A Multi-Objective Algorithm based upon Particle Swarm Optimisation, an Efficient Data Structure and Turbulence.

Jonathan E. Fieldsend
Department of Computer Science
University of Exeter
Exeter
EX4 4QF. UK
J.E.Fieldsend@exeter.ac.uk

Sameer Singh
Department of Computer Science
University of Exeter
Exeter
EX4 4QF. UK
S.Singh@exeter.ac.uk

Abstract

This paper introduces a Multi-Objective Algorithm (MOA) based upon the Particle Swarm Optimisation (PSO) heuristic. It utilises an order put upon members of non-dominated sets by the recent *dominated tree* data structure to facilitate the choosing of a 'best' global individual for each member of the swarm, in order to direct their velocities. The method is validated using several test functions, and is shown to be significantly better than two existing MOAs. The benefit of perturbing the flight of particles is also highlighted, with its application demonstrated with respect to the new PSO method, and another recently developed PSO method from the literature.

1 Introduction

Frequently a number of competing objectives have to be traded against one another whilst seeking a viable solution to a given problem, often without any *a priori* knowledge of exactly how the objectives interact with one another. For instance, in product design a firm may wish to maximise the performance of an appliance whilst also trying to minimise its production cost. These two objectives cannot typically be met by a single solution, so, by adjusting the various design parameters, the firm may seek to discover what possible combinations of these two objectives are available, given a set of constraints (for instance legal requirements and size limits of the product).

The curve (for two objectives) or surface (more than two objectives) that describes the optimal trade-off possibilities between objectives is known as the Pareto front. A feasible solution lying on the Pareto front cannot improve any objective without degrading at least one of the others, and, given the constraints of the model, no solutions exist beyond the true Pareto front. The goal, therefore, of multi-objective algorithms is to locate the Pareto front of these *non-dominated* solutions.

Multi-Objective Evolutionary Algorithms (MOEAs) represent a popular approach to confronting these types of problem by using evolutionary search techniques. The use of Evolutionary Algorithms (EAs) as the tool of choice is due to such problems being typically complex, with both a large number of parameters to be adjusted, and several objectives to be optimised. In addition, EAs, which maintain a population of solutions, are able to explore several parts of the Pareto front simultaneously.

Both Genetic Algorithms (GAs) and Evolutionary Strategies (ESs) have been utilised in a number of MOEAs (for instance [2, 12, 14] and [8]) leading to the unified model proposed by Laumanns et al. [10]. However, until recently [1, 6, 11], the (relatively) new technique of Particle Swarm Optimisation (PSO) [7, 13] had not been applied to the multi-objective domain. Given the promising results reported in the uni-objective optimisation domain, the application of PSO to the multi-objective domain is a natural progression. In this paper we argue that current attempts at multi-objective PSO do not fully transfer the PSO heuristic to the multi-objective domain. We therefore introduce a new method that utilises the recent *dominated trees* data structure [3, 4] to enable the selection of an appropriate Pareto archive member to act as the global 'best' for any given particle, and also maintains a local set of 'best' solutions for each swarm member. We then demonstrate that this approach is significantly better than the method used in [1], and an Evolutionary Strategy (ES) derived from the unified model [10], which is based upon an existing MOEA [8]. In addition, we demonstrate that the inclusion of a turbulence variable within multi-objective PSO algorithms significantly increases performance.

The paper takes the following structure: in

Section 2 Pareto optimality is reviewed; in Section 3 PSO is briefly described, as are two current applications in the literature of multi-objective PSO. In Section 4 one of the data structures introduced in [4], *dominated trees*, is described, in preparation for the key role it plays in the multi-objective PSO method introduced in section 5.

A set of experiments to quantify the performance of this new multi-objective PSO algorithm, in comparison to an ES method and an existing multi-objective PSO are described in Section 6. The results of these experiments are situated in Section 7, followed by a discussion in Section 8.

2 Pareto optimality

Most recent work on MOEAs hinges on the notions of non-dominance and Pareto optimality, which are now briefly reviewed.

The multi-objective optimisation problem seeks to simultaneously extremise D objectives:

$$y_i = f_i(\mathbf{x}), \qquad i = 1, \ldots, D \tag{1}$$

where each objective y is a function of a vector $\mathbf{x}$ of N parameters or decision variables. The parameters may also be subject to the J constraints:

$$e_j(\mathbf{x}) \geq 0, \qquad j = 1, \ldots, J. \tag{2}$$

Without loss of generality it is assumed that the objectives are to be minimised, so that the optimisation problem may be expressed as:

$$\text{Minimise } \mathbf{y} = f(\mathbf{x}) = (f_1(\mathbf{x}), \ldots, f_D(\mathbf{x})) \tag{3}$$

$$\text{subject to } e(\mathbf{x}) = (e_1(\mathbf{x}), \ldots, e_J(\mathbf{x})) \geq 0 \tag{4}$$

where $\mathbf{x} = (x_1, \ldots, x_N)$ and $\mathbf{y} = (y_1, \ldots, y_D)$.

When faced with only a single objective an optimal solution is one which minimises the objective given the model constraints. However, when there is more than one objective to be minimised it is clear that solutions may exist for which performance on one objective cannot be improved without sacrificing performance on at least one other. Such solutions are said to be *Pareto optimal* and the set of all Pareto optimal solutions are said to form the Pareto front, $\mathcal{P}$.

The notion of *dominance* may be used to make Pareto optimality clearer. A decision vector $\mathbf{u}$ is said to *strictly dominate* another $\mathbf{v}$ (denoted $\mathbf{u} \prec \mathbf{v}$) iff

$$\begin{aligned} f_i(\mathbf{u}) \leq f_i(\mathbf{v}) \quad \forall i = 1, \ldots, D \quad \text{and} \\ f_i(\mathbf{u}) < f_i(\mathbf{v}) \quad \text{for some } i. \end{aligned} \tag{5}$$

Less stringently, $\mathbf{u}$ *weakly dominates* $\mathbf{v}$ (denoted $\mathbf{u} \preceq \mathbf{v}$) iff

$$f_i(\mathbf{u}) \leq f_i(\mathbf{v}) \quad \forall i = 1, \ldots, D. \tag{6}$$

A set of M decision vectors W is said to be a *non-dominated set* (an estimated Pareto front $\mathcal{E}$) if no member of the set is dominated by any other member:

$$W_i \nprec W_j \quad \forall i, j = 1, \ldots, M. \tag{7}$$

3 PSO and multi-objective PSO

The PSO heuristic was first proposed by Kennedy and Eberhart [7] for the optimisation of continuous non-linear functions. Like GAs and ESs it was inspired by nature, however instead of evolutionary processes, it was the ability of birds to flock which acted as the algorithm's initial inspiration. A fixed population of solutions is used, where each solution (or particle) is represented by a point in N-dimensional space. The ith particle is commonly represented [1, 11, 13] as $X_i = (x_{i,1}, \ldots x_{i,N})$, and its performance evaluated on the given problem and stored. Each particle maintains knowledge of its best previous evaluated position, represented as $P_i = (p_{i,1}, \ldots p_{i,N})$, and also has knowledge of the single global best solution found so far, in the traditional uni-objective application indexed by g. The rate of position change of a particle then depends upon its previous local best position and the global best, and its previous velocity. For particle i this velocity is $V_i = (v_{i1}, \ldots v_{iN})$. The general algorithm for the adjustment of these velocities is:

$$v_{i,j} := wv_{i,j} + c_1 r_1 (p_{i,j} - x_{i,j}) + c_2 r_2 (p_{g,j} - x_{i,j}) \tag{8}$$

$$x_{i,j} := x_{i,j} + \chi v_{i,j}, \qquad j = 1, \ldots, N. \tag{9}$$

Where $w, c_1, c_2, \chi \geq 0$. w is the inertia of a particle, c_1 and c_2 are constraints on the velocity toward global and local best, χ is a constraint on the overall shift in position and $r_1, r_2 \sim U(0, 1)$. In [7], the final model presented has w and χ set at 1 and c_1 and c_2 are set at 2.

In order to facilitate a multi-objective approach to PSO a set of non-dominated solutions (the best individuals found so far using the search process) must replace the single global best individual in the standard uni-objective PSO case, in addition, there may be no single local best individual for each member of the swarm. Choosing both which global and which local individual to direct a swarm member's flight therefore is not trivial in the multi-objective domain. Currently three studies have attempted it.

In Hu and Eberhart [6] a considerable degree of *a priori* knowledge of the test function properties is used by their $D = 2$ multi-objecive PSO. Instead of a single *gbest* a local *lbest* is found for each swarm member, selected from the 'closest' two swarm members to an individual. The concept of closeness is calculated in terms of only one of the evaluated objective dimensions, with the selection of the local optima of the two based upon the other objective. The selection of which objective to fix first and which to optimise second is based on the knowledge of the test function design – the relatively simple objective function being fixed. A single *pbest* is maintained for each swarm member, which is only replaced when a new solution is found which is lower on all objectives (identical to the 'conservative' preservation of efficiency selection rule described by Hanne in [5]). Their model was used on a number of test functions from the literature, however no comparison was made with any other models, or the true Pareto fronts for the problems.

Parsopoulos and Vrahatis [11] introduce two methods that use a weighted aggregate approach and another that is loosely based on Schaffer's MOEA [12]. These were compared on a number of two dimensional problems. The weighted aggregate algorithms needed to be run K times to produce K estimated Pareto optimal points (meaning each run had a single global best). Although [11] states that this approach has a low computational cost, the need for a separate run for each solution found does not necessarily support this. Their final method - the Vector Evaluated Particle Swarm Optimiser (VEPSO), uses one swarm for each objective. The best particle of the second swarm was used to determine the velocities of the first swarm (act as its global best), and vice-versa. Comparison between the algorithms was qualitative (based on visual inspection of the found fronts), and no comparison was made to competitive methods in the MOEA domain. (In addition the current VEPSO model is also only designed for $D = 2$ problems.)

Coello and Lechuga [1] in comparison propose a method which is inspired by more recent developments in the MOEA literature. Two repositories are maintained in addition to the search population. One of the global best individuals found so far by the search process, Z, and one containing a single local best for each member of the swarm (as in standard PSO). A truncated archive is used to store the (global) elite individuals. This archive uses the method from [8] to separate the objective function space into a number of hypercubes (an adaptive grid), with the most densely populated hypercubes truncated if the archive exceeds its membership threshold. The archive also facilitates the selection of a global best for any particular individual in [1]. A *fitness* is given to each hypercube that contains archive members, equal to dividing 10 by the number of resident particles. Thus a more densely populated hypercube is given a lower score, an illustration of which is given in Figure 1.

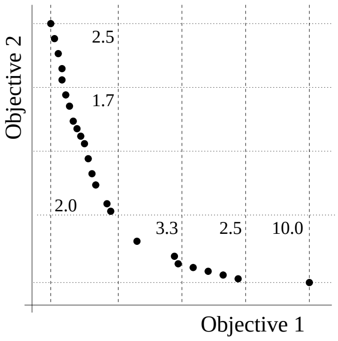

Figure 1: 2D Illustration of grid based selection scheme used in [1], with the 'fitness' of populated hypercubes highlighted.

Selection of a global best for a particle is then based on roulette wheel selection of a hypercube first (according to its score), and then uniformly choosing a member of that hypercube. This method therefore biases selection toward under-represented areas of the estimated Pareto front (unlike the original method developed in [8]). Only one local best solution is maintained for each swarm member however; if a particle X_i is evaluated and found to be mutually non-dominating with P_i, one of the two is randomly selected to be the new P_i.

The multi-objective PSO method in [1] was compared with two highly regarded MOEAs, the Pareto Archived Evolutionary Strategy (PAES) [8] and the Non-Dominated Sorting Genetic Algorithm II [2], with promising results. On the two dimensional test functions used their multi-objective PSO either outperforms or is not significantly different to the competing algorithms (using the M_1^* measure [14]).

However, even though the multi-objective PSO introduced in [1] maintains an archive of global best solutions, this study will argue that there is a better way to select from this archive than simple density based selection. That is, to base it upon which archive member the swarm individual is closest to. First a new data structure called the dominated tree, introduced in [4], will be briefly described, as this data structure facilitates the rapid selection of an appropriate archive member for this new multi-objective PSO method.

4 Dominated trees

Recent studies have highlighted the theoretic inefficiency caused by representing a non-dominated set with a limited number of solutions [5, 9]. This in turn led Fieldsend et al. [4] and Everson et al. [3] to empirically demonstrate the inefficiency caused by truncation of estimated Pareto archives in MOEAs, and develop a number of data structures to facilitate the maintenance of unconstrained archives. In this section we shall briefly describe the properties of one of these, the dominated tree. Formal algorithms and proofs of all the data structures can be found in [4], as well as descriptions on how to use them for unconstrained archive maintenance.

The dominated tree consists of a list of $L = \lceil |Z|/D \rceil$ *composite points* ordered by the weakly-dominates relation, $\preceq$:

$$\mathcal{T} = \{\mathbf{c}_L \preceq \ldots \preceq \mathbf{c}_2 \preceq \mathbf{c}_1\} \tag{10}$$

Usually, the stronger condition, $\mathbf{c}_i \prec \mathbf{c}_j \quad \text{iff } i > j$, will hold. The coordinates of each composite point are defined by (up to) D elements of Z, the *constituent points* of a composite point. An example of a dominated tree in two dimensions is shown in Figure 2.

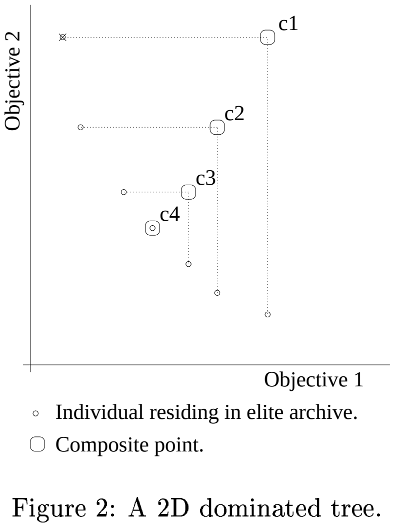

Figure 2: A 2D dominated tree.

First we regard members of the frontal set and individual(s) from the search population as points $\mathbf{y}$ in D-dimensional space. Denote by Y_i the constituent points of $\mathbf{c}_i$, namely the D-tuple defining the coordinates of $\mathbf{c}_i$; so that if

$$Y_i = \langle \mathbf{y}^{(1)}, \mathbf{y}^{(2)}, \ldots, \mathbf{y}^{(D)} \rangle \tag{11}$$

then the dth coordinate of the composite point is the dth coordinate of $\mathbf{y}^{(d)}$: $\mathbf{c}_i = \mathbf{y}_d^{(d)}$. Dominated trees are constructed to have the property such that if $\mathbf{c}_i \prec \mathbf{y}$ then all the constituent points of $\mathbf{c}_i$ (at least) weakly dominate $\mathbf{y}$:

$$\text{If} \quad \mathbf{c}_i \preceq \mathbf{y} \quad \text{then} \quad \mathbf{y}^{(d)} \preceq \mathbf{y} \quad \forall \mathbf{y}^{(d)} \in Y_i \tag{12}$$

It follows from (12) that if $\mathbf{c}_i \prec \mathbf{c}_j$ then the constituent points of $\mathbf{c}_i$ also weakly dominate $\mathbf{c}_j$.

Construction of a dominated tree from $|Z|$ points $Z = \{\mathbf{y}_m\}_{m=1}^{|Z|}$ proceeds as follows. The first composite point $\mathbf{c}_1$ is constructed by finding the individual $\mathbf{y}_m$ with maximum first coordinate; this value forms the first coordinate of the composite point:

$$\mathbf{c}_{1,1} = \max_{\mathbf{y}_m \in Z} (\mathbf{y}_{m,1}) \tag{13}$$

This individual $\mathbf{y}_m$ is now associated with $\mathbf{c}_1$ and removed from Z. Likewise the second coordinate of $\mathbf{c}_1$ is given by the maximum second coordinate of the points remaining in Z: $\mathbf{c}_{1,2} = \max_{\mathbf{y}_m \in Z \setminus \mathcal{T}} (\mathbf{y}_{m,2})$. This procedure is repeated to construct $\mathbf{c}_2$ and subsequent composite points until all elements of Z are associated with the tree. In general the dth coordinate of the ith composite point is given by:

$$\mathbf{c}_{i,d} = \max_{\mathbf{y}_m \in Z \setminus \mathcal{T}} (\mathbf{y}_{m,d}) \tag{14}$$

Note that in construction of the final composite point (that is, the composite point that dominates all other composite points) the $|Z|$ elements of Z may have been used before all the D coordinates of the final composite point $\mathbf{c}_L$ have been defined. The last remaining point in Z is reused to define the remaining coordinates (as shown in Figure 2).

It is clear from the construction of $\mathcal{T}$ that it possesses properties (10) and (12). Since (except possibly for the dominating composite point) D elements of Z are used in the construction of each composite point, the number of composite points in $\mathcal{T}$ is $L = \lceil |Z|/D \rceil$.

5 The new model

Unlike [1, 6, 11], in this study the selection of the global best for an individual in the swarm is based upon its closeness (in objective space) to an individual in the non-dominated set. Each swarm member is therefore concerned with improving a particular region of $\mathcal{E}$. This is achieved using the ordering of individuals caused by the composite point data structure discussed in Section 4. For any member of the swarm, X_i, the first non-dominated composite point, c^j, of the dominated tree is sought (i.e. where $c^j \nprec s \prec c^{j-1}$), this takes $\mathcal{O}(\lg(M+1))$ domination comparisons to find (where M is the number of composite points). The global best for an individual X_i is that archive member of the composite point c^j contributing the vertex which is less than or equal to the corresponding objective in X_i. An illustration of this is provided in Figure 3.

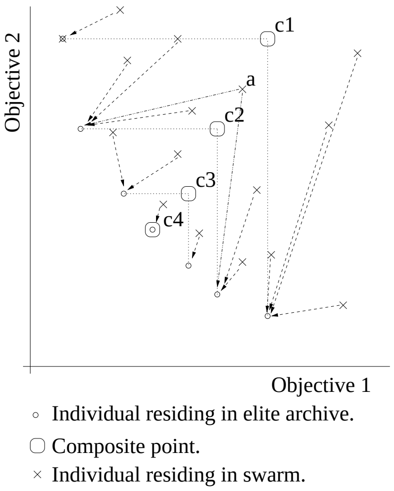
c1
a
c2
c3
c4
Objective 2
Objective 1
○ Individual residing in elite archive.
▢ Composite point.
× Individual residing in swarm.

Figure 3: Selection of global 'bests' for each swarm member.

In the case of a composite point c^j with more than one vertex less than or equal to the corresponding objectives of an individual X_i (as is illustrated in Figure 3 between composite point $c2$ and individual a) one of the vertex that meets the condition is selected at random to provide the global best (denoted in Algorithm 1 as $Z_{(i)} = (z_{(i),1}, \ldots z_{(i),N})$) for the individual X_i. The parenthesis around i in $Z_{(i)}$ in Algorithm 1 denote the individual selected from the set as the nearest to the ith member of X. They do not infer the ith individual of Z.

A set of local best individuals found is also maintained for each swarm member, instead of the single best for each member kept by [1, 6, 11] (which are prone to oscillation [3, 4, 5, 9]). The selection of a local best for an individual from the hyperset L (the set all of the local best sets) is slightly different to that used in global selection. Due to the relatively small number of estimated Pareto solution stored locally as opposed to globally, they are stored in linear lists, with uniform selection of local best $L_i = (l_{i,1}, \ldots l_{i,N})$. After initialisation of the two repositories (Z and L), the algorithm follows a standard swarm behaviour (as described in Equations 8 and 9).

Algorithm 1 Composite point based Multi-Objective PSO.

```
    Generation counter $t := 0$. Initialise the swarm population $X^t$, and update the non-dominated population $Z^t$ with non-dominated members of $X^t$.
    Initialise the local nondominated hyperset $L^t$ with members of $P^t$, $L^t_{i,1} := P^t$.
    Initialise the velocity set $V^t$, $V^t_i := 0 \; \forall i = 1, \ldots, |X|$.
1:  $t := t + 1$.
2:  Calculate new velocity of each particle. $v^t_{i,j} := wv^{t-1}_{i,j} + c_1 r_1 \left(l^{t-1}_{i,j} - x^{t-1}_{i,j}\right) + c_2 r_2 \left(z^{t-1}_{(i),j} - x^{t-1}_{i,j}\right) \; \forall i = 1, \ldots, |X|, \; \forall j = 1, \ldots, N$, where $r_1, r_2 \sim U(0,1)$.
3:  Accelerate the swarm members along their new trajectories, $X^t := X^{t-1} + \chi V^t$.
4:  Update nondominated global store $Z^t$ (see [4]), and local hyperset $L^t$.
5:  If termination rules are not met, go to 1.
```

5.1 Turbulence

During the early development of PSO [7], a stochastic variable called 'craziness' was used, such that Equation 8 read;

$$v_{i,j} := v^{\phi}_{i,j} + r_3, \tag{15}$$

where $v^{\phi}_{i,j}$ is the velocity of the jth parameter of the nearest neighbour to X_i and r_3 is the random craziness variable. As this early PSO developed and changed into the more familiar algorithms described in Section 3, this craziness parameter was dropped. In this study however we shall empirically validate the re-introduction of an extra stochastic variable within PSO (specifically in the multi-objective domain). In keeping with the overall design of the PSO, we refer to this term as turbulence (equivalent to perturbation in ES), as it reflects the change in a particle's flight which is out of its control. Where turbulence is used Equation 8 is replaced with

$$\begin{aligned} v_{i,j} := {} & wv_{i,j} + c_1 r_1 (p_{i,j} - x_{i,j}) \\ & + c_2 r_2 (p_{g,j} - x_{i,j}) + r_3. \end{aligned} \tag{16}$$

6 Experiments

The experiments in this study are designed to evaluate the new algorithm developed in this paper to existing models on a number of test functions previously described in the literature. In addition they are also designed to evaluate the benefit or otherwise of turbulence to multi-objective PSO models.

6.1 Comparative models

The first comparative model is based on the ES(1+1) PAES model of [8] with an unlimited archive. However, instead of grid based selection, the Partitioned Quasi Random Selection (PQRS) method of [4] is used. Both methods attempt to provide unbiased selection from the estimated front, PQRS is simply preferred in this case as grid knowledge need not be maintained and the method is easily integrated into the dominated and non-dominated tree framework. At each generation in PQRS one objective dimension is selected and partitioned into $Q-1$ bins of equal width (with an extra bin containing the best individual in that dimension). To select a representative from the archive first one bin (or the best solution) is selected uniformly to ensure that there is no bias toward dense areas of the front, and then an individual is uniformly selected from the bin. This is easily implemented by maintaining D balanced binary trees of the archive individuals in each objective dimension. Selection then follows randomly generating a number that lies in a chosen bin's range and selecting the nearest tree member. A more detailed description can be found in [4]. The second model is based on Coello and Lechunga's multi-objective PSO [1] with an unlimited archive and selection from PQRS. This second model uses the biased roulette wheel selection from [1] for bin selection.

6.2 Comparison measure

Results were compared using a method similar to the $\mathcal{V}$ measure from [4] and the performance measure used in [10]. $\mathcal{V}^{\mathcal{P}}$ is a measure of the multi-objective error volume that is dominated by the true Pareto front but not the estimated Pareto front. Loosely $\mathcal{V}^{\mathcal{P}}$ is the fraction of the volume of a hypercube containing $\mathcal{P}$ ($H_{\mathcal{P}}$) that is strictly dominated by $\mathcal{P}$ but is not dominated by members of $\mathcal{E}$. The $\mathcal{V}^{\mathcal{P}}$ measure is easily estimated by Monte Carlo sampling of $H_{\mathcal{P}}$ and counting the fraction of samples that are dominated exclusively by $\mathcal{P}$ and dividing by the number of samples dominated by $\mathcal{P}$. The hypercube bounds are determined by the $\mathcal{P}$ range of f_1 and the $\mathcal{P}$ range of $f_2 + 3.0$. This allows direct comparison of the $\mathcal{V}^{\mathcal{P}}$ measure across all models. The lower the $\mathcal{V}^{\mathcal{P}}$ the lower the hypercube volume exclusively dominated by the true Pareto front, and the nearer the estimated front to the true front. 250000 samples were taken for Monte Carlo estimates, and $\mathcal{P}$ was represented by 250 randomly drawn members of $\mathcal{P}$.

6.3 Test Functions

The test functions introduced in [14] are used here. The two dimensional objective functions take the form:

$$\begin{array}{lll}
\text{Minimise} & T(\mathbf{x}) & = (f_1(x_1), f_2(\mathbf{x})), \\
\text{where} & f_2(\mathbf{x}) & = g(x_2, \ldots, x_N) \\
 & & \cdot h(f_1(x_1), g(x_2, \ldots, x_N)), \\
\text{and} & \mathbf{x} & = (x_1, \ldots, x_N).
\end{array}$$

A description of the four test functions used can be found in Table 1.

#	Function
1	$f_1(x_1) = x_1$, $g(x_2, \ldots, x_N) = 1 + 9\left(\sum_{n=2}^{N} x_n\right)/(n-1)$, $h(f_1, g) = 1 - \sqrt{f_1/g}$.
2	$f_1(x_1) = x_1$, $g(x_2, \ldots, x_N) = 1 + 9\left(\sum_{n=2}^{N} x_n\right)/(n-1)$, $h(f_1, g) = 1 - (f_1/g)^2$.
3	$f_1(x_1) = x_1$, $g(x_2, \ldots, x_N) = 1 + 9\left(\sum_{n=2}^{N} x_n\right)/(n-1)$, $h(f_1, g) = 1 - \sqrt{f_1/g} - (f_1/g)\sin(10\pi f_1)$.
4	$f_1(x_1) = x_1$, $g(x_2, \ldots, x_N) =$ $1 + 10(n-1)\sum_{n=2}^{N}(x_n^2 - 10\cos(4\pi x_n))$, $h(f_1, g) = 1 - \sqrt{f_1/g}$.

Table 1: Test functions from [14] used in this study.

For test functions 1-3, $N = 30$, $x_i \in [0,1]$ and for test function 4 $N = 10$, $x_1 \in [0,1]$, $x_2, \ldots, x_N \in [-5,5]$.

6.4 Algorithm implementation

The implementation of all the models use floating point representation of parameters in the decision vectors. In order to compare the new multi-objective PSO technique, each MOA was executed 25 times on each test problem, and the resultant non-dominated solutions saved at the end of each run. For each simulation the ES was run for 4000 generations with a mutation rate of 0.2 and the PSO models were run for 200 generations with swarms of size 20. The turbulence (perturbation) variable for all models was $\sim N(0, 0.1R)$, where R is the absolute range of the model parameter. In each of the 25 different runs the algorithms were initialised from identical decision vector populations of size 20, with the non-dominated individuals residing in these populations forming the initial elite archives. Initialisation of decision vectors was from Uniform distributions, over the range of the chromosome parameters for the particular test function. The experiments were repeated with and without turbulence, with $c_1 = c_2 = \chi = 1$ and with w set at 0.4 (as used in [1]) and 0.8. Turbulence probability was fixed at 0.2. $Q = 20$.

7 Results

Table 2 shows the results of these experiments. The use of turbulence is seen to significantly increase the performance of both the multi-objective PSO algorithms across the test functions, as does the use of a higher w value. In addition the new multi-objective PSO algorithm can be seen to be significantly better than the ES method and the competing multi-objective PSO method when using turbulence and a high w on the first three test functions. The new multi-objective PSO also tends to be better than the competing multi-objective PSO even when no turbulence is used and with lower w values (sub-optimal parameter settings).

The fourth test function however, with multimodality (21^9 local Pareto-optimal fronts [14]) causes great problems for both multi-objective PSO algorithms, with none of the estimated fronts from these models anywhere near the true Pareto front[1]. This is due to the function design, where to pass through local Pareto fronts, a swarm member may have to fly in a direction opposite to its local and global best (in parameter space).

		$\mathcal{V}^{\mathcal{P}}$ (%)		
#	w , T	ES	P$'$	P$''$
1	0.4 No	**15.0** (1.3)	55.0 (6.7)	*32.3* (8.0)
	0.8 No	*15.0* (1.3)	20.1 (7.6)	**3.7** (3.8)
	0.4 Yes	15.0 (1.3)	**3.0** (0.5)	*3.6* (0.7)
	0.8 Yes	15.0 (1.3)	*1.2* (0.3)	**0.7** (0.1)
2	0.4 No	**12.0** (1.2)	65.6 (7.8)	*60.9* (8.5)
	0.8 No	**12.0** (1.2)	27.8 (5.1)	31.2 (10.2)
	0.4 Yes	12.0 (1.2)	*6.7* (0.8)	**5.8** (0.9)
	0.8 Yes	12.0 (1.2)	*4.4* (1.1)	**1.6** (0.5)
3	0.4 No	**8.2** (1.0)	32.2 (3.3)	*12.1* (2.2)
	0.8 No	*8.2* (1.0)	17.3 (4.9)	**3.1** (2.6)
	0.4 Yes	8.2 (1.0)	*2.8* (0.9)	*3.1* (0.6)
	0.8 Yes	8.2 (1.0)	*1.3* (0.5)	**0.7** (0.2)
4	0.4 No	**68.1** (16.5)	100 (0.0)	100 (0.0)
	0.8 No	**68.1** (16.5)	100 (0.0)	100 (0.0)
	0.4 Yes	**68.1** (16.5)	100 (0.0)	100 (0.0)
	0.8 Yes	**68.1** (16.5)	100 (0.0)	100 (0.0)

Table 2: Algorithm results. ES refers to the multi-objecive ES(1+1) model, P$'$ refers to the multi-objective PSO model based on [1] and P$''$ refers to the multi-objective PSO method developed in this study. Means highlighted in bold are significantly better than both competing models (using the Wilcoxon Signed Ranks Test at the 5% level, 2.5% in each tail). Means in italics are significantly better than one other competing model. # is the test function number, w the inertia and T refers to whether turbulence is present.

8 Discussion

A new method for selecting the best global and local individuals for multi-objective PSO swarm members has been proposed in this study. This new approach is based on a concept of closeness to members of the global set, and maintaining a set of local best solutions for each swarm member. It has been shown to be significantly better than the methods used in a recent alternative multi-objective PSO and an existing

[1] It is interesting to note that, although not discussed in the original study, during its presentation this problem was also noted by Coello on results not reported in [1].

MOEA. It has also been demonstrated that the use of a stochastic turbulence variable can be a significant aid to general multi-objective PSO. However this approach does have deficiencies. Clearly if there is little or no relationship between 'closeness' in objective space and 'closeness' in parameter space multi-objective PSO methods (and PSO methods in general) may experience problems.

In addition multi-objective PSO (and MOEA methods as a whole) needs additional validation on multi-objective problems of $D > 2$ dimensions. As the number of objective dimensions D increases the global best individual returned by the new selection method may not be the nearest - indeed it may only it be the $(D-1)$th nearest.

Current research interests of the authors include the validation of the methods introduced here on high D test problems, and the comparison of PSO to ES and GA approaches to multi-objective Neural Network training.

Acknowledgements

Jonathan Fieldsend gratefully acknowledges support from Invensys Climate Controls Europe and an Exeter University Studentship.

References

[1] C.A.C. Coello and M.S. Lechunga. MOPSO: A Proposal for Multiple Objective Particle Swarm Optimization. In *Proceedings of the IEEE World Congress on Computational Intelligence*, Hawaii, May 12-17, 2002. IEEE Press.

[2] K. Deb, S. Agrawal, A. Pratap, and T. Meyarivan. A Fast Elitist Non-Dominated Sorting Genetic Algorithm for Multi-Objective Optimization: NSGA-II. In *Proceedings of Parallel Problem Solving from Nature - PPSN VI*, pages 849–858. Springer, 2000.

[3] R.M. Everson, J.E. Fieldsend, and S.Singh. Full Elite Sets for Multi-Objective Optimisation. In I.C. Parmee, editor, *Adaptive Computing in Design and Manufacture V*, pages 343–354. Springer, 2002.

[4] J.E. Fieldsend, R.M. Everson, and S.Singh. Using Unconstrained Elite Archives for Multi-Objective Optimisation. *IEEE Transactions on Evolutionary Computation (Submitted)*, 2001.

[5] T. Hanne. On the convergence of multiobjective evolutionary algorithms. *European Journal of Operational Research*, 117:553–564, 1999.

[6] X. Hu and R. Eberhart. Multiobjective Optimization Using Dynamic Neighborhood Particle Swarm Optimization. In *Proceedings of the IEEE World Congress on Computational Intelligence*, Hawaii, May 12-17, 2002. IEEE Press.

[7] J. Kennedy and R. Eberhart. Particle Swarm Optimization. In *Proceedings of the Fourth IEEE International Conference on Neural Networks*, pages 1942–1948, Perth, Australia, 1995. IEEE Service Center.

[8] J.D. Knowles and D. Corne. Approximating the nondominated front using the pareto archived evolution strategy. *Evolutionary Computation*, 8(2):149–172, 2000.

[9] M. Laumanns, L. Thiele, K. Deb, and E. Zitzler. On the Convergence and Diversity-Preservation Properties of Multi-Objective Evolutionary Algorithms. Technical Report TIK-Report 108, Swiss Federal Institute of Technology Zurich (ETH), June 2001.

[10] M. Laumanns, E. Zitzler, and L. Thiele. A Unified Model for Multi-Objective Evolutionary Algorithms with Elitism. In *Proceedings of the 2000 Congress on Evolutionary Computation*, pages 46–53, Piscataway, NJ, 2000. IEEE Service Center.

[11] K.E. Parsopoulos and M.N. Vrahatis. Particle Swarm Optimization Method in Multiobjective Problems. In *Proceedings of the 2002 ACM Symposium on Applied Computing (SAC 2002)*, pages 603–607, 2002.

[12] J.D. Schaffer. Multiple objective optimization with vector evaluated genetic algorithms. In *Proceedings of the First International Conference on Genetic Algorithms*, pages 99–100, 1985.

[13] Y. Shi and R. Eberhart. Parameter Selection in Particle Swarm Optimization. In *Proceedings of the Seventh Annual Conference on Evolutionary Programming*, pages 591–601, 1998.

[14] E. Zitzler, K. Deb, and L. Thiele. Comparison of Multiobjective Evolutionary Algorithms: Empirical Results. *Evolutionary Computation*, 8(2):173–195, 2000.