Asset Aggregation in Stochastic Programming Models for Asset Liability Management

Fredrik Altenstedt
Department of Mathemathics
Chalmers University of Technology
412 96 Göteborg, Sweden

October 10, 2003

Abstract

Multi-stage stochastic linear programming is emerging as a valuable tool for addressing asset liability management problems (ALM problems). A number of researchers have demonstrated that the method used to generate the asset return scenarios used in the optimization has a major impact on the overall performance of the model. Less effort has been spent on trying to determine which assets should be used in the model. We explore the option of constraining the ALM model to use only a few linear combinations of the assets at hand, and to find the best linear combinations by optimization. Our hypothesis is that using such linear combinations may increase the performance of the model when the solution found is applied to out-of-sample scenarios. We believe our approach may be beneficial, as stochastic programming solutions have a tendency to chase spurious profits present only in the scenario tree; using artificial asset combinations should reduce this tendency, since fewer assets imply fewer spurious profit opportunities when the size of the scenario tree is held constant. We perform a number of tests on a simplified ALM problem, which show that restricting ourselves to use only a few linear combinations of assets does indeed improve performance. In fact, there is no statistically significant benefit of using more than two asset combinations for the model tested. This is the lowest number of synthetic assets we may have, while still retaining the ability to change the risk-reward trade-off.

Keywords: Stochastic programming; Asset liability management; Optimization; Finance.

1 Introduction

1.1 Background

When managing financial assets, the obvious goal of a manager is to maximize the yield of the funds invested, while keeping the investment risk low. Usually, the measure of risk is assumed to be static, that is, the goal of the investments does not change over time. However, for a large subset of investment problems, the yield of the investments is used to cover some form of liabilities. When this is the case, advantages may be gained by managing assets and liabilities in a coordinated fashion, resulting in an *asset liability management* (ALM) problem. For instance, the management of pension funds, where contributions from the fund's sponsor and yields from investments are supposed to cover future retirement liabilities, may be formulated as ALM problems. As the liabilities are dependent on the future inflation and wages of the beneficiaries of the fund, the target of the fund is uncertain and correlated with the development of the assets in which the fund invests.

Lately, stochastic programming has become a popular tool to tackle ALM problems. Stochastic programming solutions are dynamic in the sense that they explicitly account for the fact that the investment policy of the company will change when circumstances change. Taking these dynamics into

account makes it possible to outperform static methods, such as the Markowitz portfolio optimization technique (which assumes that the investment policy will not change during the horizon considered). In the literature there are several examples of stochastic programming models used for asset liability management. For instance, Bradley and Crane [3] describe a model used for managing bond portfolios as early as 1972; other models are described in Cariño et al. [5, 4, 6], who constructed an ALM model to aid a Japanese insurance company, and Dempster and Consigli [7], who applied multi-stage SP to the problem of administering assets for a retirement fund. Dert [9] and Kouwenberg [17] have done the same in a Dutch setting, whereas Høyland [13] have treated the case of pension insurance, whose ALM problem has a slightly different characteristic. Other applications include index tracking, which have been addressed by Zenios et al. [21], while Mulvey and Vladimirou [18] treat more general asset management problems.

1.2 Motivation

In all the models mentioned above, except the one in Bradley and Crane [3] and Zenios et al. [21], stochastic programming is used for strategic asset allocation, where available funds are allocated between broad categories of assets, categories such as stocks, bonds, real estate and treasury bills. In these models the assets themselves are considered to be given; how they are selected is however not apparent. In this paper we will explore whether these classes of assets should directly be used in the stochastic programming model, or if benefits may be obtained by artificially creating other classes of assets, in effect reducing the space of possible investment strategies.

This type of approach has been investigated by Gaivoronski and de Lange [11]. They use an ordinary multi-stage stochastic programming model, with the addition that free trade is only allowed at stages 0 and 1. For all stages after stage 1 the fractions of wealth invested in different assets is required to be the same as in stage 1, giving a fix-mix strategy (with different asset proportions in different parts of the tree) According to their work, locking the assets in this fashion gave a better overall performance, compared to fully dynamic stochastic programming. Well worth noting is that when Gaivoronski and de Lange optimize the dynamic/fix-mix hybrid policy, they do this using a larger scenario tree than the one used for the fully dynamic solution. The reason for using different scenario trees is that the fully dynamic model is more computationally challenging, and the authors try to determine which approach is the best when the limiting factor is the available computational time. The main difference between our work and the work by Gaivoronski and de Lange is that they try to determine if fully dynamic solutions are worth the computational effort, whereas we try to determine if the fully dynamic solution may be improved by creating artificial constraints. We feel that the solution may be improved, as too many degrees of freedom in a stochastic programming model may cause the solution to adapt to peculiarities in the scenario tree representing the underlying distribution, rather than to the general properties of the underlying random distribution the scenario tree is supposed to reflect.

In order to carry out our investigations into the asset aggregation problem, we construct a bare-bones ALM problem. The problem is designed to be as simple as possible while still retaining the basic property of an asset liability problem; the goal is to invest means which are used to honor uncertain liabilities. Our interest in this problem stems from our wish to simplify our model of a Swedish life insurance company (described in Altenstedt [1]), but for simplicity we abstain from using the full model, as its complexity might hide the question we wish to treat.

1.3 Preview

In Section 2 we give a general description of how an ALM problem is structured. In Section 3 we classify the different simplifications that need to be made in order to make the general ALM-problem computationally tractable, and motivate why sometimes further reducing the already simplified problem might give solutions that are closer to the optimum of the unreduced problem. Two of the simplifications, namely the discretization of the distribution and the aggregation of assets, are discussed in Sections 3.4 and 3.5, respectively. We give a description of the simple ALM problem used in our tests in Section

4. In Section 5 we describe the tests performed and conclusions drawn in Section 6.

2 The canonical ALM problem

As mentioned above in Section 1.1, ALM models are designed to address the problem where funds are to be invested to cover future liabilities. In reality, there exist thousands upon thousands of possible assets between which an investor may divide their means, the price of which may change at any time. In addition, trading may be performed at each instance in time (with restrictions due to the closing of markets, etcetera). In order to simplify the formulation, we assume that decisions may only be taken at a discrete and finite set of times. As this set may be arbitrarily large we do not lose any significant generality. We further assume that the asset prices and other exogenous data such as liabilities are not influenced by the actions taken by the fund. Using these assumptions, we may formulate the full ALM problem faced by the fund managers as

$$\begin{aligned}
\min_{x_0} \quad & f_0(x_0) + \mathsf{E}^P_{\xi_1}[\min_{x_1(\xi_1)} f_1(\overrightarrow{x}_1(\xi_1), \xi_1) + \mathsf{E}^P_{\xi_2|\xi_1}[\min_{x_2(\xi_2)} f_2(\overrightarrow{x}_2(\xi_2), \xi_2) \\
& + \cdots + \mathsf{E}^P_{\xi_T|\xi_{T-1}}[\min_{x_T(\xi_T)} f_T(\overrightarrow{x}_T(\xi_T), \xi_T)] \cdots]], && \text{(1a)} \\
\text{s. t.} \quad & x_t(\xi_t) \in X_t(\overrightarrow{x}_{t-1}(\xi_{t-1}), \xi_t), \qquad t = 1, \ldots, T. && \text{(1b)}
\end{aligned}$$

where we introduce the following notation:

f_t	objective at time t (typically including penalties for undesirable conditions, and rewards for assets owned),
T	number of time-stages,
ξ_t	all random information available at time t,
$x_t(\xi_t)$	decision variables at time t (typically what is bought, sold and owned),
$\overrightarrow{x}_t(\xi_t)$	all decisions made up to and including time t, $\overrightarrow{x}_t(\xi_t) = (x_0, x_1(\xi_1) \ldots, x_t(\xi_t))$,
$X_t(\overrightarrow{x}_{t-1}(\xi_{t-1}), \xi_t)$	feasible set at time t dependent on random information and earlier decisions,
P	probability measure giving the probability of different outcomes.

In this formulation we have assumed that the decision at time t is taken directly after the random information at this time becomes known. All random variables in the problem are defined on a standard probability space, $(\Omega, \mathscr{F}, P)$. Here, Ω is the space of all possible outcomes, $\mathscr{F}$ is the algebra of all possible events on this space, generated by all possible outcomes of ξ^T, and P is a probability measure, giving a probability for all events in $\mathscr{F}$.

3 Reducing the full problem

3.1 Introduction

The full problem described above need to be further reduced in order to make it computationally tractable. The reduction of the problem may be done in at least four different ways:

1: The number of points in time when decisions can be made can be reduced, for instance by allowing the fund manager to change investments and other decisions fewer times per year. This category of simplifications also include reducing the total length of the problem by reducing the horizon.

2: The objective function may be changed, in order to make it easier to evaluate. For instance, a convex function may be replaced by a piecewise linear convex approximation.

3: The number of assets in which the fund may invest may be reduced, giving a lower number of variables. Removing assets from consideration may be seen as restricting the maximum owned amount to 0 for the removed assets. Hence removing assets simply makes the feasible set of the problem smaller. In the same fashion, aggregating several assets into wider aggregate assets, such as a stock index, may be accomplished by adding linear constraints to the model, fixing the proportions of different types of assets owned to predetermined values. We will refer to aggregating assets in this fashion as creating a *synthetic asset*, as the linear combination of a number of assets might be seen as yet another asset, which is bought and sold as an ordinary asset. Again, this aggregation results in a reduction of the feasible set.

4: The random distribution of possible outcomes may be simplified. Typically, this is done by discretizing the space of possible outcomes, reducing the possible outcomes to a scenario tree. As the probabilities of different possible outcomes are given by the probability measure P, discretizing the space of possible outcomes is the same as changing the measure P to a measure $\tilde{P}$, where the support of $\tilde{P}$ consists of finitely many outcomes.

In this work we concentrate on changes of type 3 and 4, and refer to a problem instance by the pair $\{X, P\}$. Here, X is the feasible set and P the probability measure defined over the space of possible outcomes Ω. Hence, changes of type 3 and 4 will affect X and P respectively. We illustrate the importance of looking at the two restrictions in tandem with a small example.

Consider a one stage investment problem, where the random yields are discretized using a number of scenarios (we assume that the actual distribution of the yields is known). In this problem there are many different stocks and bonds available to an investor. If the set of scenarios is not good enough to accurately describe all possible outcomes of the investments, there will exist investment opportunities that seem good under the probability measure defined by the scenarios ($\tilde{P}$), while performing worse under the true probability measure (P). If the investor exploits these opportunities, he/she might end up with an investment portfolio which performs well under the measure $\tilde{P}$, but badly under the measure P. If we restrict trade only to an index of bonds and another index of stocks, the investor no longer has an opportunity to exploit these false investment opportunities, and will choose a portfolio which will perform worse under the assumed probability measure $\tilde{P}$, but better under the real probability measure P.

3.2 Errors for two stage problems

If we initially restrict ourselves to two-stage versions of the problem (1), we may define the value of the recourse problem as

$$g(x_0, \xi) := \underset{x_1 \in X_1(x_0,\xi)}{\text{minimum}} f_1(x_0, x_1, \xi). \tag{2}$$

In this section we confine ourselves to make restrictions of the feasible set only at stage 0. Hence, the value of the recourse problem is not dependent on which feasible set is chosen. Using (2) we may define the objective function in the deterministic equivalent problem as

$$F(x_0, P) := f_0(x_0) + \mathsf{E}_\xi^P[g(x_0, \xi)] \tag{3}$$

We further define an optimal stage-0 solution when using the feasible set X and the measure P as

$$x_0^*(X, P) \in \underset{x_0 \in X}{\operatorname{argmin}} F(x_0, P). \tag{4}$$

For simplicity we assume that this solution is uniquely chosen whenever the optimal set is not a singleton.

In order to quantify the errors arising from changes of type 3 and 4 we introduce the notions of *restriction error* and *discretization error*. The discretization error is taken from Pflug [19], and it is defined as

$$d_X(P, \tilde{P}) := F(x_0^*(X, \tilde{P}), P) - F(x_0^*(X, P), P).$$

The value $d_X(P, \tilde{P})$ describes the loss from using the first-stage solution obtained by using the probability distribution $\tilde{P}$ instead of the optimal first-stage solution. (This corresponds to a change of type 4 above.) Note that the discretization error is dependent on which feasible set is used.

The restriction error gives the loss from restricting the first-stage feasible set from X to $\tilde{X} \subset X$ (a change of type 3 above). It is defined as

$$r_P(X, \tilde{X}) := F(x_0^*(\tilde{X}, P), P) - F(x_0^*(X, P), P);$$

by optimality, we know that the restriction error increases monotonically with stronger and stronger restrictions on X. Note that we must specify under which probability measure the expectations are taken.

Furthermore, we define the total error from making the two changes simultaneously as

$$a(X, \tilde{X}, P, \tilde{P}) := F(x_0^*(\tilde{X}, \tilde{P}), P) - F(x_0^*(X, P), P).$$

By adding and subtracting $F(x_0^*(\tilde{X}, P), P)$ in the right hand side above we see that the total error may be expressed as the sum of two errors:

$$a(X, \tilde{X}, P, \tilde{P}) = d_{\tilde{X}}(P, \tilde{P}) + r_P(X, \tilde{X}).$$

Note that we, in this expression, must use the discretization error measured using the restricted feasible set $\tilde{X}$. This means that we may actually get a smaller total error by restricting the feasible set from $\tilde{X}$ to $\bar{X}$, as long as an increase in the restriction error is compensated by a decrease in the discretization error

Note further that while the discretization error will increase monotonically with stronger and stronger restrictions of the feasible set, the discretization error has no no such monotonicity property. It is however reasonable to assume that the discretization error will decrease with stronger restrictions on the feasible set, as a smaller set reduces the possibility to exploit imperfections in the scenario tree. Hence the total error should (naïvely illustrated) look approximately like the curve in Figure 1.

3.3 Errors for multi-stage problems

When we move to three-stage problems, the situation becomes more complex. When a multi-stage stochastic programming model is used, it is used in a rolling fashion. One set of simplifications of the feasible set and probability measure is used to obtain the first-stage solution. Once the first random outcome is revealed, a new scenario tree is generated conditionally on this outcome, and the second stage decision is obtained, possibly using another restriction of the feasible set. The first-stage solution clearly is influenced by the choice of probability measure and the choice of feasible set for the first, second and third stages. The second stage solution is influenced by the first-stage solution, the probability measure chosen, and the feasible set for the second and third stages. Note that the second stage feasible set used to obtain the first-stage solution may be different from the second stage feasible set used to obtain the second stage solution. For multi-stage problems, the situation becomes even more complicated, and hence we abstain from deriving expressions for these errors, settling for a less rigorous approach.

If we look at the problem (1), we see that we may define the values of the recourse-problems recursively as

$$g_T(\overrightarrow{x}_{T-1}, \xi_T, X, P) := \min_{x_T \in X_T(\overrightarrow{x}_{T-1}, \xi_T)} f_T(\overrightarrow{x}_{T-1}, x_T, \xi_T) \tag{5a}$$

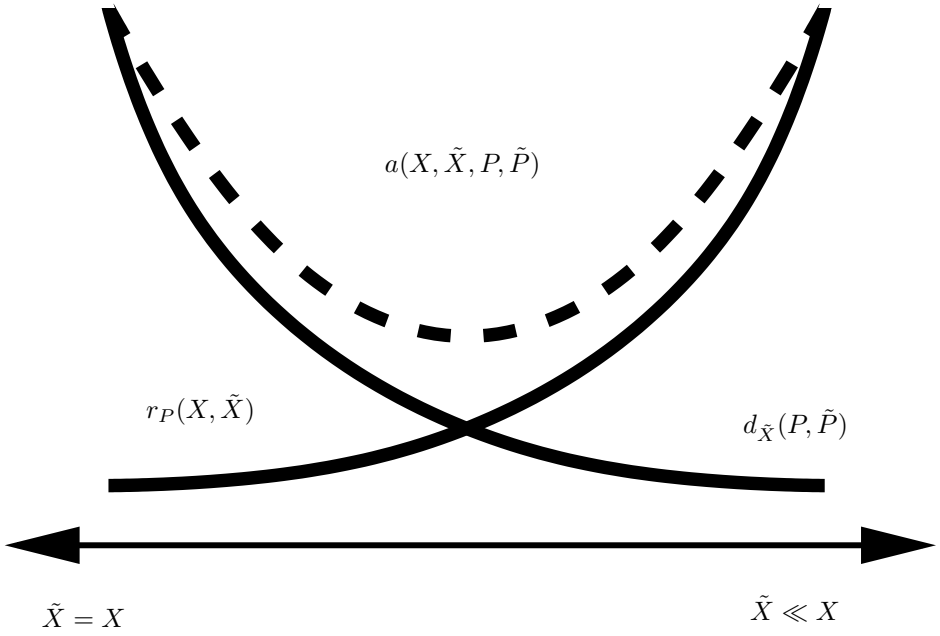

Figure 1: Changes in the restriction and discretization error.

$$g_t(\overrightarrow{x}_{t-1}, \xi_t, X, P) := \min_{x_t \in X_t(\overrightarrow{x}_{t-1}, \xi_t)} (f_t(\overrightarrow{x}_{t-1}, x_t, \xi_t) + \mathsf{E}^P_{\xi_{t+1}|\xi_t}[g_{t+1}(\overrightarrow{x}_{t-1}, x_t, \xi_{t+1}, X, P)]). \tag{5b}$$

We use this recursion to define the objective function of the deterministic equivalent problem as

$$F_0(x_0, X, P) := f_0(x_0) + \mathsf{E}^P_{\xi_1}[g_1(x_0, \xi_1, X, P)]. \tag{6}$$

In the previous section all errors in F stemmed from the probability measure used; the function g was assumed to be correctly specified. In a multi-stage problem this can no longer be true, as the choice of probability measure and feasible set will affect g_1. When we use the function F_0 to obtain a first-stage solution, we should choose our reduced set, $\tilde{X}$, and probability measure, $\tilde{P}$, so that $g_1(x_0, \xi_1, \tilde{X}, \tilde{P})$ is a good approximation of $g_1(x_0, \xi_1, X, P)$, since errors in the definition of g_1 will induce errors in the first-stage solution. In the same fashion, errors in g_2 will induce errors in g_1. Hence, at stage t, the size of the feasible set should match the quality of the scenario tree (i.e., the number of branches) at this stage if we are to obtain as good an estimate of g_t as is possible. As it is common to use scenario trees with a higher number of branches near the root, this indicates that the feasible sets should be smaller and smaller further down the scenario tree, in order for the feasible sets to match the branching of the scenario tree.

3.4 Discretization

If the space of possible outcomes Ω in (1) above is infinite, then so is the number of possible decisions $x_T(\xi_T)$ that need to be considered when solving this problem. As mentioned above in Section 3 the space of all possible random outcomes must be discretized in order to make the problem computationally tractable. Many discretization methods exist, ranging from simple ones such as random sampling

from the distribution (if the distribution is known) and random sampling augmented by variance reduction techniques (see Higle [12]), to complex schemes where the distance from the discretization $\tilde{P}$ to the probability measure P is minimized (see Pflug [19]). A good overview of different scenario tree generation techniques is given by Kaut and Wallace [16].

In this work we fit the scenario tree using optimization, as suggested by Høyland and Wallace [15], as Kouwenberg [17] has found this method to perform well without being overly complex.

3.5 Asset class selection

A fundamental part of the specification of an ALM model is the consideration of which assets or classes of assets to consider. A small number of assets will not give enough freedom to the model to find a good solution. The extreme case is a single asset, reducing the problem to simulating the development of a strategy consisting of periodically rebalancing the portfolio to a predetermined asset mix. Here, no optimization is done at all; the only feasible solution is the optimal solution. Towards the other end of the scale, adding a large number of assets will give the model more freedom, but requires us to make the scenario tree wider in order to capture the probability distributions of different investment outcomes accurately enough. A common requirement on a scenario tree is that it is free from arbitrage opportunities. In order to achieve freedom from arbitrage, the number of branches in the scenario tree at every point must be at least as many as the number of assets, accentuating the need for wider scenario trees when the number of assets increases.

Therefore, an important point of discussion when specifying our model is which assets or asset classes to choose. Traditionally, asset classes are aggregated according to category; one aggregate asset is created for stocks, one for bonds, one for real estate and so forth, possibly further divided by markets. This kind of division is used in the models described in [5, 4, 6], [13], [9], [18], and [7]. The rationale behind doing so is usually that it simplifies the problem statement when laws and regulations pose restrictions on the ownership of different asset classes. However, nothing stipulates that the optimal aggregate asset classes must consist of only one type of asset (e.g., stock or bond). Even if we wish to retain the rule of only one type of asset in an aggregate class, we still need to choose which sub-assets to aggregate, and in which proportions. We believe that gains may be made by taking into consideration not only the assets' internal correlation, but also the covariances of the aggregate classes and the correlation between aggregate classes and the liabilities. We return to this question in Sections 5.5 and 5.6.

4 A simplified ALM model

In order to test the ideas presented above, we construct a simple ALM problem. We imagine a fund manager required to cover a reserve by investing given funds in different asset classes. In addition, money flows in and out of the fund in a deterministic manner. If the funds are not sufficient to cover the reserve, a penalty is imposed. The penalty is progressive and has a number of levels (security factors). For instance, one level requires the funds to exceed 115% of the reserve, and a penalty is imposed proportionally whenever the total assets owned does not exceed this level. This allows us to specify the problem as follows:

Notation

$\hat{T}$	Time horizon.
$T = 0, 1, \ldots, \hat{T}$	Set of time-stages.
$t \in T$	Decision stage.
I	Set of asset classes.
Q	Set of penalty levels.

Variables

x_i^t	Amount of assets class i owned at time t.
y_i^{t+}	Amount of assets class i bought at time t.
y_i^{t-}	Amount of assets class i sold at time t.
z_q^t	Violation of penalty at level q at time t.
v^t	Total assets owned before trade at time t (no trading is performed at the last stage).

Parameters

p^t	Net payment inflow/outflow at time t.
γ_i	Transaction cost of asset i.
ρ_i^t	Price development of asset i from time $t-1$ to time t.
$\bar{x}_i$	Initial assets.
s_q^t	Penalty of level q at time t.
f_q	Security factor of level q.
R^t	Reserve requirement at time t.

The ALM model

$$\max \quad \mathsf{E}[v^T - \sum_{t\in T}\sum_{q\in Q} s_q^t z_q^t], \tag{7a}$$

$$\text{s. t.} \quad \sum_{i\in I}[y_i^{t-}(1-\gamma_i) - y_i^{t+}(1+\gamma_i)] = p^t, \qquad t \in T\setminus\{\hat{T}\}, \tag{7b}$$

$$y_i^{1+} - y_i^{1-} + \bar{x}_i = x_i^1, \qquad i \in I, \tag{7c}$$

$$y_i^{t+} - y_i^{t-} + \rho_i^t x_i^{t-1} = x_i^t, \qquad i \in I, t \in 2\dots\hat{T}-1, \tag{7d}$$

$$p^1 + \sum_{i\in I}\bar{x}_i = v^1, \tag{7e}$$

$$p^{t+1} + \sum_{i\in I}\rho_i^{t+1}x_i^t = v^{t+1}, \quad t \in 1\dots\hat{T}-1, \tag{7f}$$

$$z_q^t + v^t \geq f_q R^t, \quad t \in T, q \in Q, \tag{7g}$$

$$x, y, z \geq 0. \tag{7h}$$

In this problem, the objective function measures the total wealth at the terminal period minus the penalties incurred in all periods. The constraint (7b) is a cash balance constraint, guaranteeing that the net proceedings of purchases and sales equal the external inflow/outflow of capital. The asset development is modeled by (7c)–(7d) and states that the amount owned at a certain time is the amount owned at the previous time times the price development, to which trade is added. The amount of all assets owned before trade is given by the equations (7e)–(7f).

Finally, the equation (7g) states that we must cover the reserves by a certain margin, or face a penalty. Note that the last stage of the problem is simply an evaluation; no trade is done at this stage. As the restrictions of the feasible set consists of adding constraints on the assets owned after trade, no restrictions are made to the final stage, just as assumed when we derived the errors in Section 3.

4.1 Aggregating asset classes by constraints

As we wish to explore the effect of using a limited number of synthetic assets in problem (7), we explicitly add constraints to restrict which assets may be owned, creating a second version of the model described above. In this model we add constraints for all but one of the assets. The rationale

behind excluding one asset is that if we were to add constraints for all assets, we would get a set of linearly dependent constraints, making the problem infeasible if the constants c_{ik} do not sum to exactly 1 for all k. As all assets owned now will belong to a synthetic asset, the variables x_i^t are replaced by x_{ik}^t, giving the amount owned of basic asset i belonging to synthetic asset k at time t. The problem is further modified by adding the following notation:

K	The set of synthetic assets, in which we may invest.
$\hat{I}$	All basic assets except one. One asset is exempt from this set in order not to create linearly dependent constraints.
w_k^t	The total value held in synthetic asset k after trade at time t.
c_{ik}	The prescribed fraction of synthetic asset k which should be held in the basic asset i.

For a fixed value of c the model now becomes:

$$
\begin{aligned}
g(c) := \quad & \max \quad \mathsf{E}[v^T - \sum_{t\in T, q\in Q} s_q^t z_q^t], && && \text{(8a)}\\
\text{s. t.} \quad & \sum_{i\in I} y_i^{t-}(1-\gamma_i) - y_i^{t-}(1+\gamma_i) = p^t, && t\in T\setminus\{\hat{T}\}, && \text{(8b)}\\
& y_i^{1+} - y_i^{1-} + \bar{x}_i = \sum_{k\in K} x_{ik}^1, && i\in I, && \text{(8c)}\\
& y_i^{t+} - y_i^{t-} + \rho_i^t \sum_{k\in K} x_{ik}^{t-1} = \sum_{k\in K} x_{ik}^t, && i\in I, t\in 2\dots\hat{T}-1, && \text{(8d)}\\
& p^1 + \sum_{i\in I} \bar{x}_i = v^1, && i\in I, && \text{(8e)}\\
& p^{t+1} + \sum_{i\in I} \rho_i^t \sum_{k\in K} x_{ik}^t = v^{t+1}, && i\in I, t\in T\setminus\{\hat{T}\}, && \text{(8f)}\\
& z_q^t + v^t \geq f_q R^t, && t\in T, q\in Q, && \text{(8g)}\\
& w_k^t = \sum_{i\in I} x_{ik}^t, && t\in T\setminus\{\hat{T}\}, k\in K, && \text{(8h)}\\
& w_k^t c_{ik} = x_{ik}^t, && t\in T\setminus\{\hat{T}\}, i\in\hat{I}, k\in K, && \text{(8i)}\\
& x, y, z \geq 0. && && \text{(8j)}
\end{aligned}
$$

Here, the constraint (8h) aggregates the total assets owned in each of the synthetic assets and the constraint (8i) makes sure that the prescribed asset fractions are kept. As remarked above, we do not prescribe a fraction for the first basic asset, as prescribing fractions for all basic assets would give a set of linearly dependent constraints. Instead the first asset will absorb the funds not assigned to other assets, and hence a fraction of $1-\sum_{i\in\hat{I}} c_{ik}$ will be invested in the first basic asset in synthetic asset k.

Naturally, creating synthetic assets by adding extra constraints will make the problem larger and harder to solve. As an example, we may take a problem instance used later in our numerical experiments. In these, we have a 5-stage problem with 7 assets and 6400 scenarios. If we use no artificial assets, the size of problem (7) is 30,569 rows and 86,379 columns. If we add 3 artificial asset classes, the dimension of the problem increases to 67,886 rows and 116,588 columns. Note that the approach with adding constraints to define the linear combinations of assets is not entirely necessary. It would be perfectly possible to use problem (7) directly, substituting the assets I by a set of synthetic assets $\tilde{I}$, each synthetic asset having price development $\tilde{\rho}_i^t$. The constants c_{ik} would then be used to give the price development of the synthetic assets as

$$\tilde{\rho}_k^t := \sum_{i\in I} c_{ik}\rho_i^t$$

Further below, we show how the optimal value $g(c)$ [defined by (8)] might be differentiated with respect to c (under certain conditions). In the alternative formulation, we may differentiate $g(c)$ with

respect to $\tilde{\rho}$ and use basic calculus to obtain the derivatives with respect to c. The reason why we do not use this simplified formulation is twofold. Firstly, transaction costs will not be captured correctly in the reduced model formulation, as we may sell basic asset i as part of selling synthetic asset $k \in K$ while buying the same basic asset as a part of selling synthetic asset $\bar{k} \in K$. Secondly, for implementation reasons, the dual variables for the constraints (7d) and (7f) are not directly available when the model is solved using nested Benders decomposition, as these constraints span more than one time period. Both these concerns might be addressed, the first by formulating only the first stage using explicit constraints defining the linear asset combinations, and the second by improving the solver.

4.2 Optimization of synthetic assets

The problem (8) contains a number of parameters c_{ik} which define the synthetic assets. In order to determine a good set of synthetic assets, we consider the optimal value of problem (8) to be a function of these parameters and formulate the problem to

$$\begin{aligned} \underset{c}{\text{maximize}} \quad & g(c), && \text{(9a)} \\ \text{subject to} \quad & \sum_{i \in \hat{I}} c_{ik} \leq 1 - \epsilon \quad k \in K, && \text{(9b)} \\ & c_{ik} \geq 0 \qquad\quad i \in \hat{I}, k \in K. && \text{(9c)} \end{aligned}$$

The ϵ perturbation is present to avoid numerical difficulties. If the fractions of c sum to something larger than 1, the problem (7) will become infeasible, and if they sum to exactly 1 the constraints of (7) will become linearly dependent. In order to avoid these situations, we set ϵ to something a couple of orders of magnitude larger than the machine precision. Perturbing the constraint (9) in this fashion is equivalent to setting a lower bound of ϵ on the fraction of the available means that is invested in the first asset, something that will have a minor impact on our results as long as ϵ is small.

Note that the values of c are considered to be parameters in (8) but variables in (9). In order to optimize (9) with a gradient descent method, we need to differentiate $g(c)$ with respect to c, that is, differentiate the optimal value of (8) with respect to changes to elements in the constraint matrix. As is noted in Dantzig and Thapa [8] in Section 7.6, a derivative of the optimal value for the problem

$$\begin{aligned} \text{minimize} \quad & z = d^t x, && \text{(10a)} \\ \text{subject to} \quad & Ax = b, && \text{(10b)} \\ & x \geq 0, && \text{(10c)} \end{aligned}$$

with respect to the matrix coefficient A_{ij} is given by

$$\frac{\partial z^*}{\partial A_{ki}} = -\pi_k^* x_i^*,$$

subject to certain non-degeneracy conditions. In this expression, x^* and π^* denotes the optimal primal and dual solution solution to (10), respectively.

In order to optimize (9) we use a gradient projection approach combined with an Armijo step-length rule (see Bertsekas [2]) and terminate when the norm of the projected gradient of $g(c)$ falls below a certain threshold. We obtain the value of the gradient from the expressions above. In order to avoid being caught in a local minimum, the optimization procedure is restarted from a number of random positions.

5 Numerical tests

5.1 Questions

As mentioned previously, we would like to explore whether aggregating assets in different ways has a significant impact on the performance of our model, when the obtained solutions are applied to out-of-sample scenarios. The questions we pose are:

- If we reduce the number of assets allowed for trading, how many synthetic assets are appropriate?
- When assets are aggregated into wider classes, should the aggregation be applied for all stages or should the first stage be excluded?
- If we reduce the number of assets by aggregation, should this reduction be made as a partitioning, or should a single asset be allowed to belong to several classes of aggregate assets?
- When we are aggregating assets into synthetic assets, do we benefit significantly from using the procedure described in Section 4.2 compared to simply using Markowitz's method?

In the second question we only consider exempting one stage from the requirement to invest in the linear combinations. The reason for using linear combinations is to prevent the stochastic programming solution from chasing spurious profits present only in the scenario tree, resulting in bad solutions when the full underlying random distribution is considered. In the scenario trees used in this work as well as in others (see for instance Dempster and Consigli [7], and Høyland and Wallace [14]), the first stage has a larger number of branches than the rest of the scenario tree. The first stage is given a greater number of branches as the decision made in this node will be used, while the only value of solutions in later nodes are the effect these solutions have on the first stage. This means that the SP solution should have a lower tendency to chase spurious profits in the first stage of the scenario tree, and hence that it may prove beneficial not to restrict that stage.

5.2 Testing environment

In order to answer our questions, we construct a micro-world, in which there exists only 7 asset classes, which are to be used to cover one reserve requirement. The correlations and expected values for these assets are given in Table 4 and 5 in Appendix A. The correlations are taken from real-world data of Swedish assets, whereas the means are adjusted, as the rather short (10 year, 1990–2000) data sample used had higher yields than experience indicates are reasonable. (For instance, Swedish stocks yielded an average of 19% per year during the period 1990–2000, in contrast to the long term value of 9.5% for the period 1918–1990 as given by Frennberg and Hansson in [10].)

For this study, we assume that the assets and the reserves have a jointly log-normal distribution. (This assumption serves the purpose to simplify the scenario generation process.)

Scenario tree generation In order to reduce the spurious differences between different methods caused by badly specified scenario trees, we optimize the trees to fit the statistical properties of the underlying distribution. This fitting is done using Matlab, giving us a large set of optimized set of outcomes with 40, 20, 16, 10 and 4 members. When the number of descendants is larger than or equal to 16, we fit the four lowest moments, as well as the covariances between the assets. When 10 descendants are used, the three lowest moments and the covariances are fitted. With only 4 descendant nodes, the two lowest moments are fitted. A tree is then constructed by randomly picking from these sets of outcomes, to generate a tree of the desired size. If we are to have more than 40 branches in one node, we combine a number of sets of optimized outcomes. We may simplify the scenario generation process in this fashion since we assumed that the distribution of the asset yields is stationary, and hence that this distribution does not change from node to node.

Rolling horizon simulations In order to test whether there is a significant difference between different asset aggregation methods, we conduct rolling horizon simulations. In rolling horizon simulations, the effect of actually using the optimal solution of our ALM problem is investigated, using simulated scenarios. In these simulations, we use a time horizon of 4 years. The process works by generating a number of test scenarios using random sampling from the distribution of the random parameters. For each of these scenarios, we apply the following procedure:

0: Set $t = 0$ and go to step 2.

1: Use the sample-path and the state of the company just after the decision at time $t-1$ to generate the state of the company just before a decision is made at time t.

2: Use the state of the world of the current sample-path at time t to generate a scenario tree. The length of the scenario tree is adapted to last to the end of the simulation.

3: Optimize over the tree, generating a decision for the company at time t. Use this decision to determine the state of the company after the decision is made at time t. Store information of the state of the company just after the decision.

4: If $t < T$ then set $t := t + 1$ and go to 1.

5: Use the stored states of the company to determine total penalties and the terminal value of the company, which are used to evaluate the success of the scenario.

The value of a scenario is defined by the total asset value at the end of the simulation, from which we deduct all penalties incurred during the way.

In step 2 above, we stated that the length of the scenario tree should be adapted to the remaining length of the scenario. In order to reduce the length of the scenario tree, we remove the last stage and increase the number of branches in the first stage, keeping the number of scenarios constant.

The methods are tested on 3000 scenarios, for 4 periods of time. The trees used have 6400 scenarios, and the four period tree branches 16, 10, 10, 4 with shorter trees generated as previously described. In order to eliminate random contributions to the differences in performance between methods, the scenarios used and the scenario trees used are identical in all cases.

Test scenario generation and statistical test When performing rolling horizon simulation, the results of a single method will vary greatly between different test scenarios. In order to compensate for this disturbance, we will compare different methods on a scenario to scenario basis. To further reduce the errors, we will use antithetic samples for our test scenarios, and hence we will compare different methods by comparing their difference in performance over a pair of antithetic scenarios. Since the difference between the different methods used is rather small, we employ a T-test to determine if the mean difference between two methods is statistically significant. Hence we first average our results over the antithetic pairs, and then apply a pairwise T-test to the resulting samples, in order to determine if the observed difference is statistically significant. The test statistica reported in all tables is the probability of getting a larger deviation from 0 than the one obtained in the test, given that the two methods give the same mean value, evaluated over the continuous random distribution.

5.3 Question 1: on the reduction of the number of assets

Earlier we have described why it might be beneficial to reduce the number of assets held in an asset liability problem. In this section we try to determine the optimal number of linear combinations of assets to hold. We do this by testing using 1, 2, 3, 4, and 7 different asset combinations. When we are to hold 7 different assets classes, we simply use the original ALM problem (7). For the cases of 1–4 assets, we use the model described in (8).

In the optimization, the function $g(\cdot)$ in (9) is defined as the optimal value of the problem (8) evaluated over a 4 period tree. The tree over which we evaluate branches by 40, 16, 16 and 10 in the consecutive stages, giving the tree a total of 102,000 scenarios. This tree is chosen larger than the trees used in the simulations, in order to lower the impact of the choice of this tree on the overall solution.

As the problem (9) is probably not convex, we can not guarantee that the solution found is optimal. In our experiments we do however obtain approximately the same asset mixes a majority of the times when starting from a number of different random starting points. The number of local minimums does however seem to increase with the number of asset combinations, as may be expected.

Number of assets	1	2	3	4
2	0.0006			
3	0.0005	0.33		
4	0.0104	0.26	0.034	
7	0.029	0.67	0.40	0.73

Table 1: Test statistica value for different combinations.

If we observe the synthetic assets obtained in this fashion, we get the results in Figure 2. In this figure the synthetic assets are ordered in descending order according to their use in the root node of the asset search problem.

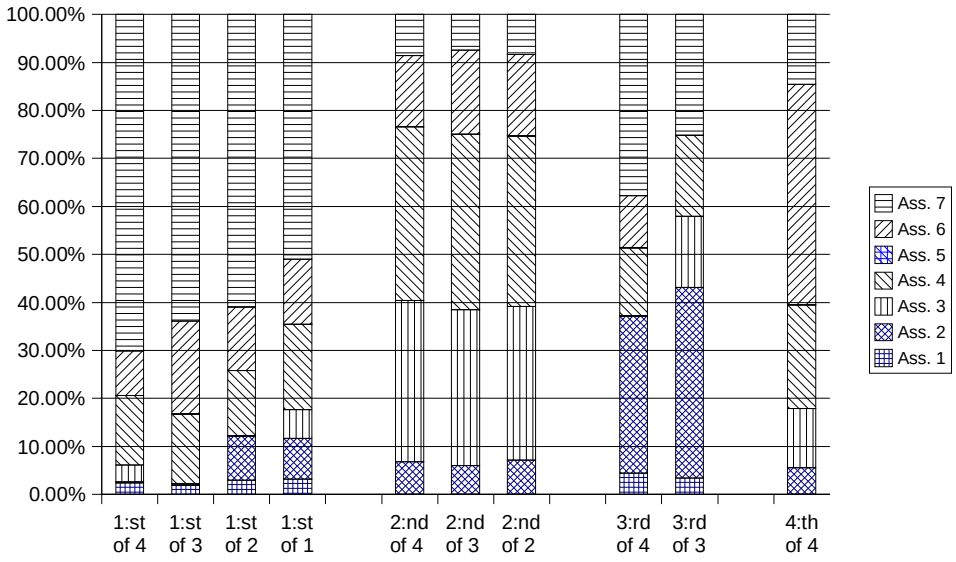

Figure 2: Funds used (ordered after decreasing use).

The funds obtained in the manner described above are then used unaltered throughout our simulations. In order to evaluate how well the chosen funds perform, we run 3000 simulations using the rolling horizon technique described in Section 5.2. The results from these runs are shown in Figure 3.

Although the number of different linear combinations is rather low, we see that we get the expected pattern of an increase in performance when the model is given a larger number of degrees of freedom, whereas the performance drops when the higher degree of freedom allows the solution to start adapting to the specific scenario tree used. As the difference between the different methods is rather small, we use the statistical test described above to generate Table 1. As might be seen from this table, not many of the differences are statistically significant when synthetic assets are applied at all stages (although the differences between using no synthetic assets and applying synthetic assets the second stage are significant).

5.4 Question 2: On when to apply asset combinations

As mentioned previously in Section 1.2 Gaivoronski and de Lange [11] experiment with an investment policy that is a hybrid between a fix-mix policy and a fully dynamic stochastic programming solution. They form this hybrid by letting the model freely choose an asset mix for the first two stages, but at the third and later stages, the asset mix i rebalanced to the asset mix chosen at the second stage.

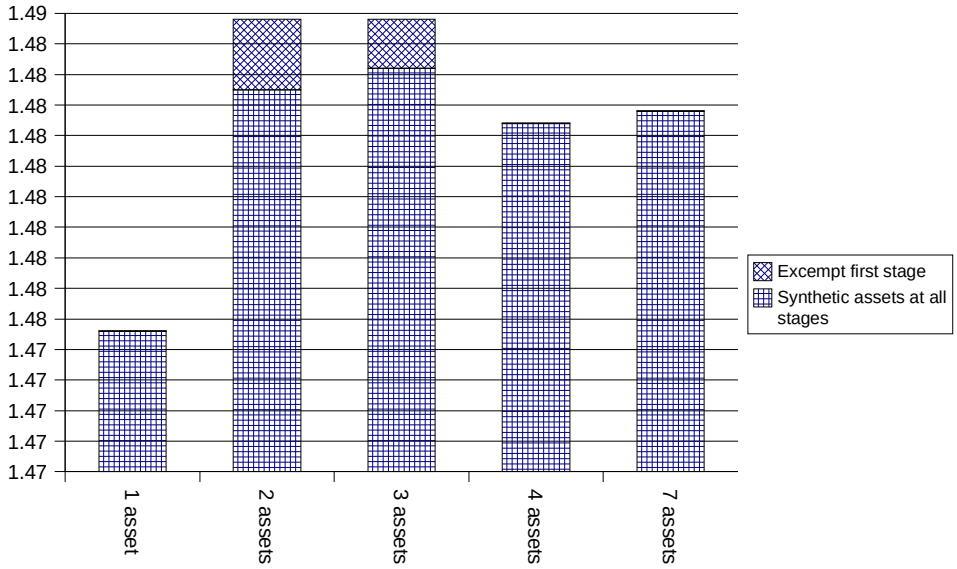

Figure 3: Results from asset reduction, simulation value as a function of the number of assets used.

Number of assets	gain	T-test statistica
2	0.0023	0.062
3	0.0016	0.17

Table 2: Gain from excluding stage 1 from synthetic assets.

Inspired by this idea, we wish to see if performance is improved if we exempt the first stage from the requirement of owning only a limited number of linear combinations of assets. We try this for 2 and 3 linear combinations, as these number of combinations gave the best results in the previous case. The results are given in Figure 3 and Table 2.

As might be seen from Table 2 our results show that performance is gained from excepting the first stage from using synthetic assets. Well worth noting is that if we compare the use of two synthetic assets applied at stage 2 to no use of synthetic assets from the previous question, the difference in objective function value is 0.0061. Using the same pairwise test as earlier, the probability of getting a larger difference given that the two strategies are equivalent is 5e-4, showing that we most certainly do gain from using synthetic assets. In Figure 3 we now see the expected pattern of a top in performance for a somewhat restricted feasible set, in between a problem with a highly restricted feasible set, and a problem with no restrictions on the feasible set.

The difference in objective function value might seem small; it corresponds to increasing the yield of the assets by 0.1% without an increase in risk. Although this difference is small, it is not negligible.

5.5 Question 3: On strict partitioning versus overlapping synthetic assets.

As mentioned earlier, when aggregating assets into larger funds which are to be used at the strategic level of the asset allocation, it is not clear that the division into aggregate classes should be made as a partitioning, with each basic asset belonging to only one synthetic asset. On the contrary, when the synthetic assets were optimized in the previous case, for the case of two funds, both funds contained

more than 5% of assets 2, 4, 6 and 7. In order to test whether forcing the synthetic assets to define partitions will make the solution better or worse, we divide the assets into high-risk (assets 3,4 and 6) and low risk (assets 1, 2, 5 and 7), based on the standard deviation of their yields. We now optimize two synthetic assets in the same fashion as in case 1, while making sure that one fund contains only high-risk assets, and one fund contains only low-risk assets by imposing constraints on c. The funds so obtained are given in Figure 4. For comparison we show the synthetic assets obtained when a partitioning is not required, and the synthetic assets are allowed to overlap.

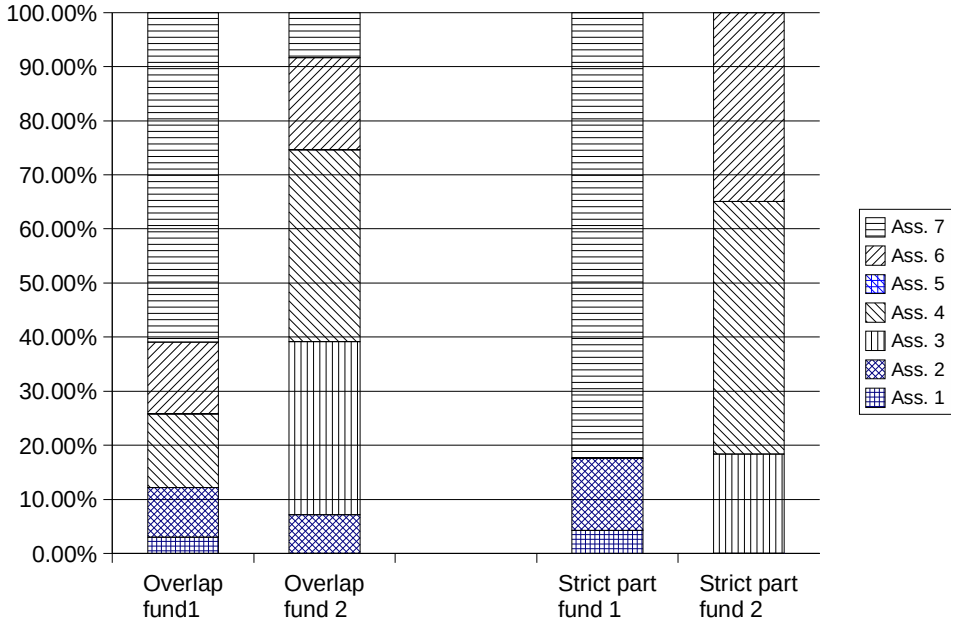

Figure 4: Funds used (partitioning versus with overlap).

As the previous case indicated that excepting the first stage from using synthetic assets was beneficial, we do so in this case as well. When we do not allow the synthetic assets to overlap, the difference in performance between the two divisions becomes $1.0e-4$ in favor of the partitioning. The T-test gives us that the probability of obtaining a larger difference if the two methods are equivalent is 0.94. Hence there does not seem to be any difference if we allow the assets to overlap or not.

5.6 Question 4: On the usefulness of optimizing aggregate assets within a scenario tree

The previous question concerned the effects of aggregating assets as a strict partitioning. The proportions of the basic assets to be included in the synthetic assets were determined via optimization, where we tried to find the synthetic assets which gave the best results when used in problem (8). As this procedure is rather complex, we would like to determine if we might get better, or at least not significantly worse results, by using the classical Markowitz optimization procedure. In order to compare asset classes created using optimization over the tree and asset classes created using the Markowitz method, we naturally need to construct asset classes using the latter method. In order to make a comparison with optimization over the tree possible, we make sure that the low-yield and the high yield assets constructed via Markowitz optimization has the same expected yield as the ones generated using strict partitioning in Section 5.5 above. The Markowitz optimization is performed in two ways. In the first case, we assume that we have one unit of money to invest in the assets, and we do this in the way that will minimize the standard deviation of our yield, given that we get the desired average

Mean value		
Markowitz, reserve	Opt. over tree	Markowitz, no reserve
1.4847	1.4848	1.4827
Statistical significance		
	Opt. over tree	Markowitz, no reserve
Markowitz, no reserve	0.08	
Markowitz, reserve	0.89	1e-4

Table 3: Test statistica value for Markowitz optimization vs. optimization over tree.

yield, an approach that will ignore the reserve requirements. In the second case, we assume that we have one unit of assets which is used to cover one unit of the reserve, and we try to minimize the standard deviation of the surplus. These two methods may be seen as using the liability hedging credit of Sharpe and Tint [20], with the weight of the liability hedging credit set to 0 and 1, respectively.

Using the two versions of Markowitz optimization, we obtain the synthetic assets given in Figure 5. As previously, we run rolling horizon simulations for the two new sets of synthetic assets, and the results from these tests are given in Table 3. We see that there is no significant difference between using the Markowitz optimization with consideration to the reserve, and optimizing over the tree. There do however seem to be a difference between optimizing over the tree and using the Markowitz method without considering the reserve.

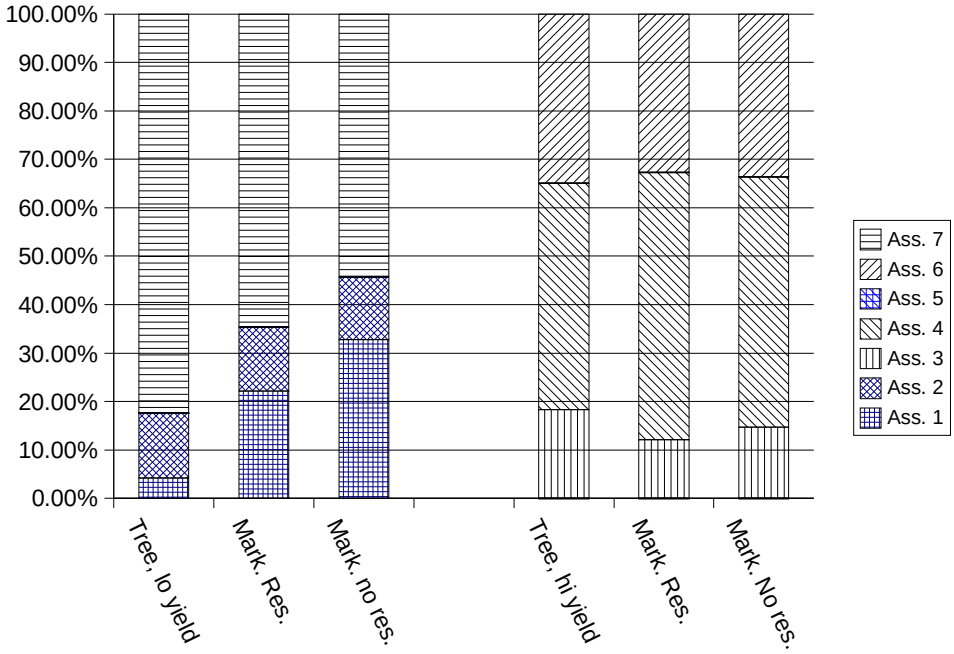

Figure 5: Funds found with Markowitz optimization, vs. funds from Section. 5.5.

6 Conclusions and further work

6.1 conclusions

In this article we have shown that reducing the number of asset classes by allowing trade in only a limited number of linear combinations of assets (termed synthetic assets) increases the performance of an asset liability model. We further found that the model should not be constrained to using the synthetic assets at the root node of the problem, as the higher number of branches commonly used at this stage should provide a good enough description of the possible random outcomes to make artificial restrictions of the feasible set superfluous, if not directly damaging.

Furthermore, the optimal number of assets was low; the best results where obtained using only two synthetic assets. In retrospect, this is not surprising. The main purpose of a multi-stage ALM model is to make it possible to change the risk-reward tradeoff in the future, depending on the state of the company and the world, and consider these future changes when todays decision is made. The simplest model allowing such changes is a model with two synthetic assets.

In addition to searching for the optimal number of synthetic assets, we examined if the aggregation into synthetic assets should be done as a partitioning, or if an overlap should be allowed. As we were not able to show any difference between no overlap and overlap, we conclude that this decision should be based on other criteria. (As it is simpler to use non-overlapping assets, we will probably do this in the future.)

The primary method of aggregating assets into synthetic assets in this work has been to find the synthetic assets which give the best objective value over an instance of the stochastic linear programming model. Our experiments do not indicate that this method outperforms the Markowitz portfolio optimization model, provided however that the latter is extended to take correlations with the reserve into account. From the experiments it is clear that there is a significant disadvantage to not include the reserve considerations when the synthetic assets are determined. If we in the future use the Markowitz model, we must however determine how the trade-off between risk and yield should be made when the synthetic assets are created. Right now the yield for the two synthetic assets were taken from the funds obtained via optimization over a scenario tree.

Well worth noting is that the increased performance we experienced when using synthetic assets is not the only benefit. As for most optimization problems, the computational prize of solving a stochastic programming problem increases with the dimension of the problem. If the number of assets in the problem is reduced, so is the size of the optimization problem. Hence aggregating assets makes it possible to increase the number of discretization points used in the scenario tree, or the number of stages in the tree, without increasing the size of the problem. As a better description will give more accurate solutions, aggregating asset classes would have a positive effect even if a solution from an aggregated problem performs just as well as an unaggregated one, since aggregating assets would allow us to solve problems with larger scenario trees without using more computational resources.

6.2 Further work

In this work, the asset fractions are enforced using artificial constraints, for reasons explained in Section 4.1. A better approach would probably be to let the model trade directly in the synthetic assets, as this significantly would reduce the number of variables.

Furthermore, as we did not find a significant advantage of using asset classes created by optimization over the scenario tree compared to using Markowitz portfolio selection method, it would probably be better to use the latter method, as it is simpler. However, it still leaves us with the problem of how to choose the value of risk used in this method. A possible approach is to use dual information from the stochastic programming ALM problem. In order to illustrate this approach, we assume that we have a simple two stage stochastic programming ALM problem with N scenarios. The problem includes only two synthetic assets and one reserve requirement. The yield of the assets for scenario i are given by the constants a_i and b_i, and the reserve for scenario i is given by s_i. In the way described in Section 4.2,

we may obtain the change in optimal objective value if we make a marginal change to the parameter a_i; we denote this change a_i^*. By summing a_i^* over i we obtain the value of marginally changing the yield of the low-yield synthetic asset. In the same fashion, the expression

$$\frac{\sum_{i=1}^{N} a_i^*(a_i - \mathsf{E}[a])}{2}$$

gives the change in the objective function value if the variance of the low yield asset is increased marginally. Similarly, we may obtain the value of marginally changing the correlation between the low-yield asset and the reserve requirement, as well as information regarding the high-yield asset. This dual data may now be used in a Markowitz optimization to find new low yield and high yield assets, which are to be used in the stochastic programming ALM problem. As the value of correlating the yield to the reserve probably would be different for the low-yield and the high-yield asset, this is equivalent to using different liability hedging credits for the two synthetic assets.

In this work we have assumed that the random distributions are stationary. If this is not the case, we may have one Markowitz problem for each node in the scenario tree, adapting the low and high yield asset to the conditions in each node, possibly obtaining better performance.

Furthermore, in this work, we have only studied how the synthetic assets should be constructed in order to prevent the solution from chasing spurious profits. However, once the synthetic assets are determined, the scenario tree need no longer provide an accurate approximation of the correlations and yields of the basic assets, all that is needed is a scenario tree which accurately describes the moments and correlations of the created synthetic assets. Hence the scenario tree may be regenerated in order to provide better information on the synthetic asset, while ignoring statistical properties regarding the basic assets which are now irrelevant.

7 Acknowledgments

This work was partially funded by Nordea life & pension.

References

[1] F. Altenstedt, *An asset liability management system for a Swedish life insurance company*, Preprint 2003:47, Chalmers University of Technology, Department of mathematics, SE-412 96 Göteborg, Sweden, 2003. Submitted to Annals of Operations Research.

[2] D. P. Bertsekas, *On the Goldstein–Levitin–Polyak gradient projection method*, IEEE Transactions on Automatic Control, AC-21 (1976), pp. 174–184.

[3] S. P. Bradley and D. B. Crane, *A dynamic model for bond portfolio management*, Management Science, 19 (1972), pp. 139–151.

[4] D. R. Cariño, D. H. Myers, and W. T. Ziemba, *Concepts, technical issues, and uses of the Russell–Yasuda Kasai financial planning model*, Operations Research, 46 (1998), pp. 450–462.

[5] D. R. Cariño and W. T. Ziemba, *Formulation of the Russell–Yasuda Kasai financial planning model*, Operations Research, 46 (1998), pp. 433–449.

[6] D. R. Cariño, W. T. Ziemba, T. Kent, D. H. Myers, C. Stacey, M. Sylvanus, A. L. Turner, and K. Watanabe, *The Russell–Yasuda Kasai model: An asset/liability model for a Japanese insurance company using multistage stochastic programming*, Interfaces, 24 (1994), pp. 29–49.

[7] G. Consigli and M. Dempster, *Dynamic stochastic programming for asset–liability management*, Annals of Operations Research, 81 (1998), pp. 131–161.

[8] G. B. Dantzig and M. N. Thapa, *Linear Programming*, vol. 1, Springer-Verlag, New York, 1997.

[9] C. Dert, *Asset liability management for pension funds; A multistage chance constrained programming approach*, PhD thesis, Erasmus University Rotterdam, 1995.

[10] P. Frennberg and B. Hansson, *Swedish stocks, bonds, bills and inflation*, Applied Financial Economics, 2 (1992), pp. 79–86.

[11] A. A. Gaivoronski and P. E. De Lange, *An asset liability management model for casualty insurers: complexity reduction vs. parameterized decision rules*, Annals of Operations Research, 99 (2000), pp. 227–250.

[12] J. L. Higle, *Variance reduction and objective function evaluation in stochastic linear programs*, INFORMS Journal on Computing, 10 (1998), pp. 236–247.

[13] K. Høyland, *Asset liability management for a life insurance company: A stochastic programming approach*, PhD thesis, Department of Economics and Technology Management, Norwegian University of Science and Technology, Trondheim, Norway, 1998.

[14] K. Høyland and S. W. Wallace, *Analyzing legal regulations in the Norwegian life insurance business using a multistage asset-liability management model*, European Journal of Operational Research, 134 (2001), pp. 293–308.

[15] ——, *Generating scenario trees for multistage decision problems*, Management Science, 47 (2001), pp. 295–307.

[16] M. Kaut and S. W. Wallace, *Evaluation of scenario-generation methods for stochastic programming.*
Available from `http://www.iot.ntnu.no/~mkaut/CV_and_study/SG_evaluation.pdf`.

[17] R. Kouwenberg, *Scenario generation and stochastic programming models for asset liability management*, European Journal of Operational Research, 134 (2001), pp. 279–292.

[18] J. M. Mulvey and H. Vladimirou, *Stochastic network programming for financial planning problems*, Mangement Science, 38 (1992), pp. 1642–1664.

[19] G. Pflug, *Scenario tree generation for multiperiod financial optimization by optimal discretisation*, Mathematical Programming, Series B, 89 (2001), pp. 251–271.

[20] W. F. Sharpe and L. G. Tint, *Liabilities-a new approach*, the journal of portfolio management, (1990), pp. 5–10.

[21] K. J. Worzel, C. Vassiadou-Zeniou, and S. A. Zenios, *Integrated simulation and optimization models for tracking indices of fixed-income securities*, Operations Research, 42 (1994), pp. 223–233.

A Simulation Data

A.1 Simulation parameters

$\bar{x}_i$	$\{0.1, 0.2, 0.2, 0.1, 0.1, 0.1, 0.1\}$
γ_i	$0.005, i \in I$
p^t	$0.06, t \in 0, \ldots, \hat{T}$
f_q	$\{1.15, 1.06, 1.02, 1.00\}$
s_q	$\{1.0, 1.0, 2.0, 2.0\}$

A.2 Assets

The assets used in the simulations have the following means and standard deviations:

	reserve	a1	a2	a3	a4	a5	a6	a7
mean	11.01	6.21	7.38	12.48	11.37	4.59	8.19	6.18
Std.	1.88	5.26	9.46	24.81	18.09	0.43	16.06	3.52

Table 4: Mean and standard deviation of assets used (%).

	reserve	a1	a2	a3	a4	a5	a6
a1	0.49381						
a2	0.18627	0.18943					
a3	0.45105	0.45642	0.20575				
a4	0.33364	0.33703	0.58166	0.67816			
a5	0.56793	0.61823	0.11801	0.26933	0.19448		
a6	0.48187	0.48950	0.03016	0.15257	0.01430	0.29368	
a7	0.19084	0.19593	0.17999	0.21120	0.09766	0.16223	0.17333

Table 5: Correlations of assets used.