Solving Nonlinear Portfolio Optimization Problems with the Primal-Dual Interior Point Method

Jacek Gondzio Andreas Grothey

April 2, 2004

MS-04-001

Solving Nonlinear Portfolio Optimization Problems with the Primal-Dual Interior Point Method*

Jacek Gondzio† Andreas Grothey‡

School of Mathematics
The University of Edinburgh
Mayfield Road, Edinburgh EH9 3JZ
United Kingdom.

March 31st, 2004

*Supported by the Engineering and Physical Sciences Research Council of UK, EPSRC grant GR/R99683/01.
†Email: J.Gondzio@ed.ac.uk, URL: http://maths.ed.ac.uk/~gondzio/
‡Email: A.Grothey@ed.ac.uk, URL: http://maths.ed.ac.uk/~agr/

Solving Nonlinear Portfolio Optimization Problems with the Primal-Dual Interior Point Method

Abstract

Stochastic programming is recognized as a powerful tool to help decision making under uncertainty in financial planning. The deterministic equivalent formulations of these stochastic programs have huge dimensions even for moderate numbers of assets, time stages and scenarios per time stage. So far models treated by mathematical programming approaches have been limited to simple linear or quadratic models due to the inability of currently available solvers to solve NLP problems of typical sizes. However stochastic programming problems are highly structured. The key to the efficient solution of such problems is therefore the ability to exploit their structure. Interior point methods are well-suited to the solution of very large nonlinear optimization problems. In this paper we exploit this feature and show how portfolio optimization problems with sizes measured in millions of constraints and decision variables, featuring constraints on semi-variance, skewness or nonlinear utility functions in the objective, can be solved with the state-of-the-art solver.

1 Introduction

Stochastic programming is recognized as an important tool in financial planning. Reformulated as the deterministic equivalent problems the stochastic programs become treatable by standard mathematical programming approaches. Unfortunately these deterministic equivalent formulations lead to problems with huge dimensions even for moderate sizes of event trees used to model the uncertainty. The advances in large scale optimization during the last decade have made these problems tractable. However current practice of portfolio optimization is still limited to simple formulations of linear programming (LP) or quadratic programming (QP) type.

In this paper we show that the mathematical programming methodology is ready to tackle the huge problems arising from portfolio optimization, even if general nonlinear constraints or objective terms are present.

Ever since Markowitz [13, 14] put forward the argument for a mean-variance model, different formulations of the portfolio optimization problems have been suggested by practitioners. However, the mean-variance model may lead to inferior conclusions, that is some optimal solutions may be stochastically dominated by other feasible solutions [17, 16]. Ogryczak and Ruszczynski [17] proved that the standard semideviation (the square root of the semivariance) is, under some additional conditions, consistent with second degree stochastic dominance. Therefore one should replace the usual objective in the Markowitz model with a new one composed of mean and semideviation. Alternatively one can model the risk preference by a constraint on semideviation (or semivariance). Logarithmic utility functions of the final wealth have been recognized since the 1950's for their attractive theoretical properties [10]. The justification for neglecting third moments have been questioned by Konno et al. [11, 12] because experimental data shows that stocks are non-symmetrically distributed. They argue to include skewness into the objective function to express the investors preference for a positive deviation from the mean in the case of non-symmetric distributions of asset returns. All these modifications require the introduction of nonlinearities either to the objective function or to the constraints and lead to deterministic equivalent formulations that are of nonlinear programming (NLP) type.

The last decade has seen a rapid improvement of methods to solve large scale stochastic programs. However most of these are only applicable in a very special setting. Nested Benders Decomposition approaches [1, 18] are limited to LP formulations. Linear algebra approaches such as [2, 19] are usually limited to very special structures resulting for example from constraints on the allowed type of recurrence relation. The aim of this paper is to show that by using a modern general structure exploiting interior point method such as our Object-Oriented Parallel Solver OOPS [7, 6] the solution of very large, general nonlinear stochastic programs is feasible. Furthermore this method is not limited to a special setting but can be applied to almost any conceivable constraint or objective function in stochastic programming.

In the following section we recall the standard mean-variance Asset and Liability Management (ALM) problem to introduce our notation. In Section 3 we discuss several extensions of the basic model that lead to nonlinear programming problems. In Section 4 we discuss their structure and comment briefly on how this can be exploited by a modern interior point solver. In Section 5 we present computational evidence that suggest that nonlinear stochastic problems are now tractable with state-of-the-art implementations. In Section 6 we give our conclusions.

2 Mean-variance Asset Liability Management Problem

We will describe a multi-stage Asset and Liability Management model which uses the mean to measure the reward and the variance to measure the risk. Our description of the problem follows those in [8, 20, 21]. We are concerned with finding the optimal way of investing into assets $j = 1, \ldots, J$. The returns on assets are uncertain. An initial amount of cash b is invested at $t = 0$ and the portfolio may be rebalanced at discrete times $t = 1, \ldots, T$. The objective is to maximize the expectation of the final value of the portfolio at time $T + 1$ and minimize the associated risk measured with the variance of the final wealth. We denote the decision variables at time t by x_t.

The uncertainty in the process is described by an *event tree*: discrete random events ω_t are observed at times $t = 0, \ldots, T$; each of the ω_t has only a finite number of possible outcomes. For each sequence of observed events $(\omega_0, \ldots, \omega_t)$, we expect one of only finitely many possible outcomes for the next observation ω_{t+1}. This branching process creates a tree rooted at the initial event ω_0. Let L_t be the set of nodes representing past observations $(\omega_0, \ldots, \omega_t)$ at time t, L_T the set of final nodes (leaves) and $L = \bigcup_t L_t$ the complete node set. In what follows a variable $i \in L$ will denote nodes, with $i = 0$ corresponding to the root and $\pi(i)$ denoting the predecessor (parent) of node i as indicated in Figure 1. Further p_i is the total probability of reaching node i, i.e. on each level set the probabilities sum up to one.

Let v_j be the value of asset j, and c_t the transaction cost. It is assumed that the value of the assets will not change throughout time and a unit of asset j can always be bought for $(1+c_t)v_j$ or sold for $(1-c_t)v_j$. Instead a unit of asset j held in node i (coming from node $\pi(i)$) will generate extra return $r_{i,j}$. Denote by $x^h_{i,j}$ the units of asset j held at node i and by $x^b_{i,j}, x^s_{i,j}$ the transaction volume (buying, selling) of this asset at this node, respectively. Similarly $x^h_{t,j}, x^b_{t,j}, x^s_{t,j}$ are the random variables describing the holding, buying and selling of asset j at time stage t. We assume that we start with zero holding of all assets but with funds b to invest. Further we assume that one of the assets represents cash, i.e. the available funds are always fully invested.

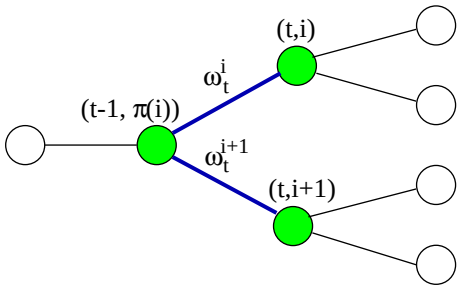


Figure 1: Event tree.

The standard constraints on the investment policy can be expressed as follows: Cash balance constraints describe possible buying and selling actions within a scenario while taking transaction costs into account:

$$\begin{array}{rcll} \sum_j (1+c_t) v_j x^b_{i,j} & = & \sum_j (1-c_t) v_j x^s_{i,j} & \forall i \neq 0 \\ \sum_j (1+c_t) v_j x^b_{0,j} & = & b. & \end{array} \tag{1}$$

Each scenario is linked to its parent through inventory constraints: these are balance constraints on asset holdings (taking into account the random return on asset):

$$(1+r_{i,j}) x^h_{\pi(i),j} = x^h_{i,j} - x^b_{i,j} + x^s_{i,j}, \quad \forall i \neq 0, j. \tag{2}$$

In the original Markowitz portfolio optimization problem [13, 14] two objectives are considered: maximization of the final wealth and minimization of the associated risk. In the multistage context, the final wealth y is simply expressed as the expected value of the final portfolio converted into cash [20]

$$y = I\!E((1-c_t) \sum_{j=1}^{J} v_j x^h_{T,j}) = (1-c_t) \sum_{i \in L_T} p_i \sum_{j=1}^{J} v_j x^h_{i,j}. \tag{3}$$

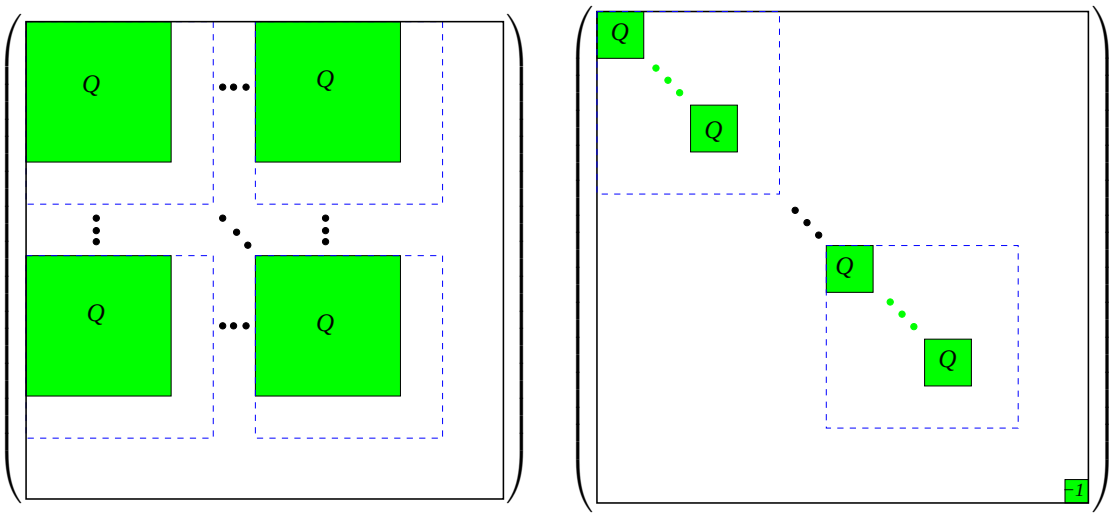
The risk is conventionally expressed as the variance of the return:

$$\mathrm{Var}((1-c_t) \sum_{j=1}^{J} v_j x^h_{T,j}) \quad = \quad \sum_{i \in L_T} p_i [(1-c_t) \sum_j v_j x^h_{i,j} - y]^2. \tag{4}$$

In the classical Markowitz portfolio optimization problem these two objectives are combined into a single one of the following form

$$f(x) = I\!E(F) - \rho \mathrm{Var}(F), \tag{5}$$

where F denotes the final portfolio converted into cash (3) and ρ is a scalar expressing investor's attitude to risk. Thus in the classical model we would maximize (5) subject to constraints (1), (2) and (3). Such a problem is a convex quadratic program. Indeed, all its constraints are linear and its objective is a concave quadratic function of x. The variance (4) gathers all $x^h_{T,j}$ (assets held in the last time period) and thus produces a quadratic form $x^T Q x$ with a very large number of nonzero entries in matrix Q. Gondzio and Grothey [6] proposed a reformulation of the problem which exploits partial separability of the variance and leads to a non-convex formulation of the



$\sum_{i\in L_T} p_i[(1-c_t)\sum_j v_j x^h_{i,j} - y]^2$
dense, convex QP

$\sum_{i\in L_T} p_i(1-c_t)^2[\sum_j v_j x^h_{i,j}]^2 - y^2$
sparse, nonconvex QP

Figure 2: Hessian matrices in two formulations of the variance.

problem but with a much higher degree of sparsity in the quadratic term. The reformulation exploits another representation of the variance (using $\mathrm{Var}(X) = I\!E(X^2) - I\!E(X)^2$):

$$\begin{aligned}\mathrm{Var}((1-c_t)\sum_{j=1}^{J} v_j x^h_{T,j}) &= I\!E((1-c_t)^2[\sum_j v_j x^h_{T,j}]^2) - I\!E((1-c_t)\sum_j v_j x^h_{T,j})^2 \\ &= \sum_{i\in L_T} p_i(1-c_t)^2[\sum_j v_j x^h_{i,j}]^2 - y^2.\end{aligned}$$

The non-convex formulation uses one additional decision variable y in the objective but produces better block-sparsity properties in the associated Hessian matrix as displayed in Figure 2.

3 Extensions of Asset Liability Management Problem

The mean-variance model does not satisfy the second order stochastic dominance condition [17]. This is a serious drawback of the classical Markowitz model. Furthermore, this model penalizes equally the overperformance and the underperformance of the portfolio which is counter-intuitive for a portfolio manager. The model presented in the previous section may be easily extended to take into account only a semi-variance (downside risk).

To allow more flexibility for the modelling we introduce two more (nonnegative) variables s_i^+, s_i^- per scenario $i \in L_t$ as the positive and negative variation from the mean and add the constraint

$$(1-c_t)\sum_{j=1}^{J} v_j x^h_{i,j} + s_i^+ - s_i^- = y, \quad i \in L_T \tag{6}$$

into the model. These variables might not strictly be needed in all described circumstances. Their purpose is primarily to show that extensions to the mean-variance model can be easily incorporated and lead to structured sparse problems. Using these additional variables the variance can be expressed much simpler as

$$\mathrm{Var}(X) = \sum_{i \in L_t} p_i (s_i^+ - s_i^-)^2 = \sum_{i \in L_t} p_i ((s_i^+)^2 + (s_i^-)^2), \tag{7}$$

as long as $(s_i^+)^2, (s_i^-)^2$ are not both positive at the solution. The formulation with these slack variables would further improve the sparsity of the Hessian matrix. Actually, the corresponding Hessian would be diagonal, that is, the quadratic problem would be separable. However, in this formulation, a considerable number of linear equality constraints need to be added to the problem. This new formulation allows an easy extension in which semivariance $I\!E[(X - I\!EX)_-^2]$ is used to measure downside risk

$$\mathrm{sVar}(X) = \sum_{i \in L_t} p_i (s_i^+)^2. \tag{8}$$

Using the notation introduced above the standard Markowitz model can be expressed as

$$\begin{array}{ll} \max_{x,y,s \geq 0} & y - \rho[\sum_{i \in L_T} p_i ((s_i^+)^2 + (s_i^-)^2)] \\ \text{s.t.} & (1), (2), (3), (6), \end{array} \tag{9}$$

which is a convex quadratic programming problem. We will consider the following variations which lead to nonlinear problem formulations.

- A constraint on risk (measured by the variance): We capture the investors risk-adversity directly in a form of a (nonlinear) constraint, i.e.

$$\begin{array}{ll} \max_{x,y,s \geq 0} & y \\ \text{s.t.} & \sum_{i \in L_t} p_i ((s_i^+)^2 + (s_i^-)^2) \leq \rho \\ & (1), (2), (3), (6). \end{array} \tag{10}$$

- A constraint on downside risk (measured by the semi-variance):

$$\begin{array}{ll} \max_{x,y,s \geq 0} & y \\ \text{s.t.} & \sum_{i \in L_t} p_i (s_i^+)^2 \leq \rho \\ & (1), (2), (3), (6). \end{array} \tag{11}$$

- A logarithmic utility function as the objective:

$$\begin{array}{ll} \max_{x,y,s \geq 0} & \log(1 + y) \\ \text{s.t.} & \sum_{i \in L_t} p_i (s_i^+)^2 \leq \rho \\ & (1), (2), (3), (6). \end{array} \tag{12}$$

- An objective including skewness: Konno et al. [11] suggest using the third moment in the model to capture the investors preference towards positive deviations in the case of non-symmetric distribution of returns for some assets

$$\begin{array}{ll} \max_{x,y,s \geq 0} & y + \gamma \sum_{i \in L_t} p_i (s_i^+ - s_i^-)^3 \\ \text{s.t.} & \sum_{i \in L_t} p_i ((s_i^+)^2 + (s_i^-)^2) \leq \rho \\ & (1), (2), (3), (6). \end{array} \tag{13}$$

All these formulations are nonlinear programming problems. While many of them have been recognized for their theoretical properties, solving them has not been possible due to the complexities involved with nonlinear programming. The few algorithms that there are, are highly adapted to a particular problem and not easily applicable to more general settings [2, 19]. We will show that by using a structure exploiting interior point method, these formulations can now be solved by standard optimization software. Unlike other approaches, our general interior point solver is still applicable after changes to the formulations as long as some general block-structure properties are present. We will point these out in the next section.

4 Structure of Nonlinear ALM Formulations

To solve the nonlinear ALM variations (10) through (13) we use a textbook sequential quadratic programming (SQP) method that employs our Object-Oriented Parallel interior point Solver (OOPS) [7, 6] as the underlying QP solver. We have implemented an SQP procedure following [3]. The reader interested in more detail in the nonlinear programming algorithms should consult the classic book of Fletcher [5] or newer books on the subject [4, 15].

An SQP method to solve $\min_x f(x)$ s.t. $g(x) \leq 0$, generates primal and dual iterates $x^{(k)}, \lambda^{(k)}$ and at each step solves the quadratic programming problem

$$\begin{array}{ll} \min_{\Delta x} & \nabla f(x^{(k)})^T \Delta x + \frac{1}{2} \Delta x^T Q \Delta x \\ \text{s.t.} & A\Delta x \leq -g(x^{(k)}), \end{array} \tag{14}$$

where $A = \nabla g(x^{(k)})$ and $Q = \nabla^2 f(x^{(k)}) + \sum_i \lambda^{(k)} \nabla^2 g_i(x^{(k)})$ are the Jacobian of the constraints and the Hessian of the Lagrangian respectively.

Our interior point based QP solver OOPS can efficiently exploit virtually any nested block-structure of the system matrices A and Q. We refer the reader to [6] for a detailed description of exploitable structures and the algorithms used to exploit them. For the purposes of this discussion a subset of the exploitable structures consists of matrices that are nested combinations of primal or dual block-angular and bordered block-diagonal matrices. We recall these well-known sparsity patterns in Figure 3.

$$\begin{bmatrix} A_1 & & & & \\ & A_2 & & & \\ & & \ddots & & \\ & & & A_n & \\ B_1 & B_2 & \cdots & B_n & B_{n+1} \end{bmatrix}, \quad \begin{bmatrix} A_1 & & & C_1 \\ & A_2 & & C_2 \\ & & \ddots & \vdots \\ & & A_n & C_n \end{bmatrix}, \quad \begin{bmatrix} A_1 & & & C_1 \\ & A_2 & & C_2 \\ & & \ddots & \vdots \\ & & A_n & C_n \\ B_1 & B_2 & \cdots & B_n & B_{n+1} \end{bmatrix}.$$

Figure 3: Sparsity patterns of typical exploitable structures.

The matrices Q and A in the asset liability problems display such nested block-sparse structures. We will start by analysing the structures of matrices A and Q for problem (9) and afterwards emphasize modifications needed for formulations (10) -(13).

Let us denote by $x_i = (x^s_{i,1}, x^b_{i,1}, x^h_{i,1}, \ldots, x^s_{i,J}, x^b_{i,J}, x^h_{i,J}, s^+_i, s^-_i)$ all variables associated with the i-th scenario, and define matrices

$$\tilde{A}=\begin{pmatrix} 1 & -1 & 1 & & & & & 0 & 0 \\ & & & \ddots & & & & \vdots & \vdots \\ & & & & 1 & -1 & 1 & 0 & 0 \\ -c^s_1 & c^b_1 & 0 & \cdots & -c^s_J & c^b_J & 0 & 0 & 0 \\ 0 & 0 & -c^s_1 & \cdots & 0 & 0 & -c^s_J & -1 & 1 \end{pmatrix},\ B_i=\begin{pmatrix} 0 & 0 & 1+r_{i,1} & & & & & 0 & 0 \\ & & & \ddots & & & & \vdots & \vdots \\ & & & & 0 & 0 & 1+r_{i,J} & 0 & 0 \\ 0 & 0 & 0 & \cdots & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & \cdots & 0 & 0 & 0 & 0 & 0 \end{pmatrix},$$

where $c^b_j = (1 + c_t)v_j$ and $c^s_j = (1 - c_t)v_j$. With this notation, the dynamics of the system (constraints (2), (1), (6)) are described by the equation

$$B_i x_{\pi(i)} + \tilde{A} x_i = 0,$$

where $\pi(i)$ is again the ancestor node of node i in the event tree (see Fig. 1).

Assemble the vectors $x_i, i \in L$ and y into a vector $x = (x_{\sigma(0)}, x_{\sigma(1)}, \ldots, x_{\sigma(|L|-1)}, y)$, where σ is a permutation of the nodes $0, \ldots, |L|-1$ in a reverse depth-first order. The Jacobian matrix A has the form

$$\left(\begin{array}{ccccc|ccccc|ccc|c} \tilde{A} & & & B_i & & & & & & & & & 0 & e \\ & \ddots & & \vdots & & & & & & & & & \vdots & \vdots \\ & & \tilde{A} & B_i & & & & & & & & & 0 & e \\ & & & A & & & & & & & & & B_i & \\ \hline & & & & \ddots & & & & & & & & \vdots & \\ \hline & & & & & \tilde{A} & & & B_i & & & & 0 & e \\ & & & & & & \ddots & & \vdots & & & & \vdots & \vdots \\ & & & & & & & \tilde{A} & B_i & & & & 0 & e \\ & & & & & & & & \tilde{A} & & & & B_i & \\ \hline & & & & & & & & & & & & \tilde{A} & \\ \hline d_i & \cdots & d_i & 0 & \cdots & d_i & \cdots & d_i & 0 & & & & -1 & \end{array}\right). \tag{15}$$

The last row in this matrix corresponds to constraint (3) and the entries in the last column (vectors e) link the expected value y with x_i through constraint (6):

$$\begin{aligned} d_i \in I\!R^{1\times(3J+2)} &: (d_i)_{3j} = (1 - c_t)p_i v_j, \quad j = 1, \ldots, J \\ e \in I\!R^{(3J+2)\times 1} &: e_{J+2} = 1, e_i = 0, i \neq J + 2. \end{aligned}$$

The Hessian matrix Q has a block-diagonal structure $Q = \mathrm{diag}(Q_{\sigma(0)}, Q_{\sigma(1)}, \ldots, Q_{\sigma(|L|-1)}, -1)$, where each block $Q_i \in I\!R^{(3J+2)\times(3J+2)}$ has entries

$$Q_i \ : \ \begin{cases} (Q_i)_{3J+1,3J+1} = (Q_i)_{3J+2,3J+2} = -2\rho p_i & i \in L_T \\ Q_i = 0, & i \notin L_T \end{cases}$$

Q is therefore a diagonal matrix, but this fact is not explicitly exploited by the solver (except in the form of general sparsity). Any block-diagonal (or indeed block-bordered diagonal) matrix Q is supported by the solver. Matrix A displays a complicated structure which is a superposition of the dual block-angular pattern resulting from dynamics and uncertainty and the row-border (constraint (3)) which was introduced to allow the exploitation of separability in the variance. This structure is in the set of structures supported by OOPS as outlined earlier.

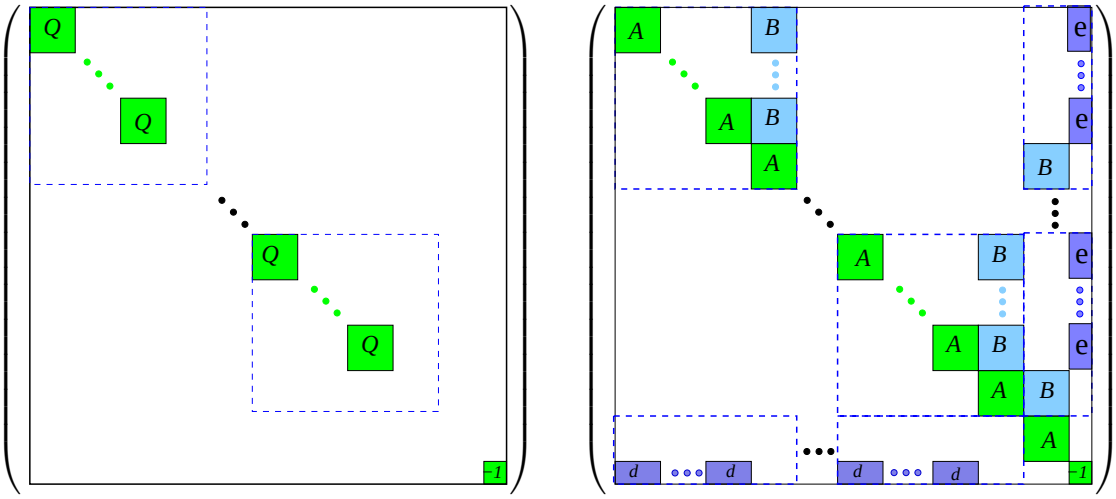


Figure 4: Sparsity patterns of Q and A.

The sparsity patterns of Hessian matrix Q and Jacobian matrix A are displayed in Figure 4.

The alternative formulations (10)-(13) lead to problems of similar structure. The constraint on the semivariance in (11) and in (12) or the constraint on the variance in (10) contribute diagonal elements to Hessian matrix and a row of similar type as (3) to Jacobian matrix. This adds an additional row (as the d-row in (15)) linking otherwise independent scenarios. Finally, the skewness-term in the objective of (13) will result in Q_i blocks containing 2x2 block entries in the positions corresponding to variables s_i^+, s_i^-.

In all cases the formulations result in a block sparse structure of the problem, whose system matrices A, Q are of nested block-bordered diagonal form. The sparsity patterns of matrices A and Q in subsequent iterations of the sequential quadratic procedure will not change although the numerical values in these matrices may change. Such nested block-structures can be exploited by modern implementations of interior point methods such as OOPS [6].

5 Numerical Results

We will now present the computational results that underpin our claim that very large nonlinear portfolio optimization problems are now within scope of modern structure exploiting implementations of general mathematical programming algorithms. To perform these tests we have used the NLP extension of our object-oriented interior point solver OOPS [6].

We have tried this solver on the three variants (11), (12) and (13) of the Asset and Liability Management problem discussed in the previous sections. All test problems are randomly generated using symmetric scenario tree with 3-4 periods and between 24-70 realizations per period (Blocks). The data for the 20-50 assets used are also generated randomly. However we draw the reader's attention to the important issue of scenario tree generation [9]. The statistics of test problems are summarized in Table 1. As can be seen problem sizes increase to just over 10 million decision variables.

Problem	Stages	Blocks	Assets	Total Nodes	constraints	variables
ALM1	3	70	40	4971	208.713	606.322
ALM2	4	24	25	14425	388.876	1.109.525
ALM3	4	40	50	65641	3.411.693	9.974.152
ALM4	4	55	20	169456	3.724.953	10.500.112

Table 1: Asset and Liability Management Problems: Problem Statistics.

Problem	1 proc		2 procs		4 procs		8 procs	
	iter	time (s)	time (s)	speed-up	time (s)	speed-up	time (s)	speed-up
ALM1	35	568	258	2.20	141	4.02	92	6.11
ALM2	30	1073	516	2.08	254	4.21	148	7.27
ALM3	41	17764	9028	1.97	4545	3.91	2390	7.43
ALM4	43	18799	9391	2.00	4778	3.93	2459	7.64

Table 2: Results for variant involving semi-variance.

The computational results of OOPS for the three variants of the ALM problem are collected in Tables 2, 3 and 4. All computations were done on the SunFire 15K machine at Edinburgh Parallel Computing Centre (EPCC). This machine features 48 UltraSparc-III processors running at 900MHz and 48GB of shared memory. Since communication between processors is made with MPI we expect these results to generalise to a more loosely linked network of processors such as PCs linked via Ethernet. All problems were solved to an optimality tolerance of 10^{-5}.

All problems can be solved in a reasonable time and with a reasonable amount of interior point iterations - the largest problem needing just over 7 hours on a single 900MHz processor. The good parallel efficiency of OOPS carries over to these nonlinear problems, achieving a speed-up of up to 7.64 on 8 processors (that is a parallel efficiency of 95.5%). While we recognize that not many institutions will be able to solve ALM problems on a dedicated parallel machine, we expect these results to carry over to a network of Ethernet linked PCs which is certainly an affordable computing platform - keeping in mind that such a parallel environment can be installed at the fraction of the cost of a parallel machine with similar characteristics.

The solution statistics obtained with the parallel implementation are reported in Tables 2, 3 and 4. For runs on one processor we report the total number of interior point iterations (Newton steps) when solving all quadratic programs in the sequential quadratic programming scheme and the CPU time. For parallel runs we report the solution times and the speed-ups. On occassions superlinear speed-ups have been recorded. We believe that this is due to avoiding the memory paging when more processors are used.

5.1 Comparison with CPLEX

We wish to make the point that a structure exploiting solver is an absolute requirement to attempt solving very large stochastic nonlinear programming problems. To demonstrate this we have compared OOPS with the barrier code of CPLEX 7.0. CPLEX on its own has only the capability to solve QPs, however it is possible to use the CPLEX library rather than OOPS to solve the individual QPs arising within the SQP method. The solution statistics for CPLEX and OOPS on the first QP that has to be solved for problem formulation (11) are reported

Problem	1 proc	2 procs		4 procs		8 procs	
	iter	time (s)	time (s)	speed-up	time (s)	speed-up	time (s)	speed-up
ALM1	25	448	214	2.09	110	4.07	72	6.22
ALM2	31	1287	618	2.08	306	4.20	179	7.19
ALM3	56	25578	13044	1.96	6650	3.85	3566	7.17
ALM4	60	24414	12480	1.96	6275	3.89	3338	7.31

Table 3: Results for logarithmic utility function.

Problem	1 proc	2 procs		4 procs		8 procs	
	iter	time (s)	time (s)	speed-up	time (s)	speed-up	time (s)	speed-up
ALM1	50	820	390	2.10	208	3.94	130	6.31
ALM2	43	1466	715	2.05	396	3.70	207	7.08
ALM3	65	28678	14393	1.99	7144	4.01	3845	7.46
ALM4	62	23664	11963	1.98	6131	3.86	3097	7.64

Table 4: Results for variant involving skewness.

in Table 6. The statistics of these problems are collected in Table 5. Unfortunately we do not posses a CPLEX license for the SunFire machine, so these results were obtained on a smaller SunEnterprise Server featuring a 400MHz UltraSparc-II CPU with 2GB of memory. Consequently problem sizes are smaller than those reported earlier. (Solvers do not exploit parallelism in these runs.) As can be seen from the statistics of these runs the QP solution times of OOPS are significantly smaller than those for CPLEX. Also the peak memory requirements of OOPS are up to about a factor of 5 smaller than those of CPLEX. Noticably the advantage of OOPS is more pronounced on problems with a large number of blocks. We believe this is due to the structure exploiting features of OOPS which are a must when attempting to solve problems of the sizes presented.

6 Conclusion

We have presented a case for solving nonlinear portfolio optimization problems by general purpose structure exploiting interior point solver. We have concentrated on three variations of the classical mean-variance formulations of an Asset and Liability Management problem each leading to a nonlinear programming problem. While these variations have been recognized for some time for their theoretical value, received wisdom is that these models are out of scope for mathematical programming methods. We have shown that in the light of recent progress in structure exploiting interior point solvers, this is no longer true. Indeed nonlinear ALM problems with several millions of variables can now be solved by a general solver in a reasonable time.

References

[1] J. R. Birge, *Decomposition and partitioning methods for multistage stochastic programming*, Operations Research, 33 (1985), pp. 989–1007.

[2] J. Blomvall and P. O. Lindberg, *A Riccati-based primal interior point solver for multistage stochastic programming*, European Journal of Operational Research, 143 (2002), pp. 452–461.

Problem	Stages	Blocks	Assets	Total Nodes	constraints	variables
QP-ALM5	4	24	12	14425	201.351	546.950
QP-ALM6	3	60	20	3661	80.483	226.862
QP-ALM7	3	80	20	6481	142.503	401.662
QP-ALM1	3	70	40	4971	208.713	606.322

Table 5: Statistics of quadratic programs used for comparison with CPLEX.

Problem	CPLEX 7.0			OOPS		
	time (s)	iter	mem (Mb)	time (s)	iter	mem (Mb)
QP-ALM5	615	28	369	406	14	325
QP-ALM6	955	16	340	159	15	136
QP-ALM7	2616	16	736	283	13	238
QP-ALM1	12236	18	1731	531	17	365

Table 6: Comparison with CPLEX 7.0.

[3] P. T. Boggs and J. W. Tolle, *Sequential quadratic programming*, Acta Numerica, (1995), pp. 1–51.

[4] A. R. Conn, N. I. M. Gould, and P. L. Toint, *Trust-Region Methods*, MPS-SIAM Series on Optimization, SIAM, Philadelphia, 2000.

[5] R. Fletcher, *Practical Methods of Optimization*, John Wiley, Chichester, second ed., 1987.

[6] J. Gondzio and A. Grothey, *Parallel interior point solver for structured quadratic programs: Application to financial planning problems*, Technical Report MS-03-001, School of Mathematics, University of Edinburgh, Edinburgh EH9 3JZ, Scotland, UK, April 2003.

[7] ——, *Reoptimization with the primal-dual interior point method*, SIAM Journal on Optimization, 13 (2003), pp. 842–864.

[8] J. Gondzio and R. Kouwenberg, *High performance computing for asset liability management*, Operations Research, 49 (2001), pp. 879–891.

[9] K. Høyland, M. Kaut, and S. W. Wallace, *A heuristic for moment-matching scenario generation*, Computational Optimization and Applications, 24 (2003), pp. 169–186.

[10] J. L. Kelly, *A new interpretation of information rate*, Bell System Technical Journal, 35 (1956), pp. 917–926.

[11] H. Konno, H. Shirakawa, and H. Yamazaki, *A mean-absolute deviation-skewness portfolio optimization model*, Annals of Operational Research, 45 (1993), pp. 205–220.

[12] H. Konno and K.-I. Suzuki, *A mean-variance-skewness portfolio optimization model*, Journal of the Operational Research Society of Japan, 38 (1995), pp. 173–187.

[13] H. M. Markowitz, *Portfolio selection*, Journal of Finance, (1952), pp. 77–91.

[14] ——, *Portfolio Selection: Efficient Diversification of Investments*, John Wiley & Sons, 1959.

[15] J. Nocedal and S. J. Wright, *Numerical Optimization*, Springer-Verlag Telos, Berlin, 1999.

[16] W. Ogryczak and A. Ruszczynski, *On consistency of stochastic dominance and mean-semideviation models*, Mathematical Programming, 89 (2001), pp. 217–232.

[17] ——, *Dual stochastic dominance and related mean-risk models*, SIAM Journal on Optimization, 13 (2002), pp. 60–78.

[18] A. Ruszczyński, *Decomposition methods in stochastic programming*, Mathematical Programming B, 79 (1997), pp. 333–353.

[19] M. Steinbach, *Hierarchical sparsity in multistage convex stochastic programs*, in Stochastic Optimization: Algorithms and Applications, S. Uryasev and P. M. Pardalos, eds., Kluwer Academic Publishers, 2000, pp. 363–388.

[20] ——, *Markowitz revisited: Mean variance models in financial portfolio analysis*, SIAM Review, 43 (2001), pp. 31–85.

[21] W. T. Ziemba and J. M. Mulvey, *Worldwide Asset and Liability Modeling*, Publications of the Newton Institute, Cambridge University Press, Cambridge, 1998.