Available online at www.sciencedirect.com
ScienceDirect
Computers & Operations Research 34 (2007) 1177–1191
computers & operations research
www.elsevier.com/locate/cor

Portfolio selection using neural networks

Alberto Fernández, Sergio Gómez*

Departament d'Enginyeria Informàtica i Matemàtiques, Universitat Rovira i Virgili, Campus Sescelades, Avinguda dels Països Catalans 26, E-43007 Tarragona, Spain

Available online 15 August 2005

Abstract

In this paper we apply a heuristic method based on artificial neural networks (NN) in order to trace out the efficient frontier associated to the portfolio selection problem. We consider a generalization of the standard Markowitz mean-variance model which includes cardinality and bounding constraints. These constraints ensure the investment in a given number of different assets and limit the amount of capital to be invested in each asset. We present some experimental results obtained with the NN heuristic and we compare them to those obtained with three previous heuristic methods. The portfolio selection problem is an instance from the family of quadratic programming problems when the standard Markowitz mean-variance model is considered. But if this model is generalized to include cardinality and bounding constraints, then the portfolio selection problem becomes a mixed quadratic and integer programming problem. When considering the latter model, there is not any exact algorithm able to solve the portfolio selection problem in an efficient way. The use of heuristic algorithms in this case is imperative. In the past some heuristic methods based mainly on evolutionary algorithms, tabu search and simulated annealing have been developed. The purpose of this paper is to consider a particular neural network (NN) model, the Hopfield network, which has been used to solve some other optimisation problems and apply it here to the portfolio selection problem, comparing the new results to those obtained with previous heuristic algorithms.

Keywords: Portfolio selection; Efficient frontier; Neural networks; Hopfield network

* Corresponding author. Tel.: +34 977 55 8508; fax: +34 977 55 9710.
E-mail address: sergio.gomez@urv.net (S. Gómez).

0305-0548/$ - see front matter © 2005 Elsevier Ltd. All rights reserved.
doi:10.1016/j.cor.2005.06.017

1. Introduction

In the portfolio selection problem, given a set of available securities or assets, we want to find out the optimum way of investing a particular amount of money in these assets. Each one of the different ways to diversify this money between the several assets is called a portfolio. For solving this portfolio selection problem, Markowitz [1] presented the so-called mean-variance model, which assumes that the total return of a portfolio can be described using the mean return of the assets and the variance of return (risk) between these assets. The portfolios that offer the minimum risk for a given level of return form what it is called the efficient frontier. For every level of desired mean return, this efficient frontier gives us the best way of investing our money.

However, the standard mean-variance model has not got any cardinality constraint ensuring that every portfolio invests in a given number of different assets, neither uses any bounding constraint limiting the amount of money to be invested in each asset. This sort of constraints are very useful in practice. In order to overcome these inconveniences, the standard model can be generalized to include these constraints.

In this paper we focus on the problem of tracing out the efficient frontier for the general mean-variance model with cardinality and bounding constraints. In previous work, some heuristic methods have been developed for the portfolio selection problem. There are some methods that use evolutionary algorithms [2–6], tabu search (TS) [2,7] and simulated annealing (SA) [2,8,9]. Here we present a different heuristic method based on artificial NN. The results obtained are compared to those obtained using three representative methods from [2] based on genetic algorithms (GA), TS and SA.

Following this introduction, in Section 2, we present the model formulation for the portfolio selection problem. Section 3 describes the Hopfield NN as well as the way to use it for solving this problem. In Section 4, we present some experimental results and, in Section 5, we finish with some conclusions.

2. Portfolio selection

First of all, as we introduce the notation that we are going to use in this paper, let us remember the well known Markowitz mean-variance model [1] for the portfolio selection problem. Let N be the number of different assets, μ_i be the mean return of asset i, σ_{ij} be the covariance between returns of assets i and j, and let $\lambda \in [0, 1]$ be the risk aversion parameter. The decision variables x_i represent the proportion of capital to be invested in asset i. Using this notation, the standard Markowitz mean-variance model for the portfolio selection problem is

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

$$\text{subject to} \quad \sum_{i=1}^{N} x_i = 1, \tag{2}$$

$$0 \leqslant x_i \leqslant 1, \quad i = 1, \ldots, N. \tag{3}$$

The case with $\lambda = 0$ represents maximising the portfolio mean return (without considering the variance) and the optimal solution will be formed only by the asset with the greatest mean return. The case with

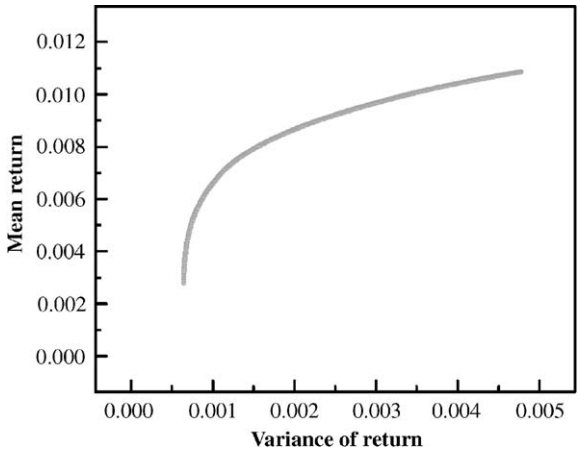

Fig. 1. Standard efficient frontier corresponding to the smallest benchmark problem.

$\lambda = 1$ represents minimising the total variance associated to the portfolio (regardless of the mean returns) and the optimal solution will typically consist of several assets. Any value of λ inside the interval (0, 1) represents a tradeoff between mean return and variance, generating a solution between the two extremes $\lambda = 0$ and 1.

Since every solution satisfying all the constraints (feasible solution) corresponds with one of the possible portfolios, from here on we will speak without distinguishing between solutions for the above problem and portfolios.

The portfolio selection problem is an instance of the family of multiobjective optimisation problems. So, one of the first things to do is to adopt a definition for the concept of optimal solution. Here we will use the Pareto optimality definition [10]. That is, a feasible solution of the portfolio selection problem will be an optimal solution (or nondominated solution) if there is not any other feasible solution improving one objective without making worse the other.

Usually a multiobjective optimisation problem has several different optimal solutions. The objective function values of all these nondominated solutions form what it is called the efficient frontier. For the problem defined in Eqs. (1)–(3), the efficient frontier is an increasing curve that gives the best tradeoff between mean return and variance (risk). In Fig. 1 we show an example of such a curve corresponding to the smallest benchmark problem described in Section 4. This efficient frontier has been computed taking 2000 different values for the risk aversion parameter λ and solving exactly the corresponding portfolio selection problems. The objective function values of the resulting solutions give the 2000 points that form the curve in Fig. 1. We call this curve the standard efficient frontier in order to distinguish it from the general efficient frontier, corresponding to the general mean-variance portfolio selection model which we describe next.

With the purpose of generalizing the standard Markowitz model to include cardinality and bounding constraints, we will use a model formulation that can be also found in [2,7,11]. In addition to the previously defined variables, let K be the desired number of different assets in the portfolio with no null investment, ε_i and δ_i be, respectively, the lower and upper bounds for the proportion of capital to be invested in asset i, with $0 \leqslant \varepsilon_i \leqslant \delta_i \leqslant 1$. The additional decision variables z_i are 1 if asset i is included in the portfolio

and 0 otherwise. The general mean-variance model for the portfolio selection problem is

$$\text{minimise} \quad \lambda \left[\sum_{i=1}^{N} \sum_{j=1}^{N} x_i \sigma_{ij} x_j \right] + (1-\lambda) \left[- \sum_{i=1}^{N} \mu_i x_i \right] \tag{4}$$

$$\text{subject to} \quad \sum_{i=1}^{N} x_i = 1, \tag{5}$$

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

$$\varepsilon_i z_i \leqslant x_i \leqslant \delta_i z_i, \quad i = 1, \ldots, N, \tag{7}$$

$$z_i \in \{0, 1\}, \quad i = 1, \ldots, N. \tag{8}$$

This formulation is a mixed quadratic and integer programming problem for which efficient algorithms do no exist. Another difference with the standard model is that in the presence of cardinality and bounding constraints the resulting efficient frontier, which we are going to call general efficient frontier, can be quite different from the one obtained with the standard mean-variance model. In particular, the general efficient frontier may be discontinuous [2,11].

3. The Hopfield network

3.1. Energy function

There are two main approaches for solving combinatorial optimisation problems using artificial NN: Hopfield networks and Kohonen's self-organizing feature maps. While the latter are mainly used in Euclidean problems, the Hopfield networks have been widely applied in different classes of combinatorial optimisation problems [12]. Although the problem at hand is not a combinatorial optimisation one, we take advantage of the fact that the objective function in Eq. (4) has the same form than the energy function in Hopfield networks and, consequently, it will be minimised if we follow the Hopfield dynamics.

The Hopfield network [13] is an artificial NN model with a single layer of neurons fully connected. That is, all the neurons are connected to each other as well as to themselves. The N variables in the problem are represented by N neurons in the network and, given that the proportion of capital to be invested in each asset i has been called x_i, the state of neuron i will be represented by x_i. Using this notation, the energy function for the Hopfield network has the following form:

$$E(x) = -\frac{1}{2} \sum_{i=1}^{N} \sum_{j=1}^{N} x_i w_{ij} x_j - \sum_{i=1}^{N} b_i x_i, \tag{9}$$

where b_i is the constant external input (bias) for neuron i and w_{ij} is the weight of the synaptic connection from neuron i to neuron j.

Now, looking at the objective function of the portfolio selection problem

$$f(x) = \lambda \left[\sum_{i=1}^{N} \sum_{j=1}^{N} x_i \sigma_{ij} x_j \right] + (1 - \lambda) \left[- \sum_{i=1}^{N} \mu_i x_i \right] \tag{10}$$

and just comparing it with the energy function, we get the values for the synaptic weights

$$w_{ij} = -2\lambda\sigma_{ij} \tag{11}$$

and the external inputs

$$b_i = (1 - \lambda)\mu_i. \tag{12}$$

3.2. Network dynamics

From here on let us consider that we work with discrete time. So, if the state of neuron i at time t is represented by $x_i(t)$, the equations that govern the dynamics of the Hopfield network are

$$x_i(t + 1) = G_i(h_i(t)), \tag{13}$$

where $h_i(t)$ is the input to neuron i at time t:

$$h_i(t) = \sum_{j=1}^{N} w_{ji} x_j(t) + b_i \tag{14}$$

and G_i is the activation function, for which we use the sigmoid:

$$G_i(h_i) = \varepsilon_i + \frac{\delta_i - \varepsilon_i}{1 + \exp(-\beta(h_i - \gamma))} \tag{15}$$

with a gain $\beta > 0$ and a centring constant γ. In our simulations γ has been assigned the middle value between the maximum and the minimum initial inputs $h_i(0)$. ε_i and δ_i are used to ensure that the outputs of the sigmoid fall inside the interval $[\varepsilon_i, \delta_i]$, as it is required by the constraint in Eq. (7).

For the rest of the subsection let us suppose, without loss of generality, that all the lower bounds take the same value ($\varepsilon_i = \varepsilon$) and all the upper bounds are also identical ($\delta_i = \delta$).

The output vector in a Hopfield network represents the solution for the problem at hand. This vector lies inside the hypercube $[\varepsilon, \delta]^N$. The stability of the network can be proved defining the so called energy function for the network and proving that its time derivative is nonincreasing.

The nonlinear nature of the Hopfield network produces multiple equilibrium points. For any given set of initial conditions $x(0)$, the symmetric Hopfield network (with $w_{ji} = w_{ij}$) will converge towards a stable equilibrium point. When the network is deterministic, the position of that point is uniquely determined by the initial conditions: all the initial conditions that lie inside the attraction field of an equilibrium point will converge asymptotically towards that point. The exact number of equilibrium points and their positions are determined by the network parameters w_{ij} and β. When the gain β is small, the number of equilibrium points is low (possibly as low as 1) and they lie inside the hypercube $[\varepsilon, \delta]^N$. Nevertheless, as the gain increases, the number of equilibrium points also increases and their positions move towards the vertices of the hypercube. When the gain tends to its extreme values ($\beta \to +\infty$), the equilibrium

points reach the hypercube vertices and are maximum in number. In this case, the energy function for the network has the form shown in Eq. (9).

In this work we update the neurons asynchronously, that is, only one neuron at a time. The neurons to be updated are selected randomly. This way of updating does not change the positions of the equilibrium points in the network, but it does change the descending path through the energy surface. So, initial conditions that originally were attracted to a particular equilibrium point, can be attracted towards a different equilibrium point when using asynchronous updating.

For solving the portfolio optimisation problem we have implemented a Hopfield network with gains β changing through time [14] and avoiding the saturation of the activation function G_i. The gains used are such that the image interval of the terms $\beta(h_i(0)-\gamma)$ in Eq. (15) widens linearly from $[-10, +10]$ to $[-20, +20]$. The initial gain values produce few equilibrium points, so, regardless of the initial conditions, the network converges towards these points. Then, as time passes, gains are linearly increased, producing energy surfaces with a higher number of equilibrium points and moving these equilibrium points towards the vertices of the hypercube $[\varepsilon, \delta]^N$.

Another problem that must be addressed is the possibility of convergence of the symmetric Hopfield network ($w_{ji} = w_{ij}$) to cycles of length 2. In order to avoid this undesired behaviour in our network dynamics, the following discrete model can be used:

$$x_i(t+1) = (1-\alpha_i)x_i(t) + \alpha_i G_i\left(\sum_{j=1}^{N} w_{ji}x_j(t) + b_i\right) \tag{16}$$

with $\alpha_i \in (0, 1]$. In [15] it is shown that periodic points that are not fix points can appear, specially when all $\alpha_i = 1$. But if synaptic weights are symmetric ($w_{ji} = w_{ij}$) and $w_{ii} > -(2-\alpha_i)/(\alpha_i\beta)$, then the above discrete model has the sequential dynamics convergent to fix points for any $\alpha_i \in (0, 1]$. Since the synaptic weights w_{ii} are fixed from the beginning and the gains β are linearly increased, given any particular pair of values w_{ii} and β, what we must do is to give a value to α_i satisfying the previous condition.

3.3. Constraints satisfaction

When solving any optimisation problem using a Hopfield network, the problem constraints usually appear in the energy function. However, in our case it is not necessary.

First, regarding the constraint $x_i \in [\varepsilon_i, \delta_i]$ in Eq. (7), we can say that it will be satisfied using as the activation function a sigmoid such as the one in Eq. (15), since its outputs already lie inside the desired interval.

To satisfy the cardinality constraint in Eq. (6) we begin our heuristic algorithm with a NN having $3K/2$ neurons that follow the already explained Hopfield dynamics. Doing so, we get a minimum for the objective function. Next thing to do is pruning the least representative neuron, that is, the one with the smallest output. Then we update this new network (with one less neuron) following the same Hopfield dynamics. These two steps, neuron pruning and objective function minimisation, are repeated until the network has exactly K neurons. These remaining neurons are a solution for our original portfolio selection problem.

We are only left to consider the constraint in Eq. (5). To satisfy this constraint we evaluate the feasibility of every portfolio using the same greedy algorithm that has been used in [2], which changes the proportions of capital x_i to be invested in each selected asset, in order to ensure, if possible, that all constraints are

satisfied. In a first step the algorithm assigns to all x_i corresponding to a selected asset its lower limit ε_i plus a fraction proportional to its current value. This ensures that all the constraints relating to the lower bounds are satisfied. In a second iterative step the algorithm takes all the selected assets exceeding their respective upper limit δ_i and fixes them up to these upper limits. Then the rest of the selected assets that are not fixed up, are given a new value for x_i ensuring the lower bounds ε_i and adding a fraction of the free portfolio proportion. This iterative process is repeated until there is no asset out of its limits.

The only thing that we have changed in this greedy algorithm is the insertion of the current candidate solution into the set H with all the Pareto optimal solutions. In [2], the current solution is added to the set H only when it decreases the best objective function value found until that moment. Afterwards, when the heuristic method finishes, all the dominated solutions are removed from H. But this policy can leave out of this set solutions which are Pareto optimal. For example, let us consider the case with a risk aversion parameter $\lambda = 1/2$. If we first evaluate a solution a with variance of return equal to 0.001 and mean return equal to 0.005, then the objective function value for a is $f(a) = -0.002$. Now, if we evaluate in second place a solution b with variance of return 0.004 and mean return 0.006, the objective function value for b is $f(b) = -0.001$, which is greater than $f(a)$, so the solution b would not be added to the set H. But, in this case, the two solutions should be included into H because both of them are Pareto optimal. What we do in our implementation of the greedy algorithm is to include in H all the evaluated solutions and, when the NN heuristic finishes, we remove from this set all the dominated solutions.

Bringing together all that we have said until now, in Algorithm 1 we show the NN heuristic used in this work.

---

function neural_network_heuristic
  $\Delta\lambda$ increment for the risk aversion parameter
  T number of iterations
  M number of portfolios in the set P
returns
  H set with all the Pareto optimal portfolios
var
  P set of portfolios
  P_{can} candidate portfolio
begin
  $H := \varnothing$
  for $\lambda := 0$ **to** 1 **by** $\Delta\lambda$ **do**
    P := initialise_portfolios_randomly(M) {K assets in each one of the M portfolios}
    evaluate_portfolios(P, H) {greedy algorithm}
    for $t := 1$ **to** T **by** $+1$ **do**
      P_{can} := select_portfolio_randomly(P)
      for $k := 3 * K/2$ **to** $K + 1$ **by** -1 **do**
        follow_Hopfield_dynamics(P_{can}) {P_{can} has k assets}
        prune_worst_neuron(P_{can})
      end for
      follow_Hopfield_dynamics(P_{can}) {P_{can} has K assets}

```
    evaluate_portfolio(P_can, H)                    {greedy algorithm}
    replace_maximum_portfolio(P_can, P)
   end for
  end for
  return H
end neural_network_heuristic

action follow_Hopfield_dynamics
  P_can    candidate portfolio
var
  R    number of repetitions
begin
  (γ, β) := study_initial_inputs(P_can)             {central input and starting gain value}
  for r := 1 to R by +1 do
   update_neuron(γ, β, P_can)                       {neuron to update selected randomly}
   increase_gain_value(β)
  end for
end follow_Hopfield_dynamics
```

Algorithm 1. Neural network heuristic.

4. Computational experiments

4.1. Definition of the experiments

In this section, we present the results obtained when searching the general efficient frontier that solves the problem formulated in Eqs. (4)–(8). We have done all the computational experiments with five sets of benchmark data that have been already used in [2–4,7,11]. These data correspond to weekly prices from March 1992 to September 1997 and they come from the indices: Hang Seng in Hong Kong, DAX 100 in Germany, FTSE 100 in UK, S&P 100 in USA and Nikkei 225 in Japan. The number N of different assets considered for each one of the test problems is 31, 85, 89, 98 and 225, respectively. The mean returns and covariances between these returns have been calculated for the data. The sets of mean returns and covariances are publicly available at *http://people.brunel.ac.uk/~mastjjb/jeb/orlib/portinfo.html*.

All the results presented here have been computed, as in Ref. [2], using the values $K = 10$, $\varepsilon_i = 0.01$ and $\delta_i = 1$ for the problem formulation, and the values $\Delta\lambda = 0.02$, $T = 1000N$ and $M = 100$ for the implementation of Algorithm 1. So we have tested 51 different values for the risk aversion parameter λ and each one of the four heuristics has evaluated $1000N$ portfolios for each value of λ, without counting initialisations.

The general efficient frontier has been computed using the former NN and our own implementation of three other heuristic algorithms presented in [2], which are based on GA, TS and SA. Our implementation of these additional heuristics uses the same parameter values than those presented in Ref. [2].

We would like to notice that the computational results presented here are not directly comparable to those presented in [2] due to the differences that exist at the moment of updating the set of Pareto optimal solutions (explained in the previous section) and some other possible statistical fluctuations.

4.2. Results

Taking the sets of Pareto optimal portfolios obtained with each heuristic we can trace out their heuristic efficient frontiers and compare them to the standard efficient frontiers. Doing so we get an upper bound of the error associated to each heuristic algorithm. We show these comparisons in Fig. 2, where the five problems are arranged by rows and the four heuristics are arranged by columns. Except for the first problem, where all four heuristics seem to obtain similar results, at least from this macroscopic point of view, the four major problems show a common behaviour. Looking at the portfolios with low mean return because of the low risk allowed (high values of the risk aversion parameter λ), we can see how the NN algorithm gets better results than the other three heuristics. But the situation changes completely when we consider low values of λ and the increase of the mean return is the main objective, regardless of the risk. The solutions obtained in the first case consist of significant investments diversified in four or more of the $K=10$ assets, whilst the solutions in the second case typically have significant investments in three or less of the $K = 10$ assets. (Here we have applied the term "significant investment" to any investment above $1/K$.)

Our explanation for the results obtained with the NN is that when the risk aversion parameter λ takes low values, the quadratic term in Eq. (10) decreases considerably and the objective function becomes almost linear, being no more a proper Hopfield energy function.

Now let us take a look at some numerical results in Table 1. First of all we have the number of points measure, which indicates how many of the portfolios evaluated by each heuristic method have persisted and finally appear in the corresponding heuristic efficient frontier. The results show that the highest numbers of points for all five problems come from TS, whilst the lowest numbers come from NN.

The number of points measure only gives an idea about the total number of solutions that appear in each heuristic efficient frontier, but it does not say anything about the quality of these solutions. Now, also in Table 1, we present three error measures, one for variances, another one for mean returns and a third global error measure. Let the pair (v_i, r_i) represent the variance and mean return of a point in a heuristic efficient frontier. Let also $\hat{v}_i$ be the variance corresponding to r_i according to a linear interpolation in the standard efficient frontier. We define the variance of return error φ_i for any heuristic point (v_i, r_i) as the value $100(\hat{v}_i - v_i)/\hat{v}_i$ (note that this quantity will always be nonnegative). The average value of all the errors φ_i for the points in a heuristic efficient frontier gives us the variance of return error shown in Table 1. In the same way, using the mean return $\hat{r}_i$ corresponding to v_i according to a linear interpolation in the standard efficient frontier, we define the mean return error ψ_i as the quantity $100(r_i - \hat{r}_i)/\hat{r}_i$. We get the average error for the mean returns computing the average of all the errors ψ_i, which appears in Table 1 as the mean return error. There is not a single heuristic that gives better results than the others in any of these two average errors. Regarding the NN results we must say that, except for the second benchmark problem, they are worse than those obtained by the other heuristics. This is so because we are considering all possible values of the risk aversion parameter as a whole. In the next subsection we will see how the NN improves considerably its results when we consider only the best solutions of each heuristic.

We also give the results obtained for a third error measure defined in [2]. It is calculated averaging the minimums between the mean return errors ψ_i and the standard deviation of return errors $\tilde{\varphi}_i$, which are similar to the variance of return errors but defined from the standard deviation of returns s_i, that is,

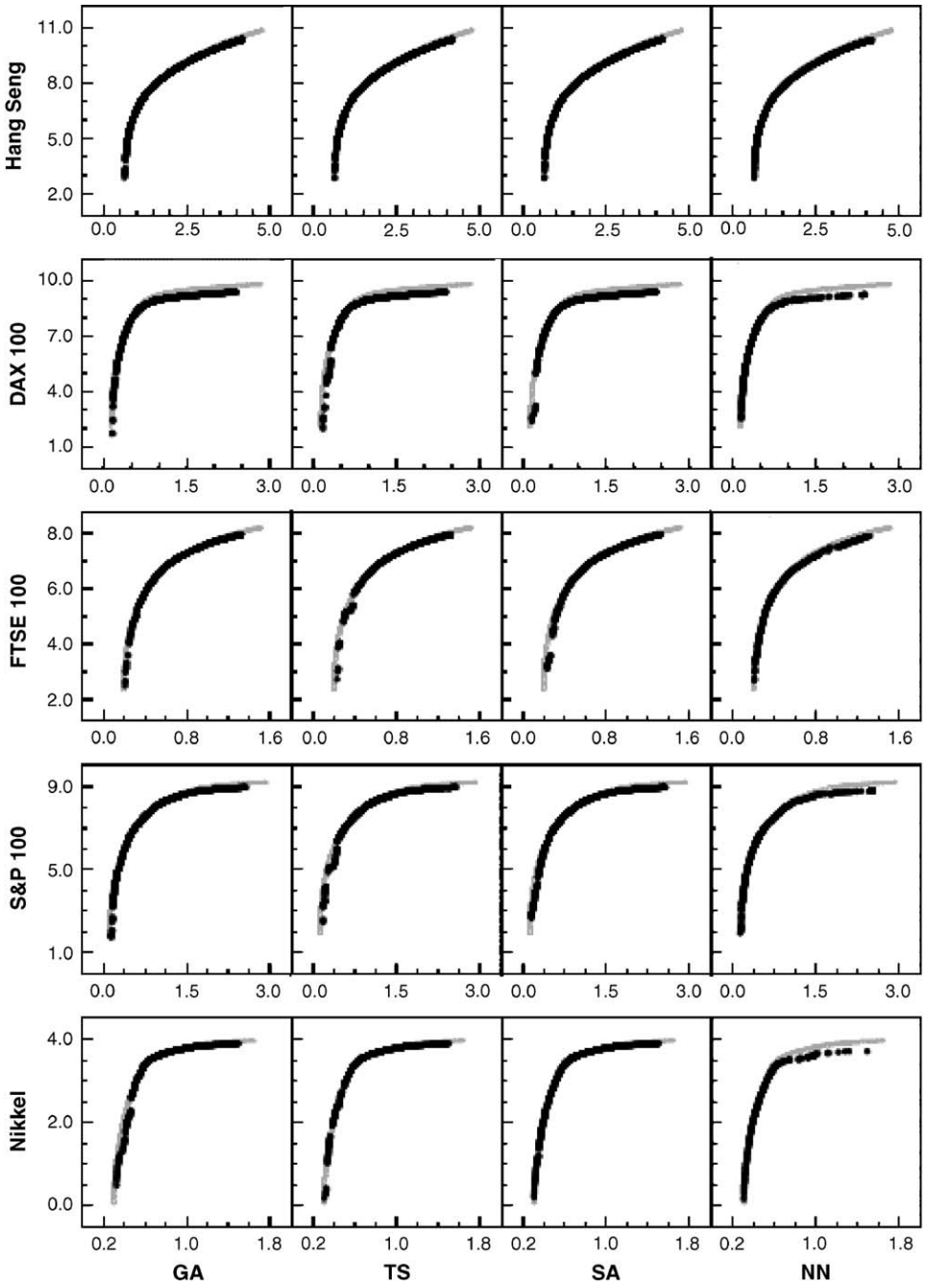

Fig. 2. Heuristic efficient frontiers. The variance of return ($\times 10^3$) is represented on the x-axis whilst the mean return ($\times 10^3$) is represented on the y-axis. The standard efficient frontiers are drawn in grey and the heuristic efficient frontiers are drawn in black.

Table 1
Numerical results

Index	Assets		GA	TS	SA	NN
Hang Seng	31	Number of points	3402	3659	2640	1108
		Variance of return error	3.9576	3.9329	3.8689	4.1039
		Mean return error	1.1926	1.1500	1.1574	1.4530
		Minimum error	1.1321	1.1237	1.1203	1.2316
		Time (s)	47	16	18	390
DAX 100	85	Number of points	1828	2292	1264	573
		Variance of return error	26.1240	24.1340	26.8588	12.5914
		Mean return error	2.6202	2.8490	2.6893	2.2060
		Minimum error	2.4457	2.6668	2.3896	1.5776
		Time (s)	162	45	47	1069
FTSE 100	89	Number of points	1284	1295	1267	426
		Variance of return error	3.3464	3.1458	3.6930	4.4663
		Mean return error	0.9300	0.8954	1.3127	1.9636
		Minimum error	0.7310	0.7357	0.9512	1.2513
		Time (s)	160	51	60	1106
S&P 100	98	Number of points	1780	2318	1779	750
		Variance of return error	7.2039	7.6219	8.1602	8.3811
		Mean return error	1.6130	1.4249	1.9672	2.6816
		Minimum error	1.3236	1.3130	1.7251	1.7922
		Time (s)	178	50	52	1211
Nikkei	225	Number of points	807	1027	984	312
		Variance of return error	4.9877	3.5724	3.4830	6.5924
		Mean return error	3.3931	1.1581	1.2144	3.1050
		Minimum error	1.1415	0.5510	0.5458	1.4737
		Time (s)	570	120	121	2827

$\tilde{\varphi}_i = 100(\hat{s}_i - s_i)/\hat{s}_i$. We present the values correspondent to the minimum error measure in Table 1 to allow some kind of comparison between our results and those of Ref. [2]. However, we prefer to consider the variance of return error and the mean return error separately, because the use of the standard deviation for the calculation of the minimum error measure does not correspond exactly with the objective function of the problem at hand.

With regard to the computation times, the TS and the SA are the most efficient algorithms, followed by the GA and finally by the NN. Anyway, all four time cost functions are linear with respect to the number of assets N since the four algorithms have evaluated $1000N$ portfolios for each value of the risk aversion parameter λ, and the rest of operations in the algorithms (except for the crossover procedure in the GA) do not depend on N. The times presented in Table 1 agree with it.

4.3. Merge process

In order to improve the results obtained separately by the four heuristic algorithms, we have merged the four heuristic efficient frontiers into a single one and we have removed from it the dominated solutions.

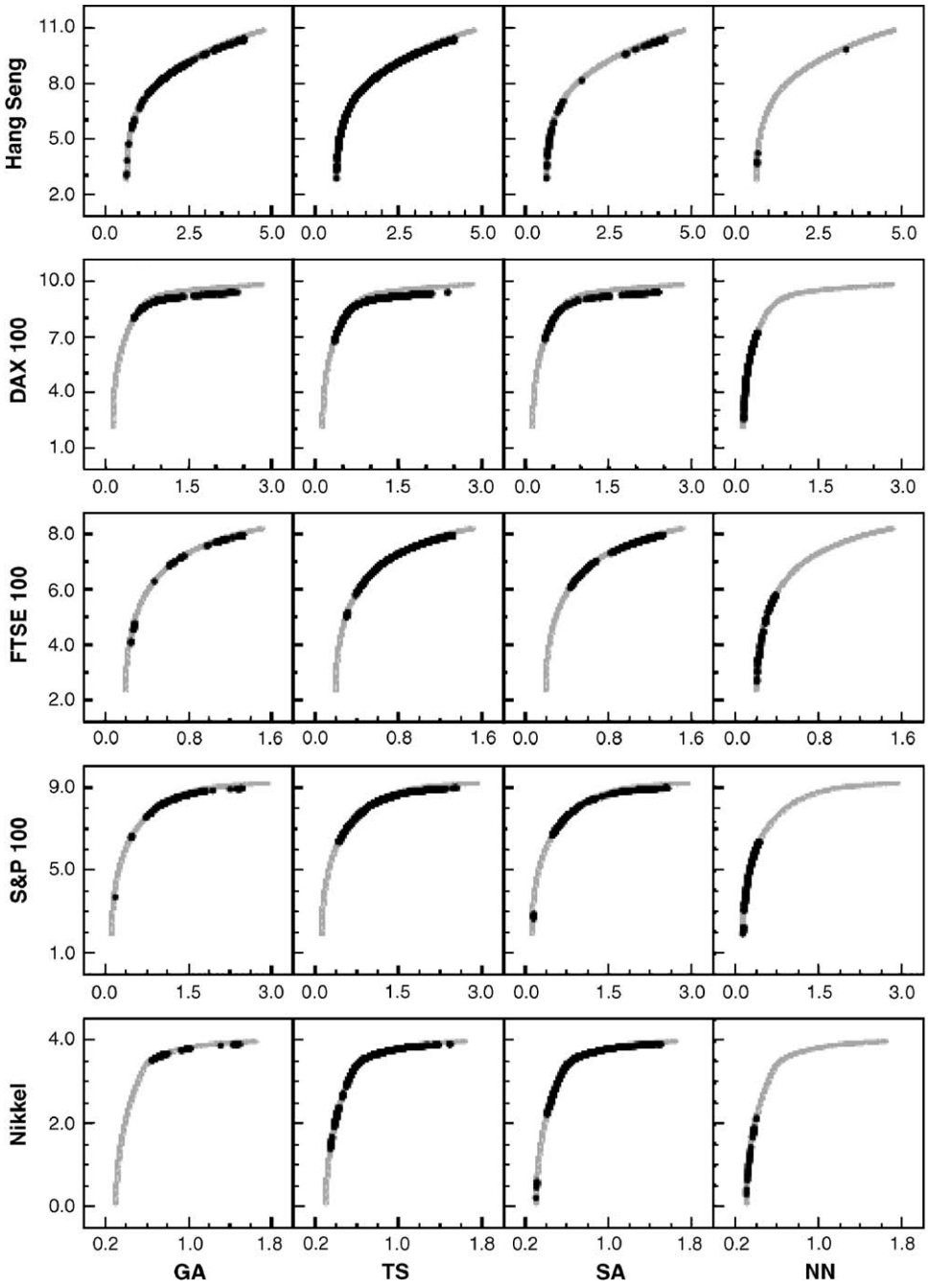

Fig. 3. Contributions to the merged efficient frontiers. The variance of return ($\times 10^3$) is represented on the x-axis whilst the mean return ($\times 10^3$) is represented on the y-axis. The standard efficient frontiers are drawn in grey and the contributions to the merged efficient frontiers are drawn in black.

Table 2
Numerical results of the merge process

Index	Assets		GA	TS	SA	NN
Hang Seng	31	Number of surviving points	1015	3340	270	5
		Surviving points percentage (%)	29.8	91.3	10.2	0.5
		Contribution percentage (%)	21.9	72.1	5.8	0.1
		Variance of return error	4.4758	3.9046	4.5314	1.2279
		Mean return error	1.2438	1.1431	1.1834	1.5304
		Minimum error	1.2321	1.1200	1.1167	0.3751
DAX 100	85	Number of surviving points	635	2088	291	222
		Surviving points percentage (%)	34.7	91.1	23.0	38.7
		Contribution percentage (%)	19.6	64.5	9.0	6.9
		Variance of return error	31.1493	23.2013	32.9453	2.3004
		Mean return error	2.6802	2.3492	1.8412	2.7545
		Minimum error	2.6802	2.3492	1.8412	1.1326
FTSE 100	89	Number of surviving points	220	1207	666	146
		Surviving points percentage (%)	17.1	93.2	52.6	34.3
		Contribution percentage (%)	9.8	53.9	29.7	6.5
		Variance of return error	5.1662	2.7487	3.8031	2.5449
		Mean return error	1.1129	0.5192	0.6501	3.1171
		Minimum error	0.9700	0.5160	0.6501	1.2544
S&P 100	98	Number of surviving points	370	2146	456	287
		Surviving points percentage (%)	20.8	92.6	25.6	38.3
		Contribution percentage (%)	11.4	65.8	14.0	8.8
		Variance of return error	6.6434	6.3510	7.3497	5.9020
		Mean return error	0.8449	0.7978	0.9405	4.7956
		Minimum error	0.8349	0.7978	0.7381	2.7908
Nikkei	225	Number of surviving points	122	943	648	53
		Surviving points percentage (%)	15.1	91.8	65.9	17.0
		Contribution percentage (%)	6.9	53.4	36.7	3.0
		Variance of return error	4.6384	3.3516	3.6212	0.7172
		Mean return error	0.6399	0.5624	0.6153	4.6097
		Minimum error	0.6399	0.5237	0.5002	0.3578

Then we have separated the resulting merged efficient frontier into the four parts that form it according to the heuristic origin of the points, getting the results shown in Fig. 3. Observe that these graphs confirm what we had already noticed in Fig. 2, which is that the NN gives better results than the other heuristics, except for the smallest benchmark problem, when we consider high values of the risk aversion parameter λ and, on the contrary, it gives worse solutions for low values of λ.

Continuing with the results of the merge process, let us now use the new merged efficient frontiers to compare the quality of the initial solutions provided by each heuristic method. We take the initial numbers of points in each heuristic efficient frontier (Table 1) and we compare them with the final numbers of points in the merged efficient frontiers that come from the corresponding heuristic. Table 2 shows the percentage of points surviving the merge process. The initial quality of TS solutions is

outstanding given that more than 90% of the portfolios provided by this algorithm survive the merge process.

Next, in Table 2, we show the contribution of each heuristic to the merged efficient frontiers. As it can be seen, more than half of the points in the merged efficient frontiers come from the TS. Nevertheless, we would like to remember that most of these points correspond to low diversification portfolios.

Also in Table 2, we give the values obtained with the three previously defined error measures when they are applied to the merged efficient frontiers. Limiting the error calculations only to the solutions that pass the merge process, we observe that the NN gets the lowest variance of return errors in the five benchmark problems, being especially good the result obtained in the second problem. With respect to the mean return error, the situation is the opposite one and the NN gives the highest error values.

Finally notice that in the GA, TS and SA columns of the same table there are several cases where the mean return error coincides with the minimum error. In these cases, the heuristics have the mean return errors always lower than the standard deviation errors.

5. Conclusions

In this work we have focused on solving the portfolio selection problem and tracing out its efficient frontier. Instead of using the standard Markowitz mean-variance model, we have used a generalization of it that includes cardinality and bounding constraints. Dealing with this kind of constraints, the portfolio selection problem becomes a mixed quadratic and integer programming problem for which no computational efficient algorithms are known.

We have developed a heuristic method based on the Hopfield neural network and we have used it to solve the general mean-variance portfolio selection model. The results obtained have been compared to those obtained using three other heuristic methods coming from the fields of genetic algorithms, tabu search and simulated annealing.

All the experimental results presented in this paper lead us to conclude that none of the four heuristic methods has outperformed the others in all kinds of investment policies. Anyway, we have observed that, when dealing with problem instances demanding proper diversification portfolios with low risk of investment, the neural network model has given better solutions than the other three heuristic methods.

Acknowledgements

This work was supported in part by CICYT projects FPA2002-04452-C02-02 and FPA2002-04208-C07-07. A.F. also thanks Universitat Rovira i Virgili for financial support.

References

[1] Markowitz H. Portfolio selection. Journal of Finance 1952;7:77–91.
[2] Chang T-J, Meade N, Beasley J, Sharaiha Y. Heuristics for cardinality constrained portfolio optimisation. Computers & Operations Research 2000;27:1271–302.
[3] Lin D, Wang S, Yan H. A multiobjective genetic algorithm for portfolio selection problem. In: Proceedings of ICOTA 2001. Hong Kong, 15–17 December 2001.

[4] Streichert F, Ulmer H, Zell A. Evolutionary algorithms and the cardinality constrained portfolio optimization problem. In: Selected papers of the international conference on operations reasearch (OR 2003). Heidelberg, 3–5 September 2003.
[5] Fieldsend J, Matatko J, Peng M. Cardinality constrained portfolio optimisation. In: Proceedings of the fifth international conference on intelligent data engineering and automated learning (IDEAL'04). Exeter, 25–27 August 2004.
[6] Xia Y, Liu B, Wang S, Lai K. A model for portfolio selection with order of expected returns. Computers & Operations Research 2000;27:409–22.
[7] Schaerf A. Local search techniques for constrained portfolio selection problems. Computational Economics 2002;20: 177–90.
[8] Gilli M, Këllezi E. Heuristic approaches for portfolio optimization. In: Sixth international conference on computing in economics and finance of the society for computational economics. Barcelona, 6–8 July 2000.
[9] Kellerer H, Maringer D. Optimization of cardinality constrained portfolios with an hybrid local search algorithm. In: MIC'2001—4th Methaheuristics international conference. Porto, 16–20 July 2001.
[10] Sawaragi Y, Nakayama H, Tanino T. Theory of multiobjective optimization. In: Bellman R, editor. Mathematics in science and engineering, vol. 176. New York: Academic Press Inc.; 1985.
[11] Jobst N, Horniman M, Lucas C, Mitra G. Computational aspects of alternative portfolio selection models in the presence of discrete asset choice constraints. Quantitative Finance 2001;1:1–13.
[12] Smith K. Neural networks for combinatorial optimization: a review of more than a decade of research. INFORMS Journal on Computing 1999;11(1):15–34.
[13] Hopfield J. Neurons with graded response have collective computational properties like those of two-state neurons. Proceedings of the National Academy of Sciences 1984;81:3088–92.
[14] Peterson C, Söderberg B. A new method for mapping optimization problems onto neural networks. International Journal of Neural Systems 1989;1:3–22.
[15] Wang X, Jagota A, Botelho F, Garzon M. Absence of cycles in symmetric neural networks. Neural Computation 1998;10(5):1235–49.