Chapter 16

Stochastic Programming: Theory and Algorithms

16.1 Introduction

In the introductory chapter and elsewhere, we argued that many optimization problems are described by uncertain parameters. There are different ways of incorporating this uncertainty. We consider two approaches: Stochastic programming in the present chapter and robust optimization in Chapter 19. *Stochastic programming* assumes that the uncertain parameters are random variables with known probability distributions. This information is then used to transform the stochastic program into a so-called *deterministic equivalent* which might be a linear program, a nonlinear program or an integer program (see Chapters 2, 5 and 11 respectively).

While stochastic programming models have existed for several decades, computational technology has only recently allowed the solution of realistic size problems. The field continues to develop with the advancement of available algorithms and computing power. It is a popular modeling tool for problems in a variety of disciplines including financial engineering.

The uncertainty is described by a certain sample space Ω, a σ-field of random events and a probability measure P (see Appendix C). In stochastic programming, Ω is often a finite set $\{\omega_1, \ldots, \omega_S\}$. The corresponding probabilities $p(\omega_k) \geq 0$ satisfy $\sum_{k=1}^{S} p(\omega_k) = 1$. For example, to represent the outcomes of flipping a coin twice in a row, we would use four random events $\Omega = \{HH, HT, TH, TT\}$, each with probability 1/4, where H stands for Head and T stands for Tail.

Stochastic programming models can include *anticipative* and/or *adaptive* decision variables. Anticipative variables correspond to those decisions that must be made *here-and-now* and cannot depend on the future observations/partial realizations of the random parameters. Adaptive variables correspond to *wait-and-see* decisions that can be made after some (or, sometimes all) of the random parameters are observed.

Stochastic programming models that include both anticipative and adaptive variables are called *recourse* models. Using a multi-stage stochastic programming formulation, with recourse variables at each stage, one can model

a decision environment where information is revealed progressively and the decisions are adapted to each new piece of information.

In investment planning, each new trading opportunity represents a new decision to be made. Therefore, trading dates where investment portfolios can be rebalanced become natural choices for decision stages, and these problems can be formulated conveniently as multi-stage stochastic programming problems with recourse.

16.2 Two Stage Problems with Recourse

In Chapter 1, we have already seen a generic form of a *two-stage stochastic linear program with recourse*:

$$\begin{array}{rrcrcl} \max_x & a^T x & + & E[\max_{y(\omega)} c(\omega)^T y(\omega)] & & \\ & Ax & & & = & b \\ & B(\omega)x & + & C(\omega)y(\omega) & = & d(\omega) \\ & x \geq 0, & & y(\omega) \geq 0. & & \end{array} \tag{16.1}$$

In this formulation, the first-stage decisions are represented by vector x. These decisions are made *before* the random event ω is observed. The second-stage decisions are represented by vector $y(\omega)$. These decisions are made *after* the random event ω has been observed, and therefore the vector y is a function of ω. A and b define deterministic constraints on the first-stage decisions x, whereas $B(\omega)$, $C(\omega)$, and $d(\omega)$ define stochastic constraints linking the recourse decisions $y(\omega)$ to the first-stage decisions x. The objective function contains a deterministic term $a^T x$ and the expectation of the second-stage objective $c(\omega)^T y(\omega)$ taken over all realizations of the random event ω.

Notice that the first-stage decisions will not necessarily satisfy the linking constraints $B(\omega)x + C(\omega)y(\omega) = d(\omega)$, if no recourse action is taken. Therefore, recourse allows one to make sure that the initial decisions can be "corrected" with respect to this second set of feasibility equations.

In Section 1.2.1, we also argued that problem (16.1) can be represented in an alternative manner by considering the *second-stage* or *recourse* problem that is defined as follows, given x, the first-stage decisions:

$$\begin{array}{rrcl} f(x,\omega) = \max & c(\omega)^T y(\omega) & & \\ & C(\omega)y(\omega) & = & d(\omega) - B(\omega)x \\ & y(\omega) \geq 0. & & \end{array} \tag{16.2}$$

Let $f(x) = E[f(x,\omega)]$ denote the expected value of this optimum. If the function $f(x)$ is available, the two-stage stochastic linear program (16.1) reduces to a deterministic nonlinear program:

$$\begin{array}{rl} \max & a^T x + f(x) \\ & Ax = b \\ & x \geq 0. \end{array} \tag{16.3}$$

Unfortunately, computing $f(x)$ is often very hard, especially when the sample space Ω is infinite. Next, we consider the case where Ω is a finite set.

Assume that $\Omega = \{\omega_1, \ldots, \omega_S\}$ and let $p = (p_1, \ldots, p_S)$ denote the probability distribution on this sample space. The S possibilities ω_k, for $k = 1, \ldots, S$ are also called *scenarios*. The expectation of the second-stage objective becomes:

$$E[\max_{y(\omega)} c(\omega)^T y(\omega)] \;=\; \sum_{k=1}^{S} p_k \max_{y(\omega_k)} c(\omega_k)^T y(\omega_k)$$

For brevity, we write c_k instead of $c(\omega_k)$, etc. Under this *scenario approach* the two-stage stochastic linear programming problem (16.1) takes the following form:

$$\begin{array}{ll} \max_x & a^T x + \sum_{k=1}^{S} p_k \max_{y_k} c_k^T y_k \\ & Ax = b \\ & B_k x + C_k y_k = d_k \quad \text{for } k = 1, \ldots S \\ & x \geq 0 \\ & y_k \geq 0 \quad \text{for } k = 1, \ldots, S. \end{array} \tag{16.4}$$

Note that there is a different second stage decision vector y_k for each scenario k. The maximum in the objective is achieved by optimizing over all variables x and y_k simultaneously. Therefore, this optimization problem is:

$$\begin{array}{lrcccccccl} \max_{x,y_1,\ldots,y_S} & a^T x & + & p_1 c_1^T y_1 & + & \ldots & + & p_S c_S^T y_S & & \\ & Ax & & & & & & & = & b \\ & B_1 x & + & C_1 y_1 & & & & & = & d_1 \\ & \vdots & & & \ddots & & & & & \vdots \\ & B_S x & & & & & + & C_S y_S & = & d_S \\ & x, & & y_1, & & \ldots & & y_S & \geq & 0. \end{array} \tag{16.5}$$

This is a deterministic linear programming problem called the *deterministic equivalent* of the original uncertain problem. This problem has S copies of the second-stage decision variables and therefore, can be significantly larger than the original problem before we considered the uncertainty of the parameters. Fortunately, however, the constraint matrix has a very special sparsity structure that can be exploited by modern decomposition based solution methods (see Section 16.4).

Exercise 16.1 Consider an investor with an initial wealth W_0. At time 0, the investor constructs a portfolio comprising one riskless asset with return R_1 in the first period and one risky asset with return R_1^+ with probability 0.5 and R_1^- with probability 0.5. At the end of the first period, the investor can rebalance her portfolio. The return in the second period is R_2 for the riskless asset, while it is R_2^+ with probability 0.5 and R_2^- with probability 0.5 for the risky asset. The objective is to meet a liability $L_2 = 0.9$ at the end of Period 2 and to maximize the expected remaining wealth W_2. Formulate a 2-stage stochastic linear program that solves the investor's problem.

Exercise 16.2 In Exercise 3.2, the cash requirement in quarter Q1 is known to be 100 but, for the remaining quarters, the company considers 3 equally likely scenarios:

	$Q2$	$Q3$	$Q4$	$Q5$	$Q6$	$Q7$	$Q8$
Scenario 1	450	100	−650	−550	200	650	−850
Scenario 2	500	100	−600	−500	200	600	−900
Scenario 3	550	150	−600	−450	250	600	−800

Formulate a linear program that maximizes the expected wealth of the company at the end of quarter Q8.

16.3 Multi Stage Problems

In a *multi-stage stochastic program with recourse*, the recourse decisions can be taken at several points in time, called stages. Let $n \geq 2$ be the number of stages. The random event ω is a vector $(o_1, \ldots, o_{n-1})$ that gets revealed progressively over time. The first-stage decisions are taken before any component of ω is revealed. Then o_1 is revealed. With this knowledge, one takes the second-stage decisions. After that, o_2 is revealed, and so on, alternating between a new component of ω beeing revealed and new recourse decisions beeing implemented. We assume that $\Omega = \{\omega_1, \ldots, \omega_S\}$ is a finite set. Let p_k be the probability of scenario ω_k, for $k = 1, \ldots, S$.

Some scenarios ω_k may be identical in their first components and only become differentiated in the later stages. Therefore it is convenient to introduce the *scenario tree*, which illustrates how the scenarios branch off at each stage. The nodes are labelled 1 through N, where node 1 is the root. Each node is in one stage, where the root is the unique node in stage 1. Each node i in stage $k \geq 2$ is adjacent to a unique node $a(i)$ in stage $k-1$. Node $a(i)$ is called the *father* of node i. The paths from the root to the leaves (in stage n) represent the scenarios. Thus the last stage has as many nodes as scenarios. These nodes are called the *terminal nodes*. The collection of scenarios passing through node i in stage k have identical components $o_1, \ldots, o_{k-1}$.

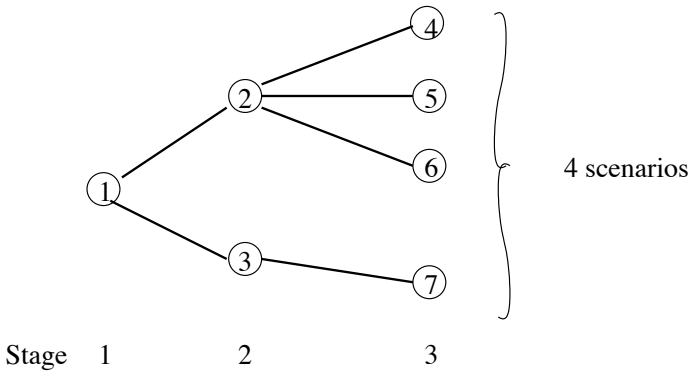


Figure 16.1: A scenario tree with 3 stages and 4 scenarios

In Figure 16.1, Node 1 is the root, Nodes 4, 5, 6 and 7 are the terminal nodes. The father of Node 6 is Node 2, in other words $a(6) = 2$.

Associated with each node i is a recourse decision vector x_i. For a node i is stage k, the decisions x_i are taken based on the information that has been revealed up to stage k. Let q_i be the sum of the probabilities p_k over all the scenarios ω_k that go through node i. Therefore q_i is the probability of node i, conditional on being in Stage k. The multi-stage stochastic program with recourse can be formulated as follows:

$$\begin{array}{lrcll} \max_{x_1,\ldots,x_N} & \sum_{i=1}^{N} q_i c_i^T x_i & & & \\ & Ax_1 & = & b & \\ & B_i x_{a(i)} + C_i x_i & = & d_i & \text{for } i = 2,\ldots,N \\ & x_i \geq 0. & & & \end{array} \tag{16.6}$$

In this formulation, A and b define deterministic constraints on the first-stage decisions x_1, whereas B_i, C_i, and d_i define stochastic constraints linking the recourse decisions x_i in node i to the recourse decisions $x_{a(i)}$ in its father node. The objective function contains a term $c_i^T x_i$ for each node.

To illustrate, we present formulation (16.6) for the example of Figure 16.1. The terminal nodes 4 to 7 correspond to scenarios 1 to 4 respectively. Thus we have $q_4 = p_1$, $q_5 = p_2$, $q_6 = p_3$ and $q_7 = p_4$, where p_k is the probability of scenario k. We also have $q_2 = p_1 + p_2 + p_3$, $q_3 = p_4$ and $q_2 + q_3 = 1$.

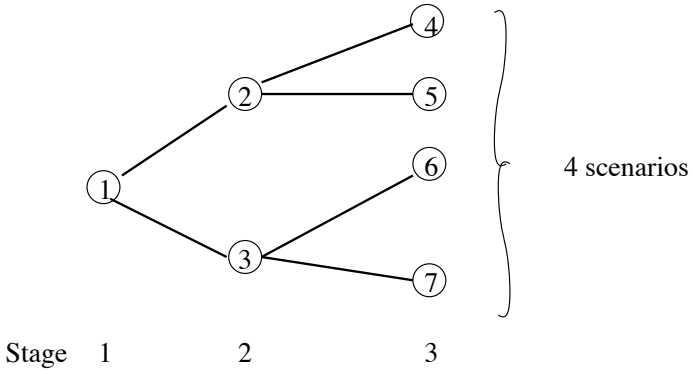
$$\begin{array}{lrrrrrrrcl} \max & c_1^T x_1 & +q_2 c_2^T x_2 & +q_3 c_3^T x_3 & +p_1 c_4^T x_4 & +p_2 c_5^T x_5 & +p_3 c_6^T x_6 & p_4 c_7^T x_7 & & \\ & Ax_1 & & & & & & & = & b \\ & B_2 x_1 & +C_2 x_2 & & & & & & = & d_2 \\ & B_3 x_1 & & +C_3 x_3 & & & & & = & d_3 \\ & & B_4 x_2 & & +C_4 x_4 & & & & = & d_4 \\ & & B_5 x_2 & & & +C_5 x_5 & & & = & d_5 \\ & & B_6 x_2 & & & & +C_6 x_6 & & = & d_6 \\ & & & B_7 x_3 & & & & +C_7 x_7 & = & d_7 \\ & & & & & & & x_i & \geq & 0. \end{array}$$

Note that the size of the linear program (16.6) increases rapidly with the number of stages. For example, for a problem with 10 stages and a binary tree, there are 1024 scenarios and therefore the linear program (16.6) may have several thousand constraints and variables, depending on the number of variables and constraints at each node. Modern commercial codes can handle such large linear programs, but a moderate increase in the number of stages or in the number of branches at each stage could make (16.6) too large to solve by standard linear programming solvers. When this happens, one may try to exploit the special structure of (16.6) to solve the model (see Section 16.4).

Exercise 16.3 In Exercise 3.2, the cash requirements in quarters Q1, Q2, Q3, Q6 and Q7 are known. On the other hand, the company considers two

equally likely (and independent) possibilities for each of the quarters Q4, Q5 and Q8, giving rise to eight equally likely scenarios. In quarter Q4, the cash inflow will be either 600 or 650. In quarter Q5, it will be either 500 or 550. In quarter Q8, it will be either 850 or 900. Formulate a linear program that maximizes the expected wealth of the company at the end of quarter Q8.

Exercise 16.4 Develop the linear program (16.6) for the following scenario tree.



16.4 Decomposition

The size of the linear program (16.6) depends on the number of decision stages and the branching factor at each node of the scenario tree. For example, a 4-stage model with 25 branches at each node has $25 \times 25 \times 25 \times 25 = 390625$ scenarios. Increasing the number of stages and branches quickly results in an explosion of dimensionality. Obviously, the size of (16.6) can be a limiting factor in solving realistic problems. When this occurs, it becomes essential to take advantage of the special structure of the linear program (16.6). In this section, we present a decomposition algorithm for exploiting this structure. It is called *Benders decomposition* or, in the stochastic programming literature, the *L-shaped method*.

The structure that we really want to exploit is that of the two-stage problem (16.5). So we start with (16.5). We will explain subsequently how to deal with the general multi-stage model (16.6). The constraint matrix of (16.5) has the following form:

$$\begin{pmatrix} A & & & \\ B_1 & C_1 & & \\ \vdots & & \ddots & \\ B_S & & & C_S \end{pmatrix}.$$

Note that the blocks $C_1, \ldots, C_S$ of the constraint matrix are only interrelated through the blocks $B_1, \ldots, B_S$ which correspond to the first-stage

decisions. In other words, once the first-stage decisions x have been fixed, (16.5) decomposes into S independent linear programs. The idea of Benders decomposition is to solve a "master problem" involving only the variables x and a series of independent "recourse problems" each involving a different vector of variables y_k. The master problem and recourse problems are linear programs. The size of these linear programs is much smaller than the size of full model (16.5). The recourse problems are solved for a given vector x and their solutions are used to generate inequalities that are added to the master problem. Solving the new master problem produces a new x and the process is repeated. More specifically, let us write (16.5) as

$$\begin{array}{rrcl} \max_x & a^T x & + & P_1(x) + \ldots + P_S(x) \\ & Ax & = & b \\ & x & \geq & 0 \end{array} \tag{16.7}$$

where, for $k = 1, \ldots S$,

$$\begin{array}{rcll} P_k(x) & = & \max_{y_k} & p_k c_k^T y_k \\ & & & C_k y_k = d_k - B_k x \\ & & & y_k \geq 0. \end{array} \tag{16.8}$$

The dual linear program of the recourse problem (16.8) is:

$$\begin{array}{rcll} P_k(x) & = & \min_{u_k} & u_k^T (d_k - B_k x) \\ & & & C_k^T u_k \geq p_k c_k. \end{array} \tag{16.9}$$

For simplicity, we assume that the dual (16.9) is feasible, which is the case of interest in applications. The recourse linear program (16.8) will be solved for a sequence of vectors x^i, for $i = 0, \ldots$. The initial vector x^0 might be obtained by solving

$$\begin{array}{rrcl} \max_x & a^T x & & \\ & Ax & = & b \\ & x & \geq & 0. \end{array} \tag{16.10}$$

For a given vector x^i, two possibilities can occur for the recourse linear program (16.8): either (16.8) has an optimal solution or it is infeasible.

If (16.8) has an optimal solution y_k^i, and u_k^i is the corresponding optimal dual solution, then (16.9) implies that

$$P_k(x^i) = (u_k^i)^T (d_k - B_k x^i)$$

and, since

$$P_k(x) \leq (u_k^i)^T (d_k - B_k x)$$

we get that

$$P_k(x) \leq (u_k^i)^T (B_k x^i - B_k x) + P_k(x^i).$$

This inequality, which is called an *optimality cut*, can be added to the current master linear program. Initially, the master linear program is just (16.10).

If (16.8) is infeasible, then the dual problem is unbounded. Let u_k^i a direction where (16.9) is unbounded, i.e. $(u_k^i)^T(d_k - B_k x^i) < 0$ and $C_k^T u_k^i \geq p_k c_k$. Since we are only interested in first-stage decisions x that lead to feasible second-stage decisions y_k, the following *feasibility cut* can be added to the current master linear program:

$$(u_k^i)^T(d_k - B_k x) \geq 0.$$

After solving the recourse problems (16.8) for each k, we have the following lower bound on the optimal value of (16.5):

$$LB = a^T x^i + P_1(x^i) + \ldots + P_S(x^i)$$

where we set $P_k(x^i) = -\infty$ if the corresponding recourse problem is infeasible.

Adding all the optimality and feasibility cuts found so far (for $j = 0, \ldots, i$) to the master linear program, we obtain:

$$\begin{array}{lrcl} \max_{x,z_1,\ldots,z_S} & a^T x & + & \sum_{k=1}^S z_k \\ & Ax & = & b \\ & z_k & \leq & (u_k^j)^T(B_k x^j - B_k x) + P_k(x^j) \quad \text{for some pairs } (j,k) \\ & 0 & \leq & (u_k^j)^T(d_k - B_k x) \qquad \text{for the remaining pairs } (j,k) \\ & x & \geq & 0. \end{array}$$

Denoting by $x^{i+1}, z_1^{i+1}, \ldots, z_S^{i+1}$ an optimal solution to this linear program we get an upper bound on the optimal value of (16.5):

$$UB = a^T x^{i+1} + z_1^{i+1} + \ldots + z_S^{i+1}.$$

Benders decomposition alternately solves the recourse problems (16.8) and the master linear program with new optimality and feasibility cuts added at each iteration until the gap between the upper bound UB and the lower bound LB falls below a given threshold. One can show that $UB - LB$ converges to zero in a finite number of iterations. See, for instance, the book of Birge and Louveaux [12], pages 159-162.

Benders decomposition can also be used for multi-stage problems (16.6) in a straightforward way: The stages are partitioned into a first set that gives rise to the "master problem" and a second set that gives rise to the "recourse problems". For example in a 6-stage problem, the variables of the first 2 stages could define the master problem. When these variables are fixed, (16.6) decomposes into separate linear programs each involving variables of the last 4 stages. The solutions of these recourse linear programs provide optimality or feasibility cuts that can be added to the master problem. As before, upper and lower bounds are computed at each iteration and the algorithm stops when the difference drops below a given tolerance. Using this approach, Gondzio and Kouwenberg [32] were able to solve an asset liability management problem with over 4 million scenarios, whose linear

programming formulation (16.6) had 12 million constraints and 24 million variables. This linear program was so large that storage space on the computer became an issue. The scenario tree had 6 levels and 13 branches at each node. In order to apply two-stage Benders decomposition, Gondzio and Kouwenberg divided the 6 period problem into a first stage problem containing the first 3 periods and a second stage containing periods 4 to 6. This resulted in 2,197 recourse linear programs, each involving 2,197 scenarios. These recourse linear programs were solved by an interior point algorithm. Note that Benders decomposition is ideally suited for parallel computations since the recourse linear programs can be solved simultaneously. When the solution of all the recourse linear programs is completed (which takes the bulk of the time), the master problem is then solved on one processor while the other processors remain idle temporarily. Gondzio and Kouwenberg tested a parallel implementation on a computer with 16 processors and they obtained an almost perfect speedup, that is a speedup factor of almost k when using k processors.

16.5 Scenario Generation

How should one generate scenarios in order to formulate a deterministic equivalent formulation (16.6) that accurately represents the underlying stochastic program? There are two separate issues. First, one needs to model the correlation over time among the random parameters. For a pension fund, such a model might relate wage inflation (which influences the liability side) to interest rates and stock prices (which influence the asset side). Mulvey [53] describes the system developed by Towers Perrin, based on a cascading set of stochastic differential equations. Simpler autoregressive models can also be used. This is discussed below. The second issue is the construction of a scenario tree from these models: A finite number of scenarios must reflect as accurately as possible the random processes modeled in the previous step, suggesting the need for a large number of scenarios. On the other hand, the linear program (16.6) can only be solved if the size of the scenario tree is reasonably small, suggesting a rather limited number of scenarios. To reconcile these two conflicting objectives, it might be crucial to use variance reduction techniques. We address these issues in this section.

16.5.1 Autoregressive model

In order to generate the random parameters underlying the stochastic program, one needs to construct an economic model reflecting the correlation between the parameters. Historic data may be available. The goal is to generate meaningful time series for constructing the scenarios. One approach is to use an autoregressive model.

Specifically, if r_t denotes the random vector of parameters in period t, an *autoregressive model* is defined by:

$$r_t = D_0 + D_1 r_{t-1} + \ldots + D_p r_{t-p} + \epsilon_t$$

where p is the number of lags used in the regression, $D_0, D_1, \ldots, D_p$ are time independent constant matrices which are estimated through statistical methods such as maximum likelihood, and ϵ_t is a vector of i.i.d. random disturbances with mean zero.

To illustrate this, consider the example of Section 8.1.1. Let s_t, b_t and m_t denote the rates of return of stocks, bonds and the money market, respectively, in year t. An autoregressive model with $p = 1$ has the form:

$$\begin{pmatrix} s_t \\ b_t \\ m_t \end{pmatrix} = \begin{pmatrix} d_1 \\ d_2 \\ d_3 \end{pmatrix} + \begin{pmatrix} d_{11} & d_{12} & d_{13} \\ d_{21} & d_{22} & d_{23} \\ d_{31} & d_{32} & d_{33} \end{pmatrix} \begin{pmatrix} s_{t-1} \\ b_{t-1} \\ m_{t-1} \end{pmatrix} + \begin{pmatrix} \epsilon_t^s \\ \epsilon_t^b \\ \epsilon_t^m \end{pmatrix} \quad t = 2, \ldots, T$$

In particular, to find the parameters d_1, d_{11}, d_{12}, d_{13} in the first equation

$$s_t = d_1 + d_{11} s_{t-1} + d_{12} b_{t-1} + d_{13} m_{t-1} + \epsilon_t^s$$

one can use standard linear regression tools that minimize the sum of the squared errors ϵ_t^s. Within an Excel spreadsheet for instance, one can use the function LINEST. Suppose that the rates of return on the stocks are stored in cells `B2` to `B44` and that, for bonds and the money market, the rates are stored in columns `C` and `D`, rows 2 to 44 as well. LINEST is an array formula. Its first argument contains the known data for the left hand side of the equation (here the column s_t), the second argument contains the known data in the right hand side (here the columns s_{t-1}, b_{t-1} and m_{t-1}). Typing `LINEST(B3:B44, B2:D43,,)` one obtains the following values of the parameters:

$$d_1 = 0.077, \quad d_{11} = -0.058, \quad d_{12} = 0.219, \quad d_{13} = 0.448.$$

Using the same approach for the other two equations we get the following autoregressive model:

$$\begin{aligned} s_t &= 0.077 - 0.058 s_{t-1} + 0.219 b_{t-1} + 0.448 m_{t-1} + \epsilon_t^s \\ b_t &= 0.047 - 0.053 s_{t-1} - 0.078 b_{t-1} + 0.707 m_{t-1} + \epsilon_t^b \\ m_t &= 0.016 + 0.033 s_{t-1} - 0.044 b_{t-1} + 0.746 m_{t-1} + \epsilon_t^m \end{aligned}$$

The option `LINEST(B3:B44, B2:D43,,TRUE)` provides some useful statistics, such as the standard error of the estimate s_t. Here we get a standard error of $\sigma_s = 0.173$. Similarly, the standard error for b_t and m_t are $\sigma_b = 0.108$ and $\sigma_m = 0.022$ respectively.

Exercise 16.5 Instead of an autoregressive model relating the rates of returns r_t, b_t and m_t, construct an autoregressive model relating the logarithms of the returns $g_t = log(1 + r_t)$, $h_t = log(1 + b_t)$ and $k_t = log(1 + m_t)$. Use one lag, i.e. $p = 1$. Solve using `LINEST` or your prefered linear regression tool.

Exercise 16.6 In the above autoregressive model, the coefficients of m_{t-1} are significantly larger than those of s_{t-1} and b_{t-1}. This suggests that these two variables might not be useful in the regression. Resolve the example assuming the following autoregressive model:

$$\begin{aligned} s_t &= d_1 + d_{13} m_{t-1} + \epsilon_t^s \\ b_t &= d_2 + d_{23} m_{t-1} + \epsilon_t^b \\ m_t &= d_3 + d_{33} m_{t-1} + \epsilon_t^m \end{aligned}$$

16.5.2 Constructing scenario trees

The random distributions relating the various parameters of a stochastic program must be discretized to generate a set of scenarios that is adequate for its deterministic equivalent. Too few scenarios may lead to approximation errors. On the other hand, too many scenarios will lead to an explosion in the size of the scenario tree, leading to an excessive computational burden. In this section, we discuss a simple random sampling approach and two variance reduction techniques: adjusted random sampling and tree fitting. Unfortunately, scenario trees constructed by these methods could contain spurious arbitrage opportunities. We end this section with a procedure to test that this does not occur.

Random sampling

One can generate scenarios directly from the autoregressive model introduced in the previous section:

$$r_t = D_0 + D_1 r_{t-1} + \ldots + D_p r_{t-p} + \epsilon_t$$

where $\epsilon_t \sim N(0, \Sigma)$ are independently distributed multivariate normal distributions with mean 0 and covariance matrix Σ.

In our example, Σ is a 3×3 diagonal matrix, with diagonal entries σ_s, σ_b and σ_m. Using the parameters $\sigma_s = 0.173$, $\sigma_b = 0.108$, $\sigma_m = 0.022$ computed earlier, and a random number generator, we obtained $\epsilon_t^s = -0.186$, $\epsilon_t^b = 0.052$ and $\epsilon_t^m = 0.007$. We use the autoregressive model to get rates of return for 2004 based on the known rates of returns for 2003 (see Table in Section 8.1.1):

$$s_{2004} = 0.077 - 0.058 \times 0.2868 + 0.219 \times 0.0054 + 0.448 \times 0.0098 - 0.186 = -0.087$$

$$b_{2004} = 0.047 - 0.053 \times 0.2868 - 0.078 \times 0.0054 + 0.707 \times 0.0098 + 0.052 = 0.091$$

$$m_{2004} = 0.016 + 0.033 \times 0.2868 - 0.044 \times 0.0054 + 0.746 \times 0.0098 + 0.007 = 0.040$$

These are the rates of return for one of the branches from node 1. For each of the other branches from node 1, one generates random values of ϵ_t^s, ϵ_t^b and ϵ_t^m and computes the corresponding values of s_{2004}, b_{2004} and m_{2004}. Thirty branches or so may be needed to get a reasonable approximation of the distribution of the rates of return in stage 1. For a problem with

3 stages, 30 branches at each stage represent 27,000 scenarios. With more stages, the size of the linear program (16.6) explodes. Kouwenberg [45] performed tests on scenario trees with fewer branches at each node (such as a 5-stage problem with branching structure 10-6-6-4-4, meaning 10 branches at the root, then 6 branches at each node in the next stage and so on) and he concluded that random sampling on such trees leads to unstable investment strategies. This occurs because the approximation error made by representing parameter distributions by random samples can be significant in a small scenario tree. As a result the optimal solution of (16.6) is not optimal for the actual parameter distributions. How can one construct a scenario tree that more accurately represents these distributions, without blowing up the size of (16.6)?

Adjusted random sampling

An easy way of improving upon random sampling is as follows. Assume that each node of the scenario tree has an even number $K = 2k$ of branches. Instead of generating $2k$ random samples from the autoregressive model, generate k random samples only and use the negative of their error terms to compute the values on the remaining k branches. This will fit all the odd moments of the distributions correctly. In order to fit the variance of the distributions as well, one can scale the sampled values. The sampled values are all scaled by a multiplicative factor until their variance fits that of the corresponding parameter.

As an example, corresponding to the branch with $\epsilon_t^s = -0.186$, $\epsilon_t^b = 0.052$ and $\epsilon_t^m = 0.007$ at node 1, one would also generate another branch with $\epsilon_t^s = 0.186$, $\epsilon_t^b = -0.052$ and $\epsilon_t^m = -0.007$. For this branch the autoregressive model gives the following rates of return for 2004:

$$s_{2004} = 0.077-0.058\times0.2868+0.219\times0.0054+0.448\times0.0098+0.186 = 0.285$$

$$b_{2004} = 0.047-0.053\times0.2868-0.078\times0.0054+0.707\times0.0098-0.052 = -0.013$$

$$m_{2004} = 0.016+0.033\times0.2868-0.044\times0.0054+0.746\times0.0098-0.007 = 0.026$$

Suppose that the set of ϵ_t^s generated on the branches leaving from node 1 has standard deviation 0.228 but the corresponding parameter should have standard deviation 0.165. Then the ϵ_t^s would be scaled down by $\frac{0.165}{0.228}$ on all the branches from node 1. For example, instead of $\epsilon_t^s = -0.186$ on the branch discussed earlier, one would use $\epsilon_t^s = -0.186\frac{0.165}{0.228} = -0.135$. This corresponds to the following rate of return:

$$s_{2004} = 0.077-0.058\times0.2868+0.219\times0.0054+0.448\times0.0098-0.135 = -0.036$$

The rates of returns on all the branches from node 1 would be modified in the same way.

Tree fitting

How can one best approximate a continuous distribution by a discrete distribution with K values? In other words, how should one choose values v_k

and their probabilities p_k, for $k = 1, \ldots, K$, in order to approximate the given distribution as accurately as possible? A natural answer is to match as many of the moments as possible. In the context of a scenario tree, the problem is somewhat more complicated since there are several correlated parameters at each node and there is interdependence between periods as well. Hoyland and Wallace [39] propose to formulate this fitting problem as a nonlinear program. The fitting problem can be solved either at each node separately or on the overall tree. We explain the fitting problem at a node. Let S_l be the values of the statistical properties of the distributions that one desires to fit, for $l = 1, \ldots, s$. These might be the expected values of the distributions, the correlation matrix, the skewness and kurtosis. Let v_k and p_k denote the vector of values on branch k and its probability, respectively, for $k = 1, \ldots, K$. Let $f_l(v, p)$ be the mathematical expression of property l for the discrete distribution (for example, the mean of the vectors v_k, their correlation, skewness and kurtosis). Each property has a positive weight w_l indicating its importance in the desired fit. Hoyland and Wallace formulate the fitting problem as

$$\begin{array}{ll} \min_{v,p} & \sum_l w_l (f_l(v,p) - S_l)^2 \\ & \sum_k p_k = 1 \\ & p \geq 0 \end{array} \tag{16.11}$$

One might want some statistical properties to match exactly. As an example, consider again the autoregressive model:

$$r_t = D_0 + D_1 r_{t-1} + \ldots + D_p r_{t-p} + \epsilon_t$$

where $\epsilon_t \sim N(0, \Sigma)$ are independently distributed multivariate normal distributions with mean 0 and covariance matrix Σ. To simplify notation, let us write ϵ instead of ϵ_t. The random vector ϵ has distribution $N(0, \Sigma)$ and we would like to approximate this continuous distribution by a finite number of disturbance vectors ϵ^k occuring with probability p_k, for $k = 1, \ldots, K$. Let ϵ_q^k denote the qth component of vector ϵ^k. One might want to fit the mean of ϵ exactly and its covariance matrix as well as possible. In this case, the fitting problem is:

$$\begin{array}{ll} \min_{\epsilon^1, \ldots, \epsilon^K, p} & \sum_{q=1}^{l} \sum_{r=1}^{l} (\sum_{k=1}^{K} p_k \epsilon_q^k \epsilon_r^k - \Sigma_{qr})^2 \\ & \sum_{k=1}^{K} p_k \epsilon^k = 0 \\ & \sum_k p_k = 1 \\ & p \geq 0 \end{array}$$

Arbitrage-free scenario trees

Approximating the continuous distributions of the uncertain parameters by a finite number of scenarios in the linear programming (16.6) typically creates modeling errors. In fact, if the scenarios are not chosen properly or if their number is too small, the supposedly "linear programming equivalent" could be far from being equivalent to the original stochastic program. One of the most disturbing aspects of this phenomenon is the possibility of creating

arbitrage opportunities when constructing the scenario tree. When this occurs, model (16.6) might produce unrealistic solutions that exploit these arbitrage opportunities. Klaassen [42] was the first to address this issue. In particular, he shows how arbitrage opportunities can be detected ex post in a scenario tree. When such arbitrage opportunities exist, a simple solution is to discard the scenario tree and to construct a new one with more branches. Klaassen [42] also discusses what constraints to add to the nonlinear program (16.11) in order to preclude arbitrage opportunities ex ante. The additional constraints are nonlinear, thus increasing the difficulty of solving (16.11). We present below Klassen's ex post check.

Recall that there are two types of arbitrage (Definition 4.1). We start we Type A. An arbitrage of Type A is a trading strategy with an initial positive cash flow and no risk of loss later. Let us express this at a node i of the scenario tree. Let r^k denote the vectors of rates of return on the branches connecting node i to its sons in the next stage, for $k = 1, \ldots, K$. There exists an arbitrage of Type A if there exists an asset allocation $x = (x_1, \ldots, x_Q)$ at node i such that

$$\sum_{q=1}^{Q} x_q < 0$$

$$\text{and} \quad \sum_{q=1}^{Q} x_q r_q^k \geq 0 \quad \text{for all } k = 1, \ldots, K.$$

To check whether such an allocation x exists, it suffices to solve the linear program

$$\begin{array}{ll} \min_x & \sum_{q=1}^{Q} x_q \\ & \sum_{q=1}^{Q} x_q r_q^k \geq 0 \quad \text{for all } k = 1, \ldots, K. \end{array} \tag{16.12}$$

There is an arbitrage opportunity of Type A at node i if and only if this linear program is unbounded.

Next we turn to Type B. An arbitrage of Type B requires no initial cash input, has no risk of a loss and a positive probability of making profits in the future. At node i of the scenario tree, this is expressed by the conditions:

$$\sum_{q=1}^{Q} x_q = 0,$$

$$\sum_{q=1}^{Q} x_q r_q^k \geq 0 \quad \text{for all } k = 1, \ldots, K$$

$$\text{and} \quad \sum_{q=1}^{Q} x_q r_q^k > 0 \quad \text{for at least one } k = 1, \ldots, K.$$

These conditions can be checked by solving the linear program

$$\begin{array}{ll} \max_x & \sum_{q=1}^{Q} x_q r_q^k \\ & \sum_{q=1}^{Q} x_q = 0 \\ & \sum_{q=1}^{Q} x_q r_q^k \geq 0 \quad \text{for all } k = 1, \ldots, K. \end{array} \tag{16.13}$$

There is an arbitrage opportunity of Type B at node i if and only if this linear program is unbounded.

Exercise 16.7 Show that the linear program (16.12) is always feasible.

Write the dual linear program of (16.12). Let u_k be the dual variable associated with the kth constraint of (16.12).

Recall that a feasible linear program is unbounded if and only if its dual is infeasible. Show that there is no arbitrage of Type A at node i if and only if there exists $u_k \geq 0$, for $k = 1, \ldots, K$ such that

$$\sum_{k=1}^{K} u_k r_q^k = 1 \quad \text{for all } q = 1, \ldots, Q.$$

Similarly, write the dual of (16.13). Let v_0, v_k, for $k = 1, \ldots, K$ be the dual variables. Write necessary and sufficient conditions for the nonexistence of arbitrage of Type B at node i, in terms of v_k, for $k = 0, \ldots, K$.

Modify the nonlinear program (16.11) in order to formulate a fitting problem at node i that contains no arbitrage opportunities.

Chapter 17

SP Models: Value-at-Risk and Conditional Value-at-Risk

In this chapter, we discuss Value-at-Risk, a widely used measure of risk in finance, and its relative Conditional Value-at-Risk. We then present an optimization model that optimizes a portfolio when the risk measure is the Conditional Value-at-Risk instead of the variance of the portfolio as in the Markowitz model. This is acheived through stochastic programming. In this case, the variables are anticipative. The random events are modeled by a large but finite set of scenarios, leading to a linear programming equivalent of the original stochastic program.

17.1 Risk Measures

Financial activities involve risk. Our stock or mutual fund holdings carry the risk of losing value due to market conditions. Even money invested in a bank carries a risk–that of the bank going bankrupt and never returning the money let alone some interest. While individuals generally just have to live with such risks, financial and other institutions can and very often must manage risk using sophisticated mathematical techniques. Managing risk requires a good understanding of quantitative risk measures that adequately reflect the vulnerabilities of a company.

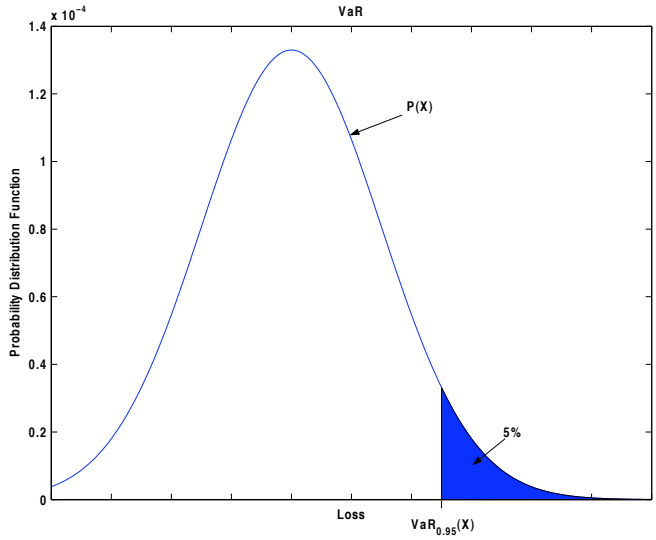
Perhaps the best-known risk measure is Value-at-Risk (VaR) developed by financial engineers at J.P. Morgan. VaR is a measure related to percentiles of loss distributions and represents the predicted maximum loss with a specified probability level (e.g., 95%) over a certain period of time (e.g., one day). Consider, for example, a random variable X that represents loss from an investment portfolio over a fixed period of time. A negative value for X indicates gains. Given a probability level α, α-VaR of the random variable X is given by the following relation:

$$\text{VaR}_\alpha(X) \quad := \quad \min\{\gamma : P(X \geq \gamma) \leq 1 - \alpha\}. \tag{17.1}$$

When the loss distribution is continuous, $\text{VaR}_\alpha(X)$ is simply the loss such that

$$P(X \leq \text{VaR}_\alpha(X)) = \alpha.$$

The following figure illustrates the 0.95-VaR on a portfolio loss distribution plot:



VaR is widely used by people in the financial industry and VaR calculators are common features in most financial software. Despite this popularity, VaR has one important undesirable property–it lacks subadditivity. Risk measures should respect the maxim "diversification reduces risk" and therefore, satisfy the following property: "The total risk of two different investment portfolios does not exceed the sum of the individual risks." This is precisely what we mean by saying that a risk measure should be a subadditive function, i.e., for a risk measure f, we should have

$$f(x_1 + x_2) \leq f(x_1) + f(x_2),\ \forall x_1, x_2.$$

Consider the following simple example that illustrates that diversification can actually increase the risk measured by VaR:

Example 17.1 *Consider two independent investment opportunities each returning a \$1 gain with probability 0.96 and \$2 loss with probability 0.04. Then, 0.95-VaR for both investments are -1. Now consider the sum of these two investment opportunities. Because of independence, this sum has the following loss distribution: \$4 with probability* $0.04 \times 0.04 = 0.0016$*, \$1 with probability* $2 \times 0.96 \times 0.04 = 0.0768$*, and -\$2 with probability* $0.96 \times 0.96 = 0.9216$*. Therefore, the 0.95-VaR of the sum of the two investments is 1, which exceeds -2, the sum of the 0.95-VaR values for individual investments.*

An additional difficulty with VaR is in its computation and optimization. When VaR is computed by generating scenarios, it turns out to be a

non-smooth and non-convex function of the positions in the investment portfolio. Therefore, when one tries to optimize VaR computed in this manner, multiple local optimizers are encountered, hindering the global optimization process.

Another criticism of VaR is that it pays no attention to the magnitude of losses beyond the VaR value. This and other undesirable features of VaR led to the development of alternative risk measures. One well-known modification of VaR is obtained by computing the *expected* loss *given* that the loss exceeds VaR. This quantity is often called *conditional Value-at-Risk* or CVaR. There are several alternative names for this measure in the finance literature including Mean Expected Loss, Mean Shortfall, and Tail VaR. We now describe this risk measure in more detail and discuss how it can be optimized using linear programming techniques when the loss function is linear in the portfolio positions. Our discussion follows parts of articles by Rockafellar and Uryasev [61, 75].

We consider a portfolio of assets with random returns. We denote the portfolio choice vector by x and the random events by the vector y. Let $f(x, y)$ denote the loss function when we choose the portfolio x from a set X of feasible portfolios and y is the realization of the random events. We assume that the random vector y has a probability density function denoted by $p(y)$.

For a fixed decision vector x, we compute the cumulative distribution function of the loss associated with that vector x:

$$\Psi(x, \gamma) \quad := \quad \int_{f(x,y)<\gamma} p(y)dy. \tag{17.2}$$

Then, for a given confidence level α, the α-VaR associated with portfolio x is given by

$$\mathrm{VaR}_\alpha(x) \quad := \quad \min\{\gamma \in I\!R : \Psi(x, \gamma) \geq \alpha\}. \tag{17.3}$$

We define the α-CVaR associated with portfolio x as:

$$\mathrm{CVaR}_\alpha(x) \quad := \quad \frac{1}{1-\alpha} \int_{f(x,y)\geq \mathrm{VaR}_\alpha(x)} f(x, y)p(y)dy. \tag{17.4}$$

Note that,

$$\begin{aligned}
\mathrm{CVaR}_\alpha(x) &= \frac{1}{1-\alpha} \int_{f(x,y)\geq \mathrm{VaR}_\alpha(x)} f(x, y)p(y)dy \\
&\geq \frac{1}{1-\alpha} \int_{f(x,y)\geq \mathrm{VaR}_\alpha(x)} \mathrm{VaR}_\alpha(x)p(y)dy \\
&= \frac{\mathrm{VaR}_\alpha(x)}{1-\alpha} \int_{f(x,y)\geq \mathrm{VaR}_\alpha(x)} p(y)dy \\
&\geq \mathrm{VaR}_\alpha(x),
\end{aligned}$$

i.e., CVaR of a portfolio is always at least as big as its VaR. Consequently, portfolios with small CVaR also have small VaR. However, in general minimizing CVaR and VaR are not equivalent.

For a discrete probability distribution (where event y_j occurs with probability p_j, for $j = 1, \ldots, n$), the above definition of CVaR becomes

$$\text{CVaR}_\alpha(x) = \frac{1}{1-\alpha} \sum_{j: f(x,y_j) \geq \text{VaR}_\alpha(x)} p_j f(x, y_j)$$

Example:

Suppose we are given the loss function $f(x, y)$ for a given decision x as $f(x, y) = -y$ where $y = 75 - j$ with probability 1 % for $j = 0, \ldots, 99$. We would like to determine the Value-at-Risk $\text{VaR}_\alpha(x)$ for $\alpha = 95\%$. We have $\text{VaR}_{95\%}(x) = 20$ since the loss is 20 or more with probability 5 %.

To compute the Conditional Value-at-Risk, we use the above formula: $\text{CVaR}_{95\%}(x) = \frac{1}{0.05}(20 + 21 + 22 + 23 + 24) \times 1\% = 22$.

Exercise 17.1 (a) Compute the 0.90-VaR and 0.90-CVaR for the rates of return of stocks between 1961 and 2003 (see Section 8.1.1 for the data).

(b) Compute the 0.90-VaR and 0.90-CVaR for the rates of return of bonds and a money market account. Again use the data of Section 8.1.1.

Exercise 17.2 Give an example showing that CVaR is not subadditive.

17.2 Minimizing CVaR

Since the definition of CVaR involves the VaR function explicitly, it is difficult to work with and optimize this function. Instead, we consider the following simpler auxiliary function:

$$F_\alpha(x, \gamma) \;:=\; \gamma + \frac{1}{1-\alpha} \int_{f(x,y) \geq \gamma} (f(x, y) - \gamma)\, p(y) dy. \qquad (17.5)$$

Alternatively, we can write $F_{\alpha,x}(\gamma)$ as follows:

$$F_\alpha(x, \gamma) \;=\; \gamma + \frac{1}{1-\alpha} \int (f(x, y) - \gamma)^+ \, p(y) dy, \qquad (17.6)$$

where $a^+ = \max\{a, 0\}$. This function, viewed as a function of γ, has the following important properties that make it useful for the computation of VaR and CVaR:

1. $F_\alpha(x, \gamma)$ is a convex function of γ.
2. $\text{VaR}_\alpha(x)$ is a minimizer over γ of $F_\alpha(x, \gamma)$.
3. The minimum value over γ of the function $F_\alpha(x, \gamma)$ is $\text{CVaR}_\alpha(x)$.

Exercise 17.3 Prove the properties of $F_{\alpha,x}(\gamma)$ stated above.

As a consequence of the listed properties, we immediately deduce that, in order to minimize $\mathrm{CVaR}_\alpha(x)$ over x, we need to minimize the function $F_\alpha(x,\gamma)$ with respect to x and γ simultaneously:

$$\min_{x\in X} \mathrm{CVaR}_\alpha(x) = \min_{x\in X,\gamma} F_\alpha(x,\gamma). \tag{17.7}$$

Consequently, we can optimize CVaR directly, without needing to compute VaR first. If the loss function $f(x,y)$ is a convex (linear) function of the portfolio variables x, then $F_\alpha(x,\gamma)$ is also a convex (linear) function of x. In this case, provided the feasible portfolio set X is also convex, the optimization problems in (17.7) are smooth convex optimization problems that can be solved using well known optimization techniques for such problems (see Chapter 5).

Often it is not possible or desirable to compute/determine the joint density function $p(y)$ of the random events in our formulation. Instead, we may have a number of scenarios, say y_s for $s = 1,\ldots,S$, which may represent some historical values of the random events or some values obtained via computer simulation. We will assume that all scenarios have the same probability. In this case, we obtain the following approximation to the function $F_\alpha(x,\gamma)$ by using the empirical distribution of the random events based on the available scenarios:

$$\tilde{F}_\alpha(x,\gamma) \;:=\; \gamma + \frac{1}{(1-\alpha)S}\sum_{s=1}^{S}\left(f(x,y_s)-\gamma\right)^+. \tag{17.8}$$

Compare this definition to (17.6). Now, the problem $\min_{x\in X}\mathrm{CVaR}_\alpha(x)$ can be approximated by replacing $F_\alpha(x,\gamma)$ with $\tilde{F}_\alpha(x,\gamma)$ in (17.7):

$$\min_{x\in X,\gamma}\ \gamma + \frac{1}{(1-\alpha)S}\sum_{s=1}^{S}\left(f(x,y_s)-\gamma\right)^+. \tag{17.9}$$

To solve this optimization problem, we introduce artificial variables z_s to replace $(f(x,y_s)-\gamma)^+$. This is achieved by imposing the constraints $z_s \geq f(x,y_s)-\gamma$ and $z_s \geq 0$:

$$\begin{array}{lrcl} \min_{x,z,\gamma} & \gamma + \frac{1}{(1-\alpha)S}\sum_{s=1}^{S} z_s & & \\ \text{s.t.} & z_s & \geq & 0,\ s=1,\ldots,S, \\ & z_s & \geq & f(x,y_s)-\gamma,\ s=1,\ldots,S, \\ & x \in X. & & \end{array} \tag{17.10}$$

Note that the constraints $z_s \geq f(x,y_s)-\gamma$ and $z_s \geq 0$ alone cannot ensure that $z_s = (f(x,y_s)-\gamma)^+ = \max\{f(x,y_s)-\gamma, 0\}$ since z_s can be larger than both right-hand-sides and be still feasible. However, since we are minimizing the objective function which involves a positive multiple of z_s, it will never be optimal to assign z_s a value larger than the maximum of the two quantities $f(x,y_s)-\gamma$ and 0, and therefore, in an optimal solution z_s will be precisely $(f(x,y_s)-\gamma)^+$, justifying our substitution.

In the case that $f(x,y)$ is linear in x, all the expressions $z_s \geq f(x,y_s)-\gamma$ represent linear constraints and therefore the problem (17.10) is a linear

programming problem that can be solved using the simplex method or alternative LP algorithms.

Other optimization problems arise naturally within the context of risk management. For example, risk managers often try to optimize a performance measure (e.g., expected return) while making sure that certain risk measures do not exceed a threshold value. When the risk measure is CVaR, the resulting optimization problem is:

$$\begin{array}{rrcl} \max_x & \mu^T x & & \\ \text{s.t.} & \text{CVaR}_{\alpha^j}(x) & \leq & U_{\alpha^j},\ j = 1, \ldots, J \\ & x \in X. & & \end{array} \tag{17.11}$$

Above, J is an index set for different confidence levels used for CVaR computations and U_{α^j} represents the maximum tolerable CVaR value at the confidence level α^j. As above, we can replace the CVaR functions in the constraints of this problem with the function $F_\alpha(x, \gamma)$ as above and then approximate this function using the scenarios for random events. This approach results in the following approximation of the CVaR-constrained problem (17.11):

$$\begin{array}{rrcl} \max_{x,z,\gamma} & \mu^T x & & \\ \text{s.t.} & \gamma + \frac{1}{(1-\alpha^j)S} \sum_{s=1}^{S} z_s & \leq & U_{\alpha^j},\ j = 1, \ldots, J, \\ & z_s & \geq & 0,\ s = 1, \ldots, S, \\ & z_s & \geq & f(x, y_s) - \gamma,\ s = 1, \ldots, S, \\ & x \in X. & & \end{array} \tag{17.12}$$

17.3 Example: Bond Portfolio Optimization

A portfolio of risky bonds might be characterized by a large likelihood of small earnings, coupled with a small chance of loosing a large amount of the investment. The loss distribution is heavily skewed and, in this case, standard mean-variance analysis to characterize market risk is inadequate. VaR and CVaR are more appropriate criteria for minimizing *portfolio credit risk*. Credit risk is the risk of a trading partner not fulfilling their obligation in full on the due date or at any time thereafter. Losses can result both from default and from a decline in market value stemming from downgrades in credit ratings. A good reference is the paper of Anderson, Mausser, Rosen and Uryasev [2].

Anderson, Mausser, Rosen and Uryasev consider a portfolio of 197 bonds from 29 different countries with a market value of $ 8.8 billion and duration of approximately 5 years. Their goal is to rebalance the portfolio in order to minimize credit risk. That is they want to minimize losses resulting from default and from a decline in market value stemming from downgrades in credit ratings (credit migration). The loss due to credit migration is simply

$$f(x, y) = (b - y)^T x$$

where b are the future values of each bond with no credit migration and y are the future values with credit migration (so y is a random vector). The

one-year portfolio credit loss was generated using a Monte Carlo simulation: 20,000 scenarios of joint credit states of obligators and related losses. The distribution of portfolio losses has a long fat tail. The authors rebalanced the portfolio by minimizing CVaR. The set X of feasible porfolios was described by the following constraints. Let x_i denote the weight of asset i in the portfolio. Upper and lower bounds were set on each x_i:

$$\begin{array}{l} l_i \le x_i \le u_i \quad i = 1, \ldots, n \\ \sum_i x_i = 1 \end{array}$$

To calculate the efficient frontier, the expected portfolio return was set to at least R:

$$\sum_i \mu_i x_i \ge R$$

To summarize, the linear program (17.10) to be solved was as follows:

$$\begin{array}{lll} \min_{x,z,\gamma} & \gamma + \frac{1}{(1-\alpha)S} \sum_{s=1}^{S} z_s & \\ \text{subject to} & z_s \ge \sum_i (b_i - y_{is}) x_i - \gamma & \text{for } s = 1, \ldots, S \\ & z_s \ge 0 & \text{for } s = 1, \ldots, S \\ & l_i \le x_i \le u_i & i = 1, \ldots, n \\ & \sum_i x_i = 1 & \\ & \sum_i \mu_i x_i \ge R & \end{array}$$

Consider $\alpha = 99\%$. The original bond portfolio had an expected portfolio return of 7.26%. The expected loss was 95 million dollars with a standard deviation of 232 million. The VaR was 1.03 billion dollars and the CVaR was 1.32 billion.

After optimizing the portfolio (with expected return of 7.26%), the expected loss was only 5 thousand dollars, with a standard deviation of 152 million. The VaR was reduced to 210 million and the CVaR to 263 million dollars. So all around, the characteristics of the portfolio were much improved. Positions were reduced in bonds from Brazil, Russia and Venezuela, whereas positions were increased in bonds from Thailand, Malaysia and Chile. Positions in bonds from Colombia, Poland and Mexico remained high and each accounted for about 5 % of the optimized CVaR.

Chapter 18

Stochastic Programming Models: Asset/Liability Management

18.1 Asset/Liability Management

Financial health of any company, and in particular those of financial institutions, is reflected in the balance sheets of the company. Proper management of the company requires attention to both sides of the balance sheet–assets and liabilities. Asset/Liability Management (ALM) offers sophisticated mathematical tools for an integrated management of assets and liabilities and is the focus of many studies in financial mathematics.

ALM recognizes that static, one period investment planning models (such as mean-variance optimization) fail to incorporate the multi-period nature of the liabilities faced by the company. A multi-period model that emphasizes the need to meet liabilities in each period for a finite (or possibly infinite) horizon is often required. Since liabilities and asset returns usually have random components, their optimal management requires tools of "Optimization under Uncertainty" and most notably, stochastic programming approaches.

We recall the ALM setting we introduced in Section 1.3.4: Let L_t be the liability of the company in year t for $t = 1, \ldots, T$. The L_t's are random variables. Given these liabilities, which assets (and in which quantities) should the company hold each year to maximize its expected wealth in year T? The assets may be domestic stocks, foreign stocks, real estate, bonds, etc. Let R_{it} denote the return on asset i in year t. The R_{it}'s are random variables. The decision variables are:

$$x_{it} = \text{ market value invested in asset } i \text{ in year } t.$$

The decisions x_{it} in year t are made after the random variables L_t and R_{it} are realized. That is, the decision problem is multistage, stochastic, with recourse. The stochastic program can be written as follows.

$$\begin{array}{rl} \max & E[\sum_i x_{iT}] \\ \text{subject to} & \\ \text{asset accumulation:} & \sum_i (1+R_{it}) x_{i,t-1} - \sum_i x_{it} = L_t \quad \text{for } t = 1, \ldots, T \\ & x_{it} \geq 0. \end{array}$$

The constraint says that the surplus left after liability L_t is covered will be invested as follows: x_{it} invested in asset i. In this formulation, $x_{0,t}$ are the fixed, and possibly nonzero initial positions in different asset classes. The objective selected in the model above is to maximize the expected wealth at the end of the planning horizon. In practice, one might have a different objective. For example, in some cases, minimizing Value at Risk (VaR) might be more appropriate. Other priorities may dictate other objective functions.

To address the issue of the most appropriate objective function, one must understand the role of liabilities. Pension funds and insurance companies are among the most typical arenas for the integrated management of assets and liabilities through ALM. We consider the case of a Japanese insurance company, the Yasuda Fire and Marine Insurance Co, Ltd, following the work of Cariño, Kent, Myers, Stacy, Sylvanus, Turner, Watanabe, and Ziemba [17]. In this case, the liabilities are mainly savings-oriented policies issued by the company. Each new policy sold represents a deposit, or inflow of funds. Interest is periodically credited to the policy until maturity, typically three to five years, at which time the principal amount plus credited interest is refunded to the policyholder. The crediting rate is typically adjusted each year in relation to a market index like the prime rate. Therefore, we cannot say with certainty what future liabilities will be. Insurance business regulations stipulate that interest credited to some policies be earned from investment income, not capital gains. So, in addition to ensuring that the maturity cash flows are met, the firm must seek to avoid interim shortfalls in income earned versus interest credited. In fact, it is the risk of not earning adequate income quarter by quarter that the decision makers view as the primary component of risk at Yasuda.

The problem is to determine the optimal allocation of the deposited funds into several asset categories: cash, fixed rate and floating rate loans, bonds, equities, real estate and other assets. Since we can revise the portfolio allocations over time, the decision we make is not just among allocations today but among allocation strategies over time. A realistic dynamic asset/liability model must also account for the payment of taxes. This is made possible by distinguishing between interest income and price return.

A stochastic linear program is used to model the problem. The linear program has uncertainty in many coefficients. This uncertainty is modeled through a finite number of scenarios. In this fashion, the problem is transformed into a very large scale linear program of the form (16.6). The random elements include price return and interest income for each asset class, as well as policy crediting rates.

We now present a multistage stochastic program that was developed for The Yasuda Fire and Marine Insurance Co., Ltd. Our presentation follows the description of the model as stated in [17].

Stages are indexed by $t = 0, 1, \ldots, T$.
Decision variables of the stochastic program:

$$\begin{aligned} x_{it} &= \text{market value in asset } i \text{ at } t, \\ w_t &= \text{interest income shortfall at } t \geq 1, \\ v_t &= \text{interest income surplus at } t \geq 1. \end{aligned}$$

Random variables appearing in the stochastic linear program: For $t \geq 1$,

$$\begin{aligned} RP_{it} &= \text{price return of asset } i \text{ from } t-1 \text{ to } t, \\ RI_{it} &= \text{interest income of asset } i \text{ from } t-1 \text{ to } t, \\ F_t &= \text{deposit inflow from } t-1 \text{ to } t, \\ P_t &= \text{principal payout from } t-1 \text{ to } t, \\ I_t &= \text{interest payout from } t-1 \text{ to } t, \\ g_t &= \text{rate at which interest is credited to policies from } t-1 \text{ to } t, \\ L_t &= \text{liability valuation at } t. \end{aligned}$$

Parameterized function appearing in the objective:

$$c_t = \text{piecewise linear convex cost function.}$$

The objective of the model is to allocate funds among available assets to maximize expected wealth at the end of the planning horizon T less expected penalized shortfalls accumulated through the planning horizon.

$$\begin{array}{rl} \max & E[\sum_i x_{iT} - \sum_{t=1}^{T} c_t(w_t)] \\ \text{subject to} & \\ \text{asset accumulation:} & \sum_i x_{it} - \sum_i (1 + RP_{it} + RI_{it}) x_{i,t-1} = F_t - P_t - I_t \quad \text{for } t = 1, \ldots, T \\ \text{interest income shortfall:} & \sum_i RI_{it} x_{i,t-1} + w_t - v_t = g_t L_{t-1} \quad \text{for } t = 1, \ldots, T \\ & x_{it} \geq 0, \;\; w_t \geq 0, \;\; v_t \geq 0. \end{array} \tag{18.1}$$

Liability balances and cash flows are computed so as to satisfy the liability accumulation relations.

$$L_t = (1 + g_t) L_{t-1} + F_t - P_t - I_t \quad \text{for } t \geq 1.$$

The stochastic linear program (18.1) is converted into a large linear program using a finite number of scenarios to deal with the random elements in the data. Creation of scenario inputs is made in stages using a tree. The tree structure can be described by the number of branches at each stage. For example, a 1-8-4-4-2-1 tree has 256 scenarios. Stage $t = 0$ is the initial stage. Stage $t = 1$ may be chosen to be the end of Quarter 1 and has 8 different scenarios in this example. Stage $t = 2$ may be chosen to be the end of Year 1, with each of the previous scenarios giving rise to 4 new scenarios, and so on. For the Yasuda Fire and Marine Insurance Co., Ltd., a problem with 7 asset classes and 6 stages gives rise to a stochastic linear program

(18.1) with 12 constraints (other than nonnegativity) and 54 variables. Using 256 scenarios, this stochastic program is converted into a linear program with several thousand constraints and over 10,000 variables. Solving this model yielded extra income estimated to about US$ 80 million per year for the company.

Exercise 18.1 Discuss the relevance of the techniques from Chapter 16 in the solution of the Yasuda Fire and Marine Insurance Co., such as scenario generation (correlation of the random parameters over time, variance reduction techniques in constructing the scenario tree), decomposition techniques to solve the large-scale linear programs.

18.1.1 Corporate Debt Management

A closely related problem to the asset/liability management (ALM) problem in corporate financial planning is the problem of debt management. Here the focus is on retiring (paying back) outstanding debt at minimum cost. More specifically, corporate debt managers must make financial decisions to minimize the costs and risks of borrowing to meet debt financing requirements. These requirements are often determined by the firm's investment decisions. Our discussion in this subsection is based on the article [24].

Debt managers need to choose the sources of borrowing, types of debts to be used, timing and terms of debts, whether the debts will be callable[1], etc., in a multi-period framework where the difficulty of the problem is compounded by the fact that the interest rates that determine the cost of debt are uncertain. Since interest rate movements can be modeled by random variables this problem presents an attractive setting for the use of stochastic programming techniques. Below, we discuss a deterministic linear programming equivalent of stochastic LP model for the debt management problem.

We consider a multi-period framework with T time periods. We will use the indices s and t ranging between 0 (now) and T (termination date, or horizon) to denote different time periods in the model. We consider K types of debt that are distinguished by market of issue, term and the presence (or absence) of call option available to the borrower. In our notation, the superscript k ranging between 1 and K will denote the different types of debt being considered.

The evolution of the interest rates are described using a scenario tree. We denote by $e_j = e_{j1}, e_{j2}, \ldots, e_{jT}, j = 1, \ldots, J$ a sample path of this scenario tree which corresponds to a sequence of interest rate events. When a parameter or variable is contingent on the event sequence e_j we use the notation (e_j) (see below).

The decision variables in this model are the following:

[1] A *callable debt* is a debt security whose issuer has the right to redeem the security prior to its stated maturity date at a price established at the time of issuance, on or after a specified date.

- $B_t^k(e_j)$: dollar amount at par[2] of debt type k **B**orrowed at the beginning of period t.
- $O_{s,t}^k(e_j)$: dollar amount at par of debt type k borrowed in period s and **O**utstanding at the beginning of period t.
- $R_{s,t}^k(e_j)$: dollar amount at par of debt type k borrowed in period s and **R**etired (paid back) at the beginning of period t.
- $S_t(e_j)$: dollar value of **S**urplus cash held at the beginning of period t.

Next, we list the input parameters to the problem:

- $r_{s,t}^k(e_j)$: interest payment in period t per dollar outstanding of debt type k issued in period s.
- f_t^k: issue costs (excluding premium or discount) per dollar borrowed of debt type k issued in period t.
- $g_{s,t}^k(e_j)$: retirement premium or discount per dollar for debt type k issued in period s, if retired in period t[3].
- $i_t(e_j)$: interest earned per dollar on surplus cash in period t.
- $p(e_j)$: probability of the event sequence e_j. Note that $p(e_j) \geq 0,\ \forall j$ and $\sum_{j=1}^{J} p(e_j) = 1$.
- C_t: cash requirements for period t, which can be negative to indicate an operating surplus.
- M_t: maximum allowable cost of debt service in period t.
- $q_t^k(Q_t^k)$: minimum (maximum) borrowing of debt type k in period t.
- $L_t(e_j)(U_t(e_j))$: minimum (maximum) dollar amount of debt (at par) retired in period t.

The objective function of this problem is expressed as follows:

$$\min \sum_{j=1}^{J} p(e_j) \left(\sum_{k=1}^{K} \sum_{t=1}^{T} \left(1 + g_{t,T}^k(e_j)\right) \left[O_{t,T}^k(e_j) - R_{t,T}^k(e_j)\right] + (1 - f_T^k) B_T^k(e_j) \right). \tag{18.2}$$

This function expresses the expected retirement cost of the total debt outstanding at the end of period T.

We complete the description of the deterministic equivalent of the stochastic LP by listing the constraints of the problem:

[2]At a price equal to the par (face) value of the security; the original issue price of a security.

[3]These parameters are used to define call options and to value the debt portfolio at the end of the planning period.

- **Cash Requirements:** For each time period $t = 1, \ldots, T$ and scenario path $j = 1, \ldots, J$:

$$C_t + S_t(e_j) = \sum_{k=1}^{K} \left\{ \left(1 - f_t^k\right) B_t^k(e_j) + (1 + i_{t-1}(e_j))\, S_{t-1}(e_j) - \sum_{s=0}^{t-1} \left[r_{s,t}^k(e_j) O_{s,t}^k(e_j) - \left(1 + g_{s,t}^k(e_j)\right) R_{s,t}^k(e_j) \right] \right\}.$$

  This balance equation indicates that the difference between cash available (new net borrowing, surplus cash from previous period and the interest earned on this cash) and the debt payments (interest on outstanding debt and cash outflows on repayment) should equal the cash requirements plus the surplus cash left for this period.

- **Debt Balance Constraints:** For $j = 1, \ldots, J$, $t = 1, \ldots, T$, $s = 0, \ldots, t-2$, and $k = 1, \ldots K$:

$$\begin{aligned} O_{s,t}^k(e_j) - O_{s,t-1}^k(e_j) + R_{s,t-1}^k(e_j) &= 0 \\ O_{t-1,t}^k(e_j) - B_{t-1}^k(e_j) - R_{t-1,t}^k(e_j) &= 0 \end{aligned}$$

- **Maximum cost of debt:** For $j = 1, \ldots, J$, $t = 1, \ldots, T$, and $k = 1, \ldots K$:

$$\sum_{s=1}^{t-1} \left(r_{s,t}^k(e_j) O_{s,t}^k(e_j) - i_{t-1}(e_j) S_{t-1}(e_j) \right) \leq M_t.$$

- **Borrowing limits:** For $j = 1, \ldots, J$, $t = 1, \ldots, T$, and $k = 1, \ldots K$:

$$q_t^k \leq B_t^k(e_j) \leq Q_t^k.$$

- **Payoff limits:** For $j = 1, \ldots, J$ and $t = 1, \ldots, T$:

$$L_t(e_j) \leq \sum_{k=1}^{K} \sum_{s=0}^{t-1} R_{s,t}^k(e_j) \leq U_t(e_j).$$

- **Nonnegativity:** For $j = 1, \ldots, J$, $t = 1, \ldots, T$, $s = 0, \ldots, t-2$, and $k = 1, \ldots K$:

$$B_t^k(e_j) \geq 0, \quad O_{s,t}^k(e_j) \geq 0, \quad R_{s,t}^k(e_j) \geq 0, \quad S_t(e_j) \geq 0.$$

In the formulation above, we used the notation of the article [24]. However, since the parameters and variables dependent on e_j can only depend on the portion of the sequence that is revealed by a certain time, a more precise notation can be obtained using the following ideas. First, let $e_j^t = e_{j1}, e_{j2}, \ldots, e_{jt}, j = 1, \ldots, J, t = 1, \ldots, T$, i.e., e_j^t represents the portion of e_j observed by time period t. Then, one replaces the expressions such as $S_t(e_j)$ with $S_t(e_j^t)$, etc.

18.2 Synthetic Options

An important issue in portfolio selection is the potential decline of the portfolio value below some critical limit. How can we control the risk of downside losses? A possible answer is to create a payoff structure similar to a European call option.

While one may be able to construct a diversified portfolio well suited for a corporate investor, there may be no option market available on this portfolio. One solution may be to use index options. However exchange-traded options with sufficient liquidity are limited to maturities of about three months. This makes the cost of long-term protection expensive, requiring the purchase of a series of high priced short-term options. For large institutional or corporate investors, a cheaper solution is to artificially produce the desired payoff structure using available resources. This is called a "synthetic option strategy".

The model is based on the following data.

W_0 = investor's initial wealth,
T = planning horizon,
R = riskless return for one period,
R_t^i = return for asset i at time t,
θ_t^i = transaction cost for purchases and sales of asset i at time t.

The R_t^i's are random, but we know their distributions.

The variables used in the model are the following.

x_t^i = amount allocated to asset i at time t,
A_t^i = amount of asset i bought at time t,
D_t^i = amount of asset i sold at time t,
α_t = amount allocated to riskless asset at time t.

We formulate a stochastic program that produces the desired payoff at the end of the planning horizon T, much in the flavor of the stochastic programs developed in the previous two sections. Let us first discuss the constraints.

The initial portfolio is

$$\alpha_0 + x_0^1 + \ldots + x_0^n = W_0.$$

The portfolio at time t is

$$x_t^i = R_t^i x_{t-1}^i + A_t^i - D_t^i \quad \text{for } t = 1, \ldots, T$$

$$\alpha_t = R\alpha_{t-1} - \sum_{i=1}^{n}(1+\theta_t^i)A_t^i + \sum_{i=1}^{n}(1-\theta_t^i)D_t^i \quad \text{for } t = 1, \ldots, T.$$

One can also impose upper bounds on the proportion of any risky asset in the portfolio:

$$0 \le x_t^i \le m_t(\alpha_t + \sum_{j=1}^{n} x_t^j),$$

where m_t is chosen by the investor.

The value of the portfolio at the end of the planning horizon is:

$$v = R\alpha_{T-1} + \sum_{i=1}^{n}(1-\theta_T^i)R_T^i x_{T-1}^i,$$

where the summation term is the value of the risky assets at time T.

To construct the desired synthetic option, we split v into the riskless value of the portfolio Z and a surplus $z \geq 0$ which depends on random events. Using a scenario approach to the stochastic program, Z is the worst-case payoff over all the scenarios. The surplus z is a random variable that depends on the scenario. Thus

$$v = Z + z$$

$$z \geq 0.$$

We consider Z and z as variables of the problem, and we optimize them together with the asset allocations x and other variables described earlier. The objective function of the stochastic program is

$$\max E(z) + \mu Z$$

where $\mu \geq 1$ is the risk aversion of the investor. The risk aversion μ is given data.

When $\mu = 1$, the objective is to maximize expected return.

When μ is very large, the objective is to maximize "riskless profit" as we defined it in Chapter 4 (Exercise 4.10).

As an example, consider an investor with initial wealth $W_0 = 1$ who wants to construct a portfolio comprising one risky asset and one riskless asset using the "synthetic option" model described above. We write the model for a two-period planning horizon, i.e. $T = 2$. The return on the riskless asset is R per period. For the risky asset, the return is R_1^+ with probability .5 and R_1^- with the same probability at time $t = 1$. Similarly, the return of the risky asset is R_2^+ with probability .5 and R_2^- with the same probability at time $t = 2$. The transaction cost for purchases and sales of the risky asset is θ.

There are 4 scenarios in this example, each occurring with probability .25, which we can represent by a binary tree. The initial node will be denoted by 0, the up node from it by 1 and the down node by 2. Similarly the up node from node 1 will be denoted by 3, the down node by 4, and the successors of 2 by 5 and 6 respectively. Let x_i, α_i denote the amount of risky asset and of riskless asset respectively in the portfolio at node i of this binary tree. Z is the riskless value of the portfolio and z_i is the surplus at node i. The linear program is:

$$
\begin{array}{rl}
\max & .25z_3 + .25z_4 + .25z_5 + .25z_6 + \mu Z \\
\text{subject to} & \\
\text{initial portfolio:} & \alpha_0 + x_0 = 1 \\
\text{rebalancing constraints:} & x_1 = R_1^+ x_0 + A_1 - D_1 \\
& \alpha_1 = R\alpha_0 - (1+\theta)A_1 + (1-\theta)D_1 \\
& x_2 = R_1^- x_0 + A_2 - D_2 \\
& \alpha_2 = R\alpha_0 - (1+\theta)A_2 + (1-\theta)D_2 \\
\text{payoff:} & z_3 + Z = R\alpha_1 + (1-\theta)R_2^+ x_1 \\
& z_4 + Z = R\alpha_1 + (1-\theta)R_2^- x_1 \\
& z_5 + Z = R\alpha_2 + (1-\theta)R_2^+ x_2 \\
& z_6 + Z = R\alpha_2 + (1-\theta)R_2^- x_2 \\
\text{nonnegativity:} & \alpha_i, x_i, z_i, A_i, D_i \geq 0.
\end{array}
$$

Example: An interesting paper discussing synthetic options is the paper of Y. Zhao and W.T. Ziemba [77]. Zhao and Ziemba apply the synthetic option model to an example with 3 assets (cash, bonds and stocks) and 4 periods (a one-year horizon with quarterly portfolio reviews). The quarterly return on cash is constant at $\rho = 0.0095$. For stocks and bonds, the expected logarithmic rates of returns are $s = 0.04$ and $b = 0.019$ respectively. Transaction costs are 0.5% for stocks and 0.1% for bonds. The scenarios needed in the stochastic program are generated using an auto regression model which is constructed based on historical data (quarterly returns from 1985 to 1998; the Salomon Brothers bond index and S&P 500 index respectively). Specifically, the auto regression model is

$$
\begin{cases}
s_t = 0.037 - 0.193 s_{t-1} + 0.418 b_{t-1} - 0.172 s_{t-2} + 0.517 b_{t-2} + \epsilon_t \\
b_t = 0.007 - 0.140 s_{t-1} + 0.175 b_{t-1} - 0.023 s_{t-2} + 0.122 b_{t-2} + \eta_t
\end{cases}
$$

where the pair (ϵ_t, η_t) characterizes uncertainty. The scenarios are generated by selecting 20 pairs of (ϵ_t, η_t) to estimate the empirical distribution of one period uncertainty. In this way, a scenario tree with 160,000 ($= 20 \times 20 \times 20 \times 20$) paths describing possible outcomes of asset returns is generated for the 4 periods.

The resulting large scale linear program is solved. We discuss the results obtained when this linear program is solved for a risk aversion of $\mu = 2.5$: The value of the terminal portfolio is always at least 4.6% more than the initial portfolio wealth and the distribution of terminal portfolio values is skewed to larger values because of dynamic downside risk control. The expected return is 16.33% and the volatility is 7.2%. It is interesting to compare these values with those obtained from a static Markowitz model: The expected return is 15.4% for the same volatility but no minimum return is guaranteed! In fact, in some scenarios, the value of the Markowitz portfolio is 5% *less* at the end of the one-year horizon than it was at the beginning.

It is also interesting to look at an example of a typical portfolio (one of the 160,000 paths) generated by the synthetic option model (the linear program was set up with an upper bound of 70 % placed on the fraction of stocks or bonds in the portfolio):

	Cash	Stocks	Bonds	Portfolio value at end of period
				100
Period 1	12%	18%	70%	103
2		41%	59%	107
3		70%	30%	112
4	30%		70%	114

Exercise 18.2 Computational exercise: Develop a synthetic option model in the spirit of that used by Zhao and Ziemba, adapted to the size limitation of your linear programming solver. Compare with a static model.

18.3 Case Study: Option Pricing with Transaction Costs

A European call option on a stock with maturity T and strike price X gives the right to buy the stock at price X at time T. The holder of the option will not exercise this option if the stock has a price S lower than X at time T. Therefore the value of a European call option is $\max(S - X, 0)$. Since S is random, the question of pricing the option correctly is of interest. The Black Scholes Merton Option Pricing model relates the price of an option to the volatility of the stock return. The assumptions are that the market is efficient and that the returns are lognormal. ¿From the volatility σ of the stock return, one can compute the option price for any strike price X. Conversely, from option prices one can compute the implied volatility σ. For a given stock, options with different strike prices should lead to the same σ (if the assumptions of the Black Scholes Merton model are correct).

The aim of the model developed in this section is to examine the extent to which market imperfections can explain the deviation of observed option prices from the Black Scholes Merton Option Pricing model. One way to measure the deviation of the Black Scholes Merton model from observed option prices is through the "volatility smile": for a given maturity date, the implied volatility of a stock computed by the Black Scholes Merton model from observed option prices at different strike prices is typically not constant, but instead often exhibits a convex shape as the strike price increases (the "smile"). One explanation for the deviation is that the smile occurs because the Black Scholes Merton model assumes the ability to rebalance portfolios without costs imposed either by the inability to borrow or due to a bid-ask spread or other trading costs. Here we will look at the effect of transaction costs on option prices.

The derivation of the Black Scholes Merton formula is through a replicating portfolio containing the stock and a riskless bond. If the market is efficient, we should be able to replicate the option payoff at time T by rebalancing the portfolio between now and time T, as the stock price evolves. Rather than work with a continuous time model, we discretize this process.

This discretization is called the binomial approximation to the Black Scholes Merton Option Pricing model. In this model, we specify a time period Δ between trading opportunities and postulate the behavior of stock and bond prices along successive time periods. The binomial model assumes that in between trading periods, only two possible stock price movements are possible.

a) There are N stages in the tree, indexed $0, 1 \ldots, N$, where stage 0 is the root of the tree and stage N is the last stage. If we divide the maturity date T of an option by N, we get that the length of a stage is $\Delta = T/N$.

b) Label the initial node k_0.

c) For a node $k \neq k_0$, let k^- be the node that is the immediate predecessor of k.

d) Let $S(k)$ be the stock price at node k and let $B(k)$ be the bond price at node k.

e) We assume that the interest rate is fixed at the annualized rate r so that $B(k) = B(k^-)e^{r\Delta}$.

f) Letting σ denote the volatility of the stock return, we use the standard parametrization $u = e^{\sigma\sqrt{\Delta}}$ and $d = 1/u$. So $S(k) = S(k^-)e^{\sigma\sqrt{\Delta}}$ if an uptick occurs from k^- to k and $S(k) = S(k^-)e^{-\sigma\sqrt{\Delta}}$ if a downtick occurs.

g) Let $n(k)$ be the quantity of stocks at node k and let $m(k)$ be the quantity of bonds at k.

18.3.1 The Standard Problem

In the binomial model, we have dynamically complete markets. This means that by trading the stock and the bond dynamically, we can replicate the payoffs (and values) from a call option. The option value is simply the cost of the replicating portfolio, and the replicating portfolio is self-financing after the first stage. This means that after we initially buy the stock and the bond, all subsequent trades do not require any additional money and, at the last stage, we reproduce the payoffs from the call option.

Therefore, we can represent the option pricing problem as the following linear program. Choose quantities $n(k)$ of the stock, quantities $m(k)$ of the bond at each nonterminal node k to

$$
\begin{array}{rl}
(5) \quad \min & n(k_0)S(k_0) + m(k_0)B(k_0) \\
\text{subject to} & \\
\text{rebalancing constraints:} & n(k^-)S(k) + m(k^-)B(k) \geq n(k)S(k) + m(k)B(k) \\
& \text{for every node } k \neq k_0 \\
\text{replication constraints:} & n(k^-)S(k) + m(k^-)B(k) \geq \max(S(k) - X, 0) \\
& \text{for every terminal node } k
\end{array}
$$

where k^- denotes the predecessor of k.

Note that we do not impose nonnegativity constraints since we will typically have a short position in the stock or bond.

Exercise 18.3 For a nondividend paying stock, collect data on 4 or 5 call options for the nearest maturity (but at least one month). Calculate the implied volatility for each option. Solve the standard problem (5) when the number of stages is 7 using the implied volatility of the at-the-money option to construct the tree.

18.3.2 Transaction Costs

To model transaction costs, we consider the simplest case where there are no costs of trading at the initial and terminal nodes, but there is a bid-ask spread on stocks at other nodes. So assume that if you buy a stock at node k, you pay $S(k)(1+\theta)$ while if you sell a stock, you receive $S(k)(1-\theta)$. This means that the rebalancing constraint becomes

$$n(k^-)S(k) + m(k^-)B(k) \geq n(k)S(k) + m(k)B(k) + |n(k) - n(k^-)|\theta S(k).$$

There is an absolute value in this constraint. So it is not a linear constraint. However it can be linearized as follows. Define two nonnegative variables:

$$\begin{aligned} x(k) &= \text{number of stocks } \textit{bought} \text{ at node } k, \text{ and} \\ y(k) &= \text{number of stocks } \textit{sold} \text{ at node } k. \end{aligned}$$

The rebalancing constraint now becomes:

$$\begin{aligned} &n(k^-)S(k) + m(k^-)B(k) \geq n(k)S(k) + m(k)B(k) + (x(k) + y(k))\theta S(k) \\ &n(k) - n(k^-) = x(k) - y(k) \\ &x(k) \geq 0, \quad y(k) \geq 0. \end{aligned}$$

Note that this constraint leaves the possibility of simultaneously buying and selling stocks at the same node. But obviously this cannot improve the objective function that we minimize in (5), so we do not need to impose a constraint to prevent it.

The modified formulation is:

$$\begin{aligned} (6) \quad \min \quad & n(k_0)S(k_0) + m(k_0)B(k_0) \\ \text{subject to} & \\ \text{rebalancing constraints:} \quad & n(k^-)S(k) + m(k^-)B(k) \geq n(k)S(k) + m(k)B(k) \\ & +(x(k) + y(k))\theta S(k) \quad \text{for every node } k \neq k_0 \\ & n(k) - n(k^-) = x(k) - y(k) \quad \text{for every node } k \neq k_0 \\ \text{replication constraints:} \quad & n(k^-)S(k) + m(k^-)B(k) \geq \max(S(k) - X, 0) \\ & \text{for every terminal node } k \\ \text{nonnegativity:} \quad & x(k) \geq 0, \quad y(k) \geq 0 \quad \text{for every node } k \neq k_0. \end{aligned}$$

Exercise 18.4 Repeat the exercise in Section 18.3.1 allowing for transaction costs, with different values of θ, to see if the volatility smile can be explained by transaction costs. Specifically, given a value for σ and for θ, calculate option prices and see how they match up to observed prices. Try $\theta = 0.001, 0.005, 0.01, 0.02, 0.05$.