Journal of the Operations Research
Society of Japan
Vol. 44, No. 3, September 2001

SOLVING LARGE SCALE MEAN-VARIANCE MODELS WITH DENSE NON-FACTORABLE COVARIANCE MATRICES

Naoya Kawadai Hiroshi Konno
Tokyo Institute of Technology

(Received March 3, 2000; Final April 13, 2001)

Abstract This paper is concerned with an efficient algorithm for solving a large-scale dense non-factorable quadratic programming problem arising in portfolio optimization consisting of a large number of assets.

A number of algorithms for quadratic programming problems have been proposed in the past. However, these methods tend to become less efficient as the rank of the covariance matrix increases. The algorithm proposed in this paper is a combination of projected steepest descent algorithm and projected variable metric algorithm. Subproblems to be solved in each step are simple linear programs which can be solved very fast, contrary to other quadratic programming algorithms which require the manipulation of large dense matrix.

Computational experiment shows that this algorithm outperforms renowned softwares when the number of assets is over one thousand.

1. Introduction

There exists a number of studies about efficient algorithms for solving a large scale mean-variance(MV) model which still plays a fundamental role in portfolio optimization. The first such algorithm was proposed by Perold [10] in 1984. He employed a multi-factor approach to reduce the rank of covariance matrix and applied a quadratic programming algorithm developed by Pang [9]. This approach can successfully solve a large-scale mean-variance model where n, the number of assets included in the model is up to a few thousand.

Later in 1992, one of the authors proposed a compact factorization method [6] for solving much larger MV models using historical/scenario data approach instead of multi-factor approach. The key observation is that when the number of historical/scenario data is small compared with the number of assets, one can derive an alternative sparse representation of the original quadratic programming problem. The resulting sparse problem can be solved either by interior point methods [12] or other efficient quadratic programming algorithms.

The success of this approach is due to the low rank property of the quadratic programming problem. In fact, when the rank of covariance matrix is small, we can apply Cholesky factorization to calculate a sparse representation of the covariance matrix. However, when the number of variables is over a few thousand, it is a very time consuming even if the rank of the matrix is relatively low.

The purpose of this paper is to propose an efficient and easy-to-implement algorithm for solving a large scale MV problem where the covariance matrix is dense and its rank is very high. Such problems can arise when we use a large number of data to estimate the covariance of random variables. Let us consider a scenario model using K factors and suppose each factor can attain l different values. Then the possible number of scenarios is l^K. When $K = 10$ and $l = 5$, this number is close to 10^7. Also, when we handle internationally diversified portfolio including stocks and bonds of several countries [5], the number of assets will be as large as 10^5. Therefore, compact factorization is doomed to fail

under these circumstances unless we impose stronger assumptions on the underlying model or data.

There exist a number of algorithms for solving general convex quadratic programming problems. Among the well known algorithms are the simplex type algorithm [14], active set method [2], dual approach [4] and more recent primal-dual interior point algorithm [12]. However, some of these algorithms require calculation of the inverse of the covariance matrix and the others have to handle a large dense system of equations, or treat the set of inequality constraint in a combinatorial way. Most of these methods are therefore not suitable for a large scale dense higher rank covariance matrix. In fact, when n is over one thousand, it takes hours to solve such MV problems by reputed mathematical programming softwares such as NUOPT.

Our algorithm is concerned with an algorithm for solving a large scale dense mean-variance model with properties above. We will propose a two-stage algorithm. The first step is to decompose the variance of the rate of the portfolio into two parts using a beta-relation established in CAPM. [1, 7, 11] We will calculate a good starting point to be used in the second stage by solving a linear programming problem. We then apply a projected steepest decent method and projected variable metric approach [3] to obtain an optimal solution. The amount of computation is small compared to other standard approaches because we do not have to calculate the inverse of the covariance matrix nor do we treat a dense linear system of equations.

In the next section, we will discuss a compact factorization approach which is commonly used for solving MV models of practical size and argue why it cannot be used to dense full rank problems. Section 3 will be devoted to the problem reduction procedure based upon CAPM. In Section 4, we will propose projected steepest descent and projected variable metric algorithm. Finally in Section 5, we will discuss computational results up to $n = 1000$.

2. Compact Representation of the Mean Variance Model

Let $R_i (i = 1, \cdots, n)$be the random variable representing the rate of return of i-th asset and let r_{it} be the rate of return of i-th asset under scenario $t (t = 1, \cdots, T)$. Also, let p_t be the probability of occurrence of scenario t. Then the expected rate of return r_i of i-th asset and the covariance σ_{ij} of the rate of return between i-th and jth asset are given by

$$r_i = \sum_{t=1}^{T} p_t r_{it}, \quad i = 1, \cdots, n, \tag{1}$$

$$\sigma_{ij} = \sum_{t=1}^{T} p_t (r_{it} - r_i)(r_{jt} - r_j), \quad j = 1, \cdots, n. \tag{2}$$

Let $\boldsymbol{x} = (x_1, \cdots, x_n)$ be a portfolio weight vector, whose element is the ratio of wealth invested in i-th asset to the entire wealth and let

$$R(\boldsymbol{x}) = \sum_{j=1}^{n} R_j x_j, \tag{3}$$

be the rate of the return of the portfolio $\boldsymbol{x}$. The mean-variance(MV)model is represented as follows

$$\left|\begin{array}{ll} \text{minimize} & f(\boldsymbol{x}) = \sum_{i=1}^{n}\sum_{j=1}^{n}\sigma_{ij}x_ix_j \\ \text{subject to} & \sum_{j=1}^{n} r_jx_j = \rho, \\ & \boldsymbol{x} \in X, \end{array}\right. \tag{4}$$

where $\rho \in [min_j r_j, max_j r_j]$ is the target expected rate of return and X is the investable set. The simplest such set is

$$X_0 = \{\boldsymbol{x} = (x_1, \cdots, x_n) | \sum_{j=1}^{n} x_j = 1, \quad 0 \leq x_j \leq u_j, j = 1, \cdots, n\}, \tag{5}$$

where u_j is the maximal purchasable units of jth asset.

Problem (4) is a convex quadratic programming with a dense covariance matrix since the rate of return of assets are highly correlated, so that $\sigma_{ij} \neq 0$ for almost all i, j. Therefore, it is very difficult to solve the problem (4) by standard QP algorithm. However, when T is much smaller than n, we can apply the following compact factorization approach [6]. By noting (2), we have

$$\begin{aligned} f(\boldsymbol{x}) &= \sum_{i=1}^{n}\sum_{j=1}^{n}\sigma_{ij}x_ix_j \\ &= \sum_{t=1}^{T}\sum_{i=1}^{n}\sum_{j=1}^{n} p_t(r_{it} - r_i)(r_{jt} - r_j)x_ix_j \\ &= \sum_{t=1}^{T} p_t \sum_{i=1}^{n}\sum_{j=1}^{n}(r_{it} - r_i)(r_{jt} - r_j)x_ix_j \\ &= \sum_{t=1}^{T} p_t\{\sum_{j=1}^{n}(r_{jt} - r_j)x_j\}^2. \end{aligned} \tag{6}$$

Let us introduce auxiliary variables

$$z_t = \sum_{j=1}^{n}(r_{jt} - r_j)x_j, \quad t = 1, \cdots, T. \tag{7}$$

Then the problem (4) can be represented as follows:

$$\left|\begin{array}{ll} \text{minimize} & \sum_{t=1}^{T} p_t z_t^2 \\ \text{subject to} & z_t = \sum_{j=1}^{n}(r_{jt} - r_j)x_j, \quad t = 1, \cdots, T, \\ & \sum_{j=1}^{n} r_jx_j = \rho, \\ & \boldsymbol{x} \in X. \end{array}\right. \tag{8}$$

This is a quadratic programming problem with only T separable convex terms.

It has been demonstrated in [12] that this problem can be solved very fast by interior point algorithms when n is over ten thousand if the number of historical/scenario data T is less than one hundred. However, it tends to become less efficient as T increases and it loses its advantage if T is larger than n.

3. Decomposition of the MV Model Using Beta-Relation

Among the well known results of CAPM(Capital Asset Pricing Model) is the so-called beta-relation between the rate of return of individual assets and the rate of return of the market portfolio.

Theorem 3.1 *Let R_j and R_M be, respectively, the rate of return of i-th asset and the market portfolio. Then the following relation holds*

$$R_j - r_0 = \beta_j(R_M - r_0) + \varepsilon_j, \quad j = 1, \cdots, n, \tag{9}$$

where ε_j is a random variable with $E[\varepsilon_j] = 0$ and r_0 is the rate of return of riskless asset. ∎

For the derivation of this relation, readers are referred to standard texts of financial economics [1]. This result was first proved by Sharpe [11] under several assumptions on the capital market and individual investors and it has been demonstrated that this relation holds in practice as a first approximation. Also, it has been proved in [7] that some of these assumptions can be relaxed.

Let $\hat{\beta}_j$ be the value of $\beta_j (j = 1, \cdots, n)$ estimated by standard least square method using historical data. Then

$$\sigma_{ij} = cov[R_i, R_j] = \hat{\beta}_i \hat{\beta}_j \sigma_M^2 + \sigma'_{ij}, \quad i, j = 1, \cdots, n, \tag{10}$$

where $\sigma'_{ij} = cov[\varepsilon_i, \varepsilon_j]$ under the common assumption $cov[R_M, \varepsilon_j] = 0, \forall j$. Therefore,

$$\begin{aligned} \sum_{i=1}^{n}\sum_{j=1}^{n} \sigma_{ij} x_i x_j &= \sum_{i=1}^{n}\sum_{j=1}^{n} \hat{\beta}_i \hat{\beta}_j \sigma_M^2 x_i x_j + \sum_{i=1}^{n}\sum_{j=1}^{n} \sigma'_{ij} x_i x_j \\ &= \sigma_M^2 (\sum_{j=1}^{n} \hat{\beta}_j x_j)^2 + \sum_{i=1}^{n}\sum_{j=1}^{n} \sigma'_{ij} x_i x_j. \end{aligned} \tag{11}$$

Since σ'_{ij} are usually much smaller than σ_M^2, the first term is expected to dominate the second term. This observation leads us to solve the following problem:

$$\left| \begin{array}{ll} \text{minimize} & \sigma_M^2 (\sum_{j=1}^{n} \hat{\beta}_j x_j)^2 \\ \text{subject to} & \sum_{j=1}^{n} r_j x_j = \rho, \\ & \boldsymbol{x} \in X, \end{array} \right. \tag{12}$$

whose optimal solution $\boldsymbol{x}^0$ is expected to give a good approximation of the optimal solution of the original problem (4). Let us note that the problem (12) is equivalent to the following linear programming problem because $\sum_{j=1}^{n} \hat{\beta}_j x_j$ is positive since $\hat{\beta}_j$'s are in the interval [0.5, 2.0] for almost all j's with possibly one or two exceptions. In particular, if we use 3 to 5 years historical data, as we did in our experiments, all $\hat{\beta}_j{}'s$ turn out to be nonnegative.

$$\left| \begin{array}{ll} \text{minimize} & \sum_{j=1}^{n} \hat{\beta}_j x_j \\ \text{subject to} & \sum_{j=1}^{n} r_j x_j = \rho, \\ & \boldsymbol{x} \in X. \end{array} \right. \tag{13}$$

4. Descent Procedure

Given a feasible solution $\boldsymbol{x}^k$,we will calculate the descent direction $\boldsymbol{d}^k$ by solving the following linear programming problem.

$$\left|\begin{array}{ll} \underset{\boldsymbol{d}^k}{\text{minimize}} & \nabla f(\boldsymbol{x}^k)\boldsymbol{d}^k \\ \text{subject to} & \boldsymbol{x}^k + \boldsymbol{d}^k \in X \\ & \sum_{j=1}^n r_j(x_j^k + d_j^k) = \rho, \end{array}\right. \tag{14}$$

and define $\boldsymbol{x}^{k+1}$ as follows:

$$\boldsymbol{x}^{k+1} \equiv \boldsymbol{x}^k + \alpha_k \boldsymbol{d}^k, \quad where \quad \alpha_k = argmin\{f(\boldsymbol{x}^k + \alpha \boldsymbol{d}^k) | 0 \leq \alpha \leq 1)\}. \tag{15}$$

Let us note that α_k can be calculated analytically since f is quadratic.

Algorithm PSD (Projected Steepest Decent Algorithm)

Step 0. Solve a linear program (13) to obtain starting solution $\boldsymbol{x}^0, k := 1$.

Step 1. Calculate $\boldsymbol{d}^k$ by solving a linear program (14).

Step 2. $\boldsymbol{x}^{k+1} = \boldsymbol{x}^k + \alpha_k \boldsymbol{d}^k$ where $\alpha_k = argmin_\alpha\{f(\boldsymbol{x}^k + \alpha \boldsymbol{d}^k) | 0 \leq \alpha \leq 1)\}$.

Step 3. If $|\boldsymbol{x}^{k+1} - \boldsymbol{x}^k| \leq \varepsilon$,then stop. Otherwise $k = k + 1$ and go to Step 1. ∎

This algorithm may not converge to an optimal solution. To guarantee convergence, we may adopt a scheme proposed by Topkis-Veinott [13]. However, the speed of convergence of this modified algorithm may not be satisfactory since it depends upon the first order approximation of the objective function.

To obtain a fast and convergent algorithm, we will switch to the projected variable metric method of Goldfarb [3] after certain number of iterations of the PSD Algorithm.

Algorithm PVM (Projected Variable Metric Algorithm)

Step 0. $H_0 = P_q$ where P_q is the projection matrix to the sets of active constraints at $\boldsymbol{x}^0$.

Step 1. $\boldsymbol{d}^k = -H_k \nabla^t f(\boldsymbol{x}^k)$.

Step 2. If $\boldsymbol{d}^k = \boldsymbol{0}$,then go to Step 4. Otherwise
$\boldsymbol{x}^{k+1} = \boldsymbol{x}^k + \alpha_k \boldsymbol{d}^k$ where α^k is defined by (15).

Step 3. If the active set at $\boldsymbol{x}^{k+1}$ is same as that at $\boldsymbol{x}^k$, then update H_k by using BFGS formula [3] and go to Step 1 by setting $k = k + 1$. If the active set changes, then update

$$H_{k+1} = H_k - \frac{H_k(\boldsymbol{a}^r)^t \boldsymbol{a}^r H_k}{\boldsymbol{a}^r H_k (\boldsymbol{a}^r)^t} \tag{16}$$

where $\boldsymbol{a}^r$ is the gradient of the new active constraint. Let P_q be the new projection matrix and go to Step 1.

Step 4. Let

$$\boldsymbol{\lambda}_q = -\nabla f(\boldsymbol{x}^k) A_q^t (A_q A_q^t)^{-1} \tag{17}$$

where the matrix A_q is the matrix of active constraints at $\boldsymbol{x}^k$. If $\boldsymbol{\lambda}_q \geq \boldsymbol{0}$ then stop(Karush-Kuhn-Tucker condition is satisfied). Otherwise choose the largest negative component of $\boldsymbol{\lambda}_q$ and remove the corresponding vector, $\boldsymbol{a}^s$ from the matrix A_q. Update

$$H_{k+1} = H_k + \frac{P_k(\boldsymbol{a}^s)^t \boldsymbol{a}^s P_k}{\boldsymbol{a}^s P_k (\boldsymbol{a}^s)^t} \tag{18}$$

$q = q - 1, k = k + 1$ and go to Step 1. ∎

If $\boldsymbol{x}^0$, the starting solution, is close to $\boldsymbol{x}^*$, then the set of active constraints is expected to remain stable. Also the direction vector of this algorithm uses the second order approximation of the objective function. Therefore, it is expected to be efficient if the solution of the Algorithm PSD is close to $\boldsymbol{x}^*$.

Remark: We may employ anti-jamming procedure such as Zoutendijk [15] to avoid jamming.

Let us summarize our algorithm for solving the problem (4) in the following steps.

Algorithm PSDVM (Projected Steepest Descent/Variable Metric Algorithm)

Step 0. Calculate $\hat{\beta}_j, j = 1, \cdots, n$.

Step 1. Solve a linear program (13) and let $\boldsymbol{x}^0$ be its optimal solution.

Step 2. Execute PSD Algorithm. Let $\hat{\boldsymbol{x}}$ be the resulting solution.

Step 3. Execute PVM Algorithm using $\hat{\boldsymbol{x}}$ as a starting solution $\boldsymbol{x}^0$. ∎

5. Numerical Experiments

We conducted a series of numerical experiments of the Algorithm PSDVM using: Genuine Intel Pentium(III) processor 500MHz, RAM 383MB with Microsoft Windows 98.We used optimization package software NUOPT3.1[1] to solve linear programming problems.Also, we used this software as a benchmark to compare the performance.

We solved the following simple MV model:

$$\left|\begin{array}{ll} \text{minimize} & \displaystyle\sum_{i=1}^{n}\sum_{j=1}^{n}\sigma_{ij}x_i x_j \\ \text{subject to} & \displaystyle\sum_{j=1}^{n} r_j x_j = \rho \\ & \displaystyle\sum_{j=1}^{n} x_j = 1 \\ & 0 \le x_j \le u_j, j = 1, \cdots, n. \end{array}\right. \tag{19}$$

where σ_{ij}'s and r_j's are calculated by using 1000 daily data in the Tokyo Stock Exchange from January, 1993 to December, 1998 and we choose u_j's from 0.05 to 0.2. Also, we used TOPIX as the market portfolio.

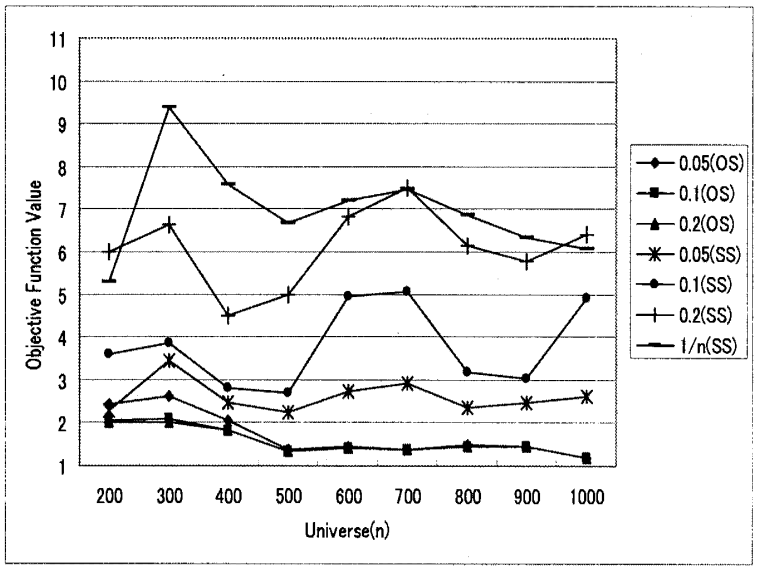
As expected, computation time for solving the problem (13) is negligibly small. In fact, it takes only 0.05 seconds even when $n = 1000$ excluding the computation time to estimate β_j's. (Calculation of β_j's for 1000 assets requires less than one second.) The quality of the starting solution of PSD(projected steepest descent) algorithm is reasonably close to the optimal solution, but not as good as we expected although $\sigma_{ij}'s$ are much smaller than ${\sigma_M}^2$.(See Figure 1 below) The quality of the starting solution calculated via (13) tends to improve as we decrease u_j's. In fact, the solution is not much different from the uniform portfolio, i.e., $x_j = 1/n, j = 1, \cdots, n$ when $u_j = 0.2$ for all j. But it is much better than uniform portfolio when $u_j = 0.05$ for all j. Also the quality of the initial solution via (13)

[1]Let us add that the new version of NUOPT4.0, which were not available at the time of our experiment, can solve large non-factorable mean-variance problems very fast using "asqp" algorithm.

tends to improve when we impose more restrictive constraints on the problem.

Also, Figure 2 shows the value of the objective function as a function of the iteration of PSD algorithm. We see from this that convergence becomes slower after 10 to 15 iterations. Let us note that the computation time for 20 iterations of PSD algorithm is less than 24 seconds even when $n = 1000$.

Figure 3 shows the computation time for calculating an optimal solution by applying PVM(projected variable metric) algorithm. The optimality condition $\boldsymbol{\lambda}_q \geq \mathbf{0}$ was satisfied after several iterations. In particular, we reached an optimal solution after 11 iterations with 58.7 CPU seconds when $n = 1000$.



OS : Optimal Solution; SS : Starting Solution

Figure 1 Quality of Starting Solution

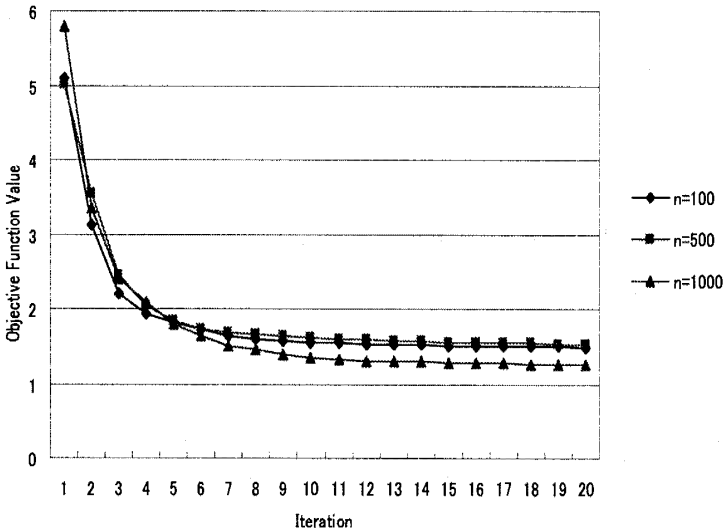


Figure 2 Result of PSD Algorithm

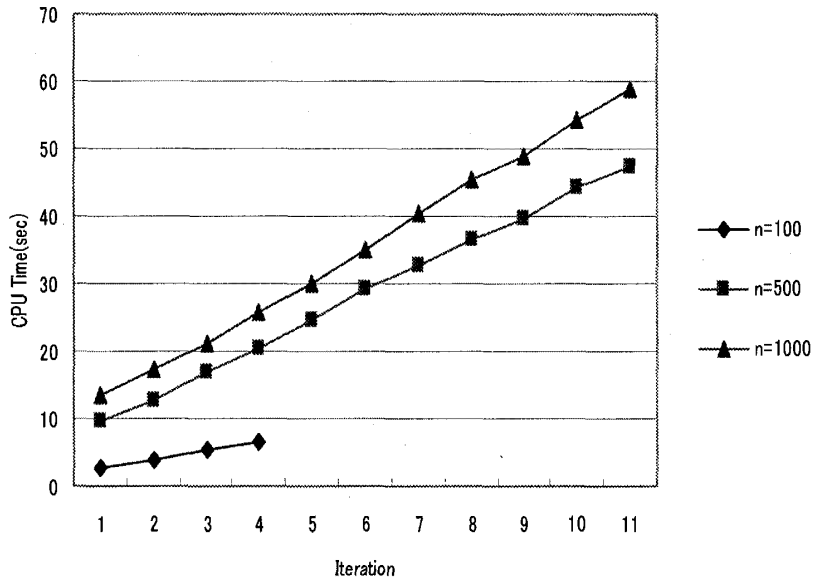


Figure 3 Result of PVM Algorithm

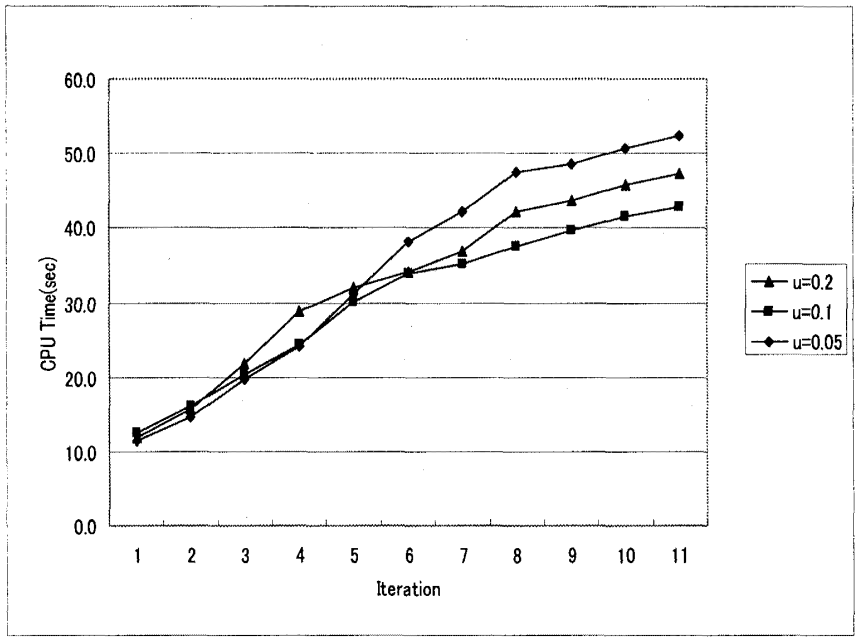


Figure 4 Result of PVM Algorithm

Figure 4 shows the computational results of PVM algorithm for various values of u_j's. Computation time is relatively insensitive to the magnitude of upper bounds.

Figure 5 shows the total computation time of our algorithm and that of NUOPT. We see from this that our algorithm outperforms NUOPT when $n \geq 400$. Also, our algorithm is more than 20 times faster than NUOPT when n is over 1000. It appears that the computation time of our algorithm is more or less linear, whereas that of NUOPT is exponential as a function of n. The amount of computation depends upon the number of linear constraints, in addition to the number of variables. However, the number of linear constraints remains small in most practical applications, so that the efficiency of our algorithm should not deteriorate too much.

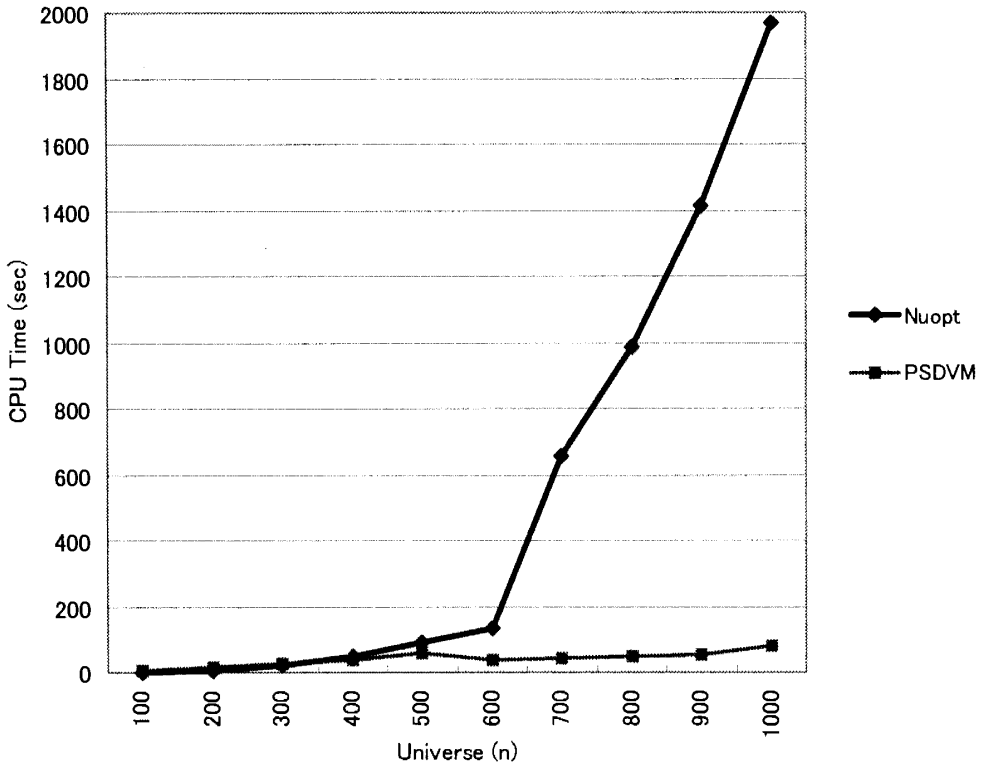


Figure 5 Comparison of PSVM and NUOPT

6. Conclusions and Future Directions of Research

We showed in this paper that the projected steepest descent/variable metric algorithm can solve a large dense non-factorable MV model very fast. The starting solution obtained by solving (13) was not very good for the tested problems contrary to our expectation. In fact, we could reach such a solution after a few iterations of the projected steepest descent procedure regardless of its starting point. However, this procedure has prone to be more effective for problem with more complicated constraint set.

One may argue that the use of the PVM procedure from the beginning would lead to a better result. However, as discussed earlier, active sets will frequently change at the earlier stage of computation, so that the use of "cheap" PSD is useful.

In this experiment, we compared our algorithm with the package software NUOPT, which is referred to as one of the most efficient softwares for solving large scale MV models and is widely used in many financial companies of Japan. Therefore, we believe it is not inadequate to use it as a benchmark for the comparison.

Large scale dense non-factorable quadratic programming problems arise when we apply sequential quadratic programming algorithm to nonlinear programming problem with many variables. Our algorithm may be used as a subprocedure for solving such class of problems.

Acknowledgements. The research of the first author was supported in part by Grant-in-Aid for Scientific Research of the Ministry of Education, Science and Culture B(2)104 50041. Also, he acknowledge the generous support of the IBJ-DL Financial Technologies, Co. and The Tokyo Trust and Banking, Co., Ltd..

References

[1] H. J. Elton and N. J. Gruber : *Modern Portfolio Theory and Investment Analysis, 4th Edition* (John Wiley & Sons, New York, 1991).

[2] P. E. Gill and W. Murray : Numerically stable method for quadratic programming. *Mathematical Programming,* **14**(1978)349-372.

[3] D. Goldfarb : Extension of davidon's variable metric method to maximization under linear constraints. *SIAM J. of Applied Mathematics,* **17**(1969)739-764.

[4] D. Goldfarb and I. Idnani : A numerically stable dual method for solving strictly convex quadratic programs. *Mathematical Programming,* **27**(1983)1-33.

[5] H. Konno and J. Li : Internationally diversified investment using an integrated portfolio model. *International J. of Theoretical and Applied Finance,* **1**(1998)145-160.

[6] H. Konno and K. Suzuki : A fast algorithm for solving large scale mean-variance models by compact factorization of covariance matrices. *J. of the Operations Research Society of Japan,* **35**(1992)93-104.

[7] H. Konno and K. Suzuki : Equilibria in the capital market with nonhomogeneous investors. *Japan J. of Industrial and Applied Mathematics,* **13**(1996)369-383.

[8] H. M. Markowitz : *Portfolio Selection : Efficient Diversification of Investments 2nd edition* (John Wiley & Sons, New York, 1990).

[9] J.-S. Pang : A new efficient algorithm for a class of portfolio selection problems. *Operations Research,* **28**(1980)754-767.

[10] A. Perold : Large scale portfolio optimization. *Management Science,* **30**(1984) 1143-1160.

[11] W. F. Sharpe : Capital asset process : a theory of market equilibrium under condition of risk. *J. of Finance,* **19**(1964)425-442.

[12] H. Takehara : An interior point algorithm for large scale portfolio optimization. *Annals of Operations Research,* **45**(1993)373-386.

[13] D. M. Topkis and A. F., Jr. Veinott : On the convergence of some feasible directions algorithms for nonlinear programming. *SIAM J. on Control,* **5**(1967)268-279.

[14] P. Wolfe : Simplex method for quadratic programming. *Econometrica,* **27**(1959)382-398.

[15] G. Zoutendijk : *Methods of Feasible Directions*(Elsevier, 1960).

Naoya Kawadai
Department of Industrial Engineering and Management
Graduate school of Decision and Technology
Tokyo Institute of Technology
2-12-1 Ookayama, Meguro-ku, Tokyo 152-8552 JAPAN
E-mail: nkawadai@me.titech.ac.jp