix
Stocks  CorpBonds  GvmtBonds  Cash
Stocks      15.500      4.689      2.673      0.000
CorpBonds   0.000     10.937      7.472      0.000
GvmtBonds   0.000      0.000      2.063      0.000
Cash        0.000      0.000      0.000      0.100
;

*--------
parameter
*--------
rbar(j)  mean returns
       /Stocks = 10.5
       GvmtBonds = 7
       CorpBonds = 8.5
       Cash = 4
       /
xmv(j)   mean-variance allocations
       /Stocks = .010849
       GvmtBonds = 0.015768
       CorpBonds = 0.00
       Cash = 0.973384
       /
utilx(j,i) second myopic policy using second utility
norm(j)    standard normal variates
return(j)  random return of asset j
policy(j,t,i) our policy from model output 2
closelev   closest whole number wealth level
wealthlev(i) wealth levels given in wealthlevels.inc;
$include /afs/ir.stanford.edu/users/h/l/hlutze/wealthlevels.inc
$include /afs/ir.stanford.edu/users/h/l/hlutze/sdppolicy.out

utilx(j,i) = policy(j,'1',i);

*-------
variables
*-------
x(j,t,i) asset allocation for second utility
xutil(j,i) myopic asset allocation for second utility
wealth(i) initial wealth index under second utility
wealthmv(i) initial wealth index for mean variance myopic
wlthutil(i) initial wealth index under second utility myopic
w beginning of period wealth for utility
wmv beginning of period wealth for mean variance
wutil beginning of period wealth for utility myopic
nw end of period wealth for utility
nwmv end of period wealth for mean variance
nwutil end of period wealth for utility myopic;

*-------
* scalars
*-------
scalar z counter for wealth levels;
scalar ww index for wealth level utility;
scalar wwutil index for wealth level utility myopic;
scalar count reverse counter for time periods to go /15/;
scalar iter iterate 1000 times;

* The number in the output file name is the same as z. This number simply
* represents the initial wealth level.

FILE out /evalout119.tab/;
pout out;
out.pw=255;

for(iter = 1 to 1000 by 1,

    z = 119;

    norm(j) = normal(0,1);
    return(j) = 1 + rbar(j)/100 + sum(jj, chol(j,jj) * norm(jj))/100;
    itemp(i) = yes$(ord(i)<=z);
    ww = card(itemp);
    wealth.fx(ii)$((ord(ii) = ww) = wealthlev(ii);
    wealthmv.fx(ii)$((ord(ii) = ww) = wealthlev(ii);
    wlthutil.fx(ii)$((ord(ii) = ww) = wealthlev(ii);
    w.l = sum(i,wealth.l(i));
    wmv.l = sum(i,wealthmv.l(i));
    wutil.l = sum(i,wlthutil.l(i));
    wealth.fx(ii)$((ord(ii) = ww) = 0;
    wealthmv.fx(ii)$((ord(ii) = ww) = 0;
    wlthutil.fx(ii)$((ord(ii) = ww) = 0;
x.fx(j,'15',i)$(ord(i) = z) = policy(j,'15',i);
xutil.fx(j,i)$(ord(i) = z) = utilx(j,i);

nw.l = w.l * sum(i,sum(j, return(j)*x.l(j,'15',i)));
nwmvl = w.l * sum(j, return(j)*xmv(j));
nwutil.l = w.l * sum(i,sum(j, return(j)*xutil.l(j,i)));

put "Level":z;
put /;
put w.l,wmv.l,wutil.l;
put /;
put nw.l,nwmvl,nwutil.l;
put /

x.fx(j,'15',i)$(ord(i) = z) = 0;
xutil.fx(j,i)$(ord(i) = z) = 0;

w.l = nw.l;
wmv.l = nwmvl;
util.l = nwutil.l;

while((count ge 2),

  norm(j) = normal(0,1);
  return(j) = 1 + rbar(j)/100 + sum(jj, chol(j,jj) * norm(jj))/100;
  ww = round(((w.l)-0.05)/0.05);
  wwutil = round(((wutil.l)-0.05)/0.05);
  if(ww>300,
    ww=300;
  );
  if(wwutil>300,
    wwutil=300;
  );
  x.fx(j,t,i)$((ord(t)=count)and(ord(i)=ww)) = policy(j,t,i);
xutil.fx(j,i)$((ord(i)=ww)) = utilx(j,i);

  nw.l = w.l * sum(t,sum(i,sum(j, return(j)*x.l(j,t,i))));
nwmvl = wmv.l * sum(j, return(j)*xmv(j));
nwutil.l = wutil.l * sum(i,sum(j, return(j)*xutil.l(j,i))));

put nw.l,nwmvl,nwutil.l;
put /;
loop(j,
  x.fx(j,t,i)$((ord(t)=count) and (ord(i)=ww)) = 0;
xutil.fx(j,i)$((ord(i)=ww)) = 0;
);
w.l = nw.l;
wmv.l = nwmvl;
util.l = nwutil.l;

  count = count - 1;
);

count = 15;