2014 International Joint Conference on Neural Networks (IJCNN)
July 6-11, 2014, Beijing, China

Enhancing MOPSO through the guidance of ANNs

Timothy Rawlins, Andrew Lewis, Jan Hettenhausen and Timoleon Kipouros

***Abstract*— In existing work, Artificial Neural Networks (ANNs) are often used to model objective functions for Multi-Objective Particle Swarm Optimisation (MOPSO) or MOPSO is used to aid in ANN-training. We instead use an ANN to guide the optimisation algorithm by deciding if a trial solution is worthy of full evaluation. This should be particularly helpful for computationally expensive calculations. We also introduce a level of scepticism to the result produced by the ANN, to account both for inaccuracy in the ANN and the loss of performance in a MOPSO if the reinitialisation of particles is too extreme.**

As a case study we used a multi-objective optimisation problem that seeks to optimise the shape of an airfoil to minimise drag and maximise lift. We evaluated several different methods for training an ANN: pre-training vs live training, continuous vs single training, and varied initial training set size. For applying the ANN's output to MOPSO we looked at various levels of scepticism and verified ANN quality before applying it.

Attainment surfaces were then used to compare the performance of guided and unguided MOPSOs. Our analysis showed the performance of guided MOPSO was significantly better than unguided MOPSO . We further analysed the results to derive guidance for selecting appropriate variations for specific problems.

I. Introduction

In this paper we introduce a novel hybrid optimisation approach, combining an Artificial Neural Network (ANN) and Multi-Objective Particle Swarm Optimisation (MOPSO), with the goal of using an ANN to increase the performance of the optimisation, where a known objective function exists, without sacrificing correctness. This is motivated, in particular, by complex industrial problems. These problems generally have existing functions to evaluate the performance of a trial solution, but these evaluations are slow (with individual evaluations taking hours or even days). It is also necessary for the results found to have a high degree of accuracy, which makes substituting approximations for the evaluation function cumbersome. During the course of optimisation, MOPSO uses promising results from trial solutions to guide the search algorithm. Without a high degree of confidence in the correctness of approximations, all trial solutions classified as "promising" or "near-optimal" must be checked. Implementing this increases code complexity and reduces the advantage of using surrogate function values. Despite this, in most current research where ANNs are used to supplement MOPSO, the ANN is used as a replacement for the objective function or a part thereof, because either no evaluation function is known, or because results do not have to be precise and the speed increase is deemed worth the loss. Examples of these techniques can be seen in the work of Amiryousefi [1], in which an ANN is used to model Deep Fat Frying and MOPSO is then used to find the optimum parameters to minimise shrinkage and fat content, or Mukhopadhyay [2] who uses a MOPSO based approach to select which solutions are classified by an ANN, and the sensitivity and specificity of the labelling are used as the objectives. Similarly, Li *et al.* use an ANN to model an Industrial Cracking Furnace and a MOPSO variant that focuses on finding the best solutions to maximise the objectives of production of 2 gases [3]. Another interesting case is the use of PSO to train an ANN which is then used as the objective function [4].

In a related area, MOPSO and ANNs are also combined by using MOPSO as a method to train an ANN, such as in the work of Qasem and Shamsuddin [5] [6] or Yusiong and Naval [7]. There is also the variant used by Feng *et al.* [8] where a MOPSO is used to select Pareto-optimal fuzzy rules used to select training set data for an ANN.

We have taken a different approach. The problems of interest are complex industrial design applications with computationally expensive object functions. However, since accurate results are crucial we do not attempt to provide a surrogate objective function but instead use an ANN as a fast estimator to determine if it is worthwhile to spend the computational power to fully evaluate a trial solution suggested by MOPSO. The parameters of the trial solution are regenerated if they are rejected. It should be emphasised that the "estimation" does *not* involve any attempt to provide an estimate of the values for objective functions, but only judge a trial solution given knowledge of previous results to provide a simple, Boolean function of "promising" or not.

However, the regeneration range is limited to the results produced by regenerating the random components of the velocity function as opposed to generating a completely new position. This limits the "diameter" of the neighbourhood in which the solution can be generated. Also, considering that an ANN's evaluation can be wrong, we make use of a scepticism-based approach, in which we ignore a negative result returned by the ANN a pre-set percentage of the time. This allows us both to avoid infinite loops when all possible regenerated parameters are unacceptable to the ANN (or if a poor training set causes the ANN to reject everything), and balance the "opinions" of the MOPSO and the ANN, in the case where there is disagreement. The structure of our problem, where there are more negative results than positive, means that there is little reason to be sceptical of positive results. However, the approach is easily altered to apply also to positive results depending on the relative costs for

Rawlins, Lewis and Hettenhausen are with Griffith University, Australia (email: tim.rawlins@griffithuni.edu.au, a.lewis@griffith.edu.au, j.hettenhausen@griffith.edu.au). Kipouros is with Cambridge University, UK (email: tk291@cam.ac.uk), and has an adjunct appointment with Griffith University's Institute for Integrated and Intelligent Systems.

978-1-4799-1484-5/14/$31.00 ©2014 IEEE

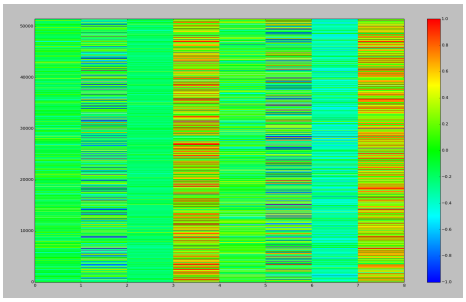

Fig. 1. Heatmap of parameter values of feasible solutions.

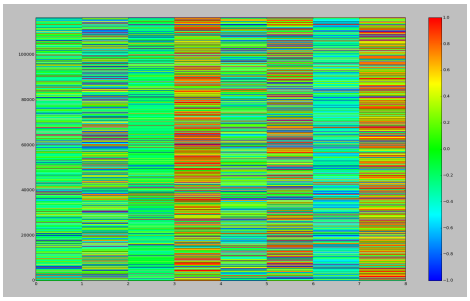

Fig. 2. Heatmap of parameter values of infeasible solutions.

other problems. This allows us a more efficient optimisation process (since less time is spent on poor results) without sacrificing accuracy, since all the results found are the result of full evaluation.

II. Problem - Optimisation of Airfoil at Stall Angle

For our test problem, we are looking at an airfoil optimisation problem. This problem has an 8-dimensional parameter space, corresponding to the change in x and y positions of 4 control points of the free form deformation technique used to manipulate the airfoil shape. The objectives were to maximised lift and minimise drag, as calculated by Xfoil [9]. To simplify the calculation, we minimised negative lift. Many solutions to this problem are considered infeasible due to violating design constraints or not functioning at an angle of attack of 15 degrees ("stall angle".) We elected to use this problem because it is reasonably complex to calculate, can serve as a a benchmark for more complex aerodynamics problems and because we have a relatively large pool of existing evaluations from prior experiments (167946 evaluations).

As using an ANN for this purpose is new, we elected to use a simple binary classification of validity: feasible points are considered positive, while infeasible points are considered negative. This results in simple labelling. Since our results are drawn from previous work we are confident there exists some underlying non-random relationship between location in parameter space and solution feasibility. Figure 1 and Figure 2 show colour maps of the parameter values of feasible and infeasible solutions respectively. In the figures, columns represent parameters, rows represent sets of values known to result in a feasible result, and the colour represents an approximation of the numeric value. It is clear from a side by side comparison of parameter values for feasible/infeasible results that consistent differences exist. For the feasible results, a number of parameters show a uniformity of parameter values over a reduced range. Infeasible results show a tendency to be more widely distributed and random; there are many ways a design can fail, and only a few ways it can excel.

In this problem there is significantly more infeasible space than feasible, as from our existing data set only 51481 trial solutions were valid, and the remainder (116465) were invalid.

It is important to note that this data is drawn from all evaluations done by complete MOPSO optimisations. If live training is performed while running MOPSO, the initial results are more likely to be infeasible and an ANN that rejects an excessive amount of (or even all) parameters is possible. This is another reason that we cannot automatically trust negative results from the ANN.

III. Tools

Table I summarizes both the initial values used and various other parameters that remain unchanged through our tests. The following sections provide further detail.

Parameter	**Value**
Neural Network	
Hidden Layers	100
Hidden Nodes	10
Regularisation Factor	1
Max Training Iterations	300
Normalisation Before Training	Yes
Initial Seed	Python `random` library
Learning Mechanism	Minimizing L2-regularised cost function using L-BFGS
Multi-Objective Particle Swarm Optimisation	
Particles	60
Iterations	100
Intertial Weight	0.4
Cognitive Weight Coefficent	2
Collective Motion Weight Coefficient	2
Archive Size	60
Problem Specific	
Initial Parameter Generation	Uniform from -1.0 to 1.0
Parameter Constraints	Constantly constrained to between -1.0 to 1.0
Regeneration	Recalculate equations of motion. For 0 scepticism see III-C.4

TABLE I

Summary of parameters values used.

A. Artificial Neutral Network

For this work we made use of the Neural Network implementation of Demšar *et al.* [10]. This is an implementation of a multilayer perceptron with parameters described in Table I. The initial seed was randomly generated for each training of the neural net.

B. Multi-Objective Particle Swarm Optimisation

MOPSO is a multi-objective extension of Particle Swarm Optimisation (PSO). PSO is an optimisation technique based on a simplified version of bird flocking. It was introduced by Kennedy and Eberhart [11]. The addition of an inertia weight term was made by Shi and Eberhart [12] to balance the role of local and global search. The typical implementation of a standard PSO can be visualised as a swarm of particles that move through (n-dimensional) parameter space guided by the location of the global best result found by the swarm, their individual memories of the location of their personal best results, and by their inertia.

MOPSO is an extension of PSO in which objective space is also n-dimensional, i.e there are two or more (often competing) objectives to optimise. A consequence of this is that there is no longer a single "best" solution but rather a set of trade-off solutions that are referred to as Pareto-optimal solutions [13]. Briefly speaking, in Pareto-dominance a solution A is said to dominate a solution B, if solution A is strictly better than solution B in at least one objective and A is at least as good as B in all other objectives. The Pareto-optimal set is that set of solutions which are not dominated by another solution in the search space of the problem. Extending PSO to multiple objectives was handled by adding an archive of Pareto-dominant solutions from which a global "best" solution was chosen using roulette wheel selection. In addition, the personal best was modified to be a single non-dominated solution [14].

Each particle in the swarm has a position and a velocity. The equations of motion that govern the movement of a particle in a swarm are:

$$\begin{aligned} V_{t+1} =& (W * V_t) + R_1 * c_1 * (PBest_t - X_t) \\ &+ R_2 * c_2 * (GBest - X_t) \end{aligned} \tag{1}$$

$$X_{t+1} = X_t + V_{t+1} \tag{2}$$

Where V is the particle velocity, W is inertial weight, R_1 and R_2 are random values between 0 and 1, c_1 is the cognitive component weight (the weight is given to the particles' own search results), c_2 is the collective component weight (the weight given to the swarm's search results), PBest is the "best" result found by the particle, GBest is a randomly selected member of the archive of best results found by the swarm and X is the particle position.

For our MOPSO implementation, we implemented a MOPSO based on the original proposal [14]. Parameters are given in Table I. We have chosen to use 60 particles and 100 iterations in order to maintain the same number of evaluations (6000) used in the reference Tabu Pareto-front provided to us for comparison [15]. We chose 60 particles specifically because past experimentation has suggest this is a value with good general performance.

C. Structure of Hybrid Algorithm

In its most basic form we use an ANN to guide the movement of particles in the MOPSO by running an ANN in parallel to the MOPSO. The initial training of the ANN is done before initialising particles in the MOPSO, if an archive is available, as these particles also need to be checked for validity. After this, live training of the ANN can be performed in parallel with the evaluation of objective function for the swarm, as the training time for the ANN is much shorter than the evaluation time.

Before a change in a particle's velocity or position is calculated, we first record the previous values. We then calculate the new velocity and position in parameter space. Before evaluation of the parameters, we pass the particle's parameters to the ANN, for classification as either valid or invalid.

If the ANN classified the particle as valid the MOPSO proceeds normally. If the particle is instead classified as invalid, a two step process takes place. Firstly, for some pre-defined scepticism percentage we treat the particle as if it is valid and allow the MOPSO to proceed normally. Secondly, if this does not occur we instead recalculate the particle's position and velocity. Because there is a random factor in the selection of the guide particle, and the variable relative weighting of the personal best result and guide particle's best result, this will result in a new set of parameters, which then must be tested. This continues until a set of parameters for the particle is accepted (either by the ANN or by scepticism).

For the initialisation of particles, because there are no previous values to regenerate from, a small change is made. In this case particles that fail both ANN and scepticism are reinitialised according to the normal rules of the MOPSO but the process is otherwise the same.

Because this approach is new, we make use of several variations of it, in order to find an appropriate version for our problem. Table II summarises the the parameters we are changing and the values they take (as well as abbreviations used).

Parameter	Values
Archive Use	a(rchive)
	l(ive)
Training Frequency	c(ontinuous)
	s(ingle)
Initial Training Set Size	500
	1800
	7000(archive only)
Scepticism	0
	0.15
	0.20
Verification	v(erification)
	n(o verification)

TABLE II

SUMMARY OF VARIABLE EXPERIMENTAL PARAMETERS, POSSIBLE VALUES AND ABBREVIATIONS

1) Initial Training from the Archive versus Live Training: An important consideration for our problem is whether establishing a well-distributed training archive in advance is

worthwhile, compared to training from results gathered into an archive by the optimisation as it progresses. A training archive cannot be assumed to exist and is often expensive to generate. However, in optimisation problems early results are often not representative of the search space in the optimal area. To take full advantage of a pre-existing archive we make sure any sample set is well distributed by requiring the distribution of valid and invalid particles matches that of the overall archive.

2) Continuous Training vs Single Training: Another consideration is whether time should be spent retraining the ANN after each iteration of the MOPSO, or if it is just as effective to train the ANN only once and use that to guide the entire optimisation. After some practical experimentation we concluded that the time to retrain the ANN after each iteration is minimal compared to the evaluation time of the problems for which it was designed. However, continuous training may not yield a benefit over single training. For example, continuous training may result in "overtraining" on the area currently being explored if the initial training set is small.

3) Size of Training Set: The "size" of the initial training set (and thus what is being tested) depends on other parameters. When training from a pre-existing archive it is how many points are drawn from the archive. For single live training, it is the size of the training set (i.e. the first n unique results found by the optimisation process), but for continuous live training, it instead acts as the minimum size of the training set; if more results are available at the time they will be used.

Before we began this experiment we had performed analysis, training an ANN with different sized training sets in order to test if training from small data sets was at all viable. From this we determined that potentially good training set sizes were between 500 (easy to obtain, reasonable results), 7000 (training sets larger than this yielded little improvement) with 1800 as a compromise.

4) Level of Scepticism: This parameter represents our level of doubt in the correctness of an an "invalid" result from the ANN. From our initial analysis we believed that the appropriate level of scepticism was based on the size of the initial training set, but we wished to verify this. We chose scepticism levels of 15% (likely to be appropriate for our large training set) and 20% (likely to be appropriate for our smaller training set.)

In order to perform a fair test we also include a 0% scepticism level. However, for practical reasons (avoiding "hanging"), we cannot have nil scepticism. We instead recalculate the value normally until we have been successively rejected a number of times equal to twice the number of entries in our archive. After this we instead use a slowly increasing Gaussian distribution around our initial position to generate new parameters, with $\sigma = 0.01 * (1 + m)$ where m is the number of evaluations in excess of twice the length of the archive, and σ is not permitted to exceed 1. Sigma has the standard meaning for generating random numbers with Gaussian distribution. This formula causes the area that the regenerated particle is likely to be in to slowly expand as the number of times it needs to be generated increases. This attempts to "escape" an infeasible region of parameter space. The cap at 1 exists to prevent overly large values which would result in exploration being "trapped" exploring only the edges of parameter space. Furthermore, to avoid hanging in the case of a universally rejecting neural net, we also accept the last set of parameters generated if 900 successive sets of parameters generated with $\sigma = 1.0$ are rejected.

5) Verification of Neural Network Before Use: With this variant we added a verification set. This set is half the size of the training set, and is used to determine if the trained ANN: 1) has predictions that have some correlation to the real results (as determine by having a Matthew's Correlation Coefficient (MMC) of > 0.1) and 2) has classified at least 1 member of the verification set as valid. If either of these conditions is not met , the ANN is not used this round, and does not count as having been trained for the purpose of single-training.

From this we can guarantee that the ANN is better than a (well-distributed) random guess and that some parameter combinations will be accepted. However, more data are required, as they are split between training and verification sets. For live training we need 50% more results to perform verification and thus the ANN is delayed further before it can begin classification.

D. Notes

Due to practical limitations, not all combinations of the above variants were tested. It is not possible to perform live optimisation with a minimum training set of 7000 results from our MOPSO, as it generates at most 6000. We also include an "unguided" ANN, which has a similar structure to our hybrid approach, but never trains an ANN and always returns valid. This is used as a baseline comparison to see how our ANN's performance compares to that of a normal MOPSO.

Our full code is available on request.

E. Attainment Surfaces

Having determined the variants to compare, a method by which to compare the results was needed. We initially experimented with some numeric methods for evaluating convergence and coverage, but these proved difficult to interpret and there was little agreement, even between numeric methods with similar goals. We attribute this to there being distinct regions of parameter space that contribute different areas of the Pareto-front in objective space. As such, some runs seem to focus on a region of the objective space that generates good drag results at the expense of lift, while other runs concentrate on an area covering good lift. Since these regions are both on the Pareto-front and have little overlap, they are mutually non-comparable.

Instead we chose to look for a more easily interpretable method, and decided to make use of a variant of attainment surfaces [16] which are a qualitative, visual comparison

method. Strictly speaking attainment surfaces use a worst case middle point to join two results. We have instead used the common practice of assuming piecewise linear interpolation between two points. This can be problematic in the case of certain deceptive or multi-modal Pareto-fronts but from our prior experience we know this is not the case in this problem. We primarily used the median (50% ,2nd Quartile) attainment surface over 4 runs of each variant to compare their average outcome. We also used the 1st Quartile (25%), 3rd Quartile (75%) and overall "Front" (100%, 4th Quartile) attainment surfaces to assess the reliability of each variant. For clarity, we note that the numeric order of the quartile points refers to the percentage of results that achieve a certain point, therefore the 1st Quartile attainment surface shows the worst performance, and the 4th Quartile the best. For consistency attainment surfaces on our diagrams are all labelled by quartile (1q = 1st Quartile, 2q = 2nd Quartile, etc.) To calculate our attainment surface all data were first normalised using the best results in each objective from the combination of our training set and all Pareto-optimal results from these experiments.

F. Process

Because our hybrid method is non-deterministic we performed 4 runs of each variation, in order to obtain some estimate of the average performance of each variant. After this an approximation of the true Pareto-front was calculated from the combination of our training set and results from all variants of the experiment (including results from unguided algorithms.) Results from the Pareto-fronts found by each run of each variant, as well as the overall Pareto-front were normalised. A normalised version of the results from a previous experiment using Tabu search was used as a benchmark for our results.

Attainment surfaces at 25%, 50%, 75% and 100% were calculated using 60 equally spaced rays emerging from the origin, to place a linear order on the results from the runs of each variant. The first quartile, median, and third quartile were then calculated normally, and the first Pareto-front encountered on each ray was used for the 100% surface. The mean of the median (2q), and the mean Interquartile Range (3q - 1q) of each variant discussed are also included with the results in Table III.

IV. Results and Analysis

Due to the large number of variants, multiple runs of each variant, and up to 60 results each with 8 parameters and 2 objectives for each run, it is not possible to include the full results (or even all attainment surface comparisons) in this paper. We will instead use a selection of the results that show both the good and poor performance of our algorithm, look at the variability of the results of particular reliable or unreliable variants, and attempt an explanation for their behaviour.

In our analysis we pay special attention to potential application of this technique to a jet engine compressor blade shape optimisation problem, in which the most interesting results are in the centre of the Pareto-front. For this purpose

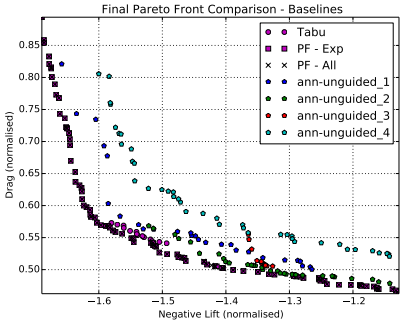

Fig. 3. Comparison of Pareto Front from this experiment to total Pareto-Front, and to our unguided ANN

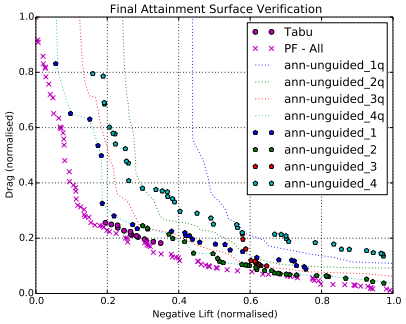

Fig. 4. The First Quartile (1q), Second Quartile (2q), Third Quartile (3q) and Fourth Quartile (4q) results for the Unguided MOPSO. Each run (after normalization) is also shown, to demonstrate the relationship between runs and the quartiles.

we consider the 50% attainment surface shows a reasonable combination of the performance and reliability of the technique. In other problems where either optimisation is greatly more important than reliability or reliability is highly important it may be more appropriate to use the third quartile or first quartile attainment surfaces respectively.

An interesting early observation is that our overall (composite) Pareto-front is derived primarily from results from the algorithms tested in this experiment with only minimal contributions from our training set of existing values, and none at all from the unguided MOPSO (Fig. 3). This strongly suggests that guidance from the ANN has helped some variants to outperform the unguided MOPSO.

As a baseline we also needed to establish the variability between runs in an unguided MOPSO. Figure 4 also demonstrates how the attainment surfaces related to the runs. Of particular note is how the lack of results with high lift in the 3rd run has degraded the high lift / high drag portion of the First Quartile attainment surface.

Comparison of the median attainment surface for each variant with the attainment surface of the unguided MOPSO further strengthens our belief that the guidance has helped. Of the 60 variants (24 each at 500 and 1800 initial training set, and 12 at 7000), 29 variants have a median attainment surface that is superior to the unguided MOPSO, 26 are competitive with it (i.e. each dominates some areas that the other does

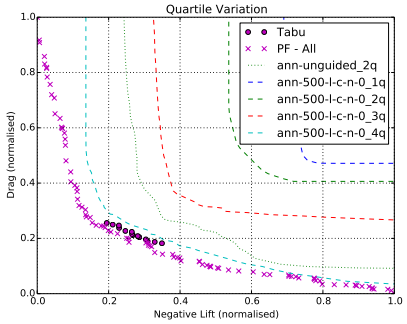

Fig. 5. The First Quartile (1q), Second Quartile (2q), Third Quartile (3q) and Fourth Quartile (4q) results for the 500 Initial Training Set, Live, Continuous, No Verification, No Scepticism variant.

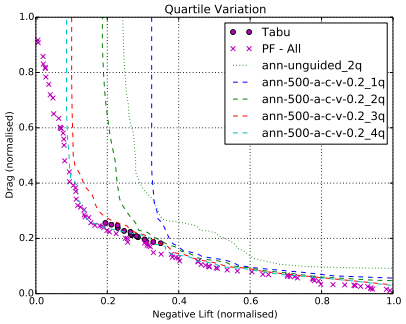

Fig. 6. The First Quartile (1q), Second Quartile (2q), Third Quartile (3q) and Fourth Quartile (4q) results for the 500 Initial Training Set, Archive, Continuous, Verification, 20% Scepticism variant.

not) and only 5 are worse. This appears to confirm that ANN guidance has provided assistance. All 5 of the strictly worse cases use nil scepticism. In addition, all used smaller (500 and 1800) training sets. For the competitive variants, the great majority used smaller training sets, and several used nil scepticism. This is generally what would be expected, as smaller initial training sets could be expected to result in a poorer network, and lacking scepticism to have a greater impact as a result of this.

Archive	Continuous	Verification	Training Set	Scepticism	Mean Median	Mean IQR
True	False	False	1800	0.2	(0.3486, 0.6151)	(-0.1631, -0.2156)
True	True	False	500	0.2	(0.3201, 0.3788)	(-0.0719, -0.1721)
True	True	False	1800	0.2	(0.3991, 0.7500)	(-0.2424, -0.6403)
True	True	False	7000	0	(0.3304, 0.3987)	(-0.0210, -0.0237)
True	True	True	500	0.2	(0.3424, 0.5732)	(-0.1197, -0.5306)
False	False	True	500	0.15	(0.3630, 0.5046)	(-0.0652, -0.1737)
False	True	False	500	0	(1.3050, 1.5655)	(-0.7607, -1.0279)
Unguided					(0.4548, 0.7077)	(-0.2563, -0.7827)

TABLE III
THE MEAN MEDIAN/2Q VALUES AND MEAN INTERQUARTILE RANGE (3Q-1Q) OF THE VARIANTS DISCUSSED

There are 4 variants with initial training sets of 500 that we find particularly interesting. These are the Live Continuous, No Verification with no scepticism (Fig. 5), the Archive Continuous with 20% scepticism both with (Fig. 6) and without verification (Fig. 7) and Live Single, Verification with 15% scepticism variants (Fig. 8). Considering first Figure 5, it shows a good front but the significant differences between the front and the quartiles make it unsuitable for practical use. These features suggest that only a single good run existed, but it was very good. This is indicated also by its poor Mean Median (indicating poor performance) and large IQR (indicating poor reliability). This strongly suggests that the good result is merely a result of random chance and should not be relied on for practical use.

Moving to consideration of the more interesting figures, it is clear from the large IQR corresponding to Figure 6 that it is less reliable than the other two variants. It also

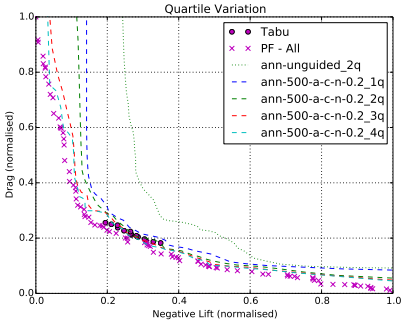

Fig. 7. The First Quartile (1q), Second Quartile (2q), Third Quartile (3q) and Fourth Quartile (4q) results for the 500 Initial Training Set, Archive, Continuous, No Verification, 20% Scepticism variant.

shows that over 4 runs it also generates a well distributed coverage of the front, but fails to track the overall Pareto-front in the best optimal lift area. Figure 7 also shows this problem but to a reduced extent. Of particular interest to us is how well it covers the central area of the true Pareto-front (where the Tabu search results are located), where it has the most even performance. It lacks the sharp inflection apparent in Figure 6, at the cost of not optimising drag as well, and the degraded values of lift for high drag cases visible in Figure 8. This makes it an interesting choice if reliability is not a high priority.

Considering particularly the median attainment surface, we see that Figure 8 clearly has the most reliable performance (which is reflected in its low IQR). In particular, the other two figures have problems reliably optimising the best lift results at this reliability threshold. This leads us to select Live Single, Verification with 15% scepticism (Fig. 8) as the best performing variant of these for our purposes, as its high degree of reliability comes with only a small cost to the total coverage of the front over multiple runs. Since the problems we wish to apply this technique to have high computational complexity we deem the higher reliability worth the small loss in convergence.

Although more of the variants based on an 1800 initial training set than those based on an initial training set of 500 perform well compared to an unguided MOPSO, the best

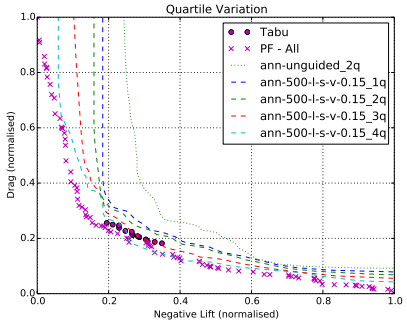

Fig. 8. The First Quartile (1q), Second Quartile (2q), Third Quartile (3q) and Fourth Quartile (4q) results for the 500 Initial Training Set, Live, Single, Verification, 15% Scepticism variant.

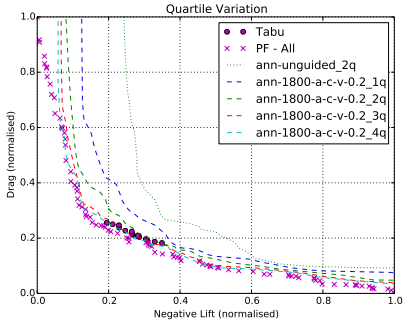

Fig. 9. The First Quartile (1q), Second Quartile (2q), Third Quartile (3q) and Fourth Quartile (4q) results for the 1800 Initial Training Set, Archive, Continuous, Verification, 20% Scepticism variant.

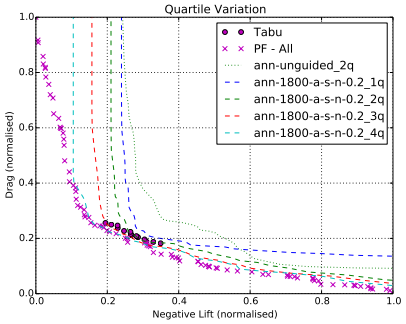

Fig. 10. The First Quartile (1q), Second Quartile (2q), Third Quartile (3q) and Fourth Quartile (4q) results for the 1800 Initial Training Set, Archive, Single, No Verification, 20% Scepticism variant.

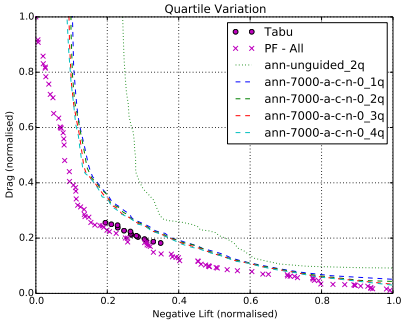

Fig. 11. The First Quartile (1q), Second Quartile (2q), Third Quartile (3q) and Fourth Quartile (4q) results for the 7000 Initial Training Set, Archive, Continuous, No Verification, No Scepticism variant.

performing variants of these sets generally do not perform as well as the best performing variants with initial training sets of 500. This behaviour can be explained by an ANN with a larger training set having stronger pre-conceived biases regarding the parameter space, which leads to better initial results but poorer optimisation near the Pareto-front (for which solutions are not present in the training set). The three variants that performed particularly well are the Archive, Continuous, 20% scepticism with verification (Fig. 9), and the Archive, Single, 20% scepticism with no verification (Fig. 10). Figure 10 is similar to Figure 6: it has a good combined front but the median attainment surface is degraded in the area we are interested in. A similar trend is also present in Figure 9. Overall this shows that for our problem there is little benefit in using an initial training set of size 1800.

Although the variants with initial training set of 7000 were reliable at outperforming the unguided MOPSO, few of them had median attainment surfaces that approached the true front. We attribute this to the same cause as for the 1800 training set variants. However, all 7000 initial training set variants outperform the unguided MOPSO and are reasonably reliable. If reliability is particularly important, the Archive, Continuous, No Verification, No Scepticism variant (Fig. 11) is both highly reliable (shown by it's low mean IQR) and has fairly good performance.

These various considerations lead us to choose the 500 Initial Training Set, Live, Single Training, 15% Scepticism with verification (Fig. 8) as the best variant overall. This variant has no need for an initial training archive (which is desirable for complex problems where it may not exist and be expensive to generate), has a small training set which, for live optimisation, means that the ANN can begin supplementing results quickly, verifies that the ANN isn't pathological before using it, does not retrain as further new results come, to avoid overtraining on the current area being explored, and has only a 15% chance of ignoring the ANN classifying a particle as infeasible. We select this variant since it is reliable, very good (though not the best) at optimisation, requires only a small training set and performs well in the middle area of the Pareto-front in which we are particularly interested.

An interesting observation is that verification doesn't seem to be very useful for this problem. Its main effect seems to be restraining the poor behaviour of poor variants (small initial training set sizes with little or no scepticism). It seems to slightly hinder well-performing variants by holding back data that could be added to the training set to generate a better neural network.

We also notice that appropriate levels of scepticism seem relatively important, with variants using higher levels of scepticism performing more reliably with small initial training sets. However, algorithms with lower, or even no scepticism perform well as long as they have large initial training sets.

This is to be expected, as ANNs trained from large training sets should be naturally more reliable.

V. Conclusion

Our results strongly suggest that using an ANN as a fast approximate evaluator to determine if a trial solution is worthwhile, can be effective at increasing the convergence of MOPSO optimisation, given the selection of appropriate parameters. Furthermore, we can perceive some relationships between the variants of guided MOPSO we are using and the problem we seek to solve. In particular, we identified a promising variant to extend to a specific, industrial optimisation problem, where the primary interest is in the middle area of the Pareto-front. This variant doesn't require an initial training set, reliably produces good optimisation results and performs well on the area of the Pareto-front in which we are most interested.

VI. Possible Future Work

There are several areas that the authors would like to explore further:

1) Testing if verification can be made more useful by increasing the Matthew's Correlation Coefficient (MMC) threshold and possibly introducing a dynamic threshold where the cut-off increases with the size of the training set, because in theory a larger training set indicates greater knowledge and thus greater predictive power.
2) Looking at the effects of a dynamic level of scepticism dependent on the size of the current training set for continuous training with particular emphasis on using the ANN's own confidence in its result.
3) Examining the potential for expert interaction, such as
   - having an expert provide a classification when confidence is low;
   - provide direction in terms of areas of parameter or objective space for the MOPSO to favour; and
   - using active learning techniques to reduce the size of initial archive required, by having an expert give preliminary classifications to parameter sets that are most likely to be useful.
4) Extending this technique to more complex problems, with particular attention to how well it functions with relatively small initial training sets, and how viable continuous training of the ANN is as the problems become more complex. This complexity could take different forms, such as a greater number of parameters, a more complicated mapping from parameters to objective space or a more finely tuned degree of classification than valid/invalid.
5) Generating a behaviour tree from the ANN trained during the optimisation in order to aid in understanding both the ANN's process and the underlying mapping between parameter and objective space i.e. the ANN acts as a "smoother" providing simpler approximations to the complex underlying mapping that behaviour trees have proven unable to handle.

The overall goal of this research is to use Artificial Intelligence techniques to aid in a variety of Multi-Objective Optimisation problems, with a particular focus on supplementing expert knowledge in order to reduce the work load on domain experts.

References

[1] M. R. Amiryousefi, M. Mohebbi, F. Khodaiyan, and M. G. Ahsaee, "Multi-objective optimization of deep-fat frying of ostrich meat plates using multi-objective particle swarm optimization (mopso)," *Journal of Food Processing and Preservation*, 2013.

[2] A. Mukhopadhyay and M. Mandal, "A hybrid multiobjective particle swarm optimization approach for non-redundant gene marker selection," in *Proceedings of Seventh International Conference on Bio-Inspired Computing: Theories and Applications (BIC-TA 2012)*. Springer, 2013, pp. 205–216.

[3] C. Li, Q. Zhu, and Z. Geng, "Multi-objective particle swarm optimization hybrid algorithm: An application on industrial cracking furnace," *Industrial & engineering chemistry research*, vol. 46, no. 11, pp. 3602–3609, 2007.

[4] G. Xu, Z.-t. Yang, and G.-d. Long, "Multi-objective optimization of mimo plastic injection molding process conditions based on particle swarm optimization," *The International Journal of Advanced Manufacturing Technology*, vol. 58, no. 5-8, pp. 521–531, 2012.

[5] S. Qasem and S. Shamsuddin, "Radial basis function network based on multi-objective particle swarm optimization," in *Mechatronics and its Applications, 2009. ISMA '09. 6th International Symposium on*, 2009, pp. 1–6.

[6] ——, "Generalization improvement of radial basis function network based on multi-objective particle swarm optimization," *J. Artif. Intell*, vol. 3, no. 1, 2010.

[7] J. P. T. Yusiong and P. C. Naval Jr, "Training neural networks using multiobjective particle swarm optimization," in *Advances in Natural Computation*. Springer, 2006, pp. 879–888.

[8] L. Feng, J. He, Q. Kong, and L. Guo, "Application of multi-objective algorithm based on particle swarm optimization in electrical short-term load forecasting," in *Power System Technology, 2006. PowerCon 2006. International Conference on*. IEEE, 2006, pp. 1–5.

[9] M. Drela and H. Youngren, "Xfoil, subsonic airfoil development system," 2008, http://web.mit.edu/drela/Public/web/xfoil/.

[10] J. Demšar, T. Curk, A. Erjavec, Črt Gorup, T. Hočevar, M. Milutinovič, M. Možina, M. Polajnar, M. Toplak, A. Starič, M. Štajdohar, L. Umek, L. Žagar, J. Žbontar, M. Žitnik, and B. Zupan, "Orange: Data mining toolbox in python," *Journal of Machine Learning Research*, vol. 14, pp. 2349–2353, 2013. [Online]. Available: http://jmlr.org/papers/v14/demsar13a.html

[11] J. Kennedy and R. Eberhart, "Particle swarm optimization," in *Neural Networks, 1995. Proceedings., IEEE International Conference on*, vol. 4, nov/dec 1995, pp. 1942 –1948 vol.4.

[12] Y. Shi and R. Eberhart, "A modified particle swarm optimizer," in *Evolutionary Computation Proceedings, 1998. IEEE World Congress on Computational Intelligence., The 1998 IEEE International Conference on*, may 1998, pp. 69 –73.

[13] K. Deb, "Multi-objective optimization," in *Search Methodologies*, E. Burke and G. Kendall, Eds. Springer US, 2005, pp. 273–316.

[14] C. Coello Coello and M. Lechuga, "Mopso: a proposal for multiple objective particle swarm optimization," in *Evolutionary Computation, 2002. CEC '02. Proceedings of the 2002 Congress on*, vol. 2, 2002, pp. 1051 –1056.

[15] D. Jaeggi, G. T. Parks, T. Kipouros, and P. J. Clarkson, "The development of a multi-objective tabu search algorithm for continuous optimisation problems," *European Journal of Operational Research*, vol. 185, no. 3, pp. 1192–1212, 2008.

[16] C. M. Fonseca and P. J. Fleming, "On the performance assessment and comparison of stochastic multiobjective optimizers," in *Parallel problem solving from nature—ppsn iv*. Springer, 1996, pp. 584–593.