Dynamic stochastic programming for asset – liability management

G. Consigli and M.A.H. Dempster
Judge Institute of Management Studies, University of Cambridge, Cambridge CB2 1AG, England
E-mail: {gc217; mahd2}@cam.ac.uk

Multistage stochastic programming – in contrast to stochastic control – has found wide application in the formulation and solution of financial problems characterized by a large number of state variables and a generally low number of possible decision stages. The literature on the use of multistage recourse modelling to formalize complex portfolio optimization problems dates back to the early seventies, when the technique was first adopted to solve a fixed income security portfolio problem. We present here the CALM model, which has been designed to deal with uncertainty affecting both assets (in either the portfolio or the market) and liabilities (in the form of scenario dependent payments or borrowing costs). We consider as an instance a pension fund problem in which portfolio rebalancing is allowed over a long-term horizon at discrete time points and where liabilities refer to five different classes of pension contracts. The portfolio manager, given an initial wealth, seeks the maximization of terminal wealth at the horizon, with investment returns modelled as discrete state random vectors. Decision vectors represent possible investments in the market and holding or selling assets in the portfolio, as well as borrowing decisions from a credit line or deposits with a bank. Computational results are presented for a set of 10-stage portfolio problems using different solution methods and libraries (OSL, CPLEX, OB1). The portfolio problem, with an underlying vector data process which allows up to 2688 realizations at the 10-year horizon, is solved on an IBM RS6000/590 for a set of twenty-four large-scale test problems using the simplex and barrier methods provided by CPLEX (the latter for either linear or quadratic objective), the predictor/corrector interior point method provided in OB1, the simplex method of OSL, the MSLiP-OSL code instantiating nested Benders decomposition with subproblem solution using OSL simplex, and the current version of MSLiP.

Keywords: asset – liability portfolio management, pension fund management, dynamic stochastic programming, linear programming, quadratic programming, simplex method, interior point method, nested Benders decomposition

1. Introduction

In this paper, we develop a generic model for the integrated dynamic management of financial assets and liabilities – *the computer-aided asset/liability management* (CALM) model – which we feel certain will provide a basis for ubiquitous model-based corporate financial planning within a decade or so. Such planning will be brought about by the realistic nature of models of this type, rapid advances in software tools and computer hardware capability, and a universal requirement to model and more tightly control financial risks in modern global corporations, from banks to manufacturers, employing a wide variety of technologies from low to high.

The CALM model has been developed from dynamic stochastic programming research which began over two decades ago with the seminal work of Bradley and Crane [8] and Lane and Hutchinson [48]. A partial list of subsequent related works includes [10, 23, 47, 58, 68], some of which will be treated in more detail in section 3.

Perhaps the most important of these papers are those of Bradley and Crane [8] – who introduced the *inventory approach* to modelling financial decisions in which each asset or liability in the model has an initiate, hold and terminate variable in each period, an invaluable aid to realistic detailed modelling, Dempster and Ireland [24] – who explored the modern information system context required for such models, and Cariño et al. [10] – who developed the first genuine *commercial* application of dynamic stochastic programming in spite of the fact that most of their predecessors were involved in prototype applications with financial institutions.

To indicate the current state-of-the-art, we present computational results for an instance of the CALM model – the *Watson model* – which is specialized to a pension fund manager's dynamic integrated asset/liability management problem in a single currency. Comparisons are made of current approaches to the numerical solution of such problems utilizing a contemporary high end workstation and the question of adequate representation of financial uncertainties in the models is addressed.

In the next section, we review the multistage recourse formulation of dynamic portfolio management problems. Section 3 introduces the CALM and Watson integrated asset/liability management models. (A detailed presentation of the CALM model is given in the appendix.) Multistage recourse models are very complex large-scale linearly constrained problems whose constraint structure grows in size linearly with the number of data paths or *scenarios* representing the uncertainties facing the decision maker. Hence, practical models must be handled with software tools which generalize – and utilize – linear programming (LP) modelling languages such as GAMS, AMPL and MODLER (used here). This is the topic of section 4, which briefly describes the STOCHGEN subroutine library [14] developed by our research group. Section 5 reviews current LP and QP solution solution techniques and software – of simplex, interior point and decomposition type – with emphasis on their features relevant to the solution of large-scale financial planning models. Our comparative computational results – showing the clear overall superiority of the nested Benders decomposition technique for these large-scale portfolio management problems – are presented in section 6. Section 7 contains conclusions and directions for further work.

2. Multistage recourse formulation of dynamic portfolio management

We consider a stochastic programming problem in the form of a multistage recourse problem (cf. Dempster [21], Ermoliev and Wets [31]) (here, boldface denotes random):

$$
\begin{aligned}
\min_{x_1 \in \mathbb{R}^{n_1}} & \left\{ f_1(x_1) + \mathbb{E}_{\boldsymbol{\omega}_2} \left[\min_{\boldsymbol{x}_2} \left(\boldsymbol{f}_2(\boldsymbol{x}_2) + \cdots + \mathbb{E}_{\boldsymbol{\omega}_T | \boldsymbol{\omega}^{T-1}} \left[\min_{\boldsymbol{x}_T} \boldsymbol{f}_T(\boldsymbol{x}_T) \right] \right) \right] \right\} \\
\text{s.t. } & A_1 x_1 = b_1, \\
& \mathbf{B}_2 \boldsymbol{x}_1 + \mathbf{A}_2 \boldsymbol{x}_2 = \boldsymbol{b}_2 \text{ a.s.}, \\
& \mathbf{B}_3 \boldsymbol{x}_2 + \mathbf{A}_3 \boldsymbol{x}_3 = \boldsymbol{b}_3 \text{ a.s.}, \\
& \ddots \\
& \mathbf{B}_T \boldsymbol{x}_{T-1} + \mathbf{A}_T \boldsymbol{x}_T = \boldsymbol{b}_T \text{ a.s.}, \\
& l_1 \le x_1 \le u_1, \\
& \boldsymbol{l}_t \le \boldsymbol{x}_t \le \boldsymbol{u}_t \text{ a.s.}, \quad t = 2, \ldots, T.
\end{aligned} \tag{1}
$$

In (1), the separable **objective** is defined by the period functionals f_t, for $t = 1$ up to the *horizon* T; $A_1 \in \mathbb{R}^{m_1 n_1}$ and $b_1 \in \mathbb{R}^{m_1}$ define deterministic constraints on the first-stage decision x_1, while, for $t = 2,\ldots,T$, $A_t : \Omega \to \mathbb{R}^{m_t n_t}$, $B_t : \Omega \to \mathbb{R}^{m_t n_{t-1}}$ and $b_t : \Omega \to \mathbb{R}^{m_t}$ define stochastic constraint regions for the recourse decisions $\boldsymbol{x}_2,\ldots,\boldsymbol{x}_T$; and $\mathbb{E}_{\boldsymbol{\omega}_t | \boldsymbol{\omega}^{t-1}}$ denotes conditional expectation of the state $\boldsymbol{\omega}_t$ of the *data process* $\boldsymbol{\omega}$ at time t with respect to the *history* $\boldsymbol{\omega}^{t-1}$ of the process up to time t, where the data process $\boldsymbol{\omega}$ may be regarded as a random vector defined in a canonical probability space $(\Omega, \mathcal{F}, P)$, with $\boldsymbol{\omega}_t := (\boldsymbol{\xi}_t, \mathbf{A}_t, \mathbf{B}_t, \boldsymbol{b}_t)$ in which $\boldsymbol{\xi}_t$ is the random parameter in the objective functional given by $f_t(\xi_t, x_t)$.

The recourse formulation (1) shows explicitly the dependence of the optimal policy or *decision process* $\boldsymbol{x}^0 := (\boldsymbol{x}_1^0, \boldsymbol{x}_2^0,\ldots,\boldsymbol{x}_T^0)$ on the realizations of the vector data process $\boldsymbol{\omega} := (\boldsymbol{\omega}_2,\ldots,\boldsymbol{\omega}_T)$ in $(\Omega, \mathcal{F}, P)$, with the sample space defined as $\Omega := \Omega_2 \times \Omega_3 \times \ldots \times \Omega_T$ and the filtration $\mathcal{F}_1 := \{0, \Omega\} \subset \mathcal{F}_2 \subset \mathcal{F}_3 \subset \ldots \subset \mathcal{F}_T := \mathcal{F}$, where $\mathcal{F}_t := \sigma\{\boldsymbol{\omega}^t\}$ is the σ-field generated by the history $\boldsymbol{\omega}^t$ of the data process $\boldsymbol{\omega}$ for $t = 2,\ldots,T$ and P is a probability measure on this space.

This model embodies the main features of a decision problem under uncertainty:

- The objective in this nested representation formalizes a sequence of optimization problems corresponding to different stages: at time 1, the decision maker has to select a decision whose outcome completely depends on the future realizations of the underlying multidimensional stochastic data process. Solution of this problem is sometimes referred to as the *here and now* problem.
- Thereafter, for each realization of the history $\boldsymbol{\omega}^t$ of the data process up to time t, a recourse problem is considered in which decisions are allowed to be a function of the observed realization (x^{t-1}, ω^t) only. At each stage, previous decisions affect current problems through the stochastic matrices $\mathbf{B}_t$, $t = 2,\ldots,T$, with the decision sequence

$$
\underset{x_1}{\text{decide}} \rightsquigarrow \underset{\omega_2}{\text{observe}} \rightsquigarrow \underset{x_2}{\text{decide}} \rightsquigarrow \ldots \rightsquigarrow \underset{\omega_T}{\text{observe}} \rightsquigarrow \underset{x_T}{\text{decide}} .
$$

The recourse decisions depend on the current state of the system as determined by previous decisions and by random events. Each decision is thus required to be *adapted* to the σ-field generated by the data process implying *nonanticipativity* in these decisions.

- The data process $\boldsymbol{\omega}$ in $(\Omega, \mathcal{F}, P)$ is defined as a discrete-time, possibly autocorrelated, vector stochastic process. A finite sample of its paths is conveniently represented as a *scenario tree*, each scenario corresponding to a trajectory of the process $\boldsymbol{\omega}^T := (\boldsymbol{\omega}_1, \boldsymbol{\omega}_2, \ldots, \boldsymbol{\omega}_T)$ at the horizon T. It is this branching structure of data paths that ensures that all decisions are made in the face of uncertainty.
- In (1) – as is generally the case in financial planning problems – both the RHS and LHS of the constraint set are scenario dependent. The generation of the sample data paths for the problem represents a crucial step for a valid specification of the stochastic optimization problem. (In section 3, we will briefly consider some of the implications of the so-called investment model (Wilkie [65]) that has been adopted for this purpose in the WATSON portfolio problems.)

Dynamic portfolio problems are easily formulated as *dynamic recourse problems* (*DRP*). This approach to model-based portfolio management was first adopted by Bradley and Crane [8] in 1973 for a portfolio problem restricted to fixed income securities. Other applications of DRP for financial planning are reported in Lane and Hutchinson [48], Kallberg et al. [43], Kusy and Ziemba [47], Dempster and Ireland [24], Mulvey and Vladimirou [58], Dantzig and Infanger [19], Zenios [68], Cariño et al. [10, 11].

In many of these examples, uncertainty takes the form of unknown future rates of return on market investments and funding sources, as well as cash flow misalignments, and the objective function is typically characterized in terms of the expected value of a linear or nonlinear utility function of wealth at the horizon and sometimes beyond.

Problem (1) may be given a more compact *dynamic programming* representation which takes advantage of the structure exhibited by the set of constraints (here Markovian or staircase, but more generally lower triangular). For each $t = 1, \ldots, T-1$, we have

$$\begin{aligned} &\min_{x_t} \, [\, f_t(x_t) + \upsilon_{t+1}(\omega^t, x^t)] \\ \text{s.t.} \quad & B_t x_{t-1} + A_t x_t = b_t, \end{aligned} \tag{2}$$

where υ_{t+1} expresses the optimal expected cost for the stages from $t+1$ to T, given the decision history $x^t := (x_1, \ldots, x_t)$ and the realized history of the random process $\omega^t := (\omega_1, \ldots, \omega_t)$. Specifically,

$$\upsilon_{t+1}(\omega^t, x^t) := \mathbb{E}_{\boldsymbol{\omega}_{t+1}|\boldsymbol{\omega}^t}\left[\min_{\mathbf{x}_{t+1}}\left(f_{t+1}(\mathbf{x}_{t+1}) + \cdots + \mathbb{E}_{\boldsymbol{\omega}_T|\boldsymbol{\omega}^{T-1}} \min_{\mathbf{x}_T} (f_T(\mathbf{x}_T)\,)\right)\right], \tag{3}$$

where the minimizations are taken subject to the appropriate constraints, which will be discussed further in section 5.

Correspondingly, at the end of each time period and on the basis of the current information, a portfolio manager selects an optimal decision in the face of the uncertainty that (s)he is now facing. This decision needs to be feasible with respect to the constraints induced by the future values of the random data process and is influenced by the current composition of the portfolio.

Due to the convexity of $\upsilon_{t+1}(\omega^t, x^t)$ with respect to x^t, problem (2) can also be formulated in the form

$$\begin{aligned} &\min_{x_t} \ f_t(x_t) + \theta_t \\ \text{s.t.} \quad &B_t x_{t-1} + A_t x_t = b_t \\ &\upsilon_{t+1}(\omega^t, x^t) \leq \theta_t. \end{aligned} \tag{4}$$

The adoption of a cutting-plane algorithm for the solution of (1) is based on the simple decomposition (4) by approximating the value function at $t+1$ by a set of cuts of the form

$$d' x_t \leq \theta_t,$$

see [6, 32, 26]. Cuts are also used to enforce constraints on current decisions which ensure that later recourse actions are feasible.

3. The CALM and Watson models

Multistage stochastic programming techniques, unlike typical stochastic control formulations, allow the representation and solution of financial problems characterized by a large number of state variables and a generally low number of possible decision stages.

Recent applications of multistage stochastic programming techniques for financial planning confirm the possibility of solving large problems with a complex structure. The Russell–Yasuda Kasai model [10] and the Towers Perrin investment system for pension plans [57] are important examples.

In the sequel, we discuss the formulation and solution of a long-term portfolio problem with uncertainty affecting price processes, interest rates and liability payments. The CALM model (Dempster [22a], see the appendix) represents a general formulation of an asset–liability model as a linearly constrained problem with an objective function explicitly taking into account the risk attitude of the portfolio manager. It is based on well-established financial theory and addresses a multicurrency portfolio problem with randomness affecting rates of return and liabilities in domestic and foreign terms.

The general CALM model assumes a portfolio manager possessing a utility function from a general class which expresses his or her attitude to possible distributions of the terminal wealth $\mathbf{W}^T$. This class of isoelastic utility functions u characterizes a risk-averse portfolio manager possessing an Arrow–Pratt relative risk measure which is inverse affine. Logarithmic utility functions, as well as certain exponential and polynomial utilities, belong to this class and a well-established theory is available with theoretical results for the case of portfolio problems formulated as dynamic programming problems under uncertainty (see e.g. [4]).

A practical dynamic portfolio management theory requires, as well as an accurate representation of the risk attitude of the portfolio manager, fast and accurate solution techniques for a decision problem involving large numbers of decision variables – so far impossible to handle in a dynamic programming framework, but currently practical for the scenario-based multistage stochastic programming DRP formulation.

For this reason and given the current state of algorithm development for large-scale nonlinear problems, we consider an objective in linear or quadratic form and we restrict ourselves to these two cases in the numerical experiments that follow, although piecewise linear/quadratic forms are also computationally tractable [46]. Specifically,

$$\mathbb{E}u(\mathbf{W}^T) := \begin{cases} \mathbb{E}\,\mathbf{W}^T & \text{in the linear case,} \\ \mathbb{E}(\mathbf{W}^T - \tilde{W})^2 & \text{in the quadratic case,} \end{cases} \tag{5}$$

where $\tilde{W}$ denotes a *target* determined a priori by the portfolio manager and the expectation is taken with respect to the probability measure assigned at the horizon to scenarios.

The *Watson model* – named after Watson & Sons Consulting Actuaries who have provided the set of financial data for the parameter specification of the problem – represents a particular instance of the CALM model for the case of pension fund management with uncertainty affecting assets, either in the portfolio or in the market, and liabilities, in the form of scenario-dependent pension payments or borrowing costs.

We consider two possible ways of generating scenarios for the test problems which are both based on the *Wilkie investment model* [65], regarded as a complex nonlinear predictor of real and nominal returns for five asset classes and pension payments on an equal number of pension schemes. In particular, either

(1) we take the simulations as provided by Watson & Sons, a set of 10-year independently generated sample paths and generate a scenario tree by “imposing” a tree-structure (as explained in section 4) which largely destroys the temporal stochastic properties of the underlying data process, or

(2) by modifying the generator as originally supplied, we generate data paths in conditional mode, correctly representing paths from the Wilkie investment model in the form of a scenario tree.

Our computational results show that the two procedures have different implications for the nature of the optimization problem and the performance of a solution method based on nested Benders decomposition. Wilkie’s model is based on actual data from the UK for the period 1924–1991 and is formulated as a set of simultaneous autoregressive equations of up to the third order in Wold recursive form, all dependent on an underlying inflation process. It generates data paths for annual returns in the UK market for (see figure 1)

(1) ordinary shares,
(2) fixed-interest irredeemable bonds,
(3) bank deposits,
(4) index-linked securities,
(5) real estate,

together with predictions of annual pension payments and an estimated *reassurance-to-close* representing future payment liabilities discounted to the horizon. All these are used to calculate the required random coefficients for the WATSON problems studied in the numerical experiments.

In order to characterize the risk faced by the portfolio manager, we show in figures 1 and 2 the marginal empirical probability densities for the different returns at the horizon resulting, respectively, from the simulation of 1,000 independent data paths and 1,024 conditional scenarios. A comparison shows the impacts of the two procedures on the marginal distributions at the horizon; with a clear path-dependent "humpy" effect in the case of the conditionally generated scenarios.

The Watson model is a dynamic A/L management optimization problem with linear constraints with general financial and accounting considerations as follows.

(a) *Cash balance constraint*

In each period cash inflows, due either to borrowing or selling decisions, must be equal to cash outflows, due to pension payments, debt reimboursements or other sources. We consider five pension funds, with an estimated payment also required at the horizon to compensate for all future liabilities.

(b) *Inventory balance constraints*

Each decision of the portfolio manager with an impact on the portfolio composition needs to be accounted for. In the Watson model – following Bradley and Crane [8] and unlike most of the other models presented in the past – we specify at each stage the composition of the portfolio for each asset category according to the period in which the investment has taken place. This is the mechanism by which both tax and regulatory constraints are easily added to the model.

(c) *Other conditions*

All decisions are constrained by upper or lower bounds on investment classes, lines of credit and specific corporate and regulatory constraints. In general these conditions have to be modelled as scenario dependent; in the first instance, however, we keep them fixed over the planning horizon.

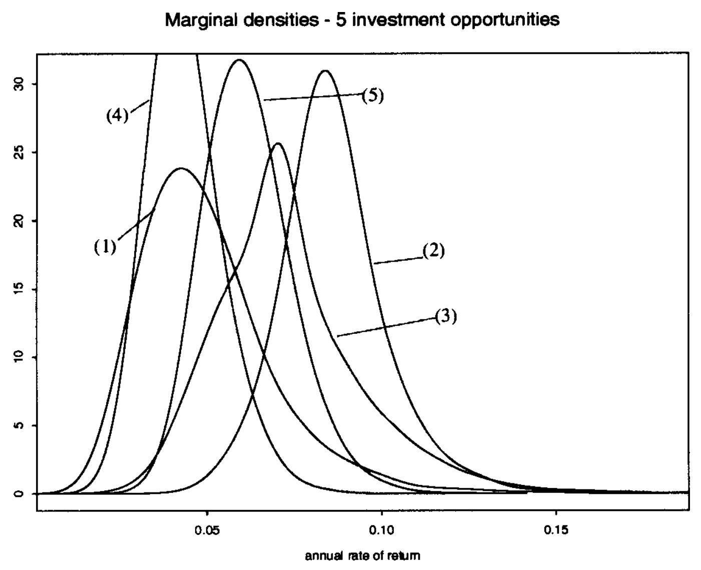

Figure 1. Return distributions: 1,000 independent simulations.

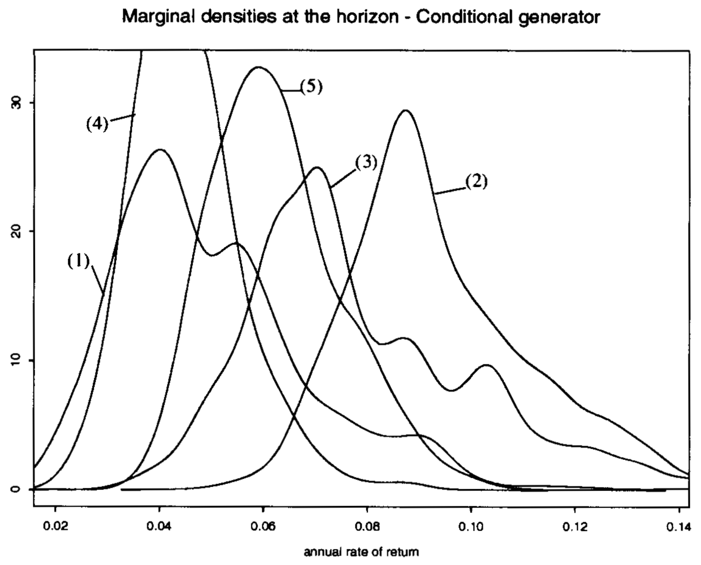

Figure 2. Return distributions: 1,024 conditional scenarios.

(d) *Terminal wealth definition*

Finally, the specification of the terminal wealth that enters the objective function represents an important aspect of the portfolio problem. In the Watson model, we include in the definition the market values at the horizon of all assets in the portfolio, together with the position with the bank, net of the terminal reassurance-to-close payments for the five pension funds.

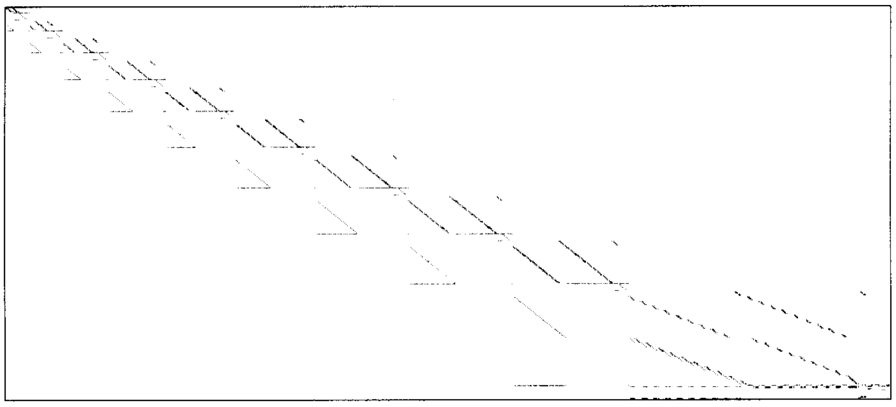

Figure 3. Non-zero entries in the WATSON coefficient matrix.

Figure 3 shows the sparsity structure of the constraint matrix for the Watson problem on a single scenario. The constraint matrix exhibits a block-bidiagonal structure as a result of the Markovian nature of the A/L management decision problem and suggests scope for decomposition methods.

Given the model formulation and a data specification defining its random coefficients, the resulting stochastic programming problem must be generated in standard input format for numerical solution. A complete specification of the problem also requires the path probabilities for the different scenarios – here equally likely, since they have been generated by simulation.

4. Model generation

A stochastic programming problem specified for numerical solution is conventionally in SMPS format (Birge et al. [7]), described in terms of three input files as: a *time file* defining the dynamic structure of the problem, a *core file* supplying all the data along one scenario path (generally regarded as the *base* scenario of the problem), and a *stoch file* containing the stochastic information of the problem, in which the scenario probabilities are also specified. The profile and dimension of the stoch file depends on the number of stages in the model as well as on the number of realizations allowed at each stage.

The data specification represents an essential step in the formulation of a stochastic problem. In a one-scenario, i.e. deterministic, problem we remain in the area of linear or quadratic programming. An MPS file generator for deterministic problems (MODLER, AMPL, MIMI, etc.) can be usefully incorporated into the design of an SMPS generator for the specification of a stochastic version of the problem. This is the approach taken in the STOCHGEN subroutine library (SSL) recently created by Corvera Poiré [14]. Its structure is shown in figure 4.

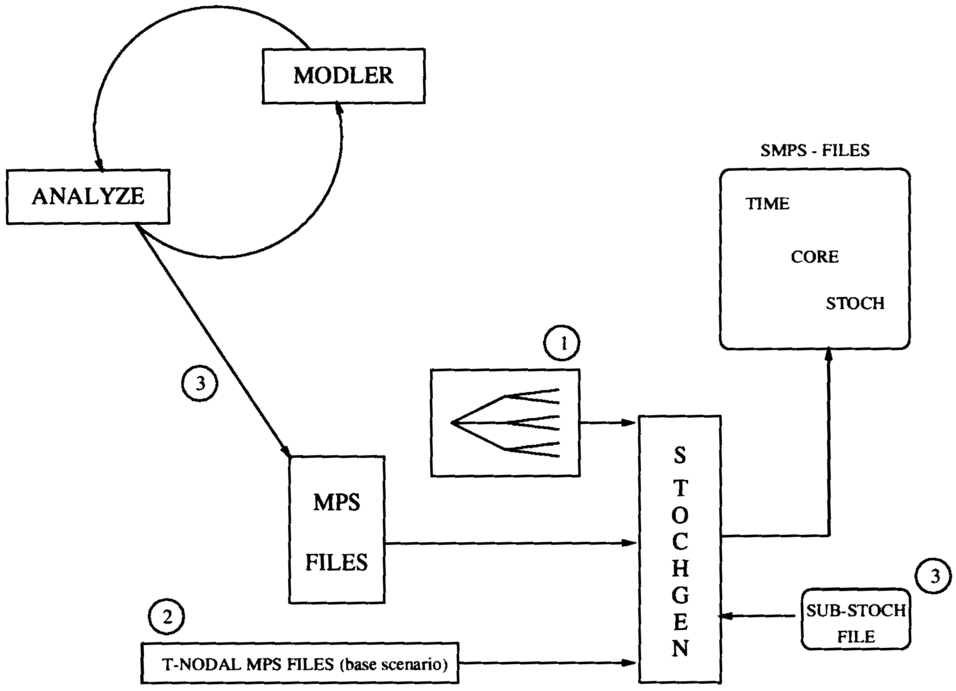

Figure 4. Model generation.

STOCHGEN takes as inputs a sequence of MPS files generated for each scenario by the modelling language, here MODLER (Greenberg [38]), together with the specification of the tree structure for the data process and a set of nodal MPS files for the base scenario, and provides the required SMPS files as output. The SSL library expands the utilization of current modelling facilities for linear programming to the stochastic case and provides insight into the understanding of dynamic stochastic programs. It is constructed to encourage the LP user to approach the field of stochastic programming and allows for the combination of trees and subtrees in an error-checked non-redundant manner.

5. Standard solution techniques

The specification of the probability measure P as discrete for the measurable sample space $(\Omega, \mathcal{F})$ allows an easy derivation [18] of the deterministic equivalent of a stochastic program with recourse for numerical solution.

We consider two possible solution procedures for the deterministic equivalent of the dynamic portfolio problem treated here:

- Direct solution of the deterministic equivalent problem in which conditional probabilities (or, equivalently, path probabilities) are used to define expectations and constraints are replicated for all data paths at each stage. Very large linear or quadratic programs are created which can be solved using, respectively, standard LP algorithms (i.e. simplex or interior point methods) or QP algorithms (pivoting, iterative or interior point based algorithms).

- Alternatively, the original problem may be decomposed into a sequence of subproblems which are computationally closely related across scenarios and nested Benders decomposition applied. This algorithm passes solution information from subproblem to subproblem forward in time, and feasibility and optimality information in the form of cuts backward in time.

Assuming for simplicity equal numbers of branches descending from each node of the decision tree in a time period, together with the specification of the corresponding conditional probabilities as $p_t^{k_2,k_3,\dots,k_t}$, $t = 2,\dots,T$, allows the formulation of the *deterministic equivalent* of (1) in the form

$$\min\left\{ f_1(x_1) + \sum_{k_2=1}^{K_2}\left[p_2^{k_2} f_2(x_2^{k_2}) + \sum_{k_3=1}^{K_3}\left[p_3^{k_2,k_3} f_3(x_3^{k_2,k_3}) + \cdots + \sum_{k_T=1}^{K_T}\left[p_T^{k_2,\dots,k_T} f_T(x_T^{k_2,\dots,k_T})\right]\right]\right]\right\}$$

s.t.

$$\begin{array}{llll}
A_1x_1 & & = b_1 & \\
B_2^{k_2}x_1 + A_2^{k_2}x_2^{k_2} & & = b_2^{k_2} & k_2 = 1,\dots,K_2, \\
\quad B_3^{k_2,k_3}x_2^{k_2} + A_3^{k_2,k_3}x_3^{k_2,k_3} & & = b_3^{k_2,k_3} & k_t = 1,\dots,K_t, \\
\qquad\ddots & \ddots & \vdots & \\
\qquad\qquad B_T^{k_2,\dots,k_T}x_{T-1}^{k_2,\dots,k_{T-1}} + A_T^{k_2,\dots,k_T}x_T^{k_2,\dots,k_T} & & = b_T^{k_2,\dots,k_T} & k_t = 1,\dots,K_t, \\
 & & & t = 2,\dots,T,
\end{array}$$

$$l_1 \le x_1 \le u_1,\quad l_t \le x_t^{k_2,\dots,k_T} \le u_t,\quad t = 2,\dots,T, \tag{6}$$

in which a vector of decision variables corresponds to each node in the decision tree.

In concise matrix notation, the multistage linear problem is

$$\min_{\hat{x}} \hat{c}'\hat{x} \quad \text{s.t.} \quad \hat{A}\hat{x} = \hat{b},\ \hat{x} \ge 0,$$

where

$$\hat{c} := (c_1', (p_2^1c_2^1)', \dots, (p_2^{K_2}c_2^{K_2})', (p_3^{1,1}c_3^{1,1})', \dots, (p_3^{K_2,K_3}c_3^{K_2,K_3})', \dots, (p_T^{1,\dots,1}c_T^{1,\dots,1})', \dots, (p_T^{K_2,\dots,K_T}c_T^{K_2,\dots,K_T})')',$$

$$\hat{x} := (x_1', x_2^{1\prime}, \dots, x_2^{K_2\prime}, \dots, x_T^{1,\dots,1\prime}, \dots, x_T^{K_2,\dots,K_T\prime})',$$

$$\hat{b} := (b_1', b_2^{1\prime}, \dots, b_2^{K_2\prime}, \dots, b_T^{1,\dots,1\prime}, \dots, b_T^{K_2,\dots,K_T\prime})',$$

$$\hat{A} := \begin{pmatrix}
A_1 & 0 & \cdots & & & \cdots & & & & 0 \\
B_2^1 & A_2^1 & 0 & \cdots & & \cdots & & & & 0 \\
\vdots & & \ddots & & & \cdots & & & & \vdots \\
B_2^{K_2} & 0 & \cdots & A_2^{K_2} & 0 & & & \cdots & & 0 \\
0 & B_3^{1,1} & & & A_3^{1,1} & 0 & \cdots & & & 0 \\
\vdots & \vdots & & & & \ddots & & & & 0 \\
0 & B_3^{1,K_3} & & & \cdots & & & & \cdots & 0 \\
 & & & B_3^{K_2,K_3} & \cdots & & A_3^{K_2,K_3} & & \cdots & 0 \\
\vdots & \vdots & & & \vdots & & \vdots & & & \vdots \\
 & & & & & & B_4^{K_2,\dots,K_4} & & & \\
 & & & & \cdots & & \vdots & & \ddots & 0 \\
0 & 0 & \cdots & & & & & B_T^{K_2,\dots,K_T} & \cdots & A_T^{K_2,\dots,K_T}
\end{pmatrix}.$$

In a T-stage problem with $\Pi_{s=2}^{t} K_s$ nodal problems in stage t with associated $(m_t \times n_t)$ submatrix dimension, this results in $\hat{c} \in \mathbb{R}^{\hat{n}}$, $\hat{x} \in \mathbb{R}^{\hat{n}}$, $\hat{b} \in \mathbb{R}^{\hat{m}}$, $\hat{A} \in \mathbb{R}^{\hat{m}\hat{n}}$ (see figure 3), where $\hat{n} := n_1 + K_2(n_2 + K_3(n_3 + \cdots + K_T n_T))$ and $\hat{m} := m_1 + K_2(m_2 + K_3(m_3 + \cdots + K_T m_T))$, eventually generating *very large* problems as the number of scenarios increases.

The (LP)

$$\min_{\hat{x}} \{\hat{c}'\hat{x} \mid \hat{A}\hat{x} = \hat{b}, \hat{x} \geq 0\} \tag{7}$$

and its dual (DLP)

$$\max_{\hat{\pi}} \{\hat{b}'\hat{\pi} \mid \hat{\pi}'\hat{A} \leq \hat{c}'\} \tag{8}$$

may be considered for solution with simplex or interior point methods.

The derivation of a complete determinstic equivalent linear, or with a simple extension quadratic, program from the SMPS files generated by STOCHGEN is achieved with the STD2MPS utility created by Gassmann [32].

The main problems in solving either (7) or (8) are related to the extremely large dimensions that such an LP may attain. In the sequel, we consider different algorithms for the solution of large-scale programs and compare the results from *state-of-the-art* commercially available solvers with the performance of two versions of the MSLiP implementation of the nested Benders decomposition algorithm.

5.1. Simplex

Modern implementation of the revised simplex method, such as in IBM's OSL Release 2 [42] (1992) and CPLEX simplex [16] (1994), incorporate many of the theoretical results of research in this area, making commercially available two excellent versions of this technique for the solution of LP problems.

The performance of the simplex method, recognized as very efficient for the solution of relatively small dense problems, may be considered to depend on:

- the implementation of a *scaling* procedure, before phase 1 of the algorithm takes place, and an efficient *basis factorization* and update after the dimension of the original matrix has been reduced by *preprocessing*,
- the determination of a convenient advanced starting basis through a *crash* procedure,
- the adoption of an efficient *pricing strategy* during the solution phase, possibly with multiple pricing, in order to determine the pivot direction from the current solution point.

The scaling procedure generally results in substantial improvements in terms of the pivot path and numerical stability. The OSL and CPLEX libraries both provide a subroutine for scaling the original entries of the constraint matrix. In the solution of the WATSON problems, we always make use of the scaling option.

The presolution of LP problems reduces the dimension of the original matrix, say $\hat{A}$ of (7), by eliminating redundant rows and columns; both OSL and CPLEX have quite effective presolvers. However, CPLEX, by implementing a two-phase method with a presolver-aggregator, effects substantial reductions of the original matrix. This is the distinguishing feature of the CPLEX library which markedly improves the overall solution time for large problems – for both simplex and interior point algorithms.

Once the matrix dimension has been reduced and the columns for the basic and non-basic variables identified, the basis submatrix is factorized and inverted. Complete refactorization takes place regularly thereafter during the solution of the problem. In this respect, the performance of the algorithm may be affected by the number of factor update iterations allowed between refactorizations of the basis matrix. Levkovitz and Mitra [49] report that in order to avoid numerical instability, the *Markowitz merit count* approach, which guides pivot selection during the basis factorization, can improve the numerical properties of the solution. The CPLEX library provides a subroutine that enables the user to select the number of iterations between refactorizations and to fix the Markowitz tolerance. Similarly, OSL allows the selection of a parameter for the frequency of the basis refactorization, but no use is made of the Markowitz approach to pivot selection during basis factorization.

In order to reduce the number of iterations required to achieve the optimum, a good starting basis is essential; this is usually realized by adopting the crash procedure before the start of the solution phase. In all the test problems, we have made use of the crash subroutines provided by OSL and CPLEX.

Finally, the efficiency of the simplex algorithm depends on the pricing strategy that has been adopted in order to select the direction of the step from one vertex to the next. Both OSL and CPLEX provide a parameter for the choice of a convenient pricing method for the problem under consideration. Particularly for large sparse problems, DEVEX pricing (Harris [40]) turns out to be quite effective and can be used with both CPLEX and OSL, its idea being that the computation of the reduced cost vector for the complete set of basic variables is appropriately and efficiently scaled at each iteration. Examining the complete set of columns with any pricing method can result in unnecessary extra work when the coefficient matrix is large, so that heuristics limiting the reduced cost computations determining the pivot column may be much more efficient in this case.

In the implementation of the simplex method with OSL, we have used a pricing method based on pricing a random subset of basic columns with an automatic switch to DEVEX on the full set. Using CPLEX, the best results have been achieved by selecting DEVEX pricing for very large problems and allowing the hybrid ordinary reduced cost-DEVEX pricing method for small and medium problems. Utilizing the work of Goldfarb and Reid [35], CPLEX has also implemented a steepest edge crossing procedure which can be thought of as a logical extension of the heuristics introduced by DEVEX pricing. However, in our different tests we could not find evidence of improved solution times utilizing this method.

5.2. *Interior point method*

Interior point methods have become increasingly popular after Karmarkar's 1984 contribution [44]. We utilize a primal/dual barrier method with predictor–corrector. Following Mehrotra's study [55], the algorithm is based on approximation of primal–dual trajectories in the interior of the feasible region, as was first implemented in the OB1 code and then incorporated, with some improvements, in CPLEX. The general theory of primal/dual interior point (P/D IP) methods is due to Megiddo [54].

The two implementations of the algorithm we used differ in their coding language – FORTRAN for OB1 and C for CPLEX – and significantly in the power of the pre-processing subroutines, with a very effective presolver/aggregator implemented by CPLEX.

The search direction method adopted by both libraries (see Lustig et al. [52]) may be regarded as an attempt to generate steps in the feasible region according to first-order information, with a following "centralizing" step as a correction.

The primal/dual IP method is based on the simultaneous solution of the primal and dual problems (7) and (8) with a progressive reduction of the duality gap at each iteration. A good property of P/D IP methods is that the convergence rate for these algorithms in general is seen to be independent of the problem size.

In requiring disk storage for the primal and the dual problem simultaneously, however, current implementations of P/D IP methods are memory-intensive and this may seriously compromise the solution of very large problems. In this respect, unlike OB1, CPLEX has an efficient dynamic memory allocation system and is able to handle very large problems – although *not* the largest in this study.

The P/D IP method requires the computation at each iteration of the Cholesky factorization of a matrix of the form ADA', where D is a diagonal matrix whose nature is determined by the choice of the algorithm. This factorization absorbs most of the computational effort and the time spent on each iteration is substantially affected by the method used, which needs to be fast and memory requirement minimizing. These properties are influenced by the matrix ordering procedure before the Cholesky factor is actually determined. On the IBM/RS6000 platform used in our computations, the *multiple minimum degree* procedure (Liu [50]) has so far proved to be preferable (cf. also Lustig et al. [52]) to the *minimum local fill ordering* (Duff et al. [27]), which is also available in both the OB1 and CPLEX systems. In this respect, Berger et al. [3] have claimed that Liu's procedure – giving rise to considerable fill-in in the Cholesky factor – is in general quite inefficient, particularly for multistage stochastic programs in split-variable formulation (i.e. with explicit nonanticipative constraints on the decision variables). This evidence lead to the development of an alternative ordering heuristic termed *tree-dissection*. Comparing the above-mentioned methods, however, we have so far had empirical confirmation of the superiority for large sparse matrices of multiple minimum degree, due largely to the fact that solution times for problems formulated with explicit nonanticipative constraints are in general extremely slow relative to the so-called *compact* formulation of (6).

This conclusion also holds in the case of the quadratic problems solved by the CPLEX interior point QP solver, which was beta-tested on the Watson test problems. In this case, despite the increased density of the Cholesky factor and a significant increase in the fill-in for the different test-problems, the ordering procedure appears to be very effective and the solution times, given in table 3, are competitive with the performance of the best LP solvers.

In IP algorithms, the determination of a starting point again follows the reduction of the original coefficient matrix by the elimination of redundant rows and columns. As opposed to the presolver-aggregator implemented by CPLEX, OB1 bases the pre-processing of the problem on simple inferences; the reductions achieved by OB1 for the different problems are generally considerably less than those achieved by the OSL and the CPLEX presolvers.

Levkovitz and Mitra [49] report results of experiments on the determination of an efficient starting point by selecting different starting points in the dual space, but this remains, in general, an open problem.

5.3. Nested Benders decomposition

The nested Benders decomposition method implemented by MSLiP (Gassmann [32–34]) specifically applies to the solution of multistage stochastic linear programs in the DRP form. Unlike the methods considered above, it does not require the simultaneous determination of a solution for the complete deterministic equivalent problem. Instead, it involves an efficient decomposition procedure by extending to the multistage case the Benders decomposition method established for the two-stage case.

A good introduction to the application of Benders decomposition to stochastic linear programs can be found for two- or three-stage problems in, respectively, Van Slyke and Wets [63] and Birge [6] and, for the multistage case, in Gassmann [33].

The idea behind nested Benders decomposition is to express at each stage t, as the impact of stages $t+1,\ldots,T$, the expected future costs in terms of a variable θ_t and by “cuts” – linear necessary conditions for both *feasibility* and *optimality* – expressed only in terms of the current stage variables. It is thus a special type of cutting plane algorithm in which the convex constraint $\theta_t \geq \upsilon_{t+1}$ in (4) is replaced by a polyhedral description which is updated as more information becomes available from the dual problems at the descendants of current nodes. In the MSLiP implementation, each nodal subproblem is solved with a (1974) simplex subroutine created by Pfefferkorn and Tomlin [33].

The efficiency of nested Benders decomposition depends on the number of subproblems to be solved – each one corresponding to a node in the scenario tree, the performance in solving each subproblem and the order in which the nodes of the tree are visited.

The number of subproblems to solve depends in general on the number of cuts generated by dual information. A high number of feasibility cuts is likely to cause numerical instability and eventually jeopardizes the convergence of the implemented algorithm (see below).We show in section 6 that the performance of the nested Benders cutting plane method depends critically on the stochastic properties of the problem in terms of the number of feasibility cuts generated. Scenarios produced in conditional mode lead to a lower level of stochasticity as measured by the *expected value of perfect information* (EVPI) value at the root node of a problem (Dempster and Gassmann [23], Dempster and Thompson [26]) and imply a significant decrease in the number of cuts generated and the overall solution times.

In order to improve the numerical properties of the algorithm, the OSL Release 2 simplex solver has been implemented in the MSLiP code, introducing the possibility of preprocessing the subproblems and selecting adequate crash procedures (Thompson [62]). As demonstrated in the following section, this has improved the stability of the method without affecting solution times. If stages are aggregated to create larger subproblems more efficiently solved by OSL simplex, considerable speed-ups result with MSLiP-OSL (see Dempster and Thompson [25, 26]).

Three alternative approaches may be adopted for selecting the sequence of subproblems to be solve. In the context of deterministic dynamic problems, Wittrock [67] suggested that the fastest way to pass information between nodal subproblems would be to change direction as little as possible and to move along the chain until a direction is stopped, either because the horizon has been reached or because an infeasible problem has been found. This method applied to trees is referred to as the *fast forward fast back* (FFFB) approach. Alternatively, a “forward first” or “backward first” approach as described by Gassmann [33] may be considered. The FFFB protocol has been shown to give the best results and has been adopted in our experiments.

6. Computational results

Results are presented for different solution methods applied to two sets of 10-stage WATSON problems with "independent" and conditionally generated scenarios (see section 3). All problems were run in double precision on an IBM RS6000/590 with 128 MB of main memory and 2.5 GB of disk running under AIX 3.2.5.

By expanding the sample space $\Omega := \Pi_{t=1}^{10}\Omega_t$ of the multidimensional random process, we can generate successive problems of increasing complexity and derive a comparative evaluation of the algorithms discussed in section 5. A check on the actual randomness, or *stochasticity*, of the problems is made by computing the relative EVPI (as a ratio with respect to the objective value) at the root node of each problem (see table 2).

In table 1, we give details of the two sets, of 11 and 13 test problems, respectively, that have been generated by increasing the number of scenarios – both "independently" and conditionally generated. These 24 problems represent the basis of the comparative evaluation of the solution algorithms for our long-term portfolio problem. Statistics are reported regarding the original dimension of the complete deterministic equivalent problem and the reduction achieved by preprocessing using the different libraries.

Thanks to a very rich set of financial data and the reliability of the SSL problem generator (given the memory and hard-disk constraints of the IBM platform), we have been able to generate 10-stage problems with up to 2688 possible realizations of the process at the horizon. Additional sample paths give increasing confidence in the representation of the portfolio problem, for which we eventually wish to study the sensitivity of the optimal policy to very small variations in the parameters underlying market rates of return and borrowing conditions.

In figure 5, we depict the almost linear relationship between the dimension of the WATSON problems (expressed in terms of the number of variables) and the number of scenarios.

We have tested the performance of the preprocessing subroutines provided by OSL, OB1, and CPLEX for the different problems. Both OSL and CPLEX achieve a significative reduction in the dimension of the original problems, but the presolver/aggregator of CPLEX proves to be extremely effective, with a remarkable impact on solution times.

From a numerical viewpoint, long-term portfolio decision problems such as the WATSON problems result in stochastic programs whose general form is characterized by:

- A very sparse coefficient matrix, with entries representing financial returns and cash inflows and outflows per unit of value.
- A generally small feasible region, with most of the constraints specified by upper and lower bounds on borrowing and investment decisions binding at each stage, effectively reducing the size of the set of feasible policies to be a direct function of the initial wealth of the decision maker.

Table 1

Definition of WATSON test problems.

Number of scenarios	16	32	64	128	256
Tree branching structure	$2^4\ 1^5$	$2^5\ 1^4$	$2^6\ 1^3$	$2^7\ 1^2$	$2^8\ 1^1$
Number of nodes	111	191	319	511	767
Original matrix	4573×8401	8413×15553	15101×28097	26237×49153	43517×82177
Entries	21,368	39,848	72,648	128,648	218,888
Density	0.0556%	0.0305%	0.0171%	0.00997%	0.00612%
OSL reduced matrix	2886×5961	5318×10953	9478×19465	16134×33033	25606×52233
Entries	15,359	28,511	51,359	88,863	144,927
Density	0.0893%	0.0489%	0.0278%	0.01667%	0.01083%
OB1 reduced matrix	3970×706	7362×8430	13314×23366	23298×40326	38914×65798
Entries	19,525	36,437	66,389	117,269	198,421
Density	0.06964%	0.05871%	0.02134%	0.01248%	0.00775%
CPLEX reduced matrix	837×3897	1630×7230	3101×13025	5981×22753	11517×37889
Entries	16,669	30,799	55,099	94,255	154,475
Density	0.5119%	0.2613%	0.1341%	0.0693%	0.0354%

Number of scenarios	512	768	1024	1152
Tree branching structure	2^9	$3^1\ 2^8$	$4^1\ 2^8$	$3^2\ 2^7$
Number of nodes	1023	1534	2045	2299
Original matrix	67069×128001	100598×191994	134127×255987	150869×287949
Entries	351,228	516,784	689,156	775,288
Density	0.00406%	0.00267%	0.002007%	0.001784%
OSL reduced matrix	35846×72713	53768×109067	71690×145421	80642×163577
Entries	244,765	367,140	489,515	550,653
Density	0.00939%	0.00626%	0.00469%	0.004174%
OB1 reduced matrix	60418×97798	90624×146692	120830×195586	135918×220006
Entries	314,389	471,572	628,755	707,285
Density	0.00532%	0.00354%	0.00266%	0.002365%
CPLEX reduced matrix	21757×58113	32636×87167	43513×116221	48950×130733
Entries	239,403	359,106	478,801	538,650
Density	0.0189%	0.0126%	0.00941%	0.00842%

Number of scenarios	1536	1920	2304	2688
Tree branching structure	$4^1\ 3^1\ 2^7$	$5^1\ 3^1\ 2^7$	$6^1\ 3^1\ 2^7$	$7^1\ 3^1\ 2^7$
Number of nodes	3065	383	4597	5363
Original matrix	201155×383927	251441×479905	301828×575983	352114×671961
Entries	1,033,268	1,292,258	1,578,528	1,841,528
Density	0.001338%		0.001071%	0.0009073%
OSL reduced matrix	107522×218101	134402×272626	out of memory	out of memory
Entries	734,199	917,745	–	–
Density	0.003131%	0.002504	–	–
OB1 reduced matrix	181222×293338	226526×366652	out of memory	out of memory
Entries	943,039	1,178,193	–	–
Density	0.001774%	0.001418%	–	–
CPLEX reduced matrix	65266×174309	81582×217886	out of memory	out of memory
Entries	718,193	897,750	–	–
Density	0.00631%	0.00505%	–	–

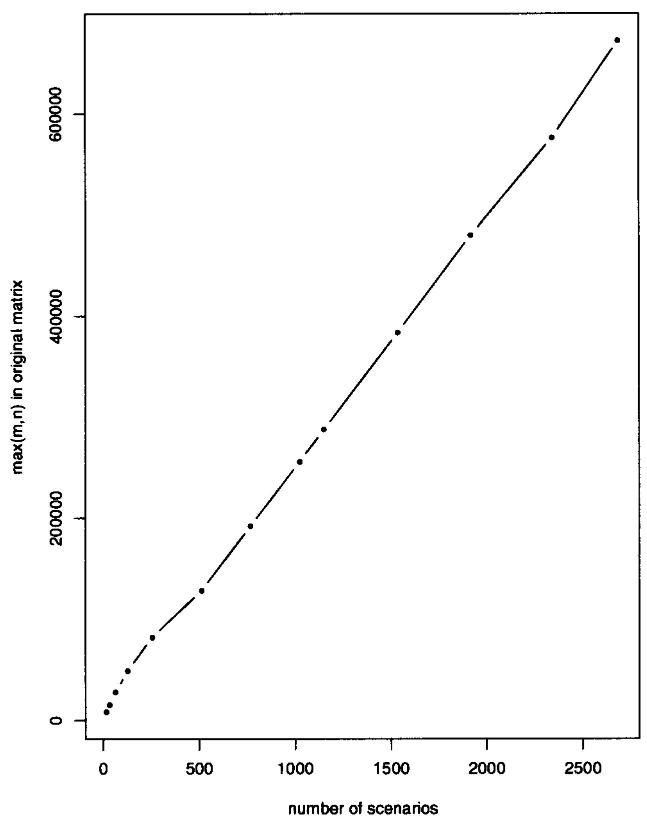

Figure 5. WATSON test problem dimensions.

- At each stage, the information provided by the possible subsequent realizations of the random data process strongly affects the state of the system (in the Watson case identified with the current portfolio value process).

Interior point methods are particularly efficient in this setting. For the different test problems, CPLEX barrier converges to the solution in a number of iterations which is independent of the size of the problem. This property – generally attributed to P/D IP methods – is also possessed by MSLiP and MSLiP-OSL; in both cases, the number of iterations is computed on the basis of the number of times the *master* (root-node) problem is solved in the course of the solution procedure.

A major advantage of nested Benders decomposition relative to P/D IP methods is that a negligible amount of memory needs to be allocated during the solution procedure. The WATSON.10.1920 problem, for instance, has not been solved by OB1 because of the lack of memory space, whereas CPLEX barrier, thanks to its dynamic memory allocation, has succeeded in finding an optimum, although requiring roughly 85% of the total memory (128 MB) available. On the other hand, for the same problem, MSLiP and MSLiP-OSL needed only about 21% of the memory to be allocated.

Table 2

WATSON numerical results: independent scenario generation.

Number of scenarios	16	32	64	128	256	512
Obj (£ m)	2158.7519	1866.398	2310.51	1637.8125	2000.838	1959.635
Relative EVPI	12.64%	28.31%	30.43%	42.75%	36.63%	44.63%
OSL simplex	11.6”	47”	188”	707”	2253”	6510”
no. iterations	1141	2696	5721	11747	23089	45457
OB1 pred./corr.	12.1”	27.1”	63.4”	128.2”	210.7”	394.9”
no. iterations	41	52	68	80	73	85
CPLEX simplex	6.37”	17.6”	56.4”	196.1”	674.7”	2600.2”
no. iterations	2257	4623	10623	20126	36315	50512
CPLEX barrier	5.77”	11.79”	26.27”	58.9”	125.8”	248.8”
no. iterations	23	26	35	44	50	63
CPLEX QP barrier	5.1”	16.3”	28.6”	74.3”	113.3”	400.4”
no. iterations	19	31	28	46	34	67
MSLiP-OSL	9.87”	20.03”	43.5”	83.0”	157.9”	204.6”
no. iterations	6	7	9	11	12	12
MSLiP	9.93”	19.5”	37.6”	65.2”	131.2”	202.9”
no. iterations	6	8	11	12	22	12
Number of scenarios	768	1024	1152	1536	1920	
Obj (£ m)	1813.105	1926.787	1687.92	1790.276	1778.954	–
Relative EVPI	45.33%	46.53%	63.45%	61.86%	62.13%	
OB1 pred./corr.	658.6”	1002.2”	1111.1”	2280.7”	n.a.	–
no. iterations	96	111	108	123	–	
CPLEX simplex	5380.5”	–	–	–	–	–
no. iterations	72206	–	–	–	–	
CPLEX barrier	397.5”	573.6”	591.5”	833.4”	1137.6”	–
no. iterations	66	70	63	67	73	
CPLEX QP barrier	510.1”	735.7”	926”	2135.8''	n.a.	–
no. iterations	55	60	67	90	–	
MSLiP-OSL	303.1”	413.8”	937.9”	876.25”	1079.74”	–
no. iterations	10	12	7	15	18	
MSLiP	304.4''	402.4''	unsolved	unsolved	unsolved	–
no. iterations	7	15	–	–	–	

OSL simplex has been tested up to the solution of the 512-scenario problem and CPLEX simplex up to 768 scenarios; at these points, the sizes of the problem made simplex already very inefficient. All other problems, up to system sustainability, have been tried with MSLiP, MSLiP-OSL, CPLEX barrier and OB1.

The numerical results shown in table 2 refer to the complete set of “independent” problems generated with the original Watson simulator. Solution accuracy agreement of the various methods is displayed in the number of decimal places shown for each problem in the objective value (at least 2 decimal places) (see also table 3).

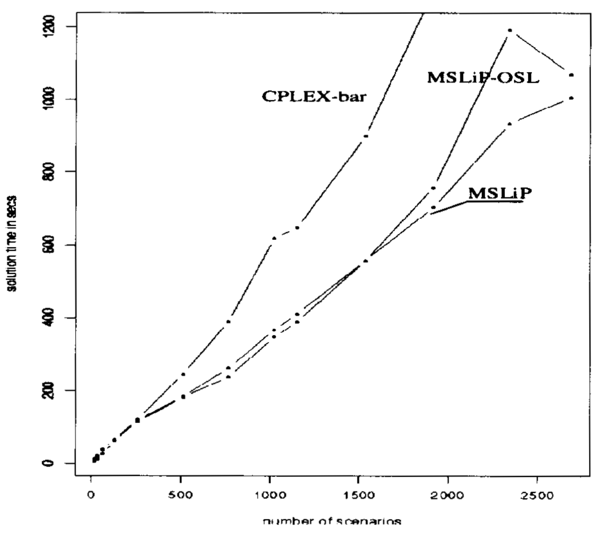

Figure 6. WATSON problem comparative solution times: conditional scenarios.

Table 3

WATSON numerical results: conditional scenario generation.

Number of scenarios	16	32	64	128	256	512	768
Obj (£ m)	2401.875	2167.624	2027.4293	1879.432	1841.2985	1787.1977	1780.126
Relative EVPI	10.47%	13.85%	19.51%	24.85%	28.19%	32.87%	30.26%
CPLEX barrier	5.89”	10.94”	25.71”	60.37”	120.34”	245.52”	389.55”
no. iterations	28	27	33	45	48	62	63
MSLiP	10.23”	18.28”	35.92”	66.44”	118.08”	184.47”	261.93”
no. iterations	9	8	9	12	11	10	10
MSLiP-OSL	12.48”	21.42”	37.51”	64.58”	113.28”	182.14”	238.01”
no. iterations	8	8	9	8	9	11	9
Number of scenarios	1024	1152	1536	1920	2304	2688	–
Obj (£ m)	1791.176	1725.53	1693.474	1769.45	1664.66	1679.921	–
Relative EVPI	39.20%	33.83%	45.37%	37.04%	47.48%	45.00%	–
CPLEX barrier		618.20”	647.71”	898.54”	1305.74”	–	–
no. iterations	75	74	74	82			
MSLiP	367.94”	411.30”	557.71”	704.03”	933.69”	1006.59”	–
no. iterations	11	11	12	12	16	15	
MSLiP-OSL	347.85”	389.23”	557.41”	758.85”	1192.18”	1070.84”	–
no. iterations	10	10	11	12	17	13	

All solution methods have been tested on the “independent” problems shown in table 2, while only the more efficient solvers are considered in the case of problems shown in table 3 with scenarios generated in conditional mode.

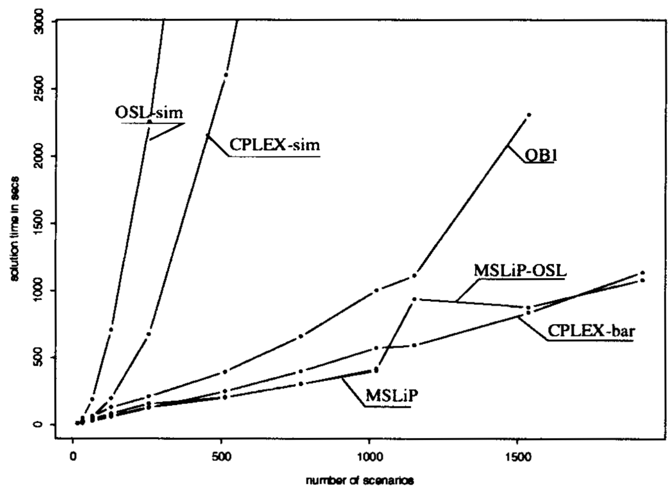

Figure 7. WATSON problem comparative solution times: independent scenarios.

Table 4

Performance of MSLiP and MSLiP-OSL on selected test problems: independent versus dependent scenario generation.

Number of scenarios	128	256	512	1024	1536	1920
MSLiP						
Solution time:						
• Independent scenarios	65.16”	131.07”	202.88”	402.42”	unsolved	unsolved
• Conditional scenarios	66.44”	118.18”	184.47”	367.94”	557.71”	704.03”
No. iterations:						
• Independent scenarios	12	22	12	15	–	–
• Conditional scenarios	12	11	10	11	12	12
Optimality cuts/total:						
• Independent scenarios	1697/1699	2842/2887	3762/3795	7375/7469	–	–
• Conditional scenarios	1549/1549	2403/2405	2973/2977	5922/5922	8652/8652	10756/10763
No. of subproblems:						
• Independent scenarios	7398	19696	15605	35247	–	–
• Conditional scenarios	7805	10090	12503	25988	38134	51646
MSLiP-OSL						
solution time:						
• Independent scenarios	82.99”	157.9”	204.64”	413.76”	876.25”	1079.74”
• Conditional scenarios	64.58”	113.3”	182.14”	347.85”	557.41”	758.85”
No. iterations:						
• Independent scenarios	11	13	12	12	15	18
• Conditional scenarios	8	9	11	10	11	12
Optimality cuts/total:						
• Independent scenarios	1474/1474	2262/2273	2855/2855	5723/5729	8726/8880	10942/11153
• Conditional scenarios	1365/1365	2043/2043	2577/2577	5330/5330	7595/7595	9734/9734
No. of subproblems:						
• Independent scenarios	6708	11700	13977	27936	59570	77657
• Conditional scenarios	5104	8290	12741	23570	41874	52299

There are evident variations in the values of both the objective and the relative EVPI which appear to be independent of the number – but clearly not the nature – of randomly generated scenarios. A priori, we expect more conservative strategies and lower terminal wealths as a result of increasing number of scenarios! Decomposition methods, unlike solution methods based on the complete certainty equivalent problem, are affected by the stochastic properties of the underlying random process.

Tables 2 and 3 show the impact of the relative EVPI values at the root node on the different procedures: reduced randomness of the conditional problems improves speed of the solution and stability of the decomposition method, making MSLiP much faster then CPLEX barrier. Very large problems – previously unsolved by MSLiP – can be solved with conditionally generated scenarios since the number of feasibility cuts needed decreases.

The IP-QP solver of CPLEX (which was made available to us by CPLEX for beta-testing) appears to be competitive with respect to the other solvers (cf. table 2) and becomes relatively slow only for extremely large problems. The WATSON.10. 1920 QP problem could not be solved, again because of lack of memory. The target terminal wealth $\tilde{W}$, for a utility function of the form $(W - \tilde{W})^2$ has been defined for all test problems to be slightly higher then the corresponding linear optimal value, in order to make sensible comparisons between solution times. The solution times reported in table 2 thus refer to a set of problems that have been run in order to have explicit comparisons with LP solution times. In general, for problems with a quadratic objective and the same number of scenarios as with a linear objective, a reduction of the target value $\hat{W}$ implies a decrease in solution times and in some cases, an IP-QP solution time even lower than the corresponding LP solution time by the CPLEX IP-LP solver.

The speed of convergence of MSLiP (either version) to the optimum depends in general on the number of subproblems to solve and this, in turn, may depend on the number of cuts generated before the algorithm terminates. The original version of MSLiP has proven unable to solve very large problems because of looping due to the excessive generation of feasibility cuts. On the other hand, MSLiP-OSL, thanks to the accuracy of the subproblem solver, proves a fast and efficient solver on such problems. Table 4 allows more insight into the comparative performance of the two implementations of nested Benders decomposition in the two cases of scenarios conditionally or unconditionally generated.

Summarizing, we see that for nested Benders decomposition a reduced level of stochasticity, regardless of the implementation considered, results in:

(i) a lower number of feasibility cuts generated throughout the solution procedure,

(ii) a significant reduction of the number of subproblems solved,

(iii) faster CPU times with an almost linear relationship between solution time and problem dimension.

7. Conclusions

We have shown how a dynamic portfolio problem can be conveniently represented as a discrete time linearly constrained stochastic optimization problem which can be formulated as a dynamic (multistage) stochastic program with recourse in which the principal decisions correspond to portfolio rebalancing over time.

The main steps in the formulation and solution of such decision problems have been described, with an emphasis on both the financial assumptions implied by the representation adopted and the extended numerical work needed for correct model generation and solution of the resulting stochastic optimization problem. Many sophisticated tools are available for generating, solving, analyzing and simulating such models, eventually resulting in representing the full corporate and regulatory environment in them (as, for example, is the case in the Russell–Yasuda Kasai model [10]).

We conclude with some directions for future research under active pursuit by our group at the time of writing:

- The EVPI value computed at each node of the scenario tree [23, 26] provides a useful measure of the randomness embedded in the remaining stochastic decision problem at each stage. An EVPI-based sampling procedure interfaced with the STOCHGEN subroutine library has been designed by Dempster and Corvera Poiré [13–15], which can be effective in the solution of financial planning problems typically characterized as highly stochastic systems. A marginal EVPI-based [22] sampling procedure is also under development.
- The two implementations of nested Benders decomposition (MSLiP and MSLiP-OSL) tested provide very efficient solvers in the case of linear problems. An extension of the method to handle nonlinear objectives has been developed which can significantly extend the applicability of the method to general linearly constrained decision problems with convex objective functions representing arbitrary utility functions – i.e. attitudes to risk – in financial planning problems.
- The solution of these very large and complex problems needs to be followed by a detailed computer-based analysis of the results in order to supply conveniently represented information to the decision maker. This is a necessary step towards the implementation of a user-friendly decision support system and can be expected to make extensive use of visualization and other sophisticated software tools [24].
- The general implications and advantages of the DRP formulation studied here relative to other portfolio management paradigms – such as the well-established static Markowitz representation which still dominates the financial services industry or the dynamic stochastic control formulation with continuous rebalancing [9] – also represents an area of increasing interest. We shall shortly be in a position to report on such a comparative study conducted under the auspices of the Frank Russell Company.

Acknowledgements

The research reported here represents one output of a long-term research project conducted by the Finance Research Group, first at Dalhousie University, subsequently at the University of Essex and now at the University of Cambridge. The second author is grateful for considerable financial support over the years from these universities, from the Natural Science and Engineering Research Council of Canada, and the Engineering and Physical Sciences Research Council of the UK. Our colleagues, Horand Gassmann, Xavier Corvera Poiré and Robin Thompson, have provided both crucial software development support for the results reported here and lively discussion regarding their presentation. We owe a sincere debt of gratitude to Professor David Wilkie of Watson Wyatt Limited, who suggested the pension fund managers' problem resulting in the Watson version of the CALM model and the WATSON problems and who generously gave us his time, simulation data from the Wilkie investment model and finally his FORTRAN code for its generation. We also wish to acknowledge with pleasure the supply and tuning of the OB1 Fortran code by Professor David Shanno of Rutgers University.

Appendix: The CALM model

Notation

T	horizon.
$s, t = 1, \dots, T+1$	time periods.
$i = 1, \dots, I$	asset type.
$j = 1, \dots, J$	liability type.
$k = 1, \dots, K$	riskless instrument type.
$\omega = 1, \dots, \lvert\Omega\rvert$	data paths (scenarios).
Ω^t	all distinct paths at time t ($\Omega := \Omega^T$).
ω^t	data path history to the beginning of period t.

Decision variables

$x^+_{it}(\omega^t)$	amount purchased of asset i in period t.
$x_{ist}(\omega^t)$	amount held of asset i in period t which was purchased in period $s \le t$.
$x^-_{ist}(\omega^t)$	amount sold of asset i in period t which was purchased in period $s < t$.
$y^+_{jt}(\omega^t)$	amount incurred of liability j in period t .
$y_{jst}(\omega^t)$	amount held of liability j in period t which was incurred in period $s \le t$.
$y^-_{jst}(\omega^t)$	amount discharged of liability j in period t which was incurred in period $s < t$.

$z^+_{kt}(\omega^t)$	amount held of riskless asset k in period t.
$z^-_{kt}(\omega^t)$	amount owed of riskless asset k in period t.
$\delta_{x^+_{it}}(\omega^t)$, $\delta_{x^-_{ist}}(\omega^t)$	binary action of buying (selling) asset i in period t ($s < t$).
$\delta_{y^+_{jt}}(\omega^t)$, $\delta_{y^-_{jst}}(\omega^t)$	binary action of incurring (discharging) liability j in period t ($s < t$).
w	net wealth at the beginning of period $T + 1$.

Parameters

$r_{ist}(\omega^t)$	cash return in period t on asset i purchased in period $s < t$ and held in period $t - 1$.
$s_{jst}(\omega^t)$	unit cost in period t of liability j incurred in period $s < t$ and held in period $t - 1$.
$r^+_{kt}(\omega^t)$	return in period t on riskless asset k held in period $t - 1$.
$r^-_{kt}(\omega^t)$	unit cost of borrowing riskless asset k in period $t - 1$.
$e_{it}(\omega^t)$, $e_{jt}(\omega^t)$	lump sum transaction cost of purchasing (incurring) asset i (liability j) in period t.
$f_{it}(\omega^t)$, $f_{jt}(\omega^t)$	unit cash outflow (inflow) upon purchasing (incurring) asset i (liability j) period t.
$g_{ist}(\omega^t)$, $g_{jst}(\omega^t)$	unit cash inflow (outflow) upon selling (discharging) in period t asset i (liability j) bought (incurred) in period $s < t$.
$h_{ist}(\omega^t)$, $h_{jst}(\omega^t)$	lump sum transaction cost of selling (discharging) in period t asset i (liability j) bought (incurred) in period $s < t$.
$\rho_{it}(\omega^t)$, $\rho_{jt}(\omega^t)$	exchange rate appropriate to asset i (liability j).
$\rho_{kt}(\omega^t)$	(riskless asset k) held in period $t - 1$.
$\upsilon_{is}(\omega)$, $\upsilon_{js}(\omega)$	market value at the horizon ($T + 1$) of asset i (liability j) purchased (incurred) in period $s \leq T$.
$\underline{X_{it}}$, $\overline{X_{it}}$	limits on investment on asset i in period t.
$\underline{Y_{jt}}$, $\overline{Y_{jt}}$	limits on incurring liability j in period t.
$\overline{Z_{kt}}$	short position limit on riskless asset k in period t.
$\overline{X^+_t}$, $\overline{Y^+_t}$	maximum new investment (liabilities) in period t.
$\overline{Y_t}$	maximum liability in period t.

Objective

$\max \mathbb{E}u(\boldsymbol{w})$ (expected utility of terminal wealth).

$$u \in C^2(\mathbb{R}) \qquad \frac{u'(\boldsymbol{w})}{-u''(\boldsymbol{w})} := a\boldsymbol{w} + b.$$

Constraints

$$\sum_{s=1}^{T}\left[\sum_{i=1}^{I} \boldsymbol{v}_{is}\boldsymbol{x}_{isT} - \sum_{j=1}^{J} \boldsymbol{v}_{js}\boldsymbol{y}_{jsT}\right]$$

$$+ \sum_{k=1}^{K}[(1+\boldsymbol{r}^{+}_{k(T+1)})\boldsymbol{z}^{+}_{kT} - (1+\boldsymbol{r}^{-}_{k(T+1)})\boldsymbol{z}^{-}_{kT}] = \boldsymbol{w} \quad \text{(terminal wealth).}$$

$$\sum_{i=1}^{I}\rho_{it}\left[-\boldsymbol{e}_{it}\boldsymbol{\delta}_{x^{+}_{it}} - \boldsymbol{f}_{it}\, bf\, x^{+}_{it} + \sum_{s=1}^{t-1}(\boldsymbol{r}_{ist}\boldsymbol{x}_{is(t-1)} + \boldsymbol{g}_{ist}\boldsymbol{x}^{-}_{ist} - \boldsymbol{h}_{ist}\boldsymbol{\delta}_{x^{-}_{ist}})\right]$$

$$-\sum_{j=1}^{J}\rho_{jt}\left[\boldsymbol{e}_{jt}\boldsymbol{\delta}_{y^{+}_{jt}} - \boldsymbol{f}_{jt}\boldsymbol{y}^{+}_{jt} + \sum_{s=1}^{t-1}(\boldsymbol{s}_{jst}\boldsymbol{y}_{js(t-1)} + \boldsymbol{g}_{jst}\boldsymbol{y}^{-}_{jst} + \boldsymbol{h}_{jst}\boldsymbol{\delta}_{y^{-}_{jst}})\right]$$

$$+\sum_{k=1}^{K}\rho_{kt}[(1+\boldsymbol{r}^{+}_{kt})\boldsymbol{z}^{+}_{k(t-1)} - (1+\boldsymbol{r}^{-}_{kt})\boldsymbol{z}^{-}_{k(t-1)} - \boldsymbol{z}^{+}_{kt} + \boldsymbol{z}^{-}_{kt}] = 0 \quad \text{(cash balance)}$$

$$t = 1,\ldots,T+1.$$

$\boldsymbol{x}_{itt} - \boldsymbol{x}^{+}_{it} = 0$ (asset purchase inventory balance)

$\boldsymbol{y}_{jtt} - \boldsymbol{y}^{+}_{jt} = 0$ (liability incurrence inventory balance)

$i = 1,\ldots,I,\ j = 1,\ldots,J,\ t = 1,\ldots,T.$

$\boldsymbol{x}_{ist} - \boldsymbol{x}_{is(t-1)} + \boldsymbol{x}^{-}_{ist} = 0$ (asset sale inventory balance)

$\boldsymbol{y}_{jst} - \boldsymbol{y}_{js(t-1)} + \boldsymbol{y}^{-}_{jst} = 0$ (liability discharge inventory balance)

$i = 1,\ldots,I,\ j = 1,\ldots,J,\ s = 1,\ldots,t-1,\ \ t = 1,\ldots,T+1.$

$\boldsymbol{x}^{+}_{i(T+1)} = 0 \quad \boldsymbol{y}^{+}_{j(T+1)} = 0$ (no horizon decisions)

$\boldsymbol{x}^{-}_{is(T+1)} = 0 \quad \boldsymbol{y}^{-}_{js(T+1)} = 0$

$i = 1,\ldots,I,\ j = 1,\ldots,J,\ s = 1,\ldots,T.$

$\underline{X_{it}}\,\boldsymbol{\delta}_{x^{+}_{it}} \le \boldsymbol{x}^{+}_{it} \le \overline{X_{it}}\,\boldsymbol{\delta}_{x^{+}_{it}}$ (investment limits by type)

$\underline{Y_{jt}}\,\boldsymbol{\delta}_{y^{+}_{jt}} \le \boldsymbol{y}^{+}_{jt} \le \overline{Y_{jt}}\,\boldsymbol{\delta}_{y^{+}_{jt}}$ (liability limits by type)

$i = 1,\ldots,I,\ j = 1,\ldots,J,\ t = 1,\ldots,T.$

$0 \le \boldsymbol{z}^{+}_{kt}$

$0 \le \boldsymbol{z}^{-}_{kt} \le \overline{Z_{kt}}$ (short position limit by type)

$k = 1,\ldots,K,\ t = 1,\ldots,T.$

$$\sum_{i=1}^{I} \mathbf{x}_{it}^{+} \leq \overline{X_t^{+}} \qquad \text{(maximum new investments)}$$

$$\sum_{j=1}^{J} \mathbf{y}_{jt}^{+} \leq \overline{Y_t^{+}} \qquad \text{(maximum new liabilities)}$$

$$\sum_{j=1}^{J} \sum_{s=1}^{t} \mathbf{y}_{jst} \leq \overline{Y_t} \qquad \text{(maximum liability per period)}$$

$$t = 1,\ldots,T.$$

NB: Boldface denotes random entities. All constraints hold almost surely, i.e. with probability 1.

References

[1] I. Adler, N.K. Karmarkar, M.G.C. Resende and G.Veiga, An implementation of Karmarkar's algorithm for linear programming, Mathematical Programming 44(1989)297–335.

[2] J.F. Benders, Partitioning procedures for solving mixed-variables programming problems, Numerische Mathematik 4(1962)238–252.

[3] A.J. Berger, J.M. Mulvey, E. Rothberg and R. Vanderbei, Solving multistage stochastic programs using tree dissection, Statistics and Operations Research Research Report, Princeton University, Princeton, NJ, 1995.

[4] D.P. Bertsekas, *Dynamic Programming*, Prentice-Hall, Englewood Cliffs, NJ, 1987.

[5] P. Billingsley, *Probability and Measure*, 2nd ed., Wiley, New York, 1980.

[6] J.R. Birge, Decomposition and partitioning methods for multistage stochastic linear programs, Operations Research 33(1985)989–1007.

[7] J.R. Birge, M.A.H. Dempster, H.I. Gassmann, E.A. Gunn, A.J. King and S. Wallace, A standard input format for multiperiod stochastic linear programs, Mathematical Programming Society, Committee on Algorithms Newsletter 17 (1987)1–20.

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

[9] M.J. Brennan, E.S. Schwartz and R. Lagnado, Strategic asset allocation, Working Paper, University of California, Los Angeles, March, 1995.

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

[11] D.R. Cariño and W.T. Ziemba, Formulation of the Russell–Yasuda Kasai financial planning model, Research Report, Frank Russell Company, Tacoma, Washington, May 1995.

[12] D.R. Cariño, D.H. Myers and W.T. Ziemba, Concepts, technical issues, and uses of the Russell–Yasuda Kasai financial planning model, Research Report, Frank Russell Company, Tacoma, Washington, May 1995.

[13] G. Consigli and M.A.H. Dempster, Solving dynamic portfolio problems using stochastic programming, submitted to Zeitschrift für Angewandte Mathematik und Mechanik, *Proceedings of the GAMM96 Conference*, Charles University, Prague, May 1996.

[14] X. Corvera Poiré, *STOCHGEN User's Manual*, Department of Mathematics, University of Essex, 1995.

[15] X. Corvera Poiré, Model generation and sampling algorithms for dynamic stochastic programming, Ph.D. Thesis, Department of Mathematics, University of Essex, Colchester, UK, 1995.

[16] CPLEX Optimization, Inc., *Using the CPLEX Callable Library, Version 3.0*, Incline Village, NE, USA, 1994.

[17] G.B. Dantzig, M.A.H. Dempster and M.J. Kallio (eds.), *Large-Scale Linear Programming*, Vol. 1, IIASA CP-81-S1, Laxenburg, Austria, 1981.

[18] G.B. Dantzig and A. Madansky, On the solution of two-stage linear programs under uncertainty, *Proceedings of the 4th Symposium on Mathematical Statistics and Probability*, Vol. 1, University of California, Berkeley, 1961, pp. 165–176.

[19] G.B. Dantzig and G. Infanger, Multi-stage stochastic linear programs for portfolio optimization, Annals of Operations Research 45(1993)59–76.

[20] M.A.H. Dempster (ed.), *Stochastic Programming*, Academic Press, London, 1980.

[21] M.A.H. Dempster, Stochastic programming: An introduction, in [20], pp. 3–59.

[22] M.A.H. Dempster, On stochastic programming: II. Dynamic problems under risk, Stochastics 25 (1988)15–42.

[22a] M.A.H. Dempster, CALM: A stochastic MIP model, Finance Research Group Internal Report, Department of Mathematics, University of Essex, 1993.

[23] M.A.H. Dempster and H.I. Gassmann, Stochastic programming: Using the expected value of perfect information to simplify the decision tree, *Proceedings of the 15th IFIP Conference on System Modelling and Optimization*, IFIP, Zurich, 1991, pp. 301–303.

[24] M.A.H. Dempster and A. Ireland, Object oriented model integration in a financial decision support system, Decision Support Systems 7(1991)329–340.

[25] M.A.H. Dempster and R.T. Thompson, Parallelization and aggregation of nested Benders decomposition, *Proceedings of the APMOD95 Conference*, Brunel University of West London, 1996, Annals of Operations Research, this volume.

[26] M.A.H. Dempster, R.T. Thompson, EVPI-based importance sampling solution procedures for multistage stochastic linear programmes on parallel MIMD architectures, *Proceedings of the POC96 Conference*, Versailles, 1996, Annals of Operations Research, to appear.

[27] I.S. Duff, A. Erisman and J. Reid, *Direct Methods for Sparse Matrices*, Clarendon Press, Oxford, 1986.

[28] J. Dupacova, Stochastic programming models in banking, Working Paper, International Institute for Applied Systems Analysis, Laxenburg, Austria, 1991.

[29] J. Dupacova, Multistage stochastic programs: The state-of-the-art and selected bibliography, Kybernetika 31(1995)151–174.

[30] Yu. Ermoliev and R.J-B. Wets (eds.), *Numerical Techniques for Stochastic Optimization*, Springer, Berlin, 1988.

[31] Yu. Ermoliev and R.J-B. Wets, Stochastic programming, an introduction, in [30], pp. 1–32.

[32] H.I.Gassmann, An algorithm for the multistage stochastic linear programming problem, Ph.D. Thesis, Faculty of Commerce, University of British Columbia, Vancouver, 1987.

[33] H.I. Gassmann, MSLiP: A computer code for the multi-stage stochastic linear programming problem, Mathematical Programming 47(1990)407–423.

[34] H.I. Gassmann, *MSLiP82 User's Guide*, School of Business Administration, Dalhousie University, Halifax, Canada, 1992.

[35] D. Goldfarb and J.K. Reid, A practical steepest edge simplex algorithm, Mathematical Programming 12(1977)361–371.

[36] C.G. Gonzaga, Path following methods for linear programming, SIAM Review 34(1992)167–227.

[37] H.J. Greenberg, A computer-assisted analysis system for mathematical programming models and solutions: A user's guide for ANALYZE, Mathematics Department, University of Colorado at Denver, March, 1994.

[38] H.J. Greenberg, A primer for MODLER: Modelling by object-driven linear elemental relations, Mathematics Department, University of Colorado at Denver, June, 1992.

[39] H.J. Greenberg, A primer for RANDMOD: A system for randomizing modifications to an instance of a linear program, Mathematics Department, University of Colorado at Denver, March, 1992.

[40] P.M. Harris, Pivot selection method of the DEVEX LP Code, Mathematical Programming 5(1973) 1–28.

[41] G. Infanger, *Planning Under Uncertainty: Solving Large-Scale Stochastic Linear Programs*, Boyd & Fraser, Danvers, MA, 1994.

[42] International Business Machines Co. *Optimization Subroutine Library Guide and Reference*, Release 2, Document SC23-0519-03, Armonk, NY, 1992.

[43] J.G. Kallberg, R.W. White and W.T. Ziemba, Short term financial planning under uncertainty, Management Science 28(1982)670–682.

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

[45] A.J. King, Stochastic programming problems: Examples from the literature, in [30], pp. 543–567.

[46] A.J. King, Strategic asset allocation with stochastic programming, presented at the *UNICOM Conference: Applications of Novel/Newly Emerging Techniques to Financial Systems*, USA Embassy, London, November, 1994.

[47] M.I. Kusy and W.T. Ziemba, A bank asset and liability management model, Operations Research 34(1986)356–376.

[48] M. Lane and P. Hutchinson, A model for managing a certificate of deposit portfolio under uncertainty, in [20], pp. 473–493.

[49] R. Levkovitz and G. Mitra, Solution of large-scale linear programs: A review of hardware, software and algorithmic issues, in: *Optimization in Industry*, eds. T.A. Ciriani and R.C. Leachman, Wiley, New York, 1993, pp. 139–171.

[50] J. Liu, Modification of the minimum-degree algorithm by multiple elimination, ACM Transactions on Mathematical Software 11 (1985)141–153.

[51] D.G. Luenberger., *Linear and Nonlinear Programming*, Addison-Wesley, 1984.

[52] I.J. Lustig, R.E. Marsten and D. Shanno, On implementing Mehrotra's predictor–corrector interior point algorithm for linear programming, SIAM Journal on Optimization 2(1992)735–749.

[53] I.J. Lustig, J.M. Mulvey and T.J. Carpenter, Formulating two-stage stochastic programs for interior point methods, Operations Research 39(1991)757–770.

[54] N. Megiddo, Pathways to the optimal set in linear programming, in: *Progress in Mathematical Programming: Interior Points and Related Methods*, Springer, New York, 1989, pp. 131–158.

[55] S. Mehrotra, On the implementation of a primal-dual interior point method, SIAM Journal on Optimization 2(1992)575–601.

[56] J.M. Mulvey, Integrative asset-liability planning using large-scale stochastic optimization, Technical Report, Statistics and Operations Research Series, School of Engineering and Applied Science, Princeton University, 1992.

[57] J.M. Mulvey, Generating scenarios for the Towers Perrin investment system, Technical Report, Statistics and Operations Research Series, School of Engineering and Applied Science, Princeton University, March, 1995.

[58] J.M. Mulvey and H. Vladimirou, Stochastic network optimization models for investment planning, Annals of Operations Research 20(1989)187–217.

[59] C.E. Pfefferkorn and T.A. Tomlin, Design of a linear programming system for ILLIAC-V, Technical Report SOL 76-8, Department of Operations Research, Stanford University, and Technical Report 5487, NASA-Ames Institute for Advanced Computation, Sunnyvale, CA, 1976.

[60] R.T. Rockafellar and R.J-B. Wets, Scenarios and policy aggregation in optimization under uncertainty Maths of OR 16(1991)119–147.

[61] A. Ruszczyński, Augmented Lagrangian decomposition for sparse convex optimization, Working Paper, IIASA, Laxenburg, Austria, 1992.

[62] R.T. Thompson, *MSLiP-OSL User's Guide*, Judge Institute of Management Studies, University of Cambridge, UK, 1997.

[63] R. Van Slyke and R.J-B. Wets, L-shaped linear programs with application to optimal control and stochastic programming, SIAM Journal of Appl. Math. 17(1969)638–663.

[64] R.J-B. Wets, Large scale linear programming techniques, in [30], pp. 65–93.
[65] A.D. Wilkie, More on a stochastic asset model for actuarial use, presented to the Institute of Actuaries, London, 24.4.1995.
[66] A.D. Wilkie, Stochastic investment models, Lecture Notes, Summer School on Financial Mathematics for Actuaries, Oxford, 21–23 March, 1995.
[67] R.J. Wittrock, Dual nested decomposition of staircase linear programs, Mathematical Programming Study 24(1985)65–86.
[68] S. Zenios, Asset-liability management under uncertainty: The case of mortgage-backed securities, Research Report, Hermes Lab for Financial Modeling and Simulation, The Wharton School, University of Pennsylvania, 1992.
[69] S.A. Zenios (ed.), *Financial Optimization*, Cambridge University Press, Cambridge, 1993.