Heuristics for cardinality constrained portfolio optimisation

T.-J. Chang

N. Meade

J.E. Beasley

Y.M. Sharaiha

May 1998

{t.j.chang, n.meade, j.beasley,y.sharaiha}@ic.ac.uk

The Management School

Imperial College

London SW7 2AZ

England

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.

Keywords: portfolio optimisation, efficient frontier

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 [27,28] 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 [30]). 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,...,N)

σ_{ij} be the covariance between assets i and j (i=1,...,N; j=1,...,N)

R be the desired expected return

Then the decision variables are:

w_i the proportion ($0 \le w_i \le 1$) held of asset i (i=1,...,N)

and using the standard Markowitz mean-variance approach [10,11,27,28,35] 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

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

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

$$0 \le w_i \le 1 \qquad i=1,...,N \tag{4}$$

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

This formulation (equations (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 (equations (1)-(4)) can be expressed in terms of the correlation ρ_{ij} between assets i and j ($-1 \le \rho_{ij} \le +1$) and the standard deviations s_i, s_j in returns for these assets since $\sigma_{ij}=\rho_{ij}s_is_j$.

2.2 Efficient frontier

By resolving the above QP (equations (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 this curve as the unconstrained efficient frontier (UEF). One such UEF is shown in Figure 1 for assets drawn from the UK FTSE market index.

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

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

subject to

$$\sum_{i=1}^{N} w_i = 1 \quad (6)$$

$$0 \le w_i \le 1 \qquad i=1,...,N \quad (7)$$

In equation (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 equation (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 (equations (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 (equation (5)), which we wish to minimise, becomes 0.25[risk]-0.75[return]. Considering Figure 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 equations (5)-(7) we can trace out *exactly* the same efficient frontier curve as we would obtain by solving equations (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 [24] 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 [21]). Simaan [36] 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 [22] who extended the MAD approach to include skewness in the objective function. Konno and Suzuki [23] considered a mean-variance objective function extended to include skewness. Negative semi-variance, discussed by Markowitz [28], 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 [2] suggested a mixed quadratic and modulus objective function, soluble by QP;

(b) for multi-period optimisation, Mulvey and Vladimirou [32] used stochastic network programming and Yoshimoto [38] used nonlinear programming.

2.4 <u>Constrained problem</u>

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,...,N) if any of asset i is held

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

where we must have $0 \le \varepsilon_i \le \delta_i \le 1$ (i=1,...,N). In practice ε_i represents a "min-buy" for asset i and δ_i limits the exposure of the portfolio to asset i.

Introducing zero-one decision variables:

z_i = 1 if any of asset i (i=1,...,N) is held

= 0 otherwise

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

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

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

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

$$\varepsilon_i z_i \le w_i \le \delta_i z_i \qquad i=1,...,N \tag{12}$$

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

Equation (8) minimises the total variance (risk) associated with the portfolio whilst equation (9) ensures that the portfolio has an expected return of R. Equation (10) ensures that the proportions add to one whilst equation (11) ensures that exactly K assets are held. Equation (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 is held (z_i=0) its proportion w_i is zero. Equation (13) is the integrality constraint.

This formulation (equations (8)-(13)) is a mixed-integer nonlinear (quadratic) programming problem for which, in contrast to the unconstrained case, effective algorithms do not exist. For more on solving mixed-integer nonlinear programs see [7,8,13,18].

We would comment here that we have explicitly chosen to formulate this problem with equality (rather than an inequality ≤) with respect to the number of assets in the portfolio in equation (11). This is because an equality in equation (11) explicitly enables us to gain information about the tradeoff associated with different values of K (see Section 5.5 below).

Speranza [37] considered a cardinality constrained portfolio optimisation problem, but with the risk associated with the portfolio being measured by the deviation of the return below average, rather than by variance. She gave a mixed-integer linear programming formulation together with an heuristic algorithm. Computational results were presented for problems involving up to 20 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 results for the optimal solution of problems involving up to 150 assets. See also Borchers and Mitchell [8].

2.5 Practical constraints

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

(a) Class constraints

Let Γ_m, m=1,...,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:

$$L_m \leq \sum_{i \in \Gamma_m} w_i \leq U_m \quad m=1,...,M \tag{14}$$

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). By simply enumerating (to an appropriate number of decimal places) all possible values for the weights w_i we can construct the feasible region associated with the QP (equations (1)-(4)) for this example. This is shown in Figure 2.

The UEF is the upper left boundary of this feasible region and will (in general) contain portfolios with varying numbers of assets. Typically the maximum return portfolio will contain just a single asset (in our example asset 1) whilst the minimum risk portfolio will contain a relatively high number of assets (in our example all four assets with w_1=0.0847, w_2=0.3364, w_3=0.3412 and w_4=0.2377).

Contrast now the feasible region for the cardinality constrained problem (equations (8)-(13)) involving exactly two assets (K=2) also shown in Figure 2 (where ε_i=0, δ_i=1, i=1,2,3,4). In this case the feasible region reduces to six line segments (one segment for each pair of assets). Moreover the efficient frontier for this example is no longer just the upper left boundary of this feasible region, instead it is the *discontinuous* curve shown in Figure 3.

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.

We note in passing that, as we would expect, imposing the cardinality constraint means that for certain levels of return we incur a greater risk than in the unconstrained case (compare the UEF in Figure 2 with Figure 3).

3.2 Weighting

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

minimise
$$\lambda[\sum_{i=1}^{N}\sum_{j=1}^{N} w_i w_j \sigma_{ij}] - (1-\lambda)[\sum_{i=1}^{N} w_i \mu_i] \tag{15}$$
subject to
$$\sum_{i=1}^{N} w_i = 1 \tag{16}$$
$$\sum_{i=1}^{N} z_i = K \tag{17}$$
$$\varepsilon_i z_i \le w_i \le \delta_i z_i \qquad i=1,...,N \tag{18}$$
$$z_i \in [0,1] \qquad i=1,...,N \tag{19}$$

It would be natural to believe that by varying λ we could use this program (equations (15)-(19)) 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 Figure 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 Figure 3.

In other words, even if we were able to solve equations (15)-(19) exactly for as many values of λ as we wished there will always be portions of the CCEF shown in Figure 3 that can *never* be found by such an approach, i.e. they are effectively (mathematically) invisible to an exact approach based upon weighting.

Figure 3 highlights those portions of the CCEF that are invisible to an exact approach based upon weighting.

3.3 Minimum proportion constraints

To illustrate the effect of imposing a nonzero minimum proportion the feasible region for the case where ε_i=0.24 and δ_i=1, (i=1,2,3,4) is shown in Figure 4, where that figure has been plotted explicitly disregarding the cardinality constraint (equation (11)). It is clear from that figure that again the 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 effect shown in Figure 4 is 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 (equations (15)-(19)) presented in Section 3.2 above. The reason for using this weighted formulation is essentially two-fold:

(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;

(b) attempting to design a computationally effective heuristic that directly addresses the non-weighted formulation (equations (8)-(13)) is difficult because of the requirement that the portfolio expected return is exactly R (equation (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 [19]. 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 [1,4,31,33,34].

Arnone et al [3] 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 [26] 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 [3].

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 ($0 \leq s_i \leq 1$) $i \in Q$. 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 take s_i as proportional 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 (equation (18)) 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.

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 constraints relating to the upper limits δ_i are satisfied. Note here that Algorithm 1 can be viewed as an heuristic for solving the QP equations (15)-(19) 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).

We used a steady-state strategy with 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, each taken from one 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 (ε_i+s_i) of a randomly selected asset i. This mutation corresponds to decreasing or increasing this value by 10%.

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 [14] and Hansen [17]. 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 [1,15,33].

Glover et al [16] 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.

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 is feasible.

The procedure first chooses the best solution from a set of 1000 randomly generated solutions 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 number of neighbours that 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 [29] but its use for optimisation problems originated with Kirkpatrick et al [20] and Cerny [9].

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 [1,33].

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.

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, 100MHz, 48MB main memory).

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 [5,6], 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, ε_i=0 and δ_i=1 (i=1,...,N)).

In order to calculate the exact UEF we used the Numerical Algorithms Group (NAG) library routine E04NAF to solve the QP (equations (1)-(4)) for 2000 different return (R, equation (2)) values, taken between the return value associated with the minimum risk and the return value associated with the maximum return.

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

(a) took the portfolios associated with the values V(λ) as given by our heuristics (see Algorithm 1)

(b) computed the percentage deviation of each portfolio from the (linearly interpolated) exact UEF.

This measurement of error via linear interpolation of the UEF is simply a convenient computational device to approximate the (continuous) UEF from a finite (but reasonably large) number of distinct points on the UEF.

More precisely let (x_i,y_i) be the discrete (x-coordinate, y-coordinate) values on our UEF. For a portfolio with (x^*,y^*) let j correspond to $y_j=\min[y_i|y_i\geq y^*]$ and k correspond to $y_k=\max[y_i|y_i\leq y^*]$ (i.e. y_j and y_k are the closest y-coordinates bracketing y^*). Then the value x^{**} associated with the x-direction linearly interpolated point on the UEF with $y=y^*$ is $x^{**}=x_k+(x_j-x_k)[(y^*-y_k)/(y_j-y_k)]$. A convenient percentage error measure in the x-direction is then $|100(x^*-x^{**})/x^{**}|$ (note here that no error is calculated if either j or k do not exist). A similar expression can be derived for linear interpolation in the y-direction and in the computational results reported later we take as the error measure for each (x^*,y^*) the minimum of the x-direction, y-direction errors. Note here that in order to work in commensurate units we used the portfolio standard deviation (rather than variance) in computing these error measures.

With regard to all the computational results reported here 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 2. 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 2 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 it appropriate to judge the effectiveness of our heuristics solely using the $V(\lambda)$ values. This contrasts with Section 5.2 above where these values 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.

The 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 $r_j \geq r_i$ and $v_j \leq v_i$ 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 was done in Section 5.2 above.

The results for our heuristic algorithms with K=10 and ε_i=0.01, δ_i=1 (i=1,...,N) 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 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 3 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 2). 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 3.

5.4 Discussion

There are a number of points which can be made with respect to Table 3 and these are discussed in this section.

In Table 3 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 3 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 to the UEF. In order to decide which portfolio of assets to buy however, investors examining the efficient frontier need a good distribution of points over this frontier to enable them to make an informed decision.

This distribution of points over the frontier is illustrated numerically in Table 4. 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 favour 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 (equations (15)-(19)) with the given value of λ; or

(b) minimising 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

One interesting aspect of our heuristics is that they can easily be used to illustrate the tradeoff that occurs with respect to K. These tradeoff curves are illustrated in Figure 5 for the DAX data set (which has the highest mean (pooled) error in Table 3) using the pooled results for all three heuristics. In that figure we have plotted the curves for K=2,3,4 and 5 (ε_i=0.01 and δ_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.

6. CONCLUSIONS

In this paper we have considered the problem of calculating the efficient frontier for the cardinality constrained portfolio optimisation 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] E.H.L. Aarts and J.K. Lenstra (editors), *Local search in combinatorial optimization*. Wiley, (1997).

[2] C.J. Adcock and N. Meade, A simple algorithm to incorporate transaction costs in quadratic optimisation. *European Journal of Operational Research* 79 (1994) 85-94.

[3] S. Arnone, A. Loraschi and A. Tettamanzi, A genetic approach to portfolio selection. *Neural Network World* 6 (1993) 597-604.

[4] T. Bäck, D.B. Fogel and Z. Michalewicz (editors), *Handbook of evolutionary computation*. Oxford University Press, (1997).

[5] J.E. Beasley, OR-Library: distributing test problems by electronic mail. *Journal of the Operational Research Society* 41 (1990) 1069-1072.

[6] J.E. Beasley, Obtaining test problems via Internet. *Journal of Global Optimization* 8 (1996) 429-433.

[7] B. Borchers and J.E. Mitchell, An improved branch and bound algorithm for mixed integer nonlinear programs. *Computers & Operations Research* 21 (1994) 359-367.

[8] B. Borchers and J.E. Mitchell, A computational comparison of branch and bound and outer approximation algorithms for 0-1 mixed integer nonlinear programs. *Computers and Operations Research* 24 (1997) 699-701.

[9] V. Cerny, Thermodynamical approach to the travelling salesman problem: an efficient simulation algorithm. *Journal of Optimization Theory and Applications* 45 (1985) 41-51.

[10] H. Dahl, A. Meeraus and S.A. Zenios, Some financial optimization models:I risk management. In *Financial optimization*, S.A. Zenios (editor), 3-36 (1993) Cambridge University Press.

[11] E.J. Elton and M.J. Gruber, *Modern portfolio theory and investment analysis*. Wiley, (1995).

[12] B.R. Feiring, W.L. Wong, M. Poon and Y.C. Chan, Portfolio selection in downside risk optimization approach - application to the Hong Kong stock market. *International Journal of Systems Science* 25 (1994) 1921-1929.

[13] C.A. Floudas, *Nonlinear and mixed-integer optimization: fundamentals and applications*. Oxford University Press, (1995).

[14] F. Glover, Future paths for integer programming and links to artificial intelligence. *Computers & Operations Research* 13 (1986) 533-549.

[15] F.W. Glover and M. Laguna, *Tabu search*. Kluwer Academic Publishers, (1997).

[16] F. Glover, J.M. Mulvey and K. Hoyland, Solving dynamic stochastic control problems in finance using tabu search with variable scaling. In *Meta-heuristics: theory & applications*, I.H. Osman and J.P. Kelly (editors), 429-448 (1996) Kluwer Academic Publishers.

[17] P. Hansen, The steepest ascent mildest descent heuristic for combinatorial programming. Presented at the *Congress on Numerical Methods in Combinatorial*

Optimization, Capri, Italy (1986).

[18] P. Hansen, B. Jaumard and V. Mathon, Constrained nonlinear 0-1 programming. *ORSA Journal on Computing* 5 (1993) 97-119.

[19] J.H. Holland, *Adaptation in natural and artificial systems: an introductory analysis with applications to biology, control, and artificial intelligence*. University of Michigan Press, (1975).

[20] S. Kirkpatrick, C.D. Gelatt and M.P. Vecchi, Optimization by simulated annealing. *Science* 220 (1983) 671-680.

[21] H. Konno, Piecewise linear risk-function and portfolio optimization. *Journal of the Operations Research Society of Japan* 33 (1990) 139-156.

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

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

[24] H. Konno and H. Yamazaki, Mean-absolute deviation portfolio optimization model and its applications to Tokyo Stock Market. *Management Science* 37 (1991) 519-531.

[25] E.K. Lee and J.E. Mitchell, Computational experience of an interior-point SQP algorithm in a parallel branch-and-bound framework. Working paper (1997) available from the second author at Department of Mathematical Sciences, Rensselaer Polytechnic Institute, Troy, NY 12180-3590, USA.

[26] A. Loraschi, A. Tettamanzi, M. Tomassini and P. Verda, Distributed genetic algorithms with an application to portfolio selection problems. In *Artificial neural nets and genetic algorithms,* D.W. Pearson, N.C. Steele and R.F. Albrecht (editors), 384-387 (1995).

[27] H. Markowitz, Portfolio selection, *Journal of Finance* 7 (1952) 77-91.

[28] H.M. Markowitz, *Portfolio selection: efficient diversification of investments*. Wiley, (1959).

[29] N. Metropolis, A.W. Rosenbluth, M.N. Rosenbluth, A.H. Teller and E. Teller, Equation of state calculations by fast computing machines. *Journal of Chemical Physics* 21 (1953) 1087-1092.

[30] T.C. Mills, Stylized facts on the temporal and distributional properties of daily FT-SE returns. *Applied Financial Economics* 7 (1997) 599-604.

[31] M. Mitchell, *An introduction to genetic algorithms*. MIT Press, (1996).

[32] J.M. Mulvey and H. Vladimirou, Stochastic network programming for financial planning problems. *Management Science* 38 (1992) 1642-1664.

[33] C.R. Reeves (editor), *Modern heuristic techniques for combinatorial problems*. Blackwell Scientific Publications, Oxford, (1993).

[34] C.R. Reeves, Genetic algorithms for the Operations Researcher. *INFORMS Journal on Computing* 9 (1997) 231-250.

[35] A. Rudd and B. Rosenberg, Realistic portfolio optimization. In *Portfolio theory, 25 years after*, E.J. Elton and M.J. Gruber (editors), TIMS Studies in the Management Sciences 11 (1979) 21-46, North-Holland.

[36] Y. Simaan, Estimation risk in portfolio selection: the mean variance model versus the mean absolute deviation model. *Management Science* 43 (1997) 1437-1446.

[37] M.G. Speranza, A heuristic algorithm for a portfolio optimization model applied to the Milan stock market. *Computers & Operations Research* 23 (1996) 433-441.

[38] A. Yoshimoto, The mean-variance approach to portfolio optimization subject to transaction costs. *Journal of the Operations Research Society of Japan* 39 (1996) 99-117.

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

Table 1: Four asset example

evaluate(S,λ,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(λ)	is the best objective function value found for weighting parameter λ	
improved	is returned as .true. if S improves V(λ), 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:=∞
if Σ_{i∈Q} ε_i > 1 or Σ_{i∈Q} δ_i < 1 then return            /* infeasible */
L:= Σ_{i∈Q} s_i                                              /* L is the current s_i sum */
F:=1 - Σ_{i∈Q} ε_i                                           /* F is the free proportion */
w_i:=ε_i + s_iF/L ∀i∈Q              /* calculate weights to satisfy ε_i and sum to 1 */
                         /* iterative procedure to satisfy maximum proportions */
R:=∅                                /* R is a set of i whose weights are fixed at δ_i */
while there exists an i∈Q-R with w_i>δ_i do                  /* iterate until feasible */
    for all i∈Q-R if w_i>δ_i then R:=R∪[i]                   /* if w_i exceeds δ_i add to R */
    L:= Σ_{i∈Q-R} s_i                                        /* L is the current s_i sum */
    F:=1 - ( Σ_{i∈Q-R} ε_i + Σ_{i∈R} δ_i)                    /* F is the free proportion */
    w_i:=ε_i + s_iF/L ∀i∈Q-R                                 /* weights for i∈Q-R */
    w_i:=δ_i ∀i∈R                                            /* weights for i∈R */
end while
f:=λ[ Σ_{i∈Q} Σ_{i∈Q} w_iw_jσ_ij] - (1-λ)[ Σ_{i∈Q} w_iμ_i]  /* feasible solution */
s_i:=w_i-ε_i ∀i∈Q                                            /* reset s_i */
if f < V(λ) then                                             /* improved solution */
    improved:=.true.                                         /* set improved */
    V(λ):=f                                                  /* update V(λ) */
    H:=H∪S                                                   /* add S to H */
end if
end
```

Algorithm 1: Evaluation

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:=∅
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_100}           /* random initialisation, exactly K assets in each S_p */
evaluate(S_p,λ,f(S_p),V,improved,H) p=1,...,100              /* evaluate the solutions */
for t:=1 to T* do                                            /* T* iterations in all */
    select S*, S** ∈ P by binary tournament method
    crossover C:=(S*,S**) by uniform method
    A*:=S*∪S**-C                        /* find assets in parents, not in child */
    randomly choose i∈R and m=1 or 2
    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        /* 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∈R with the smallest c_j from R
    while |R|<K do                      /* add asset from parent if possible */
    if |A*|≠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∉R and set c_j:=0
    end if
    end while
                                                    /* C now has exactly K assets */
    evaluate(C,λ,f(C),V,improved,H)
    find j such that f(S_j)=max[f(S_p)|p=1,...,100]   /* find worse population member */
    S_j:=C                                  /* replace worse member with child */
end for
end for
end
```

Algorithm 2: GA heuristic

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:=∅
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 */
L_im:=0 ∀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**:=∞                                                   /* initialise best neighbour value */
    for i∈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(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 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**=∞ 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] ∀i,m                              /* reduce all tabu tenures by one */
    L_k,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

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 (seconds)	621	469	476
DAX	85	Median percentage error	0.0123	2.7816	0.0033
		Mean percentage error	0.0136	3.5645	0.0394
		Time (seconds)	10332	9546	9412
FTSE	89	Median percentage error	0.0029	3.0238	0.0426
		Mean percentage error	0.0063	3.2731	0.2012
		Time (seconds)	11672	10698	10928
S&P	98	Median percentage error	0.0085	4.2780	0.0142
		Mean percentage error	0.0084	4.4280	0.2158
		Time (seconds)	15879	14517	14367
Nikkei	225	Median percentage error	0.0084	14.2668	0.8107
		Mean percentage error	0.0085	15.9163	1.7681
		Time (seconds)	227220	210929	281588
Average		Mean percentage error	0.0114	5.6158	0.4675

Table 2: Results for the unconstrained 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%

Table 4: Contributions to the 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 (seconds)	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 (seconds)	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 (seconds)	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 (seconds)	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 (seconds)	1964		545		553		3062
Average		Mean percentage error	1.4953	1.2269	2.0483	1.9022	1.8461	1.5774	1.1569

Table 3: Results for the cardinality constrained efficient frontier