PERGAMON

Computers & Operations Research 27 (2000) 1271–1302

computers & operations research

www.elsevier.com/locate/dsw

Heuristics for cardinality constrained portfolio optimisation

T.-J. Chang[a], N. Meade[a], J.E. Beasley[a,*], Y.M. Sharaiha[b]

[a]*The Management School, Imperial College, London SW7 2AZ, UK*
[b]*Morgan Stanley Dean Witter, Quantitative Strategies, 25 Cabot Square, Canary Wharf, London E14 4QA, UK*

Received 1 May 1998; received in revised form 1 February 1999

Abstract

In this paper we consider the problem of finding the efficient frontier associated with the standard mean–variance portfolio optimisation model. We extend the standard model to include cardinality constraints that limit a portfolio to have a specified number of assets, and to impose limits on the proportion of the portfolio held in a given asset (if any of the asset is held). We illustrate the differences that arise in the shape of this efficient frontier when such constraints are present. We present three heuristic algorithms based upon genetic algorithms, tabu search and simulated annealing for finding the cardinality constrained efficient frontier. Computational results are presented for five data sets involving up to 225 assets.

Scope and purpose

The standard Markowitz mean–variance approach to portfolio selection involves tracing out an efficient frontier, a continuous curve illustrating the tradeoff between return and risk (variance). This frontier can be easily found via quadratic programming. This approach is well-known and widely applied. However, for practical purposes, it may be desirable to limit the number of assets in a portfolio, as well as imposing limits on the proportion of the portfolio devoted to any particular asset. If such constraints exist, the problem of finding the efficient frontier becomes much harder. This paper illustrates how, in the presence of such constraints, the efficient frontier becomes discontinuous. Three heuristic techniques are applied to the problem of finding this efficient frontier and computational results presented for a number of data sets which are made publicly available.

Keywords: Portfolio optimisation; Efficient frontier

* Corresponding author. Tel.: + 0171-594-9171; fax: + 0171-823-7685.
E-mail address: j.beasley@ic.ac.uk (J.E. Beasley)

0305-0548/00/$ - see front matter
PII: S0305-0548(99)00074-X

1. Introduction

Each of the larger fund management companies in the UK/US are responsible for the investment of several billion pounds/dollars. This money is invested on behalf of pension funds, unit trusts (mutual funds) and other institutions. The selection of an appropriate portfolio of assets in which to invest is an essential component of fund management. Although a large proportion of portfolio selection decisions are taken on a qualitative basis, quantitative approaches to selection are becoming more widely adopted.

Markowitz [1,2] set up a quantitative framework for the selection of a portfolio. In this framework it is assumed that asset returns follow a multivariate normal distribution. This means that the return on a portfolio of assets can be completely described by the expected return and the variance (risk). For a particular universe of assets, the set of portfolios of assets that offer the minimum risk for a given level of return form the efficient frontier. The portfolios on the efficient frontier can be found by quadratic programming (QP). The strengths of this approach are that QP solvers are available and efficient in terms of computing time. The solutions are optimal and the selection process can be constrained by practical considerations which can be written as linear constraints.

The weaknesses are of two kinds:

(1) The underlying assumption of multivariate normality is not sustainable (see, for example, Mills [3]). The distribution of individual asset returns tends to exhibit a higher probability of extreme values than is consistent with normality (statistically this is known as leptokurtosis). This departure from multivariate normality means that distribution moments higher than the first two moments (expected return and variance) need to be considered to fully describe portfolio behaviour.
(2) Integer constraints that limit a portfolio to have a specified number of assets, or to impose limits on the proportion of the portfolio held in a given asset (if any of the asset is held) cannot easily be applied. Constraints of this type are of practical significance.

This paper examines the use of three standard heuristic methods in portfolio selection. The methods considered are genetic algorithms, tabu search and simulated annealing. The attraction of these approaches is that they are effectively independent of the objective function adopted. This means that the Markowitz quadratic objective function can potentially be replaced in the light of the first set of weaknesses identified above. In addition, the imposition of integer constraints is straightforward.

In this paper the heuristics that we have developed are described and their performance compared with that of QP for the construction of the unconstrained efficient frontier (UEF). This approach allows the closeness of the heuristic solutions to optimality to be measured. The performance of the heuristic methods in constructing the efficient frontier in the presence of a constraint fixing the number of assets in the selected portfolio is demonstrated. This frontier is called the cardinality constrained efficient frontier (CCEF).

2. Formulation

In this section we formulate the cardinality constrained mean–variance portfolio optimisation problem. We first formulate the unconstrained portfolio optimisation problem and illustrate how

to calculate the efficient frontier. We then comment on the approaches presented in the literature that have used a different objective function. Finally we formulate the cardinality constrained problem.

2.1. Unconstrained problem

Let
N be the number of assets available,
μ_i be the expected return of asset i ($i = 1, \ldots, N$),
σ_{ij} be the covariance between assets i and j ($i = 1, \ldots, N; j = 1, \ldots, N$),
R^* be the desired expected return.
Then the decision variables are:
w_i the proportion ($0 \leqslant w_i \leqslant 1$) held of asset i ($i = 1, \ldots, N$)
and using the standard Markowitz mean–variance approach [1,2,4–6] we have that the unconstrained portfolio optimisation problem is

$$\text{minimise} \quad \sum_{i=1}^{N} \sum_{j=1}^{N} w_i w_j \sigma_{ij} \tag{1}$$

subject to

$$\sum_{i=1}^{N} w_i \mu_i = R^* \tag{2}$$

$$\sum_{i=1}^{N} w_i = 1 \tag{3}$$

$$0 \leqslant w_i \leqslant 1, \quad i = 1, \ldots, N \tag{4}$$

Eq. (1) minimises the total variance (risk) associated with the portfolio whilst Eq. (2) ensures that the portfolio has an expected return of R^*. Eq. (3) ensures that the proportions add to one.

This formulation (Eqs. (1)–(4)) is a simple nonlinear (quadratic) programming problem for which computationally effective algorithms exist so there is (in practice) little difficulty in calculating the optimal solution for any particular data set.

Note here that the above formulation (Eqs. (1)–(4)) can be expressed in terms of the correlation ρ_{ij} between assets i and j ($-1 \leqslant \rho_{ij} \leqslant +1$) and the standard deviations s_i, s_j in returns for these assets since $\sigma_{ij} = \rho_{ij} s_i s_j$.

2.2. Efficient frontier

By resolving the above QP (Eqs. (1)–(4)) for varying values of R^* we can trace out the efficient frontier, a smooth non-decreasing curve that gives the best possible tradeoff of risk against return, i.e. the curve represents the set of Pareto-optimal (non-dominated) portfolios. Throughout this paper we refer to this curve as the unconstrained efficient frontier (UEF). One such UEF is shown in Fig. 1 for assets drawn from the UK FTSE market index.

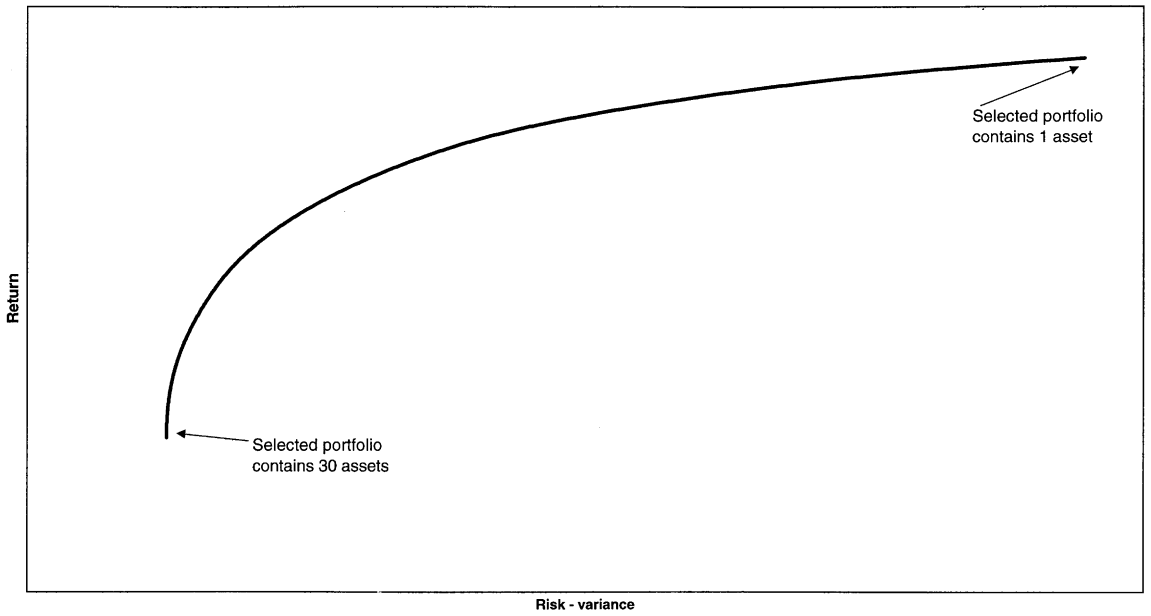

Fig. 1. An example efficient frontier.

For the unconstrained case it is standard practice to trace out the efficient frontier by introducing a weighting parameter λ ($0 \leqslant \lambda \leqslant 1$) and considering:

$$\text{minimise} \quad \lambda\left[\sum_{i=1}^{N}\sum_{j=1}^{N} w_i w_j \sigma_{ij}\right] - (1-\lambda)\left[\sum_{i=1}^{N} w_i \mu_i\right] \tag{5}$$

subject to

$$\sum_{i=1}^{N} w_i = 1, \tag{6}$$

$$0 \leqslant w_i \leqslant 1, \quad i = 1, \ldots, N. \tag{7}$$

In Eq. (5) the case $\lambda = 0$ represents maximise expected return (irrespective of the risk involved) and the optimal solution will involve just the single asset with the highest return. In Eq. (5) the case $\lambda = 1$ represents minimise risk (irrespective of the return involved) and the optimal solution will typically involve a number of assets. Values of λ satisfying $0 < \lambda < 1$ represent an explicit tradeoff between risk and return, generating solutions between the two extremes $\lambda = 0$ and $\lambda = 1$.

As before, by resolving this QP (Eqs. (5)–(7)) for varying values of λ, we can trace out the efficient frontier. To see that this is so, consider a particular value of λ, e.g. $\lambda = 0.25$. Then the objective (Eq. (5)), which we wish to minimise, becomes 0.25[risk] − 0.75[return]. Considering Fig. 1, which shows the efficient frontier as plotted by considering varying values of R^*, we could plot a series of "iso-profit" lines 0.25[risk] − 0.75[return] = Z and choose the minimum value of Z. Rearranging, these iso-profit lines are [return] = (1/3)[risk] − (4/3)Z, i.e. lines of slope (1/3) and intercept on the

y-axis of $-(4/3)Z$. Minimising Z therefore corresponds to choosing amongst these iso-profit lines of fixed slope so as to maximise the intercept on the y-axis. It is clear that this can be achieved at the (unique) point where the iso-profit line of slope (1/3) is a tangent to the efficient frontier.

Hence, by varying λ (varying the slope of the iso-profit lines) and solving Eqs. (5)–(7) we can trace out *exactly* the same efficient frontier curve as we would obtain by solving Eqs. (1)–(4) for varying values of R^*.

2.3. Other objectives

Departures from the standard Markowitz mean–variance approach presented above include the following considerations:

(a) whether variance is considered to be an adequate measure of the risk associated with the portfolio or not; and
(b) including transaction costs associated with changing from a current portfolio to a new portfolio.

Konno and Yamazaki [7] proposed that the mean absolute deviation (MAD) of portfolio returns from average (measured over a specified time period) be taken as the risk measure. This allows the portfolio selection problem to be formulated and solved via linear programming (see also [8]). Simaan [9] contends that the computational savings from the use of MAD objective functions are outweighed by the loss of information from the (unused) covariance matrix.

The possible asymmetry of returns is taken into account by Konno et al. [10] who extended the MAD approach to include skewness in the objective function. Konno and Suzuki [11] considered a mean–variance objective function extended to include skewness. Negative semi-variance, discussed by Markowitz [2], is one of several objective functions that consider downside risk only. Feiring et al. [12] use lower partial moments (a generalisation of negative semi-variance) as an objective function.

With regard to transaction costs:

(a) for a single period optimisation, Adcock and Meade [13] suggested a mixed quadratic and modulus objective function, soluble by QP;
(b) for multiperiod optimisation, Mulvey and Vladimirou [14] used stochastic network programming and Yoshimoto [15] used non-linear programming.

2.4. Constrained problem

In order to extend our formulation to the cardinality constrained case let:

K be the desired number of assets in the portfolio,

ε_i be the minimum proportion that must be held of asset i ($i = 1, \ldots, N$) if any of asset i is held,

δ_i be the maximum proportion that can be held of asset i ($i = 1, \ldots, N$) if any of asset i is held,

where we must have $0 \leqslant \varepsilon_i \leqslant \delta_i \leqslant 1$ ($i = 1, \ldots, N$). In practice ε_i represents a "min-buy" or "minimum transaction level" for asset i and δ_i limits the exposure of the portfolio to asset i. Introducing zero–one decision variables:

$$z_i = \begin{cases} 1 & \text{if any of asset } i \ (i = 1, \ldots, N) \text{ is held,} \\ 0 & \text{otherwise,} \end{cases}$$

the cardinality constrained portfolio optimisation problem is

$$\text{minimise} \quad \sum_{i=1}^{N} \sum_{j=1}^{N} w_i w_j \sigma_{ij} \tag{8}$$

subject to

$$\sum_{i=1}^{N} w_i \mu_i = R^*, \tag{9}$$

$$\sum_{i=1}^{N} w_i = 1, \tag{10}$$

$$\sum_{i=1}^{N} z_i = K, \tag{11}$$

$$\varepsilon_i z_i \leqslant w_i \leqslant \delta_i z_i, \quad i = 1, \ldots, N \tag{12}$$

$$z_i \in [0, 1], \quad i = 1, \ldots, N. \tag{13}$$

Eq. (8) minimises the total variance (risk) associated with the portfolio whilst Eq. (9) ensures that the portfolio has an expected return of R^*. Eq. (10) ensures that the proportions add to one whilst Eq. (11) ensures that exactly K assets are held. Eq. (12) ensures that if any of asset i is held ($z_i = 1$) its proportion w_i must lie between ε_i and δ_i, whilst if none of asset i is held ($z_i = 0$) its proportion w_i is zero. Eq. (13) is the integrality constraint.

The objective function (Eq. (8)), involving as it does the covariance matrix, is positive semi-definite [16–20] and hence we are minimising a convex function.

Note here that we have explicitly chosen to formulate this problem with an equality (rather than an inequality $\leqslant$) with respect to the number of assets in the portfolio (Eq. (11)). This is because if we can solve the equality constrained case then any situation involving inequalities (lower or upper limits on the number of assets in the portfolio) can be easily dealt with. For example if we require

$$K_L \leqslant \sum_{i=1}^{N} z_i \leqslant K_U \tag{14}$$

so that the number of assets in the portfolio lies between K_L and K_U ($K_L \neq K_U$) then this can be dealt with in our formulation of the problem (Eqs. (8)–(13)) by examining all values of K (Eq. (11)) between K_L and K_U, i.e. $K = K_L, K_L + 1, \ldots, K_U$.

Whilst this might seem a cumbersome approach (why be forced to explicitly examine all values of K between K_L and K_U, why not simply consider all such values implicitly in some algorithm?) we feel that this approach has merit. Our reasoning is that, in practice, the decision as to the number of assets (K) to have in the chosen portfolio is one that can only be decided by the decision-maker in the light of the tradeoffs between the three factors (risk, return, K) involved. This involves considering factors that are not captured in the model, e.g. the decision-makers attitude to risk and the ease of constructing and maintaining a portfolio involving K assets. Under our approach the decision-maker will be faced with a different CCEF for each value of K and must explicitly consider the tradeoffs involved in deciding which portfolio to adopt. An illustration of this is given in Section 5.5 below.

2.5. Previous work

There has been relatively little work presented in the literature that relates to the cardinality constrained portfolio optimisation problem.

Note first that the formulation (Eqs. (8)–(13)) presented above is a mixed-integer nonlinear (quadratic) programming problem for which, in contrast to the unconstrained case (Eqs. (1)–(4)), computationally effective algorithms do not exist. Broadly there are two general approaches that can be followed in tacking this problem:

(a) to solve the problem using the algorithms available for solving mixed-integer nonlinear programs (e.g. see [21–24]);
(b) to replace the quadratic Markowitz measure of risk (Eq. (8)) by a measure of risk that is a linear function (or equivalently a linearisable function) enabling the array of algorithms available for mixed-integer linear programming to be used.

2.5.1. Quadratic risk

With regard to the first of these approaches work has been presented by Bienstock [16,17] and Lee and Mitchell [25].

Bienstock [16,17] considered a similar cardinality constrained portfolio optimisation problem to that formulated above. He presented a number of valid inequalities (cuts) for the problem and presented a branch and cut algorithm based on disjunctive cuts. Computational results were presented for both sequential and parallel implementations of his algorithm involving up to 3897 assets.

Lee and Mitchell [25] considered a similar cardinality constrained portfolio optimisation problem to that formulated above. Their approach is based upon an interior point nonlinear solver using a network of loosely coupled workstations in a distributed (parallel) environment. They presented computational results for problems involving up to 150 assets. See also Borchers and Mitchell [22].

2.5.2. Linear risk

With regard to the second of these approaches work has been presented by Speranza [26], Mansini and Speranza [27], Kellerer et al. [28], and Young [29].

Speranza [26] considered a cardinality constrained portfolio optimisation problem, but with the risk associated with the portfolio being measured by the mean absolute deviation of the return below average (negative semi-MAD), rather than by variance. She gave a mixed-integer linear programming formulation together with a heuristic algorithm. Computational results were presented for problems involving up to 20 assets.

Mansini and Speranza [27] used the same risk measure as in [26] and considered the problem where the assets must be bought in integer multiples of a certain specified amount (the minimum lot). Three heuristics for the problem, based upon first solving the linear programming relaxation, were presented and computational results given for two data sets involving 244 and 277 assets.

Kellerer et al. [28] used the same risk measure as in [26] and considered the problem where a fixed cost is incurred if investment in an asset exceeds a certain limit. They also considered

minimum lots (see [27]). Two heuristics for the problem, based upon first solving the linear programming relaxation, were presented and computational results given for one data set involving 244 assets.

Young [29] indicated how minimum transaction levels could be incorporated into a formulation where the risk (as modelled by the minimum historical return on the portfolio) is limited. No computational results for such problems were presented however.

2.6. Practical constraints

There are a number of constraints that can be added to our constrained formulation (Eqs. (8)–(13)) to better reflect practical portfolio optimisation.

(a) *Class constraints*: Let Γ_m, $m = 1, \ldots, M$, be M sets of assets that are mutually exclusive, i.e. $\Gamma_i \cap \Gamma_j = \emptyset$, $\forall i \neq j$. Class constraints limit the proportion of the portfolio that can be invested in assets in each class. Let L_m be the lower proportion limit and U_m be the upper proportion limit for class m then the class constraints are:

$$L_m \leqslant \sum_{i \in \Gamma_m} w_i \leqslant U_m, \quad m = 1, \ldots, M. \tag{15}$$

Such constraints typically limit the "exposure" of the portfolio to assets with a common characteristic. For example typical classes might be oil stocks, utility stocks, telecommunication stocks, etc. The heuristics presented in this paper do not deal with constraints of this type.

(b) *Assets in the portfolio*: Assets which must be in the portfolio (at some proportion between ε_i and δ_i yet to be determined) can be accommodated in our formulation simply by setting z_i to one for any such asset i. Although we do not present it below the changes required to our heuristics to deal with this are trivial.

3. Constrained efficient frontier

One aspect of constrained portfolio optimisation that appears to have received no attention in the literature is the fact that in the presence of constraints of the type we have considered above the efficient frontier is markedly different from the UEF. In this section we illustrate this.

3.1. Cardinality constraints

In order to illustrate the effect of cardinality constraints we consider the small four asset example problem shown in Table 1 (drawn from the FTSE data set we consider later). Consider the cardinality constrained problem (Eqs. (8)–(13)) for this data involving exactly two assets ($K = 2$), where $\varepsilon_i = 0, \delta_i = 1$, $i = 1, 2, 3, 4$. By simply enumerating (to an appropriate number of decimal places) all possible values for the proportions w_i we can construct all feasible combinations of two assets. Fig. 2 shows these feasible combinations, six line segments are shown there (one segment for

Table 1
Four asset example

Asset	Return (weekly)	Standard deviation	Correlation matrix			
			1	2	3	4
1	0.004798	0.046351	1	0.118368	0.143822	0.252213
2	0.000659	0.030586		1	0.164589	0.099763
3	0.003174	0.030474			1	0.083122
4	0.001377	0.035770				1

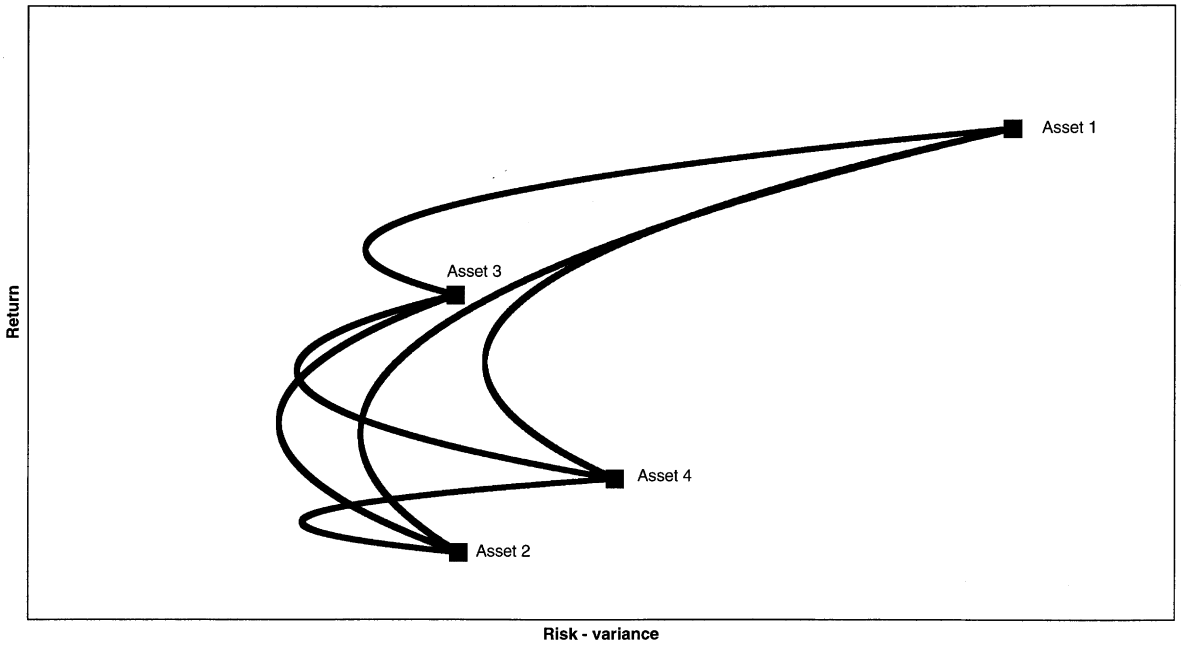

Fig. 2. Feasible combinations of two assets.

each pair of assets). For example the line segment between asset 3 and asset 1 (the top line segment in Fig. 2) represents portfolios composed of these two assets in varying proportions.

Fig. 2 represents the universe of possible portfolios composed of some combination of two assets. Now it is plain that certain portfolios shown in Fig. 2 are dominated. For example all portfolios on the line segment between asset 4 and asset 1 are dominated (since for any portfolio on that line segment there exists another portfolio with less risk but greater return). If we eliminate from Fig. 2 all portfolios which are dominated we will be left with the set of efficient portfolios, in other words the efficient frontier for this cardinality constrained example.

This cardinality constrained efficient frontier is shown in Fig. 3. Note that it is a *discontinuous* curve, unlike the continuous curve we saw before for the UEF. In other words, in the presence of cardinality constraints the efficient frontier may become discontinuous, where the discontinuities imply that there are certain returns which no rational investor would consider (since there exist portfolios with less risk and greater return). Throughout this paper we refer to the cardinality constrained efficient frontier as the CCEF.

3.2. Weighting

As for the unconstrained case in Section 2.2 above introduce a weighting parameter λ ($0 \leqslant \lambda \leqslant 1$) and consider:

$$\text{minimise} \quad \lambda\left[\sum_{i=1}^{N}\sum_{j=1}^{N} w_i w_j \sigma_{ij}\right] - (1-\lambda)\left[\sum_{i=1}^{N} w_i \mu_i\right] \tag{16}$$

subject to

$$\sum_{i=1}^{N} w_i = 1, \tag{17}$$

$$\sum_{i=1}^{N} z_i = K, \tag{18}$$

$$\varepsilon_i z_i \leqslant w_i \leqslant \delta_i z_i, \quad i = 1, \ldots, N, \tag{19}$$

$$z_i \in [0,1], \quad i = 1, \ldots, N. \tag{20}$$

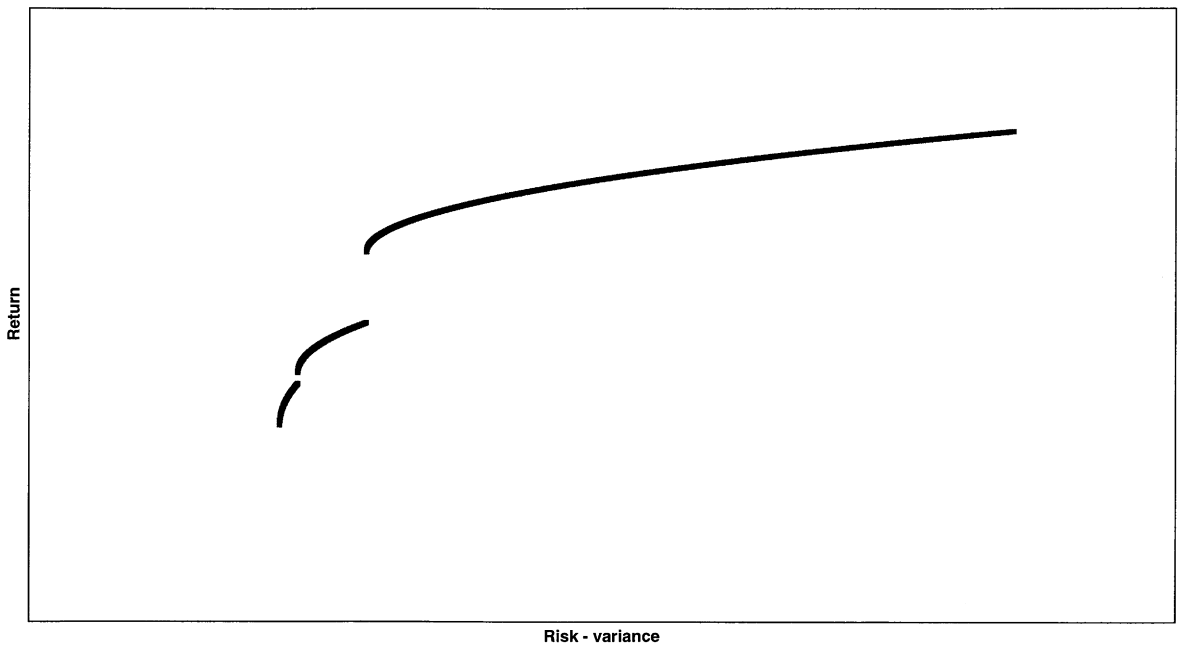

Fig. 3. Cardinality constrained efficient frontier.

It would be natural to believe that by varying λ we could use this program (Eqs. (16)–(20)) to trace out the CCEF in an exactly analogous way as can be done in the unconstrained case for the UEF. In fact this is not so.

To see why recall that the argument presented in Section 2.2 above relied upon straight lines tangential to the efficient frontier which maximised their intercept on the y-axis. Consider the upper part of the middle segment of the discontinuous CCEF shown in Fig. 3. Any straight line tangential to this can obtain a greater intercept on the y-axis by moving to be tangential to the top segment of the discontinuous CCEF shown in Fig. 3.

In other words, even if we were able to solve Eqs. (16)–(20) *exactly* for as many values of λ as we wished there will always be portions of the CCEF shown in Fig. 3 that can *never* be found by such an approach, i.e. they are effectively (mathematically) invisible to an exact approach based upon weighting.

Fig. 4 shows those portions of the CCEF that are invisible to an exact approach based upon weighting whilst Fig. 5 shows those portions of the CCEF that are visible to an exact approach based upon weighting.

3.3. Minimum proportion constraints

To illustrate the effect of imposing a non-zero minimum proportion the efficient frontier for the case where $\varepsilon_i = 0.24$ and $\delta_i = 1$ $(i = 1, 2, 3, 4)$ is shown in Fig. 6, where that figure has been plotted explicitly disregarding the cardinality constraint (Eq. (11)). It is clear from that figure that again the

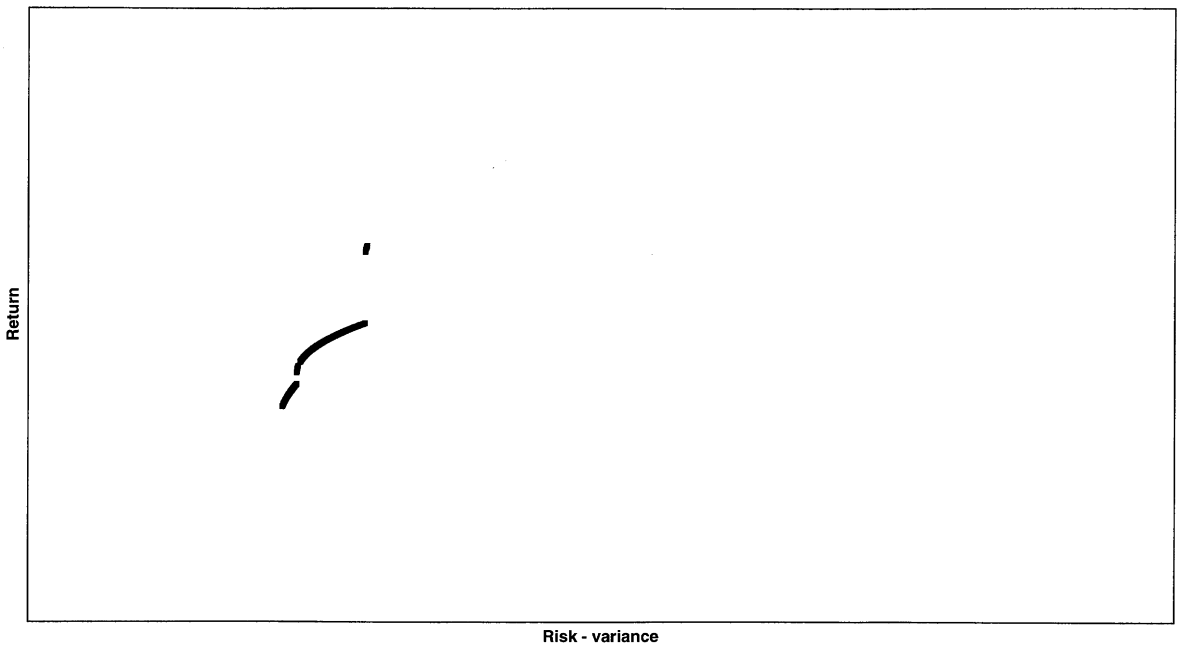

Fig. 4. Cardinality constrained efficient frontier – invisible portions.

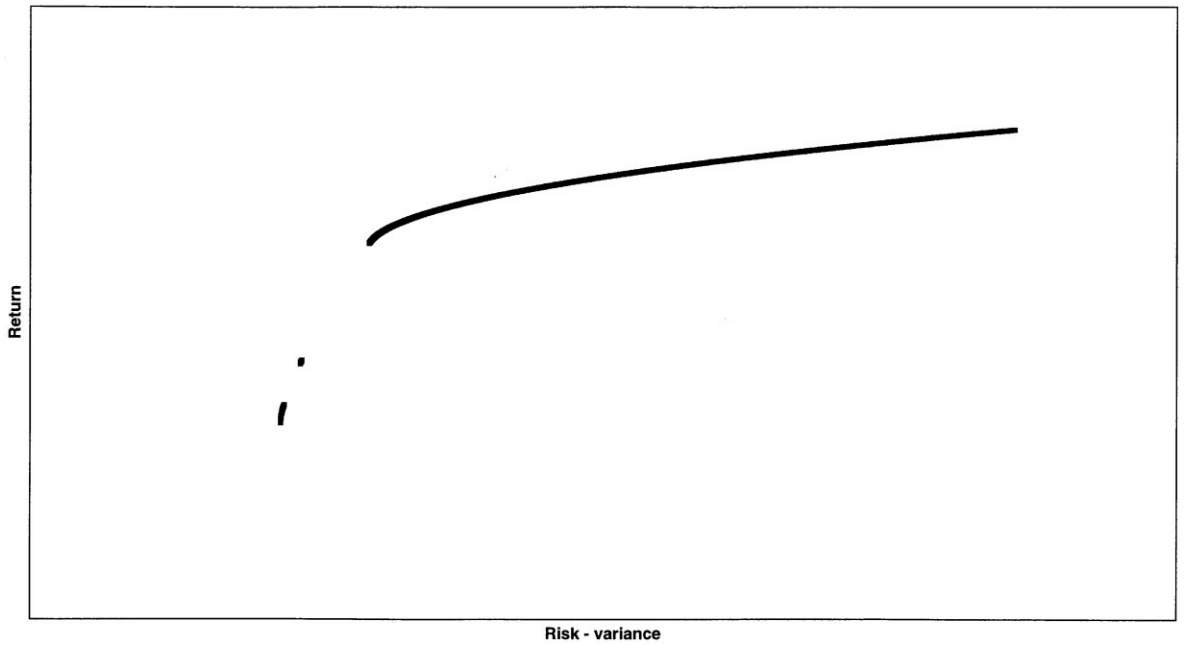

Fig. 5. Cardinality constrained efficient frontier – visible portions.

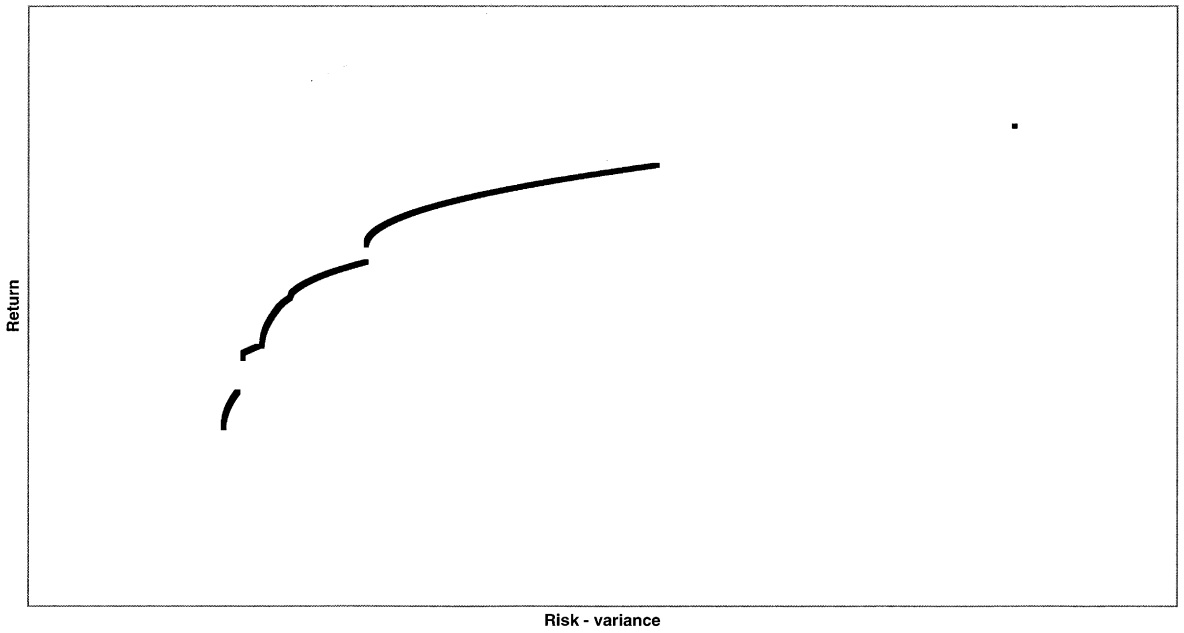

Fig. 6. Efficient frontier with a minimum proportion of 0.24.

efficient frontier is discontinuous and has portions that are invisible to an exact approach based upon weighting. Note here that the ε_i values chosen do not implicitly induce a cardinality constraint, so the effects shown in Fig. 6 are a direct consequence of the minimum proportion constraints.

3.4. Summary

To summarise this section then we have shown through a small numeric example that if cardinality constraints and/or minimum proportion constraints are present:

(a) the efficient frontier may be discontinuous;
(b) the efficient frontier may contain portions that are invisible to an exact approach based upon weighting.

4. Heuristic algorithms

In this section we outline the three heuristic algorithms based upon genetic algorithms, tabu search and simulated annealing that we have developed for finding the CCEF. We also discuss here any application of these techniques to portfolio optimisation previously reported in the literature.

Note here that all of our heuristics use the weighted formulation (Eqs. (16)–(20)) presented in Section 3.2 above. We had two reasons for using this weighted formulation:

(a) even though solving the weighted formulation exactly has the implication that some portions of the CCEF are invisible (Section 3 above), in a heuristic approach involving investigating many different solutions it is possible to gain information about such portions; and
(b) attempting to design a computationally effective heuristic that directly addresses the non-weighted formulation (Eqs. (8)–(13)) is difficult because of the requirement that the portfolio expected return is exactly R^* (Eq. (9)).

With respect to the first of these two reasons Fig. 7 shows the CCEF, as found by the heuristics presented below, for the four asset example ($K = 2$) considered previously. Comparing this figure with Fig. 3, which was the exact CCEF as calculated by enumerating all possible combinations, it is clear that the two curves are effectively identical. Hence, for this example, our heuristics have found all portions (visible or invisible) of the CCEF.

With respect to the second of these reasons it might appear that we would be in a better position to design a heuristic algorithm if the equality constraint relating to return (Eq. (9)) were changed to an inequality, i.e. to

$$\sum_{i=1}^{N} w_i \mu_i \geqslant R^*. \tag{21}$$

If this were done then the CCEF (found by parametrically varying R^* and eliminating dominated portfolios) would be precisely as before.

However, our view is that it is the explicit presence of this constraint in the problem that is the root cause of algorithmic design problems. Whether the constraint is present as an equality, or as

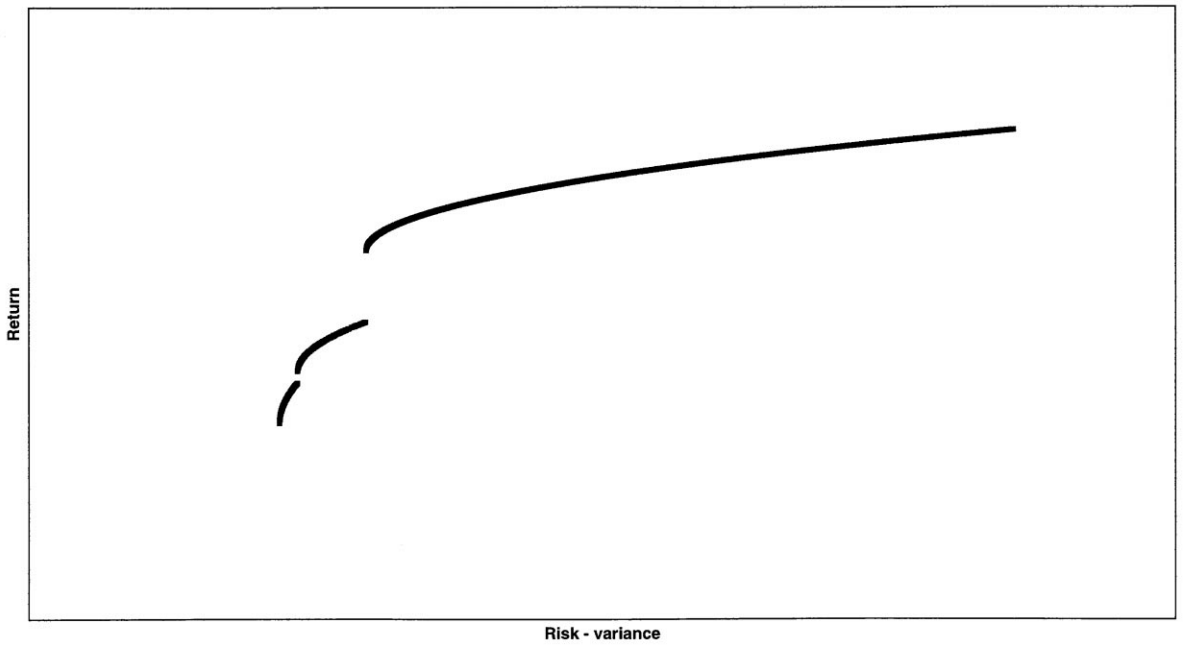

Fig. 7. Cardinality constrained efficient frontier as found by the heuristics.

an inequality, is a secondary issue. Hence we believe that subsuming the expected return constraint into the objective function via a weighting approach (as is done in the weighted formulation given in Eqs. (16)–(20)) is a profitable algorithmic approach.

We note in passing that our heuristics, because they use this weighted formulation, are equally applicable to the problem when formulated with Eq. (21) in place of Eq. (9).

4.1. Genetic algorithms

A genetic algorithm (GA) can be described as an "intelligent" probabilistic search algorithm. The theoretical foundations of GAs were originally developed by Holland [30]. GAs are based on the evolutionary process of biological organisms in nature. During the course of evolution, natural populations evolve according to the principles of natural selection and "survival of the fittest". Individuals which are more successful in adapting to their environment will have a better chance of surviving and reproducing, whilst individuals which are less fit will be eliminated. This means that the *genes* from the highly fit individuals will spread to an increasing number of individuals in each successive generation. The combination of good characteristics from highly adapted parents may produce even more fit offspring. In this way, species evolve to become increasingly better adapted to their environment.

A GA simulates these processes by taking an initial population of individuals and applying genetic operators in each reproduction. In optimisation terms, each individual in the population is encoded into a string or *chromosome* which represents a possible *solution* to a given problem. The

fitness of an individual is evaluated with respect to a given objective function. Highly fit individuals or *solutions* are given opportunities to reproduce by exchanging pieces of their genetic information, in a *crossover* procedure, with other highly fit individuals. This produces new "offspring" solutions (i.e. *children*), which share some characteristics taken from both parents. Mutation is often applied after crossover by altering some genes in the strings. The offspring can either replace the whole population (*generational* approach) or replace less fit individuals (*steady-state* approach). This evaluation–selection–reproduction cycle is repeated until a satisfactory solution is found. The basic steps of a simple GA are shown below.

Generate an initial population
Evaluate fitness of individuals in the population
repeat:
    Select parents from the population
    Recombine (mate) parents to produce children
    Evaluate fitness of the children
    Replace some or all of the population by the children
until a satisfactory solution has been found.

A more comprehensive overview of GAs can be found in [31–34].

Arnone et al. [35] presented a GA for the unconstrained portfolio optimisation problem, but with the risk associated with the portfolio being measured by downside risk rather than by variance. Computational results were presented for one problem involving 15 assets.

Loraschi et al. [36] presented a distributed GA for the unconstrained portfolio optimisation problem based on an island model where a GA is used with multiple independent subpopulations (each run on a different processor) and highly-fit individuals occasionally migrate between the subpopulations. Computational results were presented for one problem involving 53 assets comparing their distributed GA with the GA presented in [35].

4.2. Genetic algorithm heuristic

In our GA heuristic the chromosome representation of a solution has two distinct parts, a set Q of K distinct assets and K real numbers s_i $(0 \leqslant s_i \leqslant 1)$ $i \in Q$. Now given a set Q of K assets a fraction $\sum_{j \in Q} \varepsilon_j$ of the total portfolio is already accounted for and so we interpret s_i as relating to the share of the *free* portfolio proportion $(1 - \sum_{j \in Q} \varepsilon_j)$ associated with asset $i \in Q$.

Not all possible chromosomes correspond to feasible solutions (because of the constraint (Eq. (19)) relating to the limits on the proportion of an asset that can be held). However, when evaluating each solution the simple procedure shown in pseudocode in Algorithm 1 was used in order to try and ensure that the evaluated solution was feasible.

To explain our representation and Algorithm 1 further suppose that we have $N = 10$, $K = 2$ and $\varepsilon_i = 0.1$ $\forall i$. One GA solution might therefore be $Q = \{3, 7\}$ and $\{s_3 = 0.9, s_7 = 0.5\}$. This means that assets 3 and 7 are in the portfolio. The free portfolio proportion $(1 - \sum_{j \in Q} \varepsilon_j) = 0.8$, since we know that each of the two assets must have a proportion in the portfolio of at least 0.1. Hence we interpret this GA representation of $\{s_3 = 0.9, s_7 = 0.5\}$ to mean that the share of the free portfolio proportion devoted to asset 3 is $s_3/(s_3 + s_7) = 0.9/1.4 = 0.6429$. Hence the proportion w_3 associated with asset 3 in the portfolio is given by $0.1 + 0.6429(0.8)$, i.e. the minimum proportion plus

the appropriate share of the free portfolio proportion, hence $w_3 = 0.6143$. Similarly the proportion w_7 associated with asset 7 is $w_7 = 0.1 + (s_7/(s_3 + s_7))0.8 = 0.1 + (0.5/1.4)0.8 = 0.3857$. Note that these values for w_3 and w_7 both satisfy the lower proportion limits and sum to one.

In Algorithm 1 we can automatically ensure that the constraints relating to the lower limits ε_i are satisfied in a single algorithmic step. However we need an iterative procedure to ensure that the

evaluate($S, \lambda, f, V, improved, H$)
S is the current solution and consists of:
  Q the set of K distinct assets in the current solution
  s_i the current value for asset $i \in Q$
λ is the current weighting parameter
f is the returned objective function value for the current solution S
$V(\lambda)$ is the best objective function value found for weighting parameter λ
improved is returned as *.true.* if S improves $V(\lambda)$, else returned as. *false.*
H is all the improved solutions found during the course of the algorithm
w_i is the proportion associated with each asset $i \in Q$

begin
improved := *.false.*
$f := \infty$
if $\sum_{i \in Q} \varepsilon_i > 1$ or $\sum_{i \in Q} \delta_i < 1$ **then return** /* infeasible */
$L := \sum_{i \in Q} s_i$ /* L is the current s_i sum */
$F := 1 - \sum_{i \in Q} \varepsilon_i$ /* F is the free proportion */
$w_i := \varepsilon_i + s_i F/L \ \forall i \in Q$ /* calculate proportions to satisfy ε_i and sum to 1 */
/* iterative procedure to satisfy maximum proportions */
$R := \emptyset$ /* R is a set of i whose proportions are fixed at δ_i */
while there exists an $i \in Q - R$ with $w_i > \delta_i$ **do** /* iterate until feasible */
  for all $i \in Q - R$ **if** $w_i > \delta_i$ **then** $R := R \cup [i]$ /* if w_i exceeds δ_i add to R */
  $L := \sum_{i \in Q - R} s_i$ /* L is the current s_i sum */
  $F := 1 - (\sum_{i \in Q - R} \varepsilon_i + \sum_{i \in R} \delta_i)$ /* F is the free proportion */
  $w_i := \varepsilon_i + s_i F/L \ \forall i \in Q - R$ /* proportions for $i \in Q - R$ */
  $w_i := \delta_i \ \forall i \in R$ /* proportions for $i \in R$ */
end while
$f := \lambda[\sum_{i \in Q} \sum_{j \in Q} w_i w_j \sigma_{ij}] - (1 - \lambda)[\sum_{i \in Q} w_i \mu_i]$ /* feasible solution */
$s_i := w_i - \varepsilon_i \ \forall i \in Q$ /* reset s_i */
if $f < V(\lambda)$ **then** /* improved solution */
  improved := *.true.* /* set improved */
  $V(\lambda) := f$ /* update $V(\lambda)$ */
  $H := H \cup S$ /* add S to H */
end if
end

Algorithm 1. Evaluation.

constraints relating to the upper limits δ_i are satisfied. Note here that Algorithm 1 can be viewed as a heuristic for solving the QP (Eqs. (16)–(20)) with a given set of K assets. Whilst, obviously, this QP could be solved optimally this would not lead to a computationally efficient heuristic (examining as we do a large number of possible solutions).

Note here that the strategy adopted in Algorithm 1, namely to change (if possible) the GA solution into a feasible solution for the original problem, is a strategy that we have used, with success, in our previous GA work [37,38].

We used a population size of 100. Parents were chosen by binary tournament selection which works by forming two pools of individuals, each consisting of two individuals drawn from the population randomly. The individuals with the best fitness, one taken from each of the two tournament pools, are chosen to be parents.

Children in our GA heuristic are generated by uniform crossover. In uniform crossover two parents have a single child. If an asset i is present in both parents it is present in the child (with an associated value s_i randomly chosen from one or other parent). If an asset i is present in just one parent it has probability 0.5 of being present in the child. Children are also subject to mutation, multiplying by 0.9 or 1.1 (chosen with equal probability) the value $(\varepsilon_i + s_i)$ of a randomly selected asset i. This mutation corresponds to decreasing or increasing this value by 10%.

We used a steady-state population replacement strategy. With this strategy each new child is placed in the population as soon as it is generated (replacing a suitably chosen member of the population). In our GA we choose to replace the member of the population with the worst objective function value.

Our complete GA heuristic is shown in pseudocode in Algorithm 2.

4.3. Tabu search

Tabu search (TS) is a local search heuristic due to Glover [39] and Hansen [40]. In TS the fundamental concept is that of a "move", a systematic operator that, given a single starting solution, generates a number of other possible solutions. In local search terms these other solutions are the "neighbourhood" of the single starting solution. Note here that these solutions may, or may not, be feasible. From the neighbourhood the best solution is chosen to become the new starting solution for the next iteration and the process repeats. This "best" solution may either be the first improving solution encountered as the move operator enumerates the neighbourhood, or it may be based upon complete enumeration of the neighbourhood.

In order to prevent cycling a list of "tabu moves" is employed. Typically this list prohibits certain moves which would lead to the revisiting of a previously encountered starting solution. This list of tabu moves is updated as the algorithm proceeds so that a move just added to the tabu list is removed from the tabu list after a certain number of iterations (the "tabu tenure") have passed. It is common to allow tabu moves to be made however if they lead to an improved feasible solution (an "aspiration criteria"). A more comprehensive overview of TS can be found in [33,41,42].

Glover et al. [43] applied TS to a portfolio optimisation problem involving rebalancing a portfolio to maintain (over time) a fixed proportion in each asset category. They used a scenario approach to model possible future asset returns. Computational results were presented for one example problem.

E is the number of λ values we wish to examine
P is the population
S^*, S^{**} are two solutions (parents) selected from the population to mate
C is the offspring of S^* and S^{**} and consists of
  R the set of K distinct assets in C
  c_i the value for asset $i \in R$
A^* is a set of assets that are in the parents, but are not in the child (together with their associated values)
T^* is the number of iterations

begin
$H := \emptyset$
for $e := 1$ to E **do**
$\lambda := (e-1)/(E-1)$ /* examine E λ values equally spaced in $[0,1]$ */
$V(\lambda) := \infty$
initialise $P := \{S_1, \ldots, S_{100}\}$ /* random initialisation, exactly K assets in each S_p */
evaluate $(S_p, \lambda, f(S_p), V, improved, H)\ p = 1, \ldots, 100$ /* evaluate the solutions */
for $t := 1$ to T^* **do** /* T^* iterations in all */
  select S^*, $S^{**} \in P$ by binary tournament method
  crossover $C := (S^*, S^{**})$ by uniform method
  $A^* := S^* \cup S^{**} - C$ /* find assets in parents, not in child */
  randomly choose $i \in R$ and $m = 1$ or 2
  if $m = 1$ **then** $c_i := 0.9(\varepsilon_i + s_i) - \varepsilon_i$ **else** $c_i := 1.1(\varepsilon_i + s_i) - \varepsilon_i$ /* mutation */
  if $c_i < 0$ then $R := R - [i]$ /* delete asset i if necessary */
  /* ensure C has exactly K assets */
  while $|R| > K$ delete the asset $j \in R$ with the smallest c_j from R
  while $|R| < K$ **do** /* add asset from parent if possible */
  if $|A^*| \neq 0$ **then**
    add to C a randomly chosen asset j from A^* /* add asset */
    $A^* := A^* - [j]$
  else
    add to C a randomly chosen asset $j \notin R$ and set $c_j := 0$
  end if
  end while
  /* C now has exactly K assets */
  evaluate$(C, \lambda, f(C), V, improved, H)$
  find j such that $f(S_j) = \max[f(S_p) | p = 1, \ldots, 100]$ /* find worse population member */
  $S_j := C$ /* replace worse member with child */
end for
end for
end

Algorithm 2. GA heuristic.

4.4. Tabu search heuristic

In our TS heuristic we used the same solution representation as in our GA heuristic, as well as Algorithm 1 in order to try and ensure that the evaluated solution was feasible.

The procedure first randomly generates 1000 solutions. Each of these solutions consisted of a set Q of K randomly generated distinct assets. Associated with each asset $i \in Q$ was a value s_i randomly generated from $(0, 1)$. Algorithm 1 was then used to evaluate each of these solutions. The best solution found was used as a starting point.

The move operator corresponds to taking all assets present in the portfolio of K assets and multiplying their values by 0.9 and 1.1. This means that the number of neighbours which we need to evaluate is $2K$. The tabu list is a matrix of $2N$ integer values which indicates for each of the N assets whether a particular move (multiplying by 0.9 or 1.1) is currently tabu or not.

Our complete TS heuristic is shown in pseudocode in Algorithm 3.

4.5. Simulated annealing

Simulated annealing (SA) originated in an algorithm to simulate the cooling of material in a heat bath [44] but its use for optimisation problems originated with Kirkpatrick et al. [45] and Cerny [46].

SA has much in common with TS in that they both examine potential moves from a single starting solution. SA incorporates a statistical component in that moves to worse solutions are accepted with a specified probability that decreases over the course of the algorithm.

This probability is related to what is known as the "temperature". More precisely, a move that worsens the objective value by Δ is accepted with a probability proportional to $e^{-\Delta/T}$, where T is the current temperature. The higher the temperature T, the higher the probability of accepting the move. Hence this probability decreases as the temperature decreases.

In SA the temperature is reduced over the course of the algorithm according to a "cooling schedule" which specifies the initial temperature and the rate at which temperature decreases. A common cooling schedule is to reduce the temperature T by a constant factor α $(0 < \alpha < 1)$ using $T = \alpha T$ at regular intervals. A more comprehensive overview of SA can be found in [33,41].

Note here that, as far as we are aware, there have been no applications of SA to portfolio optimisation reported in the literature.

4.6. Simulated annealing heuristic

Our SA heuristic is similar to our TS heuristic and can be seen in pseudocode in Algorithm 4. The initial temperature is derived from the objective value of the initial starting solution and α is set equal to 0.95. In the computational results reported later we did $2N$ iterations at the same temperature.

4.7. Comparison

In order to better compare the GA, TS and SA heuristics presented above, Table 2 compares these algorithms with respect to a number of features.

S^* is the current solution and consists of:
  Q the set of K distinct assets in S^*
  s_i the current value for asset $i \in Q$
C is the current neighbour of S^* and consists of
  R the set of K distinct assets in C
  c_i the value for asset $i \in R$
S^{**} is the best neighbour of S^* that does not involve a tabu move
V^{**} the objective function value for S^{**}
L_{im} the tabu value for asset i move m ($m = 1$ multiply by $0.9, m = 2$ multiply by 1.1), where $L_{im} = 0$ if the move is not tabu
L^* is the tabu tenure value for a move that has just been made tabu
opp(m) is the opposite move to move m (so opp(1) = 2 and opp(2) = 1)

begin
$H := \emptyset$
for $e := 1$ to E **do**
$\lambda := (e-1)/(E-1)$ /* examine E λ values equally spaced in $[0,1]$ */
$V(\lambda) := \infty$
initialise $P := \{S_1, \ldots, S_{1000}\}$ /* random initialisation, exactly K assets in each S_p */
evaluate(S_p, λ, $f(S_p)$, V, *improved*, H) $p = 1, \ldots, 1000$ /* evaluate the solutions */
find j such that $f(S_j) = V(\lambda)$ and set $S^* = S_j$ /* starting solution */
$L_{im} := 0$ $\forall i, m$ /* initialise tabu values */
$L^* := 7$ /* set tabu tenure for new move to 7 */
for $t := 1$ to T^* **do** /* T^* iterations in all */
$V^{**} := \infty$ /* initialise best neighbour value */
  for $i \in Q$ and $m := 1$ to 2 **do** /* examine neighbours of S^* */
  $C := S^*$
    /* C is neighbour of S^* corresponding to move m for asset i */
  if $m = 1$ **then** $c_i := 0.9(\varepsilon_i + s_i) - \varepsilon_i$ **else** $c_i := 1.1(\varepsilon_i + s_i) - \varepsilon_i$ /* move */
  if $c_i < 0$ **then** randomly select $j \notin R$ and set $R := R \cup [j] - [i]$, $c_j := 0$ /* add asset */
    /* C now has exactly K assets */
  evaluate(C, λ, $f(C)$, V, *improved*, H)
  if *improved* **then** $L_{im} = 0$ /* aspiration - make the move non-tabu */
  if $L_{im} = 0$ and $f(C) < V^{**}$ **then** /* improved non-tabu neighbour */
    $V^{**} := f(C)$ /* update V^{**} */
    $S^{**} := C$ /* update S^{**} */
    $k := i$ /* record move */
    $n := m$
  end if
  end for
if $V^{**} = \infty$ **then**
  finished with current value of λ /* no non-tabu moves */
else
  $S^* := S^{**}$ /* take the best neighbour found */
  $L_{im} := \max[0, L_{im} - 1]$ $\forall i, m$ /* reduce all tabu tenures by one */
  $L_{k,\mathrm{opp}(n)} := L^*$ /* tabu opposite move */
end if
end for
end for
end

Algorithm 3. TS heuristic.

```
begin
H := ∅
for e := 1 to E do
λ := (e − 1)/(E − 1)                          /* examine E λ values equally spaced in [0, 1] */
V(λ) := ∞
initialise P := {S_1, ..., S_1000}            /* random initialisation, exactly K assets in each S_p */
evaluate(S_p, λ, f(S_p), V, improved, H) p = 1, ..., 1000   /* evaluate the solutions */
find j such that f(S_j) = V(λ) and set S* = S_j             /* starting solution */
T := |V(λ)/10|                                /* initialise SA parameters */
α := 0.95
for t := 1 to T* do                           /* T* iterations in all */
for a specified number of iterations at the same temperature do
   randomly select an asset i∈Q and a move m = 1 or 2
   C := S*
                                              /* C is neighbour of S* corresponding to move m for asset i */
   if m = 1 then c_i := 0.9(ε_i + s_i) − ε_i else c_i := 1.1(ε_i + s_i) − ε_i   /* move */
   if c_i < 0 then randomly select j∉R and set R := R∪[j] − [i], c_j := 0       /* add asset */
                                              /* C now has exactly K assets */
   evaluate (C, λ, f(C), V, improved, H)
   if f(C) < f(S*) then                       /* C better than current solution S* */
      S* := C                                 /* change S* */
   else
      r := a random number drawn from [0, 1]
      if r < exp[ − (f(C) − f(S*))/T] then    /* check for accept worst solution */
         S* := C                              /* update S* */
      end if
   end if
end for
T := αT                                       /* reduce temperature */
end for
end for
end
```

Algorithm 4. SA heuristic.

5. Computational results

In this section we present computational results for the three heuristic algorithms we have presented above for finding the CCEF. Note here that all of the computational results presented in this section are for our heuristics as coded in FORTRAN and run on a Silicon Graphics Indigo workstation (R4000, 100 MHz, 48 MB main memory).

Table 2
Algorithmic features

Feature	GA	TS	SA
λ values	E λ values equally spaced in $[0,1]$	E λ values equally spaced in $[0,1]$	E λ values equally spaced in $[0,1]$
Initial solution(s)	Randomly generated	Randomly generated	Randomly generated
New solution generation	Parent selection, uniform crossover and mutation	Neighbour of current solution	Neighbour of current solution
Action if new solution has $> K$ assets	Delete smallest assets	–	–
Action if new solution has $< K$ assets	Add assets from parents if possible, else add randomly generated assets	Add randomly generated assets	Add randomly generated assets
New solution evaluation	Algorithm 1	Algorithm 1	Algorithm 1
Accept new solution	Always	If best non-tabu neighbour	If improved solution or satisfies probabilistic criterion
Action with accepted solution	Replaces member of the population	Replaces current solution	Replaces current solution
Stopping criterion	Number of iterations	Number of iterations	Number of iterations

5.1. Test data sets

To test our heuristics we constructed five test data sets by considering the stocks involved in five different capital market indices drawn from around the world. Specifically we considered the Hang Seng (Hong Kong), DAX 100 (Germany), FTSE 100 (UK), S&P 100 (USA) and Nikkei 225 (Japan). We used DATASTREAM to obtain weekly price data from March 1992 to September 1997 for the stocks in these indices. Stocks with missing values were dropped. We had 291 values for each stock from which to calculate (weekly) returns and covariances and the size of our five test problems ranged from $N = 31$ (Hang Seng) to $N = 225$ (Nikkei).

All of the test problems solved in this paper, but with the identity of each stock disguised, are publically available from OR-Library [47,48] email the message *portinfo* to *o.rlibrary@ic.ac.uk* or see *http://mscmga.ms.ic.ac.uk/jeb/orlib/portinfo.html*.

5.2. Unconstrained efficient frontier

In order to initially test the effectiveness of our GA, TS and SA heuristics we first used them to find the UEF. Adopting this approach has the advantage that (as mentioned in Section 2 above) the UEF can be exactly calculated via QP so our heuristic results can be compared with benchmark optimal solutions.

The reason for doing this comparison is simply that for the CCEF we have no way of calculating the exact efficient frontier for problems of the size we are considering, and hence no way of benchmarking our heuristics against the exact solution. We would anticipate that, unless our heuristics are able to find the UEF to a reasonable degree of accuracy, they are unlikely to be able to find the CCEF.

Note here that no amendments are required to our heuristics to calculate the UEF (simply set $K = N$, $\varepsilon_i = 0$ and $\delta_i = 1$ $(i = 1, \ldots, N)$).

5.2.1. UEF calculation

In order to calculate, to a reasonable degree of accuracy, the exact UEF we used the Numerical Algorithms Group (NAG) library routine E04NAF to solve the QP (Eqs. (1)–(4)) for a large number of different return (R^*, Eq. (2)) values, taken between the return value associated with the minimum risk and the return value associated with the maximum return. In the computational results presented below we took 2000 return (R^*) values, hence calculating 2000 distinct points on the continuous exact UEF. For adjacent points we used linear interpolation to approximate the exact UEF. This use of linear interpolation is simply a convenient computational device to approximate the (continuous) UEF from a finite (but reasonably large) number of distinct points on the UEF.

In order to compare our heuristic results against the exact frontier we:

(a) took the portfolios associated with the values $V(\lambda)$ as given by our heuristics (recall here that $V(\lambda)$ is the best objective function value found for weighting parameter λ, see Algorithm 1)
(b) computed the percentage deviation of each portfolio from the (linearly interpolated) exact UEF.

This calculation of the percentage deviation of each portfolio from the (linearly interpolated) exact UEF is not as trivial as it might at first sight appear and we consider this below.

5.2.2. Percentage deviation calculation

There are two basic issues:

(a) how we measure the percentage deviation ("distance") of a portfolio from a continuous efficient frontier; and
(b) how we quantify this percentage deviation in the case of a linearly interpolated efficient frontier.

To illustrate the first of these issues consider Fig. 8 which shows a continuous efficient frontier curve and a point (portfolio) located near it. Pictorially it appears that if we look at the "distance" of this portfolio from the efficient frontier by considering the y value (return) as fixed the portfolio is some distance from the frontier, as indicated by the horizontal line connecting the portfolio to the frontier in Fig. 8. However if we consider the x value (risk) as fixed the portfolio is nearer the frontier, as indicated by the vertical line connecting the portfolio to the frontier in Fig. 8.

Given this issue of in which direction do we look "horizontally or vertically?" we decided to resolve the issue by looking in both directions, calculating (as below) a percentage deviation in each direction and taking the minimum of these two values as the percentage deviation error measure associated with a portfolio.

With respect to the second issue mentioned above, quantifying the percentage deviation in the case of a linearly interpolated efficient frontier we, in order to work in commensurate units, used the portfolio standard deviation (rather than variance) in computing percentage deviation.

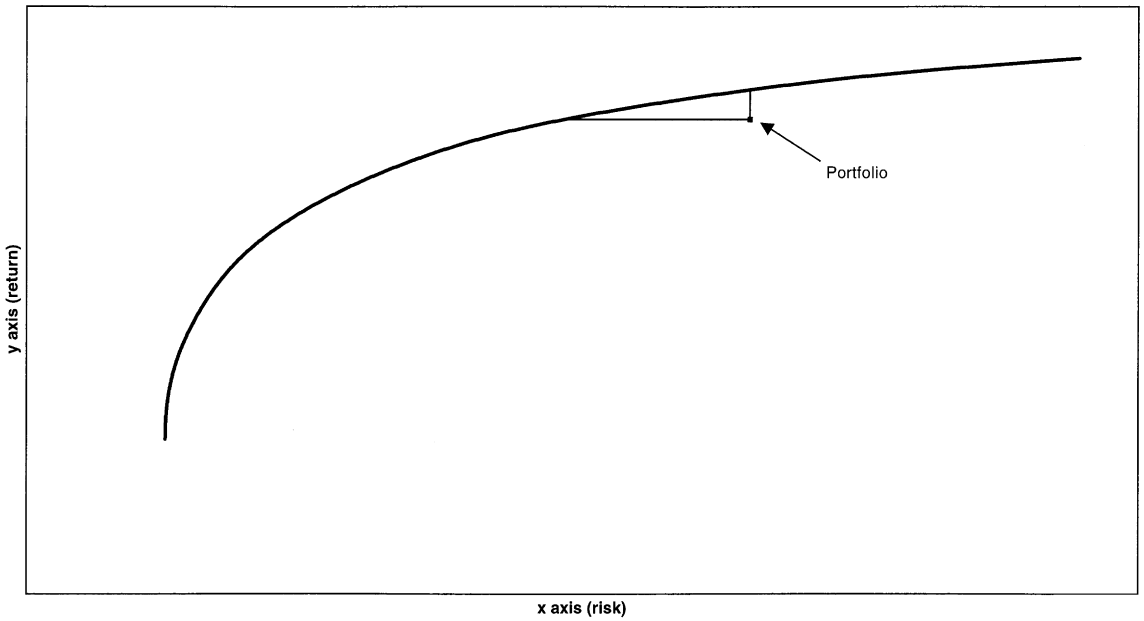

Fig. 8. Error calculation.

Hence let (x_i, y_i) be the discrete (x-coordinate: standard deviation, y-coordinate: return) values on our UEF. For a portfolio with (x^*, y^*) let j correspond to $y_j = \min[y_i \,|\, y_i \geqslant y^*]$ and k correspond to $y_k = \max[y_i \,|\, y_i \leqslant y^*]$ (i.e. y_j and y_k are the closest y-coordinates bracketing y^*). Simple geometry enables us to say that the value x^{**} associated with the x-direction linearly interpolated point on the UEF with $y = y^*$ (i.e. looking horizontally) is $x^{**} = x_k + (x_j - x_k)[(y^* - y_k)/(y_j - y_k)]$. A convenient percentage deviation error measure for this direction is then $|100(x^* - x^{**})/x^{**}|$ (note here that no value is calculated if either j or k do not exist).

To derive an expression for linear interpolation in the y-direction: let j correspond to $x_j = \min[x_i \,|\, x_i \geqslant x^*]$ and k correspond to $x_k = \max[x_i \,|\, x_i \leqslant x^*]$ (i.e. x_j and x_k are the closest x-coordinates bracketing x^*). Then y^{**} associated with the y-direction linearly interpolated point on the UEF with $x = x^*$ (i.e. looking vertically) is $y^{**} = y_k + (y_j - y_k)[(x^* - x_k)/(x_j - x_k)]$. A convenient percentage deviation error measure for this direction is then $|100(y^* - y^{**})/y^{**}|$ (note here that no value is calculated if either j or k do not exist).

As mentioned above, in the computational results reported later we take as the percentage deviation error measure for each (x^*, y^*) the minimum of the x-direction, y-direction percentage deviations.

5.2.3. Results

With regard to all the computational results reported in this paper we examined 50 different λ values ($E = 50$, Algorithms 2–4). With regard to the number of iterations T^* (see Algorithms 2–4) we used $T^* = 1000N$ for the GA heuristic, $T^* = 500(N/K)$ for the TS heuristic and $T^* = 500$ for

the SA heuristic. These values mean that (excluding initialisation) each heuristic evaluates exactly $1000N$ solutions using Algorithm 1 for each value of λ.

The results for the unconstrained efficient frontier are shown in Table 3. In that table we show, for each of our five data sets and each of our three heuristics:

(a) the median percentage error,
(b) the mean percentage error,
(c) the total computer time in seconds.

Note here that all the computer times presented in this paper exclude the time needed to calculate the error measures.

It is clear from Table 3 that our GA heuristic is best able to approximate the UEF with an average mean percentage error of 0.0114%, the SA heuristic next best with an average mean percentage error of 0.4675%, whilst the TS heuristic has an average mean percentage error of 5.6158%. For the SA heuristic the median error is noticeably smaller than the mean error, indicating a skewed error distribution with a higher probability of large errors. For the GA and TS heuristics the median and mean errors indicate a reasonably symmetric error distribution.

5.3. Cardinality constrained efficient frontier

With regard to finding the CCEF since, as discussed in Section 3 above, this frontier has portions which are invisible to an exact algorithm based on the standard λ weighting scheme we did not feel

Table 3
Results for the unconstrained efficient frontier

Index	Number of assets (N)		GA heuristic	TS heuristic	SA heuristic
Hang Seng	31	Median percentage error	0.0165	1.0718	0.0160
		Mean percentage error	0.0202	0.8973	0.1129
		Time (s)	621	469	476
DAX	85	Median percentage error	0.0123	2.7816	0.0033
		Mean percentage error	0.0136	3.5645	0.0394
		Time (s)	10,332	9546	9412
FTSE	89	Median percentage error	0.0029	3.0238	0.0426
		Mean percentage error	0.0063	3.2731	0.2012
		Time (s)	11,672	10,698	10,928
S&P	98	Median percentage error	0.0085	4.2780	0.0142
		Mean percentage error	0.0084	4.4280	0.2158
		Time (s)	15,879	14,517	14,367
Nikkei	225	Median percentage error	0.0084	14.2668	0.8107
		Mean percentage error	0.0085	15.9163	1.7681
		Time (s)	227,220	210,929	281,588
	Average	Mean percentage error	0.0114	5.6158	0.4675

it appropriate to judge the effectiveness of our heuristics solely using the portfolios associated with the best solution $V(\lambda)$ found for each value of the weighting parameter λ. This contrasts with Section 5.2 above where these portfolios were appropriate as the UEF can be traced using the standard λ weighting scheme.

Instead we also judge the effectiveness of our heuristics by taking the set H, which is all the (improved) solutions found during the course of each of our heuristics (see Algorithms 2–4). In other words, by using the history of solutions found by each of our heuristics we can gain useful information about those portions of the CCEF that are invisible to an exact approach based upon the standard λ weighting scheme.

This set H will plainly contain a number (probably a large number) of dominated solutions. However it is a simple matter to extract from H the subset of (undominated) efficient solutions using:

(a) let (r_i, v_i) be the (return, risk) values for solution $i \in H$
(b) $\forall i \in H$ if there exists $j \in H$ $(j \neq i)$ such that $r_j \geqslant r_i$ and $v_j \leqslant v_i$ then delete i from H (i.e. set $H = H - [i]$) as i is dominated by j (j has a better return for less risk)
(c) H is now the set of (undominated) efficient solutions.

This procedure can be efficiently implemented by sorting H appropriately.

Once the set H has been processed as above we can obtain an *overestimate* of the error associated with our heuristic algorithms by comparing H against the (linearly interpolated) UEF in exactly the same manner as was done in Section 5.2 above.

Note here however that, as all of our algorithms are heuristics, we can provide no guarantees as to the quality (deviation from the exact CCEF) of any particular portfolio in the set H. In particular a portfolio in H could be on the exact CCEF or could be dominated by another portfolio (not in H) which is on the exact CCEF.

The results for our heuristic algorithms with $K = 10$ and $\varepsilon_i = 0.01, \delta_i = 1$ $(i = 1, \ldots, N)$ are shown in Table 4. In that table we show, for each of our five data sets and each of our three heuristics:

(a) the median percentage error,
(b) the mean percentage error,
(c) the number of (undominated) efficient points,
(d) the total computer time in seconds.

Note that for the column labelled V we did not eliminate from $V(\lambda)$ any dominated solutions.

It can be seen from Table 4 that over our five test data sets no one of our heuristic algorithms is uniformly dominant. Although the GA heuristic performs better than the SA heuristic, which in turn performs better than the TS heuristic, the differences are not nearly as marked as they were for the UEF (Table 3). For some data sets there are considerable differences in the percentage error measures, indicating that the algorithms give significantly different results.

Hence we would envisage that a sensible approach to the cardinality constrained portfolio optimisation problem in practice would be to run all three heuristics and to pool their results in an obvious fashion (i.e. combine the three sets of undominated points given by the three algorithms together into one set and eliminate from this new set those points which are dominated). These pooled results are also shown in Table 4.

Table 4
Results for the cardinality constrained efficient frontier

Index	Number of assets (N)		GA heuristic		TS heuristic		SA heuristic		Pooled results
			V	*H*	*V*	*H*	*V*	*H*	*H*
Hang Seng	31	Median percentage error	1.2181	1.1819	1.2181	1.1992	1.2181	1.2082	1.1899
		Mean percentage error	1.0974	0.9457	1.1217	0.9908	1.0957	0.9892	0.9332
		Number of efficient points	—	1317	—	1268	—	1003	2491
		Time (s)	172		74		79		325
DAX	85	Median percentage error	2.5466	2.1262	2.6380	2.5383	2.5661	2.4675	2.4626
		Mean percentage error	2.5424	1.9515	3.3049	3.0635	2.9297	2.4299	2.1927
		Number of efficient points	—	1270	—	1467	—	1135	2703
		Time (s)	544		199		210		953
FTSE	89	Median percentage error	1.0841	0.5938	1.0841	0.6361	1.0841	0.7137	0.5960
		Mean percentage error	1.1076	0.8784	1.6080	1.3908	1.4623	1.1341	0.7790
		Number of efficient points	—	1482	—	1301	—	1183	2538
		Time (s)	573		246		215		1034
S&P	98	Median percentage error	1.2244	1.1447	1.2882	1.1487	1.1823	1.1288	1.0686
		Mean percentage error	1.9328	1.7157	3.3092	3.1678	3.0696	2.6970	1.3106
		Number of efficient points	—	1560	—	1587	—	1284	2759
		Time (s)	638		225		242		1105
Nikkei	225	Median percentage error	0.6133	0.6062	0.6093	0.5914	0.6066	0.6292	0.5844
		Mean percentage error	0.7961	0.6431	0.8975	0.8981	0.6732	0.6370	0.5690
		Number of efficient points	—	1823	—	1701	—	1655	3648
		Time (s)	1964		545		553		3062
Average		Mean percentage error	1.4953	1.2269	2.0483	1.9022	1.8461	1.5774	1.1569

Note: Number of assets in the portfolio $K = 10$, minimum proportion of any asset that must be held (if any is held) is 0.01; maximum proportion that can be held of any asset (if any is held) is 1.

V represents the portfolios associated with the best objective function value found for each of the 50 values of the weighting parameter λ examined.

H represents the set of portfolios encountered over the course of the algorithm (dominated portfolios having been removed from *H*).

Pooled results are the combination of the three *H* sets for GA, TS and SA (again with dominated portfolios being removed).

Note here that we stated before (Section 4) that using our heuristics it is possible to gain information about those portions of the CCEF that would be invisible to an exact approach based upon weighting. Fig. 7, presented before for the four asset example given in Table 1, has in fact been plotted using the pooled results from the sets *H* for each of the three heuristics.

5.4. Discussion

There are a number of points which can be made with respect to Table 4 and these are discussed in this section.

In Table 4 we have shown results for five data sets, one particular value of K and one set of values for ε_i and δ_i. Plainly for different data sets/values the results will be different. However we believe that our key point, namely that it is important to use a number of heuristics and to pool their results, is established.

As stated above, the percentage errors given in Table 4 are overestimates of the errors associated with each heuristic as they are derived from the UEF, which dominates the CCEF. One point that is important however is the distribution of these efficient points along this frontier. At the extreme a heuristic could obtain a low percentage error by finding just a few points on the CCEF near the UEF. In order to decide which portfolio of assets to buy however, a decision-maker examining the efficient frontier needs a good distribution of points over this frontier in order to make an informed decision.

This distribution of points over the frontier is illustrated numerically in Table 5. In that table we give, for each data set, for each of the three heuristics, the number of efficient points that they individually contribute to the pooled set of efficient points. For example, for the Hang Seng data it can be seen that of the 2491 pooled efficient points 953 (38.3%) are contributed by the GA heuristic, 860 (34.5%) are contributed by the TS heuristic and 733 (29.4%) are contributed by the SA heuristic (note that some points are duplicated across the heuristics). On average, across all five data sets, 39.5% are contributed by the GA heuristic, 29.6% by the TS heuristic and 32.3% by the SA heuristic. We feel that this is another argument in favor of pooling heuristic results.

Each of the undominated points found by our algorithms is a portfolio with K assets with an associated return and an associated λ. Hence there are a number of possible ways to improve our solutions by taking the assets in/out of the portfolio as fixed and:

(a) solving the weighted problem (Eqs. (16)–(20)) with the given value of λ; or
(b) minimizing the risk associated with the portfolio at its given level of return; or
(c) tracing the efficient frontier associated with the given set of K assets.

In the computational results reported here we did not however adopt any of these approaches.

5.5. Tradeoff

As noted in Section 2.4 above we believe that one merit of the heuristics given in this paper is that the decision-maker will be faced with a different CCEF for each value of K and must explicitly consider the tradeoffs between risk, return, and number of assets in the portfolio in deciding which

Table 5
Contributions to the constrained efficient frontier

Index	Number of assets (N)	Pooled results H	GA heuristic	TS heuristic	SA heuristic
Hang Seng	31	2491	953 (38.3%)	860 (34.5%)	733 (29.4%)
DAX	85	2703	1046 (38.7%)	858 (31.7%)	844 (31.2%)
FTSE	89	2538	1053 (41.5%)	616 (24.3%)	913 (36.0%)
S&P	98	2759	1202 (43.6%)	787 (28.5%)	795 (28.8%)
Nikkei	225	3648	1297 (35.6%)	1065 (29.2%)	1309 (35.9%)
Average			39.5%	29.6%	32.3%

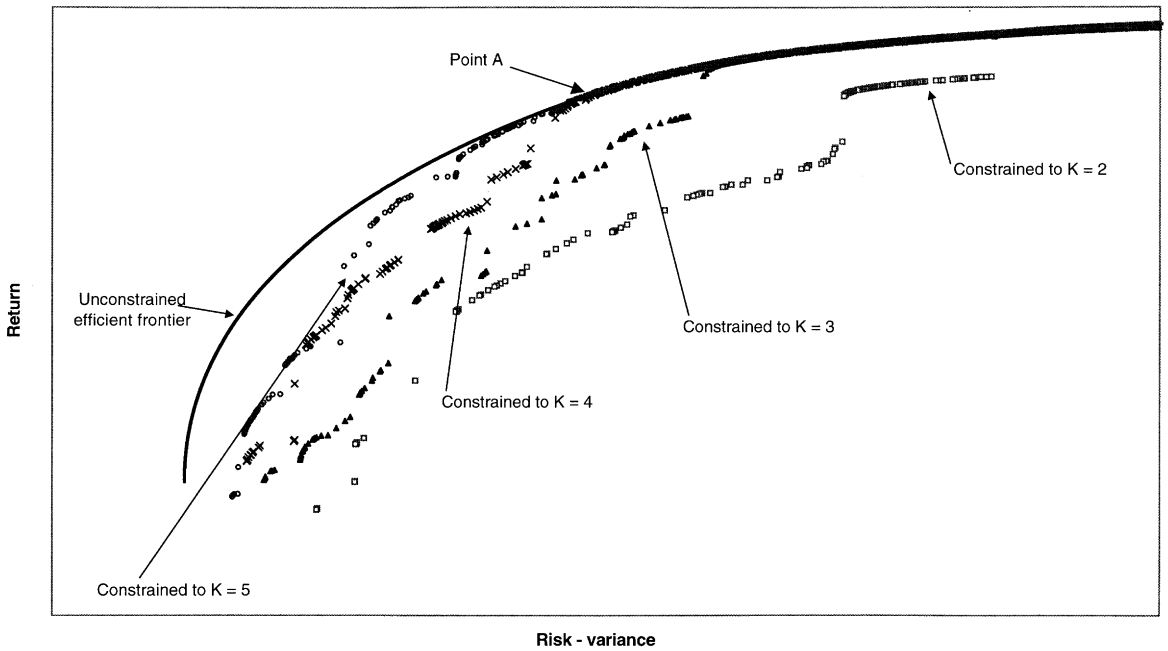

Fig. 9. DAX tradeoff curves.

portfolio to adopt. This is illustrated in Fig. 9 for the DAX data set (which has the highest mean (pooled) error in Table 4) using the pooled results for all three heuristics. In that figure we have plotted the curves for $K = 2, 3, 4$ and 5 ($\varepsilon_i = 0.01$ and $\delta_i = 1$ as before), as well as the UEF. It will be seen that, as we would expect (given the values of ε_i and δ_i), as K increases we approach the UEF.

A decision-maker presented with Fig. 9 now has an explicit pictorial representation of the possibilities open to them, and the tradeoffs involved. For example, suppose their attitude to risk and return is such that they prefer a high return, say a return corresponding to the point marked A in Fig. 9. Graphically it is clear that such a return (for the same risk) can effectively be obtained by a portfolio containing four or five assets. If the decision-maker were to prefer a portfolio containing less assets then no such portfolio containing three assets appears to exist with the same return (note the discontinuity in the CCEF for $K = 3$ at this return level). There is a portfolio giving the same return containing exactly two assets but it requires a substantial increase in risk.

We would comment here that, as far as we are aware, this paper is the first in the literature to present algorithms that allow the information shown in Fig. 9 to be produced in a computationally effective manner.

6. Conclusions

In this paper we have considered the problem of calculating the efficient frontier for the cardinality constrained portfolio optimization problem. We highlighted the differences that arise in the shape of this efficient frontier as compared with the unconstrained efficient frontier.

Computational results were presented for three heuristic algorithms based upon genetic algorithms, tabu search and simulated annealing for finding the cardinality constrained efficient frontier. These indicated that a sensible approach was to pool their results.

References

[1] Markowitz H. Portfolio selection. Journal of Finance 1952;7:77–91.
[2] Markowitz HM. Portfolio selection: efficient diversification of investments. New York: Wiley, 1959.
[3] Mills TC. Stylized facts on the temporal and distributional properties of daily FT-SE returns. Applied Financial Economics 1997;7:599–604.
[4] Dahl H, Meeraus A, Zenios SA. Some financial optimization models: I risk management. In: Zenios SA, editor. Financial optimization. 1993. p. 3–36 Cambridge: Cambridge University Press.
[5] Elton EJ, Gruber MJ. Modern portfolio theory and investment analysis. New York: Wiley, 1995.
[6] Rudd A, Rosenberg B. Realistic portfolio optimization. In: Elton EJ, Gruber MJ editors. Portfolio theory, 25 years after. TIMS Studies in the Management Sciences, vol. 11. Amsterdam: North-Holland, 1979. p. 21–46.
[7] Konno H, Yamazaki H. Mean-absolute deviation portfolio optimization model and its applications to Tokyo Stock Market. Management Science 1991;37:519–31.
[8] Konno H. Piecewise linear risk-function and portfolio optimization. Journal of the Operations Research Society of Japan 1990;33:139–56.
[9] Simaan Y. Estimation risk in portfolio selection: the mean variance model versus the mean absolute deviation model. Management Science 1997;43:1437–46.
[10] Konno H, Shirakawa H, Yamazaki H. A mean-absolute deviation-skewness portfolio optimization model. Annals of Operations Research 1993;45:205–20.
[11] Konno H, Suzuki KI. A mean-variance-skewness portfolio optimization model. Journal of the Operations Research Society of Japan 1995;38:173–87.
[12] Feiring BR, Wong WL, Poon M, Chan YC. Portfolio selection in downside risk optimization approach – application to the Hong Kong stock market. International Journal of Systems Science 1994;25:1921–9.
[13] Adcock CJ, Meade N. A simple algorithm to incorporate transaction costs in quadratic optimisation. European Journal of Operational Research 1994;79:85–94.
[14] Mulvey JM, Vladimirou H. Stochastic network programming for financial planning problems. Management Science 1992;38:1642–64.
[15] Yoshimoto A. The mean-variance approach to portfolio optimization subject to transaction costs. Journal of the Operations Research Society of Japan 1996;39:99–117.
[16] Bienstock D. Computational study of a family of mixed-integer quadratic programming problems. In: Balas E, Clausen J, editors. Integer Programming and Combinatorial Optimization: 4th International IPCO Conference, Copenhagen, Denmark, May 1995 Proceedings, Lecture Notes in Computer Science, vol. 920. Berlin: Springer, 1995.
[17] Bienstock D. Computational study of a family of mixed-integer quadratic programming problems. Mathematical Programming 1996;74:121–40.
[18] GAMS Development Corporation. http://www.gams.com/modlib/libhtml/qp4.htm, 1999.
[19] Shiryaev AN. Probability, 2nd ed. Berlin: Springer, 1996. p. 235.
[20] Wilks SS. Mathematical statistics. New York: Wiley, 1962. p. 82.
[21] Borchers B, Mitchell JE. An improved branch and bound algorithm for mixed integer nonlinear programs. Computers & Operations Research 1994;21:359–67.
[22] Borchers B, Mitchell JE. A computational comparison of branch and bound and outer approximation algorithms for 0–1 mixed integer nonlinear programs. Computers & Operations Research 1997;24:699–701.
[23] Floudas CA. Nonlinear and mixed-integer optimization: fundamentals and applications. Oxford: Oxford University Press, 1995.
[24] Hansen P, Jaumard B, Mathon V. Constrained nonlinear 0–1 programming. ORSA Journal on Computing 1993;5:97–119.

[25] Lee EK, Mitchell JE. Computational experience of an interior-point SQP algorithm in a parallel branch-and-bound framework. Working paper available from the second author at Department of Mathematical Sciences, Rensselaer Polytechnic Institute, Troy, NY 12180-3590, USA, 1997.
[26] Speranza MG. A heuristic algorithm for a portfolio optimization model applied to the Milan stock market. Computers & Operations Research 1996;23:433–41.
[27] Mansini R, Speranza MG. Heuristic algorithms for the portfolio selection problem with minimum transaction lots. Working paper available from the second author at Dip. di Metodi Quantitativi, Universita di Brescia, C.da.S Chiara 48/b, 25122 Brescia, Italy, 1997.
[28] Kellerer H, Mansini R, Speranza MG. On selecting a portfolio with fixed costs and minimum transaction lots. Working paper available from the third author at Dip. di Metodi Quantitativi, Universita di Brescia, C.da.S Chiara 48/b, 25122 Brescia, Italy, 1997.
[29] Young MR. A minimax portfolio selection rule with linear programming solution. Management Science 1998;44:673–83.
[30] Holland JH. Adaptation in natural and artificial systems: an introductory analysis with applications to biology, control, and artificial intelligence. Michigan: University of Michigan Press, 1975.
[31] Bäck T, Fogel DB, Michalewicz Z, editors. Handbook of evolutionary computation. Oxford: Oxford University Press, 1997.
[32] Mitchell M. An introduction to genetic algorithms. Cambridge, MA: MIT Press, 1996.
[33] Reeves CR, editor. Modern heuristic techniques for combinatorial problems. Oxford: Blackwell Scientific Publications, 1993.
[34] Reeves CR. Genetic algorithms for the operations researcher. INFORMS Journal on Computing 1997;9:231–50.
[35] Arnone S, Loraschi A, Tettamanzi A. A genetic approach to portfolio selection. Neural Network World 1993;6:597–604.
[36] Loraschi A, Tettamanzi A, Tomassini M, Verda P. Distributed genetic algorithms with an application to portfolio selection problems. In: Pearson DW, Steele NC, Albrecht RF, editors. Artificial neural nets and genetic algorithms, 1995. p. 384–87.
[37] Beasley JE, Chu PC. A genetic algorithm for the set covering problem. European Journal of Operational Research 1996;94:392–404.
[38] Chu PC, Beasley JE. A genetic algorithm for the multidimensional knapsack problem. Journal of Heuristics 1998;4:63–86.
[39] Glover F. Future paths for integer programming and links to artificial intelligence. Computers & Operations Research 1986;13:533–49.
[40] Hansen P. The steepest ascent mildest descent heuristic for combinatorial programming. Presented at the Congress on Numerical Methods in Combinatorial Optimization, Capri, Italy, 1986.
[41] Aarts EHL, Lenstra JK, editors. Local search in combinatorial optimization. New York: Wiley, 1997.
[42] Glover FW, Laguna M. Tabu search. Dordrecht: Kluwer Academic Publishers, 1997.
[43] Glover F, Mulvey JM, Hoyland K. Solving dynamic stochastic control problems in finance using tabu search with variable scaling. In: Osman IH, Kelly JP, editors. Meta-heuristics: theory & applications. Dordrecht: Kluwer Academic Publishers, 1996. p. 429–48.
[44] Metropolis N, Rosenbluth AW, Rosenbluth MN, Teller AH, Teller E. Equation of state calculations by fast computing machines. Journal of Chemical Physics 1953;21:1087–92.
[45] Kirkpatrick S, Gelatt CD, Vecchi MP. Optimization by simulated annealing. Science 1983;220:671–80.
[46] Cerny V. Thermodynamical approach to the travelling salesman problem: an efficient simulation algorithm. Journal of Optimization Theory and Applications 1985;45:41–51.
[47] Beasley JE. OR-Library: distributing test problems by electronic mail. Journal of the Operational Research Society 1990;41:1069–72.
[48] Beasley JE. Obtaining test problems via Internet. Journal of Global Optimization 1996;8:429–33.

T.J. Chang has a B.A. in Marine Engineering from the Chung Cheng Institute of Technology, Taiwan, and an M.Sc. in Operations Research from the Naval Postgraduate School, USA. He is currently a Ph.D. student in the Management School, Imperial College, London.

N. Meade is Reader in Management Science at Imperial College, London. His research is concerned with statistical model building and time series analysis with particular interest in applications in finance.

J.E. Beasley has a B.A. in Mathematics from Cambridge University and an M.Sc. and Ph.D. in Management Science from Imperial College, London. He is a member of the academic staff at Imperial College, London.

Y.M. Sharaiha has an M.Sc. in Engineering from UC Berkeley and a Ph.D. in Operations Research from Imperial College, London. He is currently working as a quantitative strategist at Morgan Stanley Dean Witter. His research interests and publications are in the areas of financial modelling and discrete optimisation.