

June 1992

Computationally efficient portfolio analysis

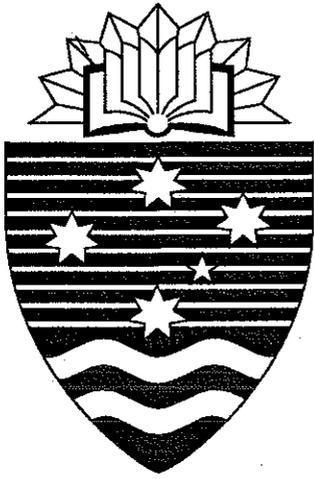
R. P. Byron

Follow this and additional works at: http://epublications.bond.edu.au/discussion_papers

Recommended Citation

Byron, R. P., "Computationally efficient portfolio analysis" (1992). *School of Business Discussion Papers*. Paper 31.
http://epublications.bond.edu.au/discussion_papers/31

This Discussion Paper is brought to you by the Bond Business School at [ePublications@bond](http://epublications.bond.edu.au). It has been accepted for inclusion in School of Business Discussion Papers by an authorized administrator of [ePublications@bond](http://epublications.bond.edu.au). For more information, please contact [Bond University's Repository Coordinator](mailto:repository@bond.edu.au).



BOND UNIVERSITY
School of Business

**DISCUSSION
PAPERS**

"Computationally Efficient Portfolio Analysis"

by

R. P. Byron

DISCUSSION PAPER NO 31

JUNE 1992

University Drive,

Gold Coast, QLD, 4229

AUSTRALIA

Computationally Efficient Portfolio Analysis

R.P.Byron
School of Business
Bond University
Queensland

Summary

A procedure is developed for handling large scale portfolio optimisation problems by combining the conjugate gradient and gradient projection methods, whilst allowing the conjugate gradient solution to venture into the infeasible region before it is wound back to the nearest face using gradient projection. The algorithm is extremely fast and can solve 400 security problems in as few as 5-6 iterations.

1. Introduction

The problem of minimising portfolio risk subject to various side conditions is not a difficult one. However, if the problem involves large numbers of securities (say 1000), issues of speed and numerical accuracy arise.

Let w_i be the share of the portfolio devoted to the i th of n securities; let e_i be the expected returns on the i th security and let A be an $n \times n$ covariance matrix of the securities' returns. The risk (variance) of the portfolio is $w'Aw$. Typically there are side conditions, which can be expressed as $Rw=s$, where R is a $k \times n$ matrix of constraints and s is a $k \times 1$ vector of constants.

For example, the portfolio manager may wish to ensure a certain level

of return (e_p) $\sum_{i=1}^n w_i e_i = e_p$. In addition, there may be the condition that the

shares sum to one $\sum_{i=1}^n w_i = 1$ and w_i be non-negative and feasible; i.e.

$0 \leq w_i \leq 1$. There could be additional restrictions of the form $\sum_{i \in I} w_i = w_0$,

implying a tilt; i.e., that a certain proportion of the portfolio be devoted to industrials. for example. The tilt constraint could take the form of an

inequality $\sum_{i \in I} w_i \geq w_0$, or a restriction on returns (or risk) from a sub-category of the portfolio.

What emerges is a quadratic programming problem of the following form: $\text{Min } w'Aw$ subject to $R_1w = s_1$, $R_2w \geq s_2$, $R_3w \leq s_3$ and $0 \leq w \leq 1$. In the absence of inequality restrictions, the Lagrangian representation is

$$(1) \quad \text{Min } \Phi = w'Aw + 1/2 \lambda(Rw - s)$$

with the first order conditions

$$(2) \quad \begin{bmatrix} A & R \\ R' & O \end{bmatrix} \begin{bmatrix} w^* \\ \lambda^* \end{bmatrix} = \begin{bmatrix} 0 \\ s \end{bmatrix}.$$

$$(3) \quad \lambda^* = -[R'A^{-1}R]^{-1}s$$

$$(4) \quad w^* = A^{-1}R[R'A^{-1}R]^{-1}s.$$

If a unique solution is to exist A must be of full rank. Given the nature of the covariance matrix, this solution, for even the simplest case, may be unsatisfactory if the estimated covariance matrix is estimated from an undersized sample. However, one approach to overcome this problem involves the use of continuity corrections, {see Theil and Fiebig, (1984)}.

Conventional quadratic programming packages are not designed to handle extreme high dimensionality problems of this sort efficiently. Rudd and Rosenberg [1979] mention an algorithm due to Von Hohenbalken [1975], which, in fact, is just a simplex linearisation method. For large problems such linearisation methods can be extremely slow. It has been pointed out [Mulvey, (1987)] that the problem has characteristics of network analysis and Mulvey provides a reduced gradient solution. Markowitz [1987] also offers an algorithm which is fast but, as it requires the bordered Hessian inverse in (2) is expensive in storage and potentially numerically unstable.

However, there is one strand in the recent optimisation literature, which point to methods of handling large scale quadratic programming problems of the type encountered in the Finance literature very easily. In particular, the reader's attention is drawn to the papers of More' and Toraldo [1989, 1991], and the preceding works of O'Leary [1980] and Calamai and More' [1987].

In the first section of the paper the modern literature on quadratic programming, as it is applied in Finance, is explored. The essential characteristics of the quadratic programming problem are noted and a simple, fast and numerically stable solution based on the conjugate gradient algorithm is proposed. This approach is explored in the final section.

2. Quadratic Programming Solutions

The general quadratic programming problem is to find w^* to minimise

$$(5) \quad \Phi = 1/2 w'Aw + a'w \text{ subject to } R_1 w = s_1 \text{ and } R_2 w \geq s_2$$

The equalities may be eliminated by transformation (or reduction). For example, if R_1 is $k \times n$ and $\rho(R_1) = k$; then w and R can be rearranged so

$$(6) \quad R_{11}w_1 + R_{12}w_2 = s_1$$

and w_1 can be substituted from Φ using

$$(7) \quad w_1 = R^{11}[s_1 - R_{12}w_2],$$

where $R^{11} = R_{11}^{-1}$. Partitioning $A = \begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix}$ and substituting out the equalities

$$(8) \quad \begin{aligned} \Phi = 1/2 w_2' [A_{22} - A_{21}R^{11}R_{12} - R_{12}'R^{11}A_{12}' + R_{12}'R^{11}A_{11}R^{11}R_{12}] w_2 \\ + [s_1'R^{11}A_{12} - s_1'R^{11}A_{11}R^{11}R_{12}] w_2 + 1/2 [A_{22} + a_1'R^{11}R_{12}] w_2 \\ + [a_1'R^{11}s_1 + 1/2 s_1'R^{11}A_{11}R^{11}s_1] \end{aligned}$$

$$(9) \quad \Phi = 1/2 w_2' B w_2 + b'w_2 + c$$

Subsequent restrictions of the form $w_2 \geq 0$ are straightforward; if the inequalities are violated then the constraint $w_{2i} = 0$ becomes active and that particular term can be eliminated from the objective function. Likewise the inequalities $w_1 \geq 0$ become $R^{11}[s_1 - R_{12}w_2] \geq 0$; and if violated

can be introduced as equality restrictions $R^{11}[s_1 - R_{12}w_2] = 0$. The cost of these restrictions, and the indicator of whether the restriction should subsequently be relaxed is provided (using the Kuhn-Tucker theorem) by the Lagrangian or the gradient associated with the restriction.

When inequality constraints are violated they switch to active equality constraints, which then means the bordered Hessian solution (2) is applicable.

$$(10) \quad \begin{bmatrix} A & R \\ R' & O \end{bmatrix} \begin{bmatrix} w^* \\ \lambda^* \end{bmatrix} = \begin{bmatrix} 0 \\ s \end{bmatrix}$$

As the inequalities slide in and out of the solution the composition of R changes, and the inverse can be updated. However; the basic problem with the inversion of the bordered Hessian and its updates, is numerical stability. Efficient code can be designed which, once the initial inverse is obtained, enables updates to be implemented quickly.

Markowitz [1987] uses the partitioned inverse above but separates out the expected returns from the remaining equality restrictions. The partitioning is thus

$$(11) \quad \begin{bmatrix} A & R & \mu \\ R' & O & O \\ \mu' & O & O \end{bmatrix} \begin{bmatrix} w^* \\ \lambda^* \\ \lambda_e \end{bmatrix} = \begin{bmatrix} 0 \\ s \\ e \end{bmatrix}$$

using notation which corresponds partly to his. R and μ correspond to the restriction matrix R_1 previously. μ are the expected returns on each of the securities, e is the expected return on the overall portfolio. The solution can be written

$$(12) \quad \begin{bmatrix} w^* \\ \lambda^* \\ \lambda_e \end{bmatrix} = M^{-1} \begin{bmatrix} 0 \\ s \\ e \end{bmatrix} = \alpha + \beta e = M^{-1} \begin{bmatrix} 0 \\ s \\ 0 \end{bmatrix} + M^{-1} \begin{bmatrix} 0 \\ 0 \\ e \end{bmatrix}$$

$$\text{where } \alpha = \begin{bmatrix} M^{12}s \\ M^{22}s \\ M^{32}s \end{bmatrix} \text{ and } \beta = \begin{bmatrix} M^{13}e \\ M^{23}e \\ M^{33}e \end{bmatrix}$$

and the superscripts indicate the appropriate inverse term in M . Since the portfolio variance is $\Phi = w'Aw$ and w relates linearly to e , through β , the relationship between Φ and e is clearly quadratic. A change in the optimal solution will result from a change in M^{12} or M^{13} resulting from the addition or removal of a restriction (e.g. an inequality becoming active or being removed from the active set). In terms of the partitioned inverse, the A matrix is being augmented or reduced - but nothing else changes. The Markowitz critical line algorithm updates M^{-1} without full reinversion. The Lagrangians indicate which restrictions are to be relaxed using the Kuhn-Tucker theorem. The Markowitz code involves array sizes are extremely large ($3n \times 3n$, for example); however, once an initial solution is found the updates can be implemented speedily.

There is no discussion in Markowitz [1987] on numerical stability issues and there are no speed comparisons with other algorithms. On numerical analysis grounds, {Fletcher, [1987, p.237]} this approach can be criticized because of error buildup and the numerical instability of the inverse. In fact, users of this algorithm do complain of rounding error problems.

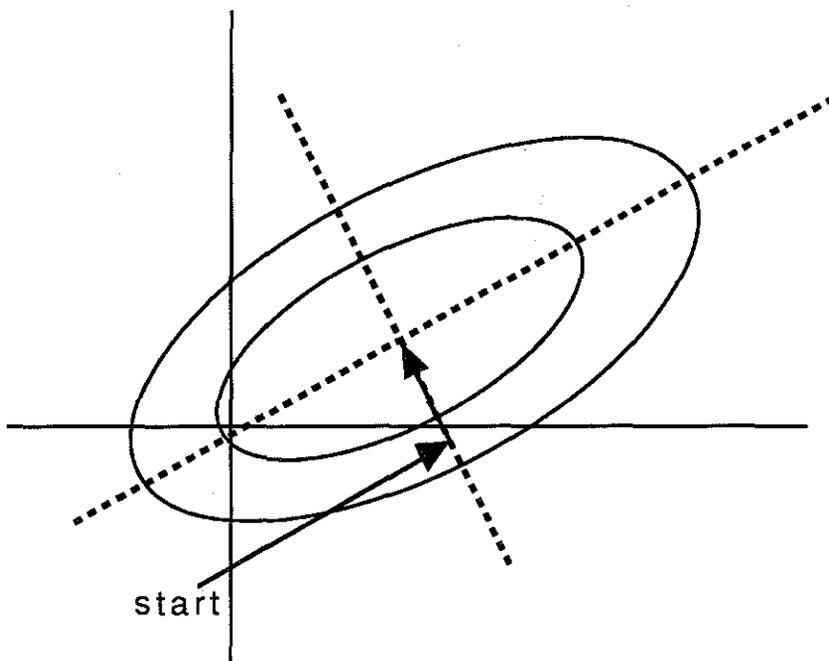
The nature of the problem appears to lend itself to solution by nonlinear network analysis. The objective function $\Phi = w'Aw$ is a quadratic form with A as the matrix of weights or penalties subject to some adding up and non-negativity conditions. Network solutions exist in the literature {see Mulvey [1987], for example} and their proponents claim considerable success in the handling of large portfolio problems.

Mulvey presents the problem cosmetically as one of network analysis, but then uses the reduced gradient method ("Truncated Newton") to achieve optimisation. It solves the constrained (Newton) equations from the quadratic form using conjugate gradients. His algorithm appears to be considerably faster than MINOS, a general nonlinear optimisation routine, its advantage is enhanced as the problem size increases. The puzzle is that Mulvey's approach is conventional; feasibility is maintained, and this suggests the introduction of equality restrictions one at a time, as the inequalities are violated. Convergence should be slow and the number of iterations large, compared to the proposals here.

3. The Conjugate Gradient Algorithm

The classical portfolio optimisation problem is a very simple example of quadratic programming. Equality (and active inequality) constraints, when eliminated, leave the objective function as a quadratic form. Probably, the best approach to minimising large *unconstrained* quadratic forms is the conjugate gradient algorithm. Examples are cited in the literature of conjugate gradient problems involving 3000-4000 variables being solved in 50 or so iterations. {see Fletcher [1987, p.85] or Reid [1971] for examples}. The trick then, is to convert the constrained problem to an unconstrained one, to exploit the advantages of the conjugate gradient method.

Figure 1



The ellipsoid in Figure 1, could be a conic section from a quadratic function with the principal axes inserted (corresponding to the eigenvectors). One version of the conjugate gradient algorithm, illustrated in Figure 1, would follow the principal axes to the optimum point in each direction. In the two dimension case it would take two search directions and six function evaluations (since the function is quadratic three function evaluations are needed in each direction to achieve the minimum). In theory, with a 100 dimension quadratic function the minimum would be achieved in

100 searches. In practice, it will involve far fewer iterations, as the algorithm will achieve large gains initially. Scaling can also be used, transforming the contours to be more spherical and this will reduce the number of iterations even further. In subsequent applications of the conjugate gradient algorithm with 100-500 securities, we typically achieved convergence for simple equality constraints in 10 or less iterations.

Several versions of the conjugate gradient method exist; the one used here is to take the first direction as the gradient with subsequent directions as

$$(13) \quad d_i = d_{i-1} + \phi g_i$$

where d_i and g_i are the i th direction and gradient and ϕ is a scalar calculated to ensure conjugacy. Conjugacy is defined as the property where $d_i' B d_j = 0$, where B is the Hessian and ϕ emerges as $\frac{g_i' g_i}{g_{i-1}' g_{i-1}}$. It is easily proved {see Byron [1977], for example} that searching these directions for a minimum leads to convergence to the global optimum in n iterations, where n is the dimension of the Hessian.

The fact that the conjugate gradient method works so well on quadratic functions gives an over-powering advantage to the conjugate gradient technique and suggests that a good approach to the problem is to redefine the objective function to unconstrained form. The equality restrictions may be eliminated by substitution and the objective function redefined as (9) i.e. as $\Phi = 1/2 w_2' B w_2 + b' w_2 + c$.

Unconstrained minimisation of Φ can lead to an infeasible solution (eg. $w_{2i} \leq 0$) and two approaches are possible. A large step approach is to allow the optimum to be attained and then constrain any negative w_2 to zero; and reapply the conjugate gradient algorithm from the previous solution with the newly active constraints. This step is a gradient projection onto the nearest constraint face and corresponds to the approach of Toraldo and More [1991]; it ensures large scale swapping of restrictions. The alternative is to follow the conjugate gradients until a bound is approach and then as the inequality becomes active introduce a new equality restriction at that point. Restrictions are thus introduced one at a time. This corresponds to Mulvey's [1987] approach.

If the restrictions are simple bounds ($w_2 \geq 0$) there is no need to redefine the objective function - but in searches those $w_2=0$ terms are not updated. After convergence the constrained zero coefficients need to be examined to see if the objective function is decreased when they become positive. This information is contained in the gradient - in fact, it is just the Lagrange multiplier from Kuhn-Tucker theory. Thus, on convergence, not only are new equalities introduced if coefficients have become infeasible, but active constraints may be relaxed. The iterations proceed until feasibility and convergence are achieved. In practice it was found that large numbers of restrictions may be deactivated and activated in the first couple of iterations of the algorithm on large problems.

Inequality constraints of the form $R_2 w \geq s_2$ or $R_3 w \leq s_3$ are also tested for feasibility at each step and added to the equality restriction set if violated. This means redefining B and b in (9), but this is computationally inexpensive. These restrictions must be checked for potential relaxation at subsequent iterations but this is easy using the Lagrangian $\lambda = \frac{\partial \Phi}{\partial s}$ which can be expressed as $w_1 = R_{11}^{-1}[s - R_{12}w_2]$, so $dw_1 = R_{11}^{-1} ds$. The effect of a feasible relaxation in s , is to induce a change in w_1 which a consequent change in Φ .

Swapping such general inequality restrictions in and out is easy if the equality restrictions are written

$$(14) \quad R_1 w_1 + R_2 w_2 = s$$

$$(15) \quad w_1 = R_1^{-1} s - R_1^{-1} R_2 w_2 = g + G w_2$$

Then

$$(16) \quad \begin{bmatrix} w_1 \\ w_2 \end{bmatrix} = \begin{bmatrix} G \\ I \end{bmatrix} w_2 + \begin{bmatrix} g \\ 0 \end{bmatrix}.$$

The quadratic form becomes

$$(17) \quad w_2' G' A G w_2 + 2 w_2' G' A g + g' A g = w_2' B w_2 + 2 w_2' b + c.$$

and the gradient is

$$(18) \quad g_i = 2 B w_{2i} + 2 b.$$

The Lagrangian solution of the bordered Hessian requires an inverse with $O(n^3)$ operations and subsequent updates, whereas the conjugate gradient method requires several $O(n^2)$ calculations at each iteration. This provides the conjugate gradient methods overwhelming advantage. Secondly, because the gradients relate to the original covariance matrix A and not its inverse, and because optimisation proceeds by search, the numerical inaccuracy associated with the Lagrange method is not a problem. The conjugate gradient method is not immune to numerical accuracy problems but restarting with a gradient, every so often, eliminates this. Line search is involved with the conjugate gradient algorithm but, because the function is quadratic this need involve no more than three function evaluations per iteration, using quadratic interpolation. In fact, even this can be eliminated and the function evaluations reduced to two per iteration adapting a result cited in Goldfeld and Quandt [1972]. Let the search direction be

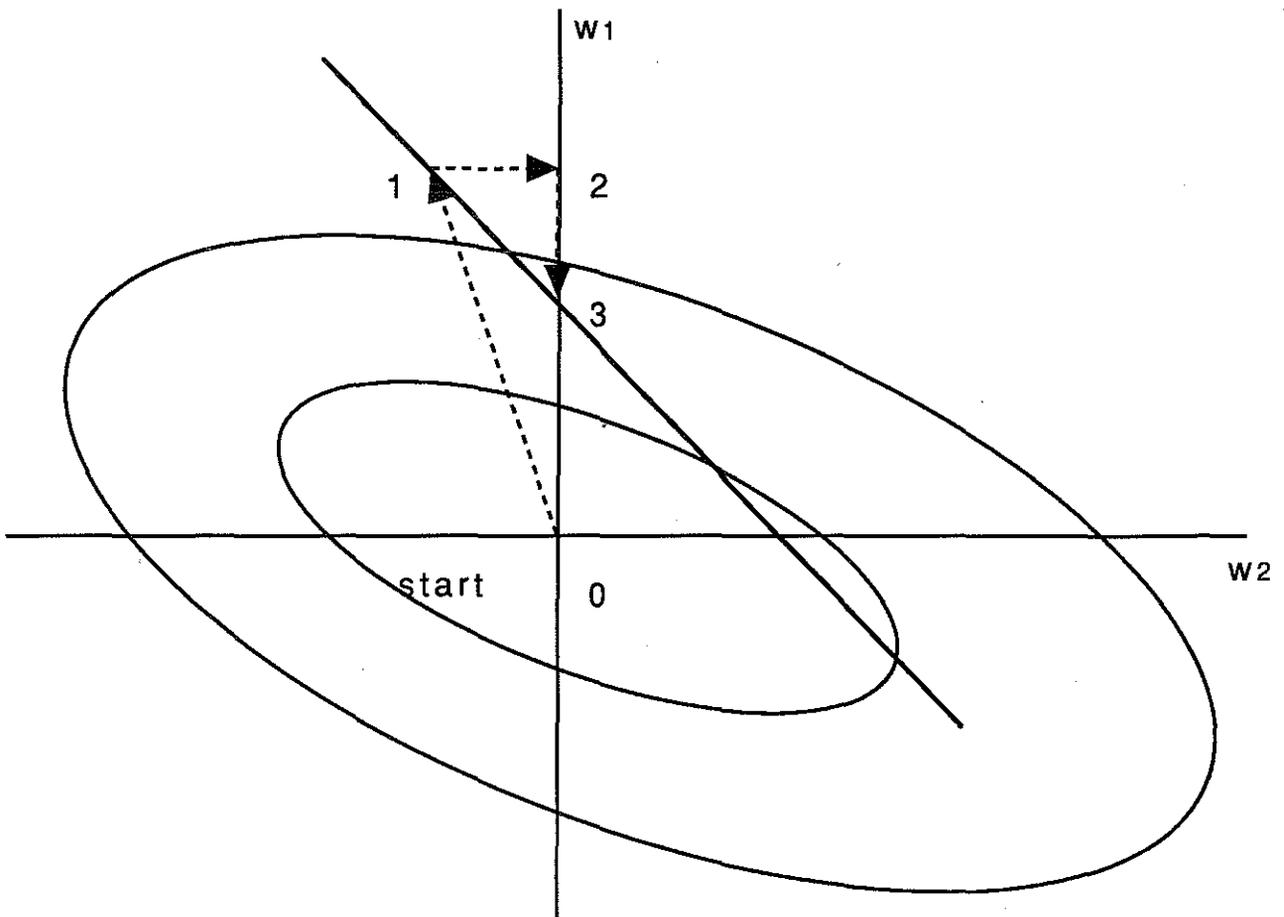
$$(19) \quad w_n = w_{n-1} + h d_{n-1}.$$

If the objective function is $w'Bw + 2 w'b + c$, then the minimising step is

$$(20) \quad h^* = - \frac{w'_{-1} B d_{-1} + b' d_{-1}}{d'_{-1} B d_{-1}}.$$

The real controversy with the approach used in this paper is not whether or not to apply the conjugate gradient algorithm, but whether to use an interior or an exterior point method. Mulvey's method maintains feasibility at all times and is interior point. What is being done here, and it corresponds to the approach of More' and Toraldo [1991] is to optimise the current objective function and then project back to the face of the feasible region in a direction orthogonal to the violated constraints. The algorithm is then restarted from this point. Graphically, it corresponds to Figure 2, starting from the initial unconstrained solution [0], proceeding to the optimal position satisfying the equality $w_1+w_2=1$, projecting orthogonally onto the face of the constraint set since the first step violated the condition $w_2 \geq 0$. In fact the orthogonal projection results in a move from [1] to [3] in one step.

Figure 2



More' and Toraldo [1991] provide two theorems {Th 5.1 and 5.2; pp.101-102}. proving that for a strictly convex non-degenerate problem the algorithm converges in a finite number of steps to the global optimum. The function we are dealing with, subject to simple linear restrictions or linear inequalities, is strictly convex. In essence ,the convergence features of the conjugate gradient algorithm and the identification properties (for local constrain faces) of the gradient projection method { see Bertsekas [1976, 1982], Dunn [1981, 1987], Calamai and More' [1987] and Burke and More' [1988] are combined. The Calamai and More' paper contains proofs of convergence for the second part of the algorithm, the use of gradient projection.

4. Trial Problems

The program was initially written in Absoft Fortran for a Mac II CX; but the code was later switched to Matlab, which is much faster. Comparisons had to be made with standard packages available on Dos machines - and crude conversions are provided where appropriate.

Consider the following problem

$$e = [4 \quad 7 \quad 8 \quad 12] \quad A = \begin{bmatrix} 10 & 2 & 4 & -5 \\ 2 & 20 & 4 & -4 \\ 4 & 4 & 40 & 10 \\ -5 & -4 & 10 & 60 \end{bmatrix}$$

Min $w'Aw$ subject to $e'w=8$ and $i'w=1$. The solution using Chang and Sullivan's [1989] quadratic programming package QSB⁺ (DOS) is

$$w_1=.2095, w_2=.3893, w_3=.0940 \text{ and } w_4=.3069 \quad \text{with} \quad \Phi = 9.27.$$

The program took 7 iterations and 1.05 seconds CPU time on an IBM PS2/70 (386/387). The author's Fortran program took 3 iterations and 1 second. The results were identical and the Mac appeared to be 60% faster than the PS2.

A more realistic problem is found in Brearly and Myers [1988, Handout E, pp.1-4] which is a 10 sector international portfolio optimisation problem. The covariance matrix is

222.010	241.738	26.716	74.023	74.425	50.779	76.765	87.910	57.752	116.518
241.738	432.640	30.514	72.717	92.664	171.309	133.952	137.446	75.878	180.336
26.716	30.514	265.690	122.967	66.015	37.034	113.970	69.242	55.746	94.214
74.023	72.717	122.967	338.560	69.552	99.286	104.954	82.506	48.245	96.968
74.425	92.664	66.015	69.552	182.250	15.336	129.168	93.987	55.404	107.865
50.779	171.309	37.034	99.286	15.336	806.560	224.701	67.024	67.990	193.120
76.765	133.952	113.970	104.954	129.168	224.701	338.560	106.389	94.392	140.760
87.910	137.446	69.242	82.506	93.987	67.024	106.389	139.240	49.772	116.348
57.752	75.878	55.746	48.245	55.404	67.990	94.392	49.772	129.960	106.590
116.518	180.336	94.214	96.968	107.865	193.120	140.760	116.348	106.590	289.000

with the expected returns

13.700 18.300 12.500 14.400 12.500 19.700 17.400 13.000 11.300 17.200.

The expected returns for the portfolio were 18% and the adding up condition was enforced. The solution for both QSB and conjugate gradients was

0 .3018 0 0 0 .1651 .2762 0 0 .2569

with $\Phi = 230.51$. QSB took 25 iterations and 6.15 seconds on the PC compared to 18 iterations and 3 seconds for the Fortran program on the Mac and 1.23 using MatLab.

A sequence of problems were then generated using artificial data involving 100-400 securities. Matlab was much superior to Fortran and only those results are cited here. The data were generated as a sequential white noise process with a lognormal distribution. The approach produces numbers which have all the characteristics of stock prices, with long series of erratically rising and then falling numbers. The data generation process is described by Benninga [1990] and by Ball and Officer [1991]. The stock prices are generated as $S_t = S_{t-1} + \exp[\mu/T + \sigma Z \sqrt{1/T}]$ with μ as the mean annual return, σ as the standard deviation of the stock and T as the number of business days in a year. Whilst this does not allow control of the correlation between different securities, it does ensure we emerge with reasonable looking stock price series for experimental purposes. In reality, the correlations (which ranged between -.20 and .20) were probably too low to be realistic; however, the data provides a useful benchmark to test algorithms.

Table 1

Performance of the Large Swap Conjugate Gradient Algorithm

Securities	100	300	400u	400s	400sv
Iterations	3	7	6	5	8
Time (secs)	9.56	92.1	229.8	194.2	172.8

QSB⁺ was unable to handle these problems; the limit was quoted as a 100 variable problem, 95% sparse. The program was modified for the 400 variable case; u refers to "unscaled" and is the program as applied to the other problems. 400s refers to the scaled version of the program; when the Hessian was scaled or conditioned to have unit diagonal elements. Finally sv refers to the trick of adjusting the convergence criterion on the conjugate gradient method so it ultimately tightens to the degree applied in

all other cases. In the present context it means the first few cycles will involve fewer iterations and fewer constraint swaps. The same solution is achieved ultimately.

5. Conclusion

The conjugate gradient algorithm is easily adapted to quadratic programming problems; the most important one probably being the portfolio optimisation problem. As has been found in other areas, the conjugate gradient algorithm is not memory hungry, is numerically stable and is extremely fast in execution. When this is combined with a method which allows large scale swapping of restrictions it results in a very fast algorithm suited for strictly convex problems; that is, for quadratic objective functions with linear equality and inequality constraints. Given the internationalisation of portfolios and the need to handle large scale portfolio optimisation problems such algorithms are becoming essential and large swap procedures will probably dominate feasible constraint methods for this class of problems.

References

- Ball, Ray and Robert R. Officer, 1991, "Try This on Your Chartist", 241-244 in Ball, Ray., Brown, Phillip., Finn, Frank.J., and Officer, Robert.R., **Share Markets and Portfolio Theory**, University of Queensland Press, Second Edition.
- Benninga, Simon, 1989, **Numerical Techniques in Finance**, MIT Press, ch 14.
- Bertsekas, Dimitri P., 1976, "On the Goldstein-Levitin-Polyak Gradient Projection Method", **IEEE Transactions of Automatic Control**, 21, 174-184.
- Bertsekas, Dimitri P., 1982, "Projected Newton Methods for Optimization Problems with Simple Constraints", **SIAM Journal of Control and Optimization**, 20, 1982, 221-246.
- Brearely, Richard A and Steward C Myers, 1988, **Principles of Corporate Finance**, (Instructor's Manual), McGraw-Hill.
- Burke, John V. and Jorge J. More', 1988, "On the Identification of Active Constraints", **SIAM Journal of Control and Optimization**, 25, 1197-1211.
- Byron, Raymond P., 1977, "Efficient Estimation and Inference for Large Econometric Systems", **Econometrica**, 45, 1499-1516.
- Calamai, Peter H. and Jorge J. More', 1987, "Projected Gradient Methods for Linearly Constrained Problems", **Mathematical Programming**, 39, 93-116.
- Chang, Yih-Long, and Robert S. Sullivan, 1989, **Quantitative Systems for Business Plus**, Prentice Hall.
- Dunn, John C., 1987, "Global and Asymptotic Convergence Rate Estimates for a Class of Projected Gradient Processes", **SIAM Journal of Control and Optimization**, 55, 203-216.
- Dunn, John C., 1987, "On the Convergence of Projected Gradient Processes to Singular Critical Points", **Journal of Optimization: Theory and Applications**, 55, 368-400.

- Fletcher, R., 1987, **Practical Methods of Optimisation**, Wiley.
- Steven M Goldfeld and Richard E.Quandt, 1977, **Nonlinear Methods in Econometrics**, North Holland.
- Karmarkar, N., 1984, "A New Polynomial Time Algorithm for Linear Programming", **Combinatorica**, 3, 373-395.
- Khachian, L.G., 1979, "A Polynomial Algorithm in Linear Programming", **Soviet Mathematics-Doklady**, 20, 191-194.
- Markowitz, Harry M., 1987, **Mean-Variance Analysis in Portfolio Choice and Capital Markets**, Basil Blackwell, Oxford
- More', Jorge J. and Geraldo Toraldo, 1989, "Algorithms for Bound Constrained Quadratic Programming Problems", **Numerische Mathematik**, 55, 377-400.
- More' Jorge J. and Geraldo Toraldo, 1991, "On the Solution of Large Quadratic Programming Problems with Bound Constraints", **SIAM Journal of Optimization**, 1, 93-113.
- Mulvey, John M., 1987, "Nonlinear Network Models in Finance", **Advances in Mathematical Programming and Financial Planning**, 1, 253-271.
- O'Leary, Daniel P., 1980, "A Generalized Conjugate Gradient Algorithm for Solving a Class of Conjugate Gradient Problems", **Linear Algebra Applications**, 34, 371-399.
- Reid, John K., 1971, "On the Method of Conjugate Gradients for the Solution of Large Sparse Systems of Equations", in **Large Sparse Sets of Equations**, (Ed. J.K.Reid), Academic Press.
- Rudd, Andrew and Barr Rosenberg, 1979, "Realistic Portfolio Optimisation", **TIMS Studies in Management Sciences**, 11, 21-46.
- Theil, Henri and Denzil Fiebig, 1984, **Exploiting Continuity: Maximum Entropy Estimation of Continuous Distributions**, Ballinger.
- Von HohenBalken, Balder., 1975, "A Finite Algorithm to Maximise Certain Pseudoconcave Functions on Polytopes", **Mathematical Programming**, .