2014 IEEE Congress on Evolutionary Computation (CEC)
July 6-11, 2014, Beijing, China

Multi-Objective Evolutionary Recurrent Neural Network Ensemble for Prediction of Computational Fluid Dynamic Simulations

Christopher Smith, John Doherty and Yaochu Jin

***Abstract*—Using a surrogate model to evaluate the expensive fitness of candidate solutions in an evolutionary algorithm can significantly reduce the overall computational cost of optimization tasks. In this paper we present a recurrent neural network ensemble that is used as a surrogate for the long-term prediction of computational fluid dynamic simulations.**

A hybrid multi-objective evolutionary algorithm that trains and optimizes the structure of the recurrent neural networks is introduced. Selection and combination of individual prediction models in the Pareto set of solutions is used to create the ensemble of predictors. Five selection methods are tested on six data sets and the accuracy of the ensembles is compared to the converged computational fluid dynamic data, as well as to the delta change between two flow conditions. Intermediate computational fluid dynamic data is used for training and the method presented can produce accurate and stable results using a third of the intermediate data needed for convergence.

I. Introduction

COMPUTATIONAL FLUID DYNAMICS (CFD) simulations are an iterative numerical process [1] used for evaluating the quality of aerodynamic designs, determining performance indicators such as lift and drag (C_L and C_D) on vehicles and aircraft [2]. CFD simulations can take many iterations to converge, resulting in significant computational expense. Simulation results may also only represent an estimation of performance if the full physics is not included.

When performing aerodynamic optimization it is desirable to use CFD to determine the fitness of candidate solutions. In addition, it is also beneficial to be able to use a global optimizer, such as an evolutionary algorithm (EA), which enables a wide design space to be explored and potentially novel design solutions to be identified. However, such an optimization process may require many hundreds (or thousands) of candidate solutions to be evaluated, resulting in a potentially very significant computational burden.

A potential means of reducing the computational expense of such an optimization process would be to use a surrogate assisted evolutionary algorithm (SAEA) [3]. A surrogate model is a computational algorithm designed to simulate the underlying function, process or system behaviour of a complex or expensive process, by building a representation based upon a limited number of sample or training values [4]. In particular, by using a surrogate model, the number of expensive CFD simulations required during aerodynamic optimization can be significantly reduced.

Christopher Smith and Yaochu Jin are with the Department of Computing, University of Surrey, Guildford, GU2 7XH, UK (email: {christopher.smith, yaochu.jin}@surrey.ac.uk). John Doherty is with the Department of Mechanical Engineering Sciences, University of Surrey, Guildford, GU2 7XH, UK (email: john.doherty@surrey.ac.uk).

This research is funded by an EPSRC DTC studentship.

Surrogate models have been used with CFD simulations to achieve optimal designs and an overall reduction in computation time has been achieved [5], [6]. Traditionally these surrogates are constructed by inputting a limited number of training values, corresponding to specific choices of design parameters (e.g. geometry design variables and angle of incidence) and the associated fitness values. The number of training values can be substantially less than the alternative approach of linking the EA optimizer directly with CFD, but this can still result in a large computational cost.

An alternative, or possibly additional approach, would be to use a surrogate model during each CFD convergence process, to predict the outcome of a fully converged CFD simulation, based on the intermediate convergence data. The aim would be to stop each CFD simulation early, before full convergence and project a possible outcome. This type of convergence based prediction surrogate could reduce the computational cost of each individual CFD calculation and hence represent a large overall computational saving for an optimization procedure, where many individual CFD solutions need to be evaluated.

CFD convergence prediction can be considered as a long term prediction problem. The process of learning the characteristics of partially converged CFD data to build a surrogate model has been implemented by Cao et al. [7] to predict the performance of turbine blade designs. A recurrent neural network (RNN) was used for this model and the CFD performance measure was predicted to within 5% of the fully converged result, using half the number of CFD iterations. We have also presented work that uses an ensemble of heterogeneous RNNs for the prediction of transonic wing aerodynamic CFD data [8]. This work showed that confident predictions can be made within 5% of the converged data, using 40% of the iterations needed for convergence. It also highlighted the importance of using an ensemble approach and monitoring accuracy and diversity. Finally, Forrester et al. [9] also used partially converged CFD data to predict converged results when optimizing a wing profile, in which a scaling factor was used to scale partially converged data to a converged value.

The convergence based prediction surrogate model in this work is an ensemble of RNNs. RNNs are designed for temporal based problems, as they have both feed-back as well as feed-forward connections [10]. This makes them ideal for predicting CFD convergence data, as the states of the neurons in the network are stored from the previous iteration step and are used to influence the prediction of data at future iterations. The value of an ensemble is that it allows individual surrogate models to be combined, which can achieve better generalization than single surrogate model use. The result is a more confident prediction [11].

In contrast to our previous work that used a gradient

978-1-4799-1488-3/14/$31.00 ©2014 IEEE

descent method to train individual RNNs in the ensemble, a hybrid multi-objective evolutionary algorithm (HMOEA) has been used to train the RNNs and determine their optimal structure. The HMOEA uses a global multi-objective evolutionary algorithm (MOEA) search technique and gradient descent (GD) local search. This approach results in a Pareto set of solutions, where each individual represents a unique RNN surrogate model.

By selecting and combining individual surrogate models from the Pareto set, a final ensemble of surrogate models can be established. Selection of ensemble members is important, as some individuals in the Pareto set may be unsuitable and sampling many of the created models can be better than sampling them all [12]. Ranking the individual models, based on some criteria or through the use of an optimization process, have been suggested as possible methods of selection [13]. Determining the best way of ranking the individual surrogate models, prior to selection, is very important. Selection based on surrogates located at the knee point of the Pareto set, surrogates with minimum fitness values and the crowding distance measure are investigated.

We hypothesize that the training method presented, as well as the selection of individuals, will be able to accurately predict converged CFD parameters, using minimal partially converged/intermediate data. Along with a direct comparison to CFD predictions, the delta change from two flow conditions will also be analyzed.

The rest of this paper is organized as follows: Section II presents the methods used for creating, selecting and combining the ensemble members. Section III introduces the data sets and Section IV presents the results achieved by the methods introduced. Section V concludes the paper and provides thoughts on future work.

II. Method

A. Ensembles - Accuracy and Diversity

An ensemble of surrogate models consists of many different models that are designed in parallel and used in combination to give a final prediction. Ensembles have been shown to provide better generalization performance than single models [11]. Ensembles can include information that is not contained in a single model [14] and each member can produce different errors. However, the creation, selection and combination of individual predictors is critical to the success of an ensemble, as each individual model needs to be both accurate and diverse [13].

There is always a trade-off between these two characteristics [15], as summarized by the Error-Ambiguity Decomposition presented by Krogh and Vedelsby [16]. This relationship shows that the generalization error of an ensemble is based on the weighted average of the individual generalization errors and the weighted average of the individuals ambiguities, also known as diversity. Creating accurate predictions is clearly very important to the success of an ensemble, but it has been said that diversity is the "holy grail" of ensemble learning [13].

Diverse ensemble members can be either implicitly or explicitly created. Different data sampling, network parameters and initialization, as well as using different learning algorithms have all been used to implicitly create diverse ensemble members [13], [15]. To explicitly create diverse neural network ensembles, the ADDEMUP [17], DIVACE [18] and Regularization [19] algorithms have been used.

The use of an MOEA to create diverse ensemble members is very attractive, as the fitness functions can be specifically chosen to optimize conflicting objectives, with the resultant Pareto set of solutions providing a trade-off between these objectives. An MOEA can be used as an indicator of which solutions to use in the ensemble and MOEAs have been used to successfully design neural networks for a variety of problems [14], [20], [21], [22]. There has not been much discussion in the literature on how to select ensemble members from the Pareto set of solutions. For example, in [20] both the weights and number of hidden neurons in a feed-forward neural network were optimized, but the final performance is based on all members in the Pareto set. Also, in [21] the errors on two different training sets are used as the conflicting objectives and it is argued that the concept of dominance in MOEA's is a form of selection, as it determines if a network should be included in the final Pareto set. Selection from the Pareto set of solutions is however not discussed. Ensembles consisting of all Pareto optimal solutions, combined using a simple average and an ensemble consisting of the weighted output of the Pareto optimal solutions were discussed in [19]. An evolutionary strategy was used to optimize the weights based on an expected error on a validation data set. The results showed that although an improvement can be made on a validation data set, using the weighted output, it did not necessarily mean that there would be an improvement on a test data set. The simple average of all Pareto optimal solutions provided the best result on a test data set.

Our study is the first to use a HMOEA to train RNNs and determine their optimal structure for CFD convergence prediction. The global search can be used to find suitable starting weight values and the local search can be used to fine tune them to their optimal value [19], [23]. The following subsections provide details of the global and local search techniques, finishing with a summary of the HMOEA. Different selection methods are then presented which are used to construct the final ensemble of predictors.

B. Global Search

Evolutionary algorithms (EAs) can be considered as multi-point search strategies that are able to "sample (a) large search space" [24] and escape local optima to find global optimum solutions [25]. EAs are stochastic search and optimization procedures that are based on the principles of natural genetics and natural selection [26]. A population of individual candidates is used, instead of one candidate solution and new solutions are created by selection, crossover and mutation operators, during a set number of generations [25]. The specific design variables that make up a solution are coded into a chromosome, which is decoded to give a fitness/quality score of how well the individual satisfies the objective function(s). Selection, based on this score, is used to determine which individuals will be used as parents to create new offspring or to determine those that will be selected for the next generation [24].

The non-dominated sorting genetic algorithm II (NSGA-II) [27] is used as the global MOEA in this work. Each RNN in

Chromosome 1	1	1	0	1	1	0	1	…	1
Chromosome 2	-0.05	2.65	1.53	5.97	0.49	0.04	-0.29	…	-4.47

Fig. 1: Chromosomes for each Recurrent Neural Network

The chromosomes are decoded to represent an individual network by placing the values of specific alleles into particular locations in the network structure. The topology of the networks is restricted to three input neurons, five hidden neurons and one output neuron. The states of the neurons from the previous time step are recalled and recurrent connections are allowed across all layers of neurons.

Fig. 2 is an example of the matrix setup used in this work, with locations below the main diagonal of the matrix representing forward connections and locations above the diagonal representing recurrent connections. Locations on the diagonal represent self recurrence. Therefore, when a connection, C_{ij} equals 1, a connection is made from neuron j to neuron i, which means neuron j is the connection start point and neuron i the connection end point, i.e. neuron i is receiving activation from neuron j. The connection highlighted in Fig. 2 shows that hidden neuron 4 (H_4) receives activation from input neuron 3 (I_3). Fig. 3 illustrates the network presented in Fig. 2, with the state of the neurons from the previous time step represented by the grey dotted circles. The solid arrows represent forward connections and the dotted arrows represent

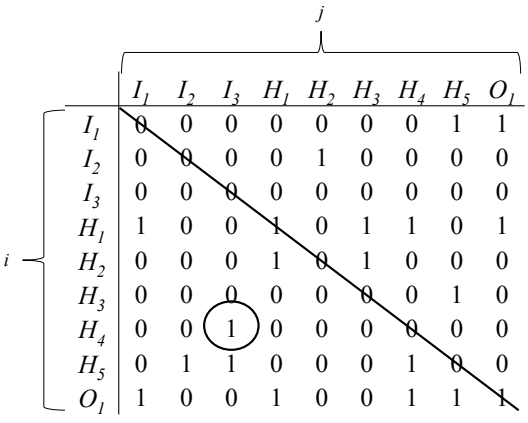

Fig. 2: Recurrent Neural Network Matrix

The two conflicting objectives are the mean squared error (MSE) on a fixed training data set and the number of connections in the network. Both are minimized and this is because

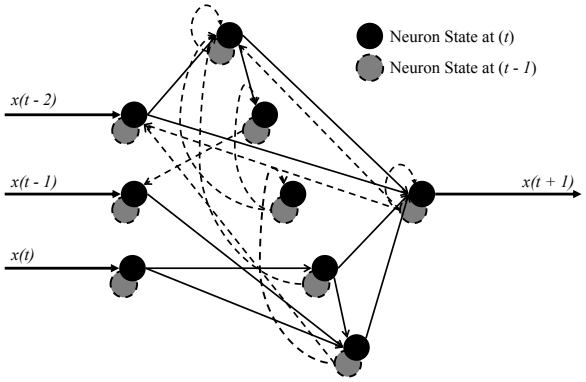

Fig. 3: An example of a Recurrent Neural Network

large complexity is the main reason behind over-fitting [19]. Different crossover and mutation operators are used for the different chromosomes and a fixed number of generations are utilized.

The created models are used to recursively predict a specific number of iterations ahead. The first three actual data points are used to start the models prediction, but then each predicted value is carried forward and used to predict the value at the next time step. This means that this work can be considered as a long-term prediction problem. All actual training data points are needed to calculate the training MSE, as the predicted values are compared to this data.

The temporal CFD data is presented to the RNN in groups of three data points. These three consecutive data points represent the CFD data at three iterations (e.g. $x(t-2)$, $x(t-1)$ and $x(t)$) and are used to predict the CFD performance measure at the next iteration (e.g. $x(t+1)$). This is also illustrated in Fig. 3.

C. Local Search

A gradient descent local search is used to fine tune the weight values of the network once it has been decoded, affecting the second chromosome of each individual. During the local search, all actual data points are presented to the RNN at once, using a batch learning technique.

The error used during the local search is the MSE calculated on all data pairs, minus a warm-up-length and is back propagated through the network to determine the change in the weights. A warm-up-length of data is taken into consideration during batch learning and is used to initialize the internal states of the neurons, so the network can converge to a "normal" dynamic state, allowing for new data to be predicted [28], [29].

The learning algorithm used is the IRPropPlus [30] and all of the neurons use the non-linear sigmoid transfer function ($tanh(v)$). This function was chosen so a non-linear system can be modeled, but it does mean that all data needs to be normalized, as the function only outputs between -1 and 1.

So any values created by the local search are compatible with the genetic algorithms crossover and mutation operators

and are within an acceptable range, the local search has a bounds check on all new design variables. The new weight values are assessed after each training epoch of the gradient descents local search and if a weight value is out of bounds, the weight values for all connections from the previous training epoch are used and the local search is stopped.

When using a local search there are several parameters that need to be considered. Firstly, when to use the local search (frequency), i.e. at which generations. Secondly, how often to use the local search (probability), i.e. which individuals and the length/duration of the local search [31].

Both Lamarckian or Baldwinian learning can be realized by the algorithm presented in this paper. Lamarckian learning allows the newly created chromosomes and associated fitness values to be passed to the individual and used by the GAs operators to create new offspring, whereas Baldwinian learning does not and only the fitness value is updated. Lamarckian learning is adopted during this work as it has been shown to outperform Baldwinian learning when evolving RNNs [32]. By using a Lamarckian search, all of the information learnt by an individual is retained and used to guide the search. If Baldwinain learning were to be adopted, the HMOEA is reliant on the GA to find the specific design parameters of the most successful individuals, with only the fitness values directing the search. This would increase the number of generations required for convergence and therefore increase the computation time.

D. Hybrid Algorithm

Algorithm 1 provides details of how the final Pareto set of solutions are generated. The parameters of the global and local search are defined at the beginning and the fitness functions are the training Mean Square Error (MSE) and total number of connections in the network (NC).

Algorithm 1 Hybrid MOEA for training RNNs

```
Step 1: Input Data Set and Normalize
Step 2: Define Global and Local Search Parameters
        Number of Generations, n
        Frequency of Local Search
        Probability of Local Search
        Duration of Local Search
        Lamarckian or Baldwinain Learning
Step 3: Initialize Chromosomes of Parent Population
Step 4: Evaluate Parents
for i Individuals do
    Decode Chromosomes
    Evaluate Fitness Functions
end for
Step 5: Optimize RNNs Structure and Parameters
for n Generations do
    for i Individuals do
        Decode Chromosomes
        if Local Search then
            Optimize Weights (batch learning)
            Boundary Check
        else Continue
        end if
        Evaluate Fitness Functions (MSE & NC)
    end for
    Select Individuals for next Generation
end for
Step 6: Assemble Final Archive
```

E. Selection and Combination

Initially, selection and combination of all individuals in the Pareto set is performed. This is because a Pareto set of solutions should be diverse and contain a lot of information. However, to select many of the individual surrogate models can be better than selecting them all [12], so once the search has been completed and a Pareto set of solutions has been established, a subset of individuals in the Pareto set are selected and combined. Fig. 4 illustrates an example of a Pareto set of solutions, where each individual represents a unique RNN model. It also illustrates the selection of some solutions that can then be used in the ensemble. In this example, these

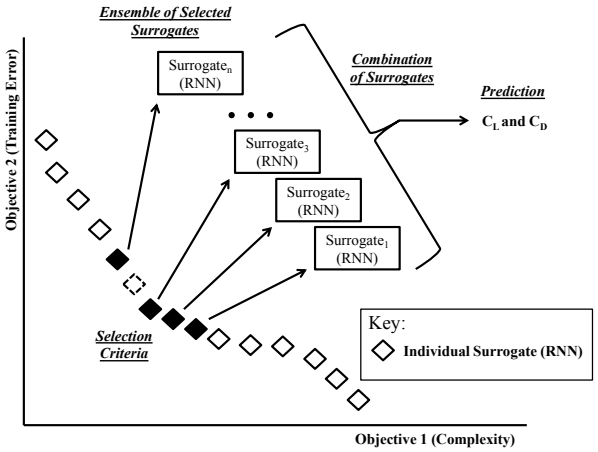

Fig. 4: Pareto Set of Solutions

Five selection methods are investigated and compared to selecting all members in the Pareto set. Prior to selection the extreme individuals in the Pareto set (surrogates with the lowest and highest complexity) are removed to avoid including surrogates with high training error and those that are more likely to cause over-fitting. The following subsections provide information on the five selection methods.

1) Subset based on Knee Point of Pareto set: The knee point of a Pareto set has been described as the region on the front that "involve(s) (a) steep trade-off between objectives" and "high marginal rates of return" [33], indicating that this region is where you get the best trade-off between objectives. The knee point of the Pareto set is also of interest because the complexity of the models in this region is most likely to match that of the data [22]. Consequently, the models in this area of the Pareto set will not exhibit over-fitting on a validation data set.

The Normalized Performance Gain (NPG) was introduced in [22] to give an indication of the knee point of the Pareto set. When there is a large change in the NPG value and it gradually drops to zero, it can be said that the model complexity matches that of the data and that this is the knee-point of the Pareto set. Equation (1) is used for calculating the NPG, where MSE_i, MSE_{i+1} and NC_i, NC_{i+1} are the MSE on the training data and the number of connections of the ith and $i+1$th Pareto optimal solutions.

$$NPG = \frac{MSE_{i+1} - MSE_i}{NC_i - NC_{i+1}} \tag{1}$$

When the solutions are ranked in the order of increasing complexity, the following relationship holds:

$$\begin{aligned} NC_{i+1} &> NC_i, \\ MSE_{i+1} &\leq MSE_i \end{aligned} \tag{2}$$

The subset will be selected from the identified knee-point and includes individuals with an increasing complexity from the knee point.

2) Single model located at Knee Point of Pareto set: The individual surrogate identified by the NPG as the knee-point of the Pareto set is selected, as this will be the individual that has the best trade-off between objectives.

3) Subset based on NSGA-II Crowding Distance: The crowding distance measure in the NSGA-II algorithm is used to sort individuals in the Pareto set. It is a measure of how crowded an individual is and is therefore a measure of the individuals diversity during the search. The value assigned to the individual is dependent on the average distance of the individuals on either side of it. The individuals in the Pareto set are ranked based on their crowding distance from largest to smallest. The top ranked individuals are selected for the subset.

4) Subset based on Training MSE: The individuals in the Pareto set are ranked based on the training MSE from smallest to largest. The top ranked individuals are selected for the subset.

5) Subset based on Number of Connections: The individuals in the Pareto set are ranked based on the number of connections in the network from smallest to largest. The top ranked individuals are selected for the subset.

Successful ensemble sizes have generally ranged from between three and five members [11], so where possible subsets of five members are used. Once the surrogates have been selected, the individual RNN models are used to recursively predict the converged performance indicator to a certain number of iterations. A simple average is used to combine the predicted values at the final iteration point.

Section III provides information on the specific data sets that have been used in this work.

III. Data Sets

Convergence data for an aircraft wing is used to test the HMOEA and selection methods presented. Converged predictions for the C_L and C_D are made at three angles of incidence (α = 1.35°, 1.65° and 6.00°). The convergence histories are from an Euler CFD method and each CFD simulation takes approximately 40 minutes to run on one CPU. Each data set has a total of 150 iterations, which is a typical level for practical use.

Results will be presented in two formats. Firstly, a comparison between the CFD converged result and the predicted results is presented for each data set and selection type, with the absolute error reported. The second format is more specific to the use of the predictors as surrogates in an optimization task. When a surrogate is used in an optimization task the accuracy of the individual predictions can be less important. As long as the surrogate can lead the search to the optimum and the order of the individuals in the population is maintained, the best individuals should still be selected [34], [35]. This means the optimizer is interested in the incremental change in performance between different designs or parameter values. It is therefore not so critical that a surrogate model can predict the exact performance of a specific design, but the same delta between different designs as the CFD simulations. Therefore, the performance delta for a change in the angle of incidence is reported for both the CFD data and the surrogate models predicted data.

TABLE I: Data Sets

Data Set	Target
1.35°C_D	0.01922
1.65°C_D	0.02176
6.00°C_D	0.10333
1.35°C_L	0.56842
1.65°C_L	0.60677
6.00°C_L	1.13679

TABLE II: Absolute Deltas

Comparison	Delta
1.35°C_D-1.65°C_D	0.00254
1.65°C_D-6.00°C_D	0.08156
1.35°C_D-6.00°C_D	0.08411
1.35°C_L-1.65°C_L	0.03835
1.65°C_L-6.00°C_L	0.53002
1.35°C_L-6.00°C_L	0.56837

Table I provides information on the target values for the six data sets and Table II the different comparisons that are made and the corresponding absolute delta values that the CFD simulations can achieve.

IV. Simulation Results and Discussion

As discussed in Section II-C the frequency, probability and duration of the local search need to be considered. A numerical investigation that considered different local search parameters was conducted to establish what the values are for one data set and the same values are then used for all other data sets. These values are; Frequency: every 10 generations, Duration: 20 epochs and Probability: 50%. The first 50 data points are used for training and a warm-up-length of 10 is used during the local search.

The weights of the connection matrices for each ensemble member are randomly initialized between -10 and 10 and the boundary check used during the local search ensures the weight values do not exceed these limits. A total of 500 generations and a population size of 100 are used. Normalization of training data is between 0 and 1. However, it should be noted that the training MSE for the global search is calculated using the original data, so the predicted values need to be converted back to the original range prior to calculating the training MSE. This is also done during the prediction phase.

Ten independent runs of each data set were performed to account for the random initial weights. The mean prediction value and standard deviation for the 10 runs, along with the absolute error to the target value, is presented in Table III. The best performing selection methods are highlighted in bold text. Table IV presents the comparisons between the deltas for different angle of incidence calculations. The absolute error between the delta achieved by the CFD simulations and the delta achieved by the surrogates is used to identify the best selection criteria for each data set.

Fig. 5 is an example of the prediction performance for a C_D data set. All surrogate model predictions are included (dashed

TABLE III: Prediction Results

		All	Selection 1	Selection 2	Selection 3	Selection 4	Selection 5
1.35° C_D	Mean Prediction	0.02279	0.02139	0.02018	0.02378	0.02564	0.02052
	Standard Deviation	0.00081	0.00148	0.00309	0.00100	0.00110	0.00115
	Absolute Error	0.00357	0.00218	**0.00097**	0.00456	0.00642	0.00131
1.65° C_D	Mean Prediction	0.02590	0.02599	0.02386	0.02663	0.02737	0.02397
	Standard Deviation	0.00168	0.00285	0.00197	0.00297	0.00272	0.00149
	Absolute Error	0.00414	0.00423	**0.00210**	0.00487	0.00561	0.00221
6.00° C_D	Mean Prediction	0.11889	0.12074	0.11951	0.11966	0.12057	0.11716
	Standard Deviation	0.00132	0.00243	0.00137	0.00185	0.00098	0.00255
	Absolute Error	0.01554	0.01741	0.01618	0.01633	0.01724	**0.01384**
1.35° C_L	Mean Prediction	0.61140	0.64138	0.63324	0.63980	0.64519	0.63259
	Standard Deviation	0.02966	0.00926	0.01267	0.01012	0.00857	0.01297
	Absolute Error	**0.04298**	0.07296	0.06482	0.07138	0.07677	0.06417
1.65° C_L	Mean Prediction	0.63887	0.67705	0.67632	0.67783	0.68368	0.66679
	Standard Deviation	0.01937	0.01069	0.01087	0.01123	0.00614	0.01251
	Absolute Error	**0.03210**	0.07028	0.06955	0.07106	0.07691	0.06002
6.00° C_L	Mean Prediction	1.10781	1.18454	1.18093	1.18636	1.19211	1.18445
	Standard Deviation	0.04198	0.01757	0.02280	0.01664	0.01959	0.01591
	Absolute Error	**0.02898**	0.04775	0.04414	0.04957	0.05532	0.04766

TABLE IV: Delta Results

		All	Selection 1	Selection 2	Selection 3	Selection 4	Selection 5
1.35°C_D-1.65°C_D	Delta	0.00312	0.00460	0.00367	0.00285	0.00173	0.00345
	Abs. Error	0.00057	0.00205	0.00113	**0.00031**	0.00081	0.00091
1.65°C_D-6.00°C_D	Delta	0.09297	0.09475	0.09565	0.09303	0.09320	0.09319
	Abs. Error	**0.01140**	0.01318	0.01409	0.01146	0.01163	0.01162
1.35°C_D-6.00°C_D	Delta	0.09608	0.09935	0.09933	0.09588	0.09493	0.09664
	Abs. Error	0.01197	0.01524	0.01522	0.01177	**0.01082**	0.01253
1.35°C_L-1.65°C_L	Delta	0.02747	0.03568	0.04309	0.03804	0.03849	0.03421
	Abs. Error	0.01088	0.00268	0.00473	0.00032	**0.00014**	0.00415
1.65°C_L-6.00°C_L	Delta	0.46894	0.50748	0.50461	0.50852	0.50843	0.51765
	Abs. Error	0.06108	0.02253	0.02540	0.02149	0.02159	**0.01236**
1.35°C_L-6.00°C_L	Delta	0.49641	0.54316	0.54770	0.54656	0.54692	0.55186
	Abs. Error	0.07196	0.02521	0.02067	0.02181	0.02145	**0.01651**

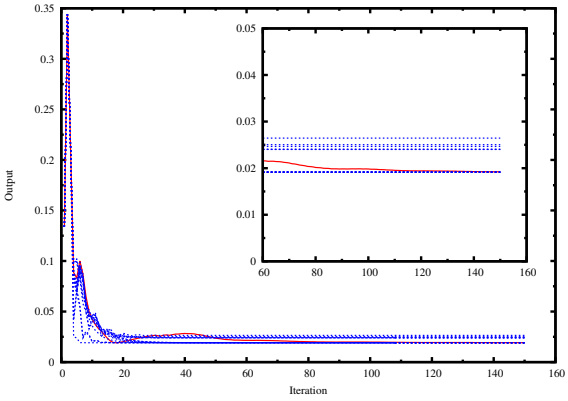

Fig. 5: CFD Prediction - 1.35°C_D

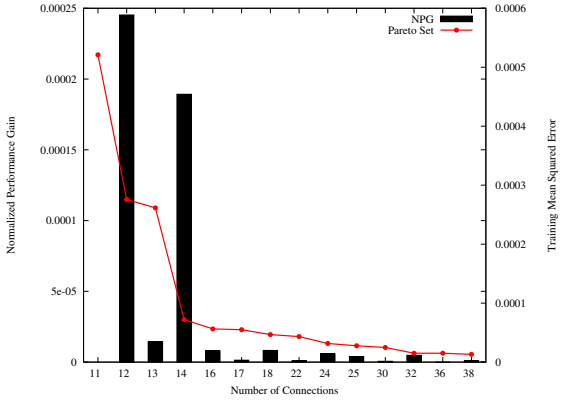

Fig. 6: NPG and Pareto Set example - 1.35°C_D

lines) along with the target data (solid line) and the enlarged plot is the final 90 iterations to show the different performance of the various surrogates. Each prediction is associated with a surrogate model in the Pareto set shown in Fig. 6. Fig. 6 also gives an example of the NPG for this setup and how it can indicate where the knee point of the Pareto set is located. Fig. 7 is an example of the prediction performance for a C_L data set, with all surrogate model predictions included (dashed lines) along with the target data (solid line).

The results presented in Table III shows that the target values can be predicted with reasonable accuracy. A selection from the Pareto set either using selection method 2 (a single model located at the knee-point of the Pareto set) or method 5 (a subset based on the number of connections) have performed best for the C_D data sets. However, using all individuals in the Pareto set has produced the best results for all C_L data sets.

This result was unexpected, as it is known that the Pareto

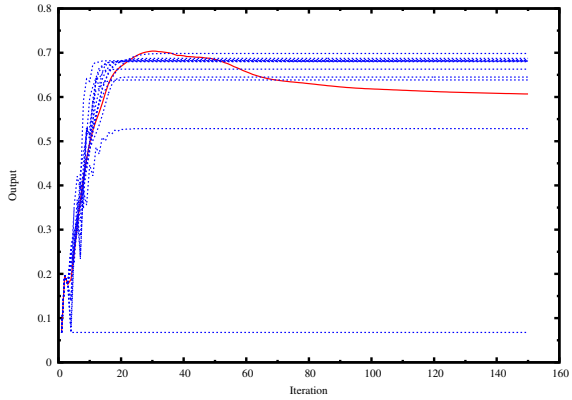

Fig. 7: CFD Prediction - 1.65°C_L

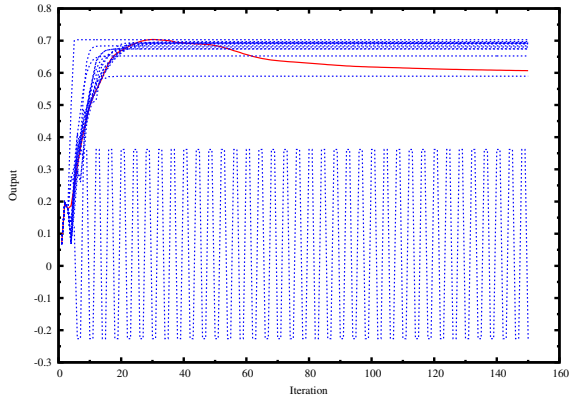

Fig. 8: Unstable Prediction example - 1.65°C_L

set will include some unsuitable models and that selection should remove these surrogates from the ensemble. An explanation as to why this has happened can be seen in Fig. 7, which shows two surrogate models that are producing very different and actually wrong predictions that are below the target (solid line). When these two predictions are included in the ensemble they add a bias to the prediction performance, shifting the mean down and closer to the target. However, when they are removed, because the other surrogates are all predicting similar results above the target, the mean is shifted up and away from the target, resulting in poorer performance.

The results presented in Table IV show that the deltas between two aerodynamic flow conditions can also be predicted reasonably well. These results are encouraging as it shows that the surrogates can still maintain the direction of design improvement and to a similar number of lift and drag counts as the CFD simulations. An explanation for this is that the absolute error for each ensemble prediction appears to be consistent, with both the lift and drag predictions being over predicted. Hence by considering the delta in lift and drag, the effect of this absolute error is reduced. The predicted delta can potentially provide a usable search direction during aerodynamic optimization, as the resultant drag delta is consistently over predicted and the lift delta under predicted.

Table IV also does not show a selection method that performs better than any of the others when considering the deltas between different flow conditions. There is also one case where using all individuals in the Pareto set provides the best result. It is clear though that when considering the deltas between different flow conditions, using a subset is generally better than using all individuals, particularly for C_L data sets.

The previously presented results show stable predictions and this behaviour was generally seen for the majority of surrogate models generated. However, there were some unstable predictions, including those that cycled with small and large amplitudes. Fig. 8 is an example of an unstable prediction for the 1.65°C_L data set. This surrogate would be a member of an ensemble that includes all individuals from the Pareto set, but would be removed by the selection process.

Further investigation is required to understand why some of the networks produce these oscillatory predictions. Also, when reviewing the structure of the networks there were some instances where connections were made, but no associated weight value is present. This may be due to the networks driving the weights to zero, indicating that a connection should not be present. However, in the future a mechanism to identify these links should be included in the model, as connections without weights means that some solutions in the Pareto set may actually be dominated solutions, as their complexity is being overstated.

V. Conclusions and Future Work

This paper has presented a framework that uses ensembles of RNNs to predict CFD convergence data using partially converged/intermediate data. A HMOEA has been presented that uses a global optimizer with local search to train the networks and optimize their structure. The result of the algorithm is a Pareto set of solutions. The selection and combination of the individuals in the Pareto set has been investigated, although it is not possible to conclude that one selection method is better than any of the others tested on the CFD datasets. Generally the predictions are very stable and the selection process should result in removal of any which are unstable.

Encouraging predictions have been made for two performance measures (C_L and C_D) at three different angles of incidence (α = 1.35°, 1.65° and 6.00°). The delta change from one flow condition to another has also been investigated and this has shown that when two surrogate models are considered together, the performance is comparable to the delta change from the CFD data. This appears to be because there is a consistent over prediction by the surrogate models.

This work has shown that an ensemble of RNNs, which are trained using a hybrid optimizer, can be used to predict converged CFD results using a third of the intermediate data. The impact of this is that a reduction in overall computation time could be achieved when an optimizer uses CFD simulations as its fitness function evaluation. Particularly when a global optimizer is used and many CFD simulations need to be evaluated.

Further work will be to investigate the consistent over and under predictions seen. Additional data sets that consider wing geometry changes will also be investigated, because it is these features that are most likely to be optimized, rather than just the angle of incidence. Different quantities of training data and

the local search parameters can also be fine tuned to improve prediction accuracy. Additional selection methods should also be investigated to see if one can be applied that is suitable for all data sets. Also, a selection method that assesses the similarity between the ensemble members should be investigated, as it can be seen that a number of the surrogates are producing similar prediction profiles. The ambiguity term of Krogh and Vedelsbys Error-Ambiguity Decomposition [16] could be used to give an indication of the diversity between members. Members with high ambiguity values should be selected, so the ensemble only includes those that are diverse.

Evolving ensembles instead of individuals could also be investigated, however it is felt that the overall computation time would increase, due to the need to evaluate the various selection schemes after each generation. Currently the ensemble members are only selected from the final archive of solutions.

Finally, the recurrent neural network ensemble based surrogates will be used in evolutionary aerodynamic design optimization of wing high-lift systems.

REFERENCES

[1] B. Massey, *Mechanics of Fluids*, 7th ed. Stanley Thornes (Publishers) Ltd, 1998.

[2] H. Versteeg and W. Malalasekera, *An Introduction to Computational Fluid Dynamics: The Finite Volume Method*. Pearson Education Limited, 2007.

[3] Y. Jin, "Surrogate-assisted evolutionary computation: Recent advances and future challenges," *Swarm and Evolutionary Computation*, vol. 1, no. 2, pp. 61 – 70, 2011.

[4] D. Gorissen, I. Couckuyt, P. Demeester, T. Dhaene, and K. Crombecq, "A surrogate modeling and adaptive sampling toolbox for computer based design," *Journal of Machine Learning Research*, vol. 11, pp. 2051–2055, 2010.

[5] A. Ribeiro, A. Awruch, and H. Gomes, "An airfoil optimization technique for wind turbines," *Applied Mathematical Modelling*, vol. 36, no. 10, pp. 4898 – 4907, 2012.

[6] R. M. Greenman and K. R. Roth, "Minimizing computational data requirements for multi-element airfoils using neural networks," *Journal of Aircraft*, vol. 36, no. 5, pp. 777–784, 1999.

[7] Y. Cao, Y. Jin, M. Kowalczykiewicz, and B. Sendhoff, "Prediction of convergence dynamics of design performance using differential recurrent neural networks," in *IEEE International Joint Conference on Neural Networks (IJCNN), World Congress on Computational Intelligence*, June 2008, pp. 528 –533.

[8] C. Smith, J. Doherty, and Y. Jin, "Recurrent neural network ensembles for convergence prediction in surrogate-assisted evolutionary optimization," in *IEEE Computational Intelligence in Dynamic and Uncertain Environments (CIDUE), Symposium Series on Computational Intelligence*, 2013, pp. 9–16.

[9] A. Forrester, N. Bressloff, and A. Keane, "Optimization using surrogate models and partially converged computational fluid dynamics simulations," in *Proceedings of the Royal Society A*, 2006, pp. 2177–2204.

[10] A. P. Engelbrecht, *Computational Intelligence: An Introduction*, 2nd ed. Wiley Publishing, 2007.

[11] L. K. Hansen and P. Salamon, "Neural network ensembles," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 12, no. 10, pp. 993–1001, October 1990.

[12] Z.-H. Zhou, J. Wu, and W. Tang, "Ensembling neural networks: Many could be better than all," *Artificial Intelligence*, vol. 137, pp. 239–263, November 2002.

[13] Z.-H. Zhou, *Ensemble Methods: Foundations and Algorithms*, ser. Chapman & Hall/CRC Data Mining and Knowledge Discovery Serie. Taylor & Francis, 2012.

[14] X. Yao and M. M. Islam, "Evolving artificial neural network ensembles," *IEEE Computational Intelligence Magazine*, vol. 3, no. 1, pp. 31–42, February 2008.

[15] G. Brown, J. Wyatt, R. Harris, and X. Yao, "Diversity creation methods: a survey and categorisation," *Journal of Information Fusion*, vol. 6, no. 1, pp. 1 – 28, 2005.

[16] A. Krogh and J. Vedelsby, "Neural network ensembles, cross validation, and active learning," in *Advances in Neural Information Processing Systems*. MIT Press, 1995, pp. 231–238.

[17] D. W. Opitz and J. W. Shavlik, "Generating accurate and diverse members of a neural-network ensemble," in *Advances in Neural Information Processing Systems*. MIT Press, 1996, pp. 535–541.

[18] A. Chandra and X. Yao, "DIVACE: Diverse and accurate ensemble learning algorithm," in *IDEAL*, ser. Lecture Notes in Computer Science, Z. Yang, H. Yin, and R. Everson, Eds. Springer Berlin Heidelberg, 2004, vol. 3177, pp. 619–625.

[19] Y. Jin, T. Okabe, and B. Sendhoff, "Neural network regularization and ensembling using multi-objective evolutionary algorithms," in *IEEE Congress on Evolutionary Computation*, vol. 1, June 2004, pp. 1–8.

[20] H. A. Abbass and R. Sarker, "Simultaneous evolution of architectures and connection weights in ANNs," *Proceedings of Artificial Neural Networks and Expert System Conference*, pp. 16–21, 2001.

[21] H. Abbass, "Pareto neuro-evolution: constructing ensemble of neural networks using multi-objective optimization," in *IEEE Congress on Evolutionary Computation*, vol. 3, 2003, pp. 2074–2080.

[22] Y. Jin and B. Sendhoff, "Pareto-based multiobjective machine learning: An overview and case studies," *IEEE Transactions on Systems, Man, and Cybernetics, Part C: Applications and Reviews*, vol. 38, pp. 397–445, 2008.

[23] X. Yao, "Evolving artificial neural networks," *Proceedings of the IEEE*, vol. 87, no. 9, pp. 1423–1447, 1999.

[24] G. Jones, "Genetic and evolutionary algorithms," 1990.

[25] R. Salomon, "Evolutionary algorithms and gradient search: similarities and differences," *IEEE Trans. Evolutionary Computation*, vol. 2, no. 2, pp. 45–55, July 1998.

[26] K. Deb, *Multi-Objective Optimization using Evolutionary Algorithms*. John Wiley and Sons, Ltd, 2001.

[27] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan, "A fast and elitist multiobjective genetic algorithm: NSGA-II," *IEEE Transactions on Evolutionary Computation*, vol. 6, no. 2, pp. 182–197, April 2002.

[28] M. Husken and P. Stagge, "Recurrent neural networks for time series classification," *Neurocomputing*, vol. 50, pp. 223 – 235, 2003.

[29] C. Igel, V. Heidrich-Meisner, and T. Glasmachers, "Shark," *Journal of Machine Learning Research*, vol. 9, pp. 993–996, 2008.

[30] C. Igel and M. Husken, "Empirical evaluation of the improved Rprop learning algorithms," *Neurocomputing*, vol. 50, pp. 105 – 123, 2003.

[31] T. A. El-Mihoub, A. A. Hopgood, L. Nolle, and A. Battersby, "Hybrid genetic algorithms: A review," *Engineering Letters*, vol. 13, no. 2, pp. 124–137, 2006.

[32] K. W. Ku and M.-W. Mak, "Exploring the effects of lamarckian and baldwinian learning in evolving recurrent neural networks," in *IEEE International Conference on Evolutionary Computation*, 1997, pp. 617–621.

[33] L. Rachmawati and D. Srinivasan, "A multi-objective evolutionary algorithm with weighted-sum niching for convergence on knee regions," in *Genetic and Evolutionary Computation Conference*. New York, NY, USA: ACM, 2006, pp. 749–750.

[34] Y. Jin, M. Husken, and B. Sendhoff, "Quality measures for approximate models in evolutionary computation," in *GECCO 2003: Proceedings of the Bird of a Feather Workshop, Genetic and Evolutionary Computation Conference*. AAAI, 2003, pp. 170–173.

[35] M. Husken, Y. Jin, and B. Sendhoff, "Structure optimization of neural networks for evolutionary design optimization," *Soft Comput.*, vol. 9, no. 1, pp. 21–28, 2005.