Convergence Based Prediction Surrogates for High-lift CFD Optimization

Christopher Smith*, John Doherty† and Yaochu Jin*
**Department of Computing / †Department of Mechanical Engineering Sciences,*
University of Surrey, Guildford, GU2 7XH, UK
{christopher.smith,john.doherty,yaochu.jin}@surrey.ac.uk

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 analyze the convergence profiles of a multi-element high-lift system, revealing insights into the flow physics of the system and how C_L varies for different numbers of flow iterations.

A hybrid multi-objective evolutionary algorithm that trains and optimizes the structure of a recurrent neural network ensemble is then introduced as a surrogate for the long-term prediction of the high-lift systems computational fluid dynamic convergence data. The intermediate data is used for training the networks and results presented show that the trends of the design space can be better predicted than the absolute magnitudes of the C_L convergence histories.

1. Introduction

Multi-element high-lift systems are used to significantly increase lift during take-off and landing. The flow features associated with high-lift aerodynamics are typically highly complex, requiring high-fidelity Computational Fluid Dynamic (CFD) simulations to adequately predict performance. The computational cost of high-lift CFD can be significant, so modeling is usually limited to convergence to a steady-state flow solution, which significantly reduces the computational burden compared to unsteady modeling. However, even these steady-state solutions can be very computationally costly, often requiring many solution iterations for adequate convergence. Key output performance indicators, such as lift (C_L) and drag (C_D) coefficient, are often used as a measure of convergence. The speed and ease with which a CFD solution will converge, can be highly dependent upon the complexity of the flow. In some cases, the CFD simulation can become numerically unstable, or exhibit oscillatory convergence behaviour, meaning an adequately converged solution cannot be obtained. Such a situation can result when the complexity of the flow is beyond the capabilities of the CFD model e.g. when applying a steady-state model to a case where the real flow contains significant regions of unsteady flow.

In addition to use of CFD for high-lift performance prediction, there is also a desire to combine CFD with use of optimization tools, to explore/discover new high-lift design concepts. Here CFD would be used to determine the fitness of the candidate solutions during an optimization search. To enable a wide range of potential concepts to be investigated, it may also be desirable to be able to use a global optimizer, such as an Evolutionary Algorithm (EA). 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). 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 [1]. In particular, by using a surrogate model, the number of expensive CFD simulations can be significantly reduced.

Surrogate models have been used with CFD simulations to achieve optimal designs and an overall reduction in computation time has been achieved [2, 3]. 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 optimization directly with CFD, but there can still be 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 after a reduced number of iterations and to predict what the result would be at full convergence. This type of convergence based prediction surrogate could reduce the computational cost of each individual CFD calculation. This would hence represent a large overall computational saving for an optimization procedure, where many individual CFD solutions need to be evaluated.

The process of learning the characteristics of partially converged CFD data to build a surrogate model has been implemented by Cao et al. [4] 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 ensembles of RNNs for the prediction of transonic wing aerodynamic CFD data [5, 6]. This work highlighted the importance of using an ensemble approach and monitoring accuracy and diversity. Encouraging results were reported, with confident predictions made using 40% [5] and 33% [6] of the iterations needed for convergence.

The convergence based prediction surrogate model in this work is an ensemble of RNNs. A hybrid multi-objective evolutionary algorithm is used to train the RNNS and determine their optimal structure. This approach results in a Pareto set of solutions, where each individual represents a unique RNN surrogate model. By selecting individual surrogate models from the Pareto set, a final ensemble of surrogate models can be established. By combining individual surrogate models, it is possible to achieve better generalization than when using a single surrogate model, resulting in a more confident prediction [7].

Section 2 introduces the specific high-lift test case used in this work, including analysis of the convergence profiles and predictions of C_L for different numbers of flow iterations. Section 3 introduces more details about the prediction method used, including the importance of ensembles and how the individual RNNS are constructed and trained. The selection of individuals that make up the ensemble of predictors will also be discussed, as well as the specific model parameters that have been used. Section 4 provides details of the simulation results and how they compare to the converged CFD data and that of the intermediate data. Section 5 discusses the results in the context of aerodynamic optimization and section 6 concludes the paper.

2. Test Case – High-lift System

This section will introduce the high-lift test case which was supplied by QinetiQ. Figure 1 shows the baseline geometry of the generic high-lift test case, incorporating a leading edge element (slat), trailing edge element (flap) and a main wing element with a flap cavity.

Existing results for a total of 200 2D RANS CFD simulations were available, which had been generated by QinetiQ as part of a previous C_L max design study. In particular, these results corresponded to a parametric study of the effect of moving the flap, whilst fixing the flap deflection, for fixed slat/main wing geometry. The position of the flap is varied in two directions, as shown in Fig. 1. FlapLap is a measure of the overlap between the flap and the trailing edge of the main wing. FlapGap provides a measure of the vertical distance between the flap and the trailing edge of the main wing.

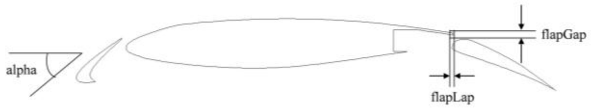


Figure 1. High-lift Test Case

The dimensions of flapLap and flapGap are a percentage of the chord of the equivalent clean wing aerofoil (high-lift devices stowed). The overall angle of incidence (alpha) is also varied. The values of flapLap, flapGap and alpha were tested at the values in Table I.

Table I. High-lift Parameters

<table>
<tr><td>flapLap</td><td colspan="2">0.000</td><td>0.00375</td><td colspan="2">0.0075</td><td>0.01125</td><td colspan="2">0.015</td></tr>
<tr><td>flapGap</td><td colspan="2">0.0075</td><td>0.011875</td><td colspan="2">0.01625</td><td>0.0206</td><td colspan="2">0.025</td></tr>
<tr><td>alpha</td><td>5 °</td><td>10 °</td><td>15 °</td><td>18 °</td><td>21 °</td><td>22 °</td><td>23 °</td><td>24 °</td></tr>
</table>

Therefore, C_L can be represented by a function of the operating parameters of the flap (flapLap, flapGap and alpha):

$$\mathbf{F(flapLap, flapGap, alpha) = C_L}$$

Each CFD simulation uses an unstructured mesh of approximately 100,000 cells, though the actual grid size varies slightly for different values of flapLap and flapGap. All simulations were run for a total of 4000 iterations. To gain understanding/insight into the underlying physics affecting the convergence of high-lift CFD, each convergence profile for C_L was analyzed. General trends between parameters were also investigated.

Initially all 200 C_L convergence profiles were plotted and analyzed together, such as the examples shown in Figs. 2 - 6. Due to the transient nature of the first phase of a CFD convergence process, it was decided to remove the first 500 iterations. The first observation made was that there was a wide range of different convergence profile types. In addition, the overall shape of some convergence profiles exhibit similar characteristics to an underdamped system (e.g. presence of overshoots/oscillations), whilst others are similar to critically damped behaviour.

These overall shape characteristics can be described as low frequency features and to get a better idea of how they varied between convergence profiles; each profile was classified into one of five categories depending on their convergence profile type. The first category grouped convergence profiles that steadily increased before flattening off to a relatively constant value. This category has been called "monotonic convergence" and an example can be seen in Fig. 2.

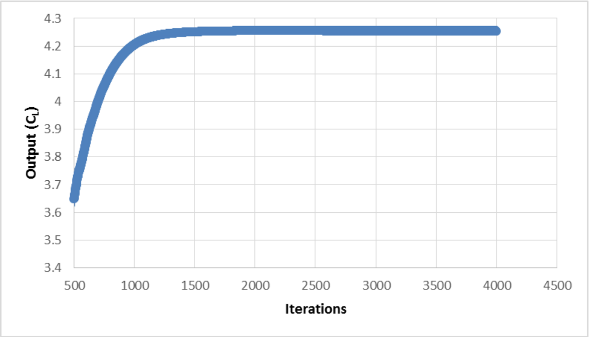


Figure 2. "monotonic convergence"

The second category includes profiles that increased to a final value, but not consistently, by initially decreasing and then increasing. Figure 3 is an example of this "complex convergence".

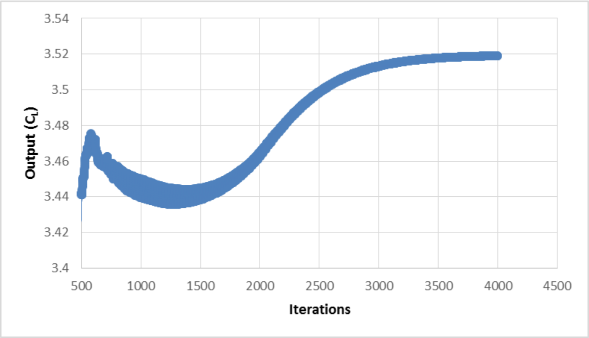


Figure 3. "complex convergence"

The third category groups profiles that behave like an underdamped system, increasing past its final converged value, before decreasing again. This category has been named as "small overshoot" and an example can be seen in Fig. 4.

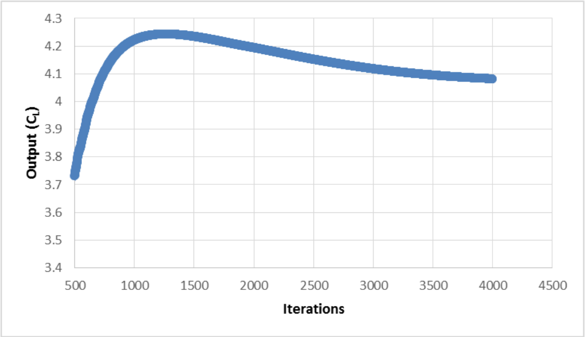


Figure 4. "small overshoot"

Category four, "large overshoot", groups profiles that are similar to the third category, but with a larger overshoot. Figure 5 is an example of this "large overshoot" profile.

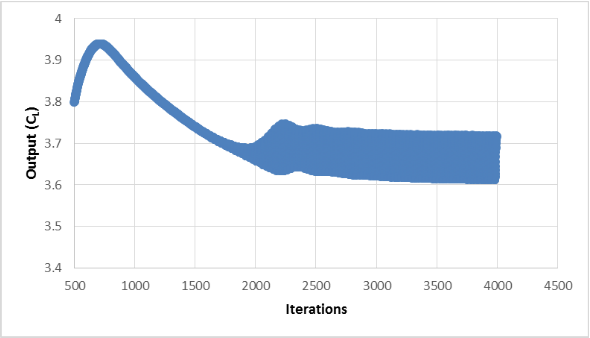


Figure 5. "large overshoot"

Finally, the fifth category, "critical convergence", behaves like a critically damped system, with the convergence profile steadily decreasing to a final value. Figure 6 is an example of this profile.

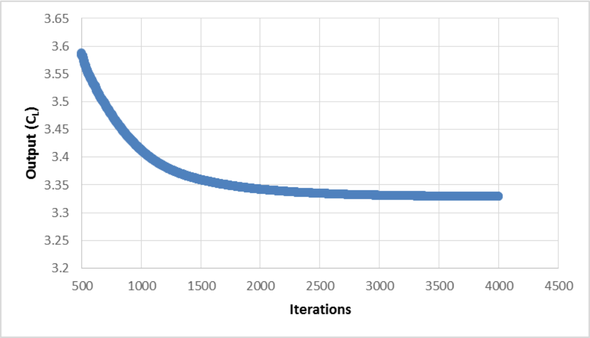


Figure 6. "critical convergence"

Figure 7 summarizes the categories assigned to each convergence profile, with results grouped into constant flapLap sets. It is clear from this plot that there are trends in the C_L convergence profiles when varying flapLap, flapGap and alpha. For example, for all flapLap values with high alpha and low flapGap, the profiles all exhibit "monotonic convergence", whereas low alpha and high flapGap all exhibit "critical convergence".

Generally, the lower alphas have more varied profiles, although there seems to be a consistent transition through the different categories as alpha is increased and flapGap is decreased. When keeping flapGap and alpha constant and increasing flapLap, there are no major changes in the profiles. This is evident from each group of flapLap having similar behaviour to one another. Another overall point from this analysis is that the complex and overshoot convergence histories could be described as having longer transient phases, which may be indicative of the flow being more complex, such as due to more extensive regions of flow separation, or perhaps indicating that the real flow would exhibit more significant unsteadiness.

		Angle of Incidence							
flapLap	flapGap	5	10	15	18	21	22	23	24
0	0.0075	🟢	🟢	🟢	🟢	🟩	🟩	🟩	🟩
0	0.011875	🟨	🟨	🟨	🟩	🟩	🟩	🟩	🟩
0	0.01625	🟧	🟧	🟧	🟧	🟨	🟩	🟩	🟩
0	0.0206	🟥	🟥	🟧	🟧	🟨	🟨	🟩	🟩
0	0.025	🟥	🟥	🟥	🟧	🟨	🟨	🟩	🟩
0.00375	0.0075	🟢	🟢	🟢	🟩	🟩	🟩	🟩	🟩
0.00375	0.011875	🟨	🟧	🟨	🟩	🟩	🟩	🟩	🟩
0.00375	0.01625	🟧	🟧	🟧	🟨	🟩	🟩	🟩	🟩
0.00375	0.0206	🟥	🟥	🟧	🟧	🟨	🟨	🟩	🟩
0.00375	0.025	🟥	🟥	🟥	🟧	🟨	🟨	🟩	🟩
0.0075	0.0075	🟢	🟢	🟢	🟩	🟩	🟩	🟩	🟩
0.0075	0.011875	🟧	🟧	🟨	🟩	🟩	🟩	🟩	🟩
0.0075	0.01625	🟧	🟧	🟧	🟨	🟩	🟩	🟩	🟩
0.0075	0.0206	🟥	🟥	🟧	🟧	🟨	🟨	🟩	🟩
0.0075	0.025	🟥	🟥	🟥	🟧	🟨	🟨	🟩	🟩
0.01125	0.0075	🟢	🟩	🟩	🟩	🟩	🟩	🟩	🟩
0.01125	0.011875	🟧	🟧	🟨	🟩	🟩	🟩	🟩	🟩
0.01125	0.01625	🟧	🟧	🟧	🟨	🟩	🟩	🟩	🟩
0.01125	0.0206	🟥	🟥	🟧	🟧	🟨	🟨	🟩	🟩
0.01125	0.025	🟥	🟥	🟧	🟧	🟨	🟨	🟩	🟩
0.015	0.0075	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩
0.015	0.011875	🟧	🟧	🟨	🟩	🟩	🟩	🟩	🟩
0.015	0.01625	🟥	🟧	🟧	🟨	🟩	🟩	🟩	🟩
0.015	0.0206	🟥	🟥	🟧	🟨	🟨	🟩	🟩	🟩
0.015	0.025	🟥	🟥	🟧	🟧	🟨	🟨	🟩	🟩

🟩 = Monotonic Convergence
🟢 = Complex Convergence
🟨 = Small Overshoot
🟧 = Large Overshoot
🟥 = Critical Convergence

Figure 7. Low Frequency Features

The red square border in Fig. 7 indicates where the largest overall value of C_L (C_{Lmax}) is located across all the available parameter combinations. Interestingly, the largest value of C_L for all combinations of flapLap and flapGap was located at an angle of incidence of either 22^o or 23^o. In addition, these largest values of C_L also tend to have monotonic convergence profiles, whilst the lower lift cases tend to have more complex convergence profiles. This behaviour can potentially be explained by considering the associated high-lift aerodynamic flow. For each combination of flapLap and flapGap, the specific high-lift geometry shape will have an associated incidence value, which results in the largest local value of C_L. At this best local condition, the high-lift aerodynamic flow is more likely to be attached, which is a relatively simpler situation to resolve numerically in CFD. Away from this best local condition, flow may have more extensive flow separation. This represents more complex flow physics, which is more difficult to numerically resolve, leading to slower and less stable CFD convergence. In summary, this would suggest that a good C_L design and incidence combination will result in a fast and simple convergence history.

A number of profiles also exhibited 'high frequency' localized features, where the convergence profiles did not converge to a steady state solution. Figure 5 is an example of a convergence history with high frequency features. Figure 8 summarizes those profiles that exhibited high frequency oscillations in their convergence histories. It is clear from this summary that only profiles with overshoot or critical convergence also exhibit high frequency oscillations. This again suggests these situations are more difficult to resolve/converge, as discussed previously, perhaps indicating the flow is separated and unsteady in nature.

It is clear that a lot of information about the flow physics of a high-lift system can be extracted from the convergence histories, further supporting the assumption that intermediate data can be used to build a surrogate model to predict aerodynamic performance.

However, the resulting prediction needs to be an improvement over simply stopping the CFD simulation after a smaller number of iterations. Therefore, the values of C_L were analyzed for different numbers of CFD iterations. In each case, the value of C_{Lmax} over the entire design space was identified and the corresponding values of flapLap, flapGap and Alpha recorded. In addition, C_L was derived from the average of the final 200 iterations of the CFD simulation, following industry best practice.

		Angle of Incidence							
flapLap	flapGap	5	10	15	18	21	22	23	24
0	0.0075	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩
0	0.011875	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩
0	0.01625	🟥	🟩	🟩	🟩	🟩	🟩	🟩	🟩
0	0.0206	🟥	🟥	🟥	🟥	🟥	🟥	🟩	🟩
0	0.025	🟥	🟥	🟩	🟩	🟩	🟩	🟩	🟩
0.00375	0.0075	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩
0.00375	0.011875	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩
0.00375	0.01625	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩
0.00375	0.0206	🟩	🟥	🟥	🟥	🟥	🟩	🟩	🟩
0.00375	0.025	🟩	🟥	🟥	🟩	🟩	🟩	🟩	🟩
0.0075	0.0075	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩
0.0075	0.011875	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩
0.0075	0.01625	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩
0.0075	0.0206	🟩	🟥	🟥	🟥	🟥	🟩	🟩	🟩
0.0075	0.025	🟩	🟥	🟥	🟩	🟩	🟩	🟩	🟩
0.01125	0.0075	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩
0.01125	0.011875	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩
0.01125	0.01625	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩
0.01125	0.0206	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩
0.01125	0.025	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩
0.015	0.0075	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩
0.015	0.011875	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩
0.015	0.01625	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩
0.015	0.0206	🟩	🟥	🟥	🟩	🟩	🟩	🟩	🟩
0.015	0.025	🟩	🟩	🟥	🟩	🟩	🟩	🟩	🟩

🟩 = No Oscillations

🟥 = High-Frequency Oscillations

Figure 8. High Frequency Features

Analysis of the CFD data based on 4000 iterations (i.e. C_L averaged over 3801–4000 iterations) shows that C_{Lmax}≈4.254, occurring at flapLap=0.00, flapGap=0.011875 and alpha=22°:

$$\mathbf{F(0.00,\ 0.011875,\ 22) \approx 4.254}$$

These coordinates and C_L value will be used as the comparison for all other analysis and this location is highlighted by the red square in Figs. 7 and 8. Analysis of the CFD averaged over 1801 – 2000 iterations show that C_{Lmax} occurs at:

$$\mathbf{F(0.00,\ 0.01625,\ 22) \approx 4.255}$$

This is clearly a different location to the one identified when the CFD simulations were run for 4000 iterations and is actually a slightly higher value. Therefore, if calculations had been stopped at 2000 iterations, the wrong location would be predicted, with a small overestimate of the true C_{Lmax} value. Analysis of the CFD data from 1500 iterations indicates that C_{Lmax} occurs at the same location as for 2000 iterations, but again slightly increases the C_{Lmax} value:

$$\mathbf{F(0.00,\ 0.01625,\ 22) \approx 4.259}$$

Analysis of the CFD data from 1000 iterations shows that C_{Lmax} occurs at:

$$\mathbf{F(0.00,\ 0.0206,\ 22) \approx 4.199}$$

This is again a different location from 4000 iterations and is moving further away in terms of flapGap, whilst the value of C_{Lmax} is now reduced. Finally, analysis of the CFD data for up to 500 iterations indicates that C_{Lmax} occurs at:

$$\mathbf{F(0.00,\ 0.025,\ 18) \approx 3.591}$$

This again is a different location to 4000 iterations, but unlike the previous analysis where the changes were only in flapGap, at this number of iterations the alpha value has now also changed. This suggests that even the gross aerodynamic flowfield incidence effects have not settled down by this number of iterations.

From this analysis it can be concluded that the effect of flapLap and alpha are resolved relatively quickly (reduced number of iterations), since the movement of C_{Lmax} is restricted to FlapGap, even after only 1000 iterations. However, the simulations need more iterations to fine tune and converge flapGap, suggesting that the performance of this particular high-lift test case is very sensitive to this parameter.

The difference in the absolute values between different convergence profiles is relatively small. However, when a surrogate is used as part of an optimization search, the accuracy of the individual predictions can be less important. In particular, provided the surrogate can direct the search to the optimum and hence the ranking of the individuals in the population is maintained, then the best individuals should still be selected [8, 9]. This means the optimizer will use the incremental change in performance between different designs or parameter values. It is therefore not so critical that a surrogate model can predict the same absolute performance of a design, as the CFD simulations, but rather the correct delta/trends between different designs.

This concludes the analysis of the data set. Section 3 will introduce the prediction method used.

3. Method - Prediction

In this section we will describe the RNN ensemble modeling technique used to predict aerodynamic performance using intermediate CFD data.

Neural networks are nonlinear models used to approximate solutions to complex problems and can be used to model any nonlinear function. They acquire knowledge of the system or environment they are embedded in through observations and use them to train the network [10]. Recurrent neural networks (RNNs) are dynamical systems that are specifically designed for temporal problems, as they have both feed-back as well as feed-forward connections. More specifically, a form of memory is incorporated in RNNs, which makes them ideal for predicting CFD convergence data, as the states of the neurons from previous iteration steps are stored and used to influence the prediction of future iterations.

The overall structure of a RNN consists of synaptic connections between the inputs, hidden and output layers of neurons. Knowledge is represented in a network by the values of these synaptic connections. The states of the neurons are dependent on these free parameters, the inputs to the neurons and the states of the neurons at previous time steps [11]. A RNN can have copies of any neuron in the network from the previous time-step and they can be used to influence the prediction of data at future iterations. The objective of learning is to train the network by adjusting the connection weight values, over several training epochs, to reduce the output error of the network. Training moves the error towards a minimum point on the error surface, which has the free parameters of the network as its coordinates [10]. An example of the RNN structure used in this work can be seen in Fig. 11.

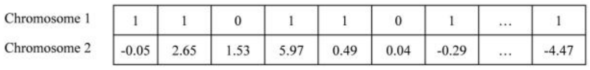
An ensemble of surrogate models consists of many different models that are designed in parallel and used in combination to give a final prediction. Ensembles have been shown to provide better generalization performance than single models [7]. Ensembles can include information that is not contained in a single model [12] 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 [14], as summarized by the Error-Ambiguity Decomposition presented by Krogh and Vedelsby [15]. 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].

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 [12, 16, 17, 18]. A hybrid MOEA (HMOEA) is made up of global and local search components, with the global search used to find suitable starting weight values in the network and the local search used to fine tune them to their optimal value [19, 20]. A HMOEA has been used in this work to build and train the individual RNNs, which are then used in the ensemble. The following sections will provide information on the specific techniques used.

Evolutionary algorithms (EAs) can be considered as multi-point search strategies that are able to “sample (a) large search space” [21] and escape local optima to find global optimum solutions [22]. EAs are stochastic search and optimization procedures that are based on the principles of natural genetics and natural selection [23]. 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 [22]. 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 [21].

The non-dominated sorting genetic algorithm II (NSGA-II) [24] is used as the global MOEA in this work. Each RNN in the population is encoded using two chromosomes. The first of Boolean type to represent the structure of the networks and the second is of real values to represent the weights of the networks. Figure 9 represents how the two chromosomes are linked and that specific alleles represent specific connections. This means that when a Boolean connection is present, the corresponding weight value is used by the network. The direct method of representing the network structure, as described in [20] is used, with every possible connection represented in the chromosomes.

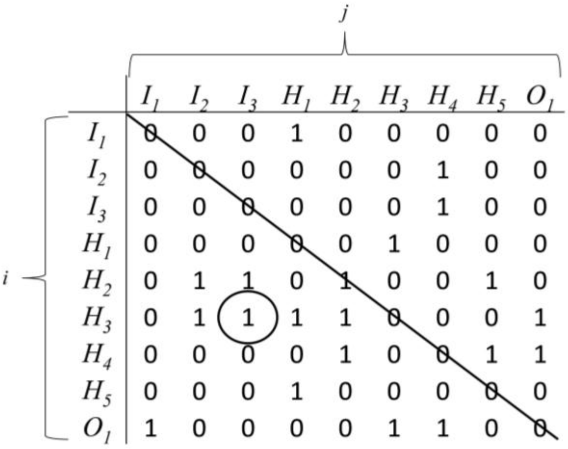


Chromosome 1	1	1	0	1	1	0	1	...	1
Chromosome 2	-0.05	2.65	1.53	5.97	0.49	0.04	-0.29	...	-4.47

Figure 9. 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.

Figure 10 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. 10 shows that hidden neuron 3 (H_3) receives activation from input neuron 3 (I_3). Figure 11 illustrates the network presented in Fig. 10, 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 recurrent connections, which originate from the neurons at (*t-1*).



	I_1	I_2	I_3	H_1	H_2	H_3	H_4	H_5	O_1
I_1	0	0	0	1	0	0	0	0	0
I_2	0	0	0	0	0	0	1	0	0
I_3	0	0	0	0	0	0	1	0	0
H_1	0	0	0	0	0	1	0	0	0
H_2	0	1	1	0	1	0	0	1	0
H_3	0	1	1	1	1	0	0	0	1
H_4	0	0	0	0	1	0	0	1	1
H_5	0	0	0	1	0	0	0	0	0
O_1	1	0	0	0	0	1	1	0	0

Figure 10. Recurrent Neural Network Matrix

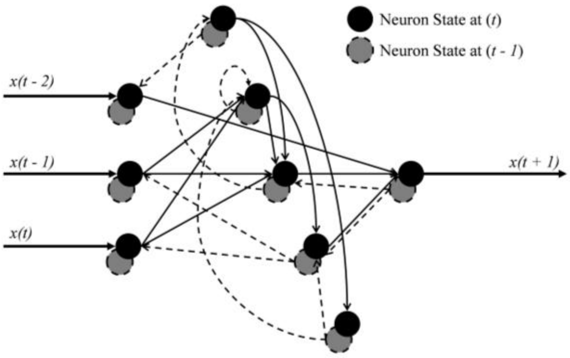


Figure 11. Example of Recurrent Neural Network

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 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 individual models are specifically trained for long-term prediction by training them to recursively predict a certain number of iterations ahead during the training phase. Three consecutive data points are presented to the RNN, which represent the CFD data at three iterations (e.g. *x(t-2)*, *x(t-1)* and *x(t)*). These are used to predict the CFD at the next iteration (e.g. *x(t+1)*) and this is also illustrated in Fig. 11.

Only the first three known data points are used as an initial input to the model and after each prediction the predicted value is fed back and used as input for prediction of data at the next iteration. The input vector only has three consecutive data points, so the predicted values replace the data in the input vector once it has been predicted.

The model performance is not determined until all iterations in the training data set have been predicted. The predicted values are compared to the known data and a MSE value is calculated. This means that although only the data at the next iteration is being predicted, the model is not a single step ahead predictor, which is common in most time-series prediction tasks, as the model recursively predicts a certain number of iterations before the fitness is determined. This training method of predicting several steps ahead is important to the long term prediction of the CFD data because after the training phase the model will be used to recursively predict the remaining convergence history data (test data). It is hypothesized that a model designed to exclusively predict only one step ahead may not be suitable, where as a model that is specifically designed to predict many steps ahead should be more appropriate.

Also, CFD convergence histories do not have repeating features in the training data that are then seen in the test data. This too is not typical of a time-series prediction task, so makes this long term prediction task even more challenging, as the trend of the convergence history needs to be learnt using a very small amount of data, which may not be representative of the future distribution of data.

A gradient descent local search is used to fine tune the weight values of the network once it has been decoded, prior to calculating the individuals fitness values and affecting the second chromosome of the 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 [25, 26].

The learning algorithm used is the IRPropPlus [27] 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 [28].

Both Lamarckian and 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 [29]. By using a Lamarckian search, all of the information learnt by an individual is retained and used to guide the search. If Baldwinian 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.

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. Different selection methods is not the focus of this paper, however combining all members in the Pareto set or selecting some based on a certain criteria have proved successful [19, 6]. However, selecting individuals can be advantageous as some individuals may not provide suitable predictions and it has been shown that to sample many of the created models can be better than sampling them all [30]. Figure 12 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 selected surrogates are combined to give a final prediction of the CFD lift coefficient C_L.

A simple selection based on the individual models performance on a validation test set will be performed, with the five best models being selected to form the ensemble.

A convergence history can be considered as an evolving curve, where the last data points are the more reliable/converged. Also, because the initial phase of a CFD convergence history is transient, using the early data to train the networks can cause it to misinterpret the underlying function. Therefore, the first 800 iterations of each CFD convergence history has been discarded.

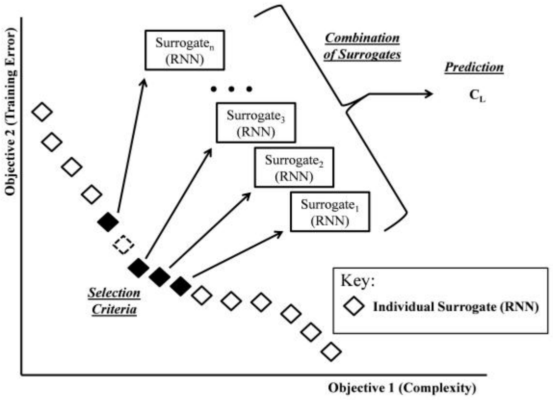


Figure 12. Pareto Set of Solutions

Initial investigations have shown that to use a RNN to recursively predict many data points, when only giving it the first three, can cause the model problems. Therefore, the remaining 3200 data points are sampled at every 5th point, resulting in a data set of 640 points.

The first 30 data points, which corresponds to iterations 801 – 950, are used for training. The next 10 data points, corresponding to iterations 951 – 1000, are used as a validation data set, where selection of the individual models in the ensemble is based on the performance of the models on this validation data. The remaining 600 data points correspond to iterations 1001 - 4000 and are not used during training or validation.

Predictions are made up to the 640th data point (4000th iterations) and, after selection, the output of the five models are combined, using a simple average of each model output at each data point. The last 40 data points, corresponding to iterations 3801 – 4000, from the ensemble are averaged and compared to the converged CFD data average over iterations 3801 – 4000. An additional comparison to see if the prediction model is making an improvement over just stopping the CFD simulations at 1000 iterations (the end of the validation set) is also made, by comparing the predicted results with those of the CFD data averaged over iterations 801 – 1000.

Each simulation is run 10 times to take account of the random elements of the HMOEA. Table II presents the parameter values used by the prediction model and Section 4 presents the results achieved for predictions around C_L max, predictions for fixed flapLap and predictions of a wing polar.

Table II. Model Parameters

Global Search Parameters	
Number of Generations	500
Population Size	100
Local Search Parameters	
Frequency	10
Duration	20
Probability	50%
Warm Up Length	25% (8 data points)
Other Parameters	
Data Normalization	-0.1 – 0.1
Bounds Range	-10 - 10

4. Simulation Results

4.1 Predictions around C_{Lmax}

As previously mentioned, the analysis of the CFD data from 3801 – 4000 iterations shows that C_{Lmax} occurs at:

$$\mathbf{F(0.00,\ 0.011875,\ 22) \approx 4.254}$$

And analysis of the CFD data from 801 – 1000 iterations shows that C_L max occurs at:

$$\mathbf{F(0.00,\ 0.0206,\ 22) \approx 4.199}$$

Figures 13 and 14 are contour plots of how C_L varies with flapGap and alpha in this area and therefore represent what the design space looks like around C_{Lmax} at 4000 and 1000 iterations respectively. It is clear from this analysis that the design space is different.

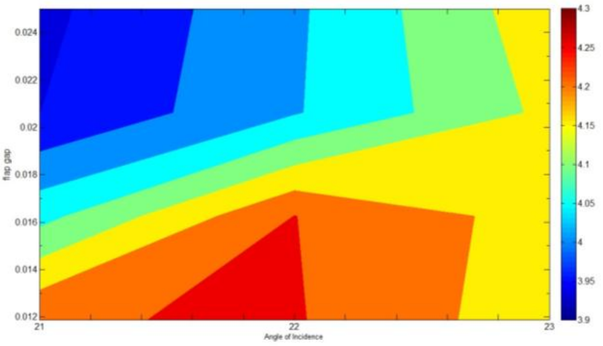


Figure 13. Design Space at 4000 iterations

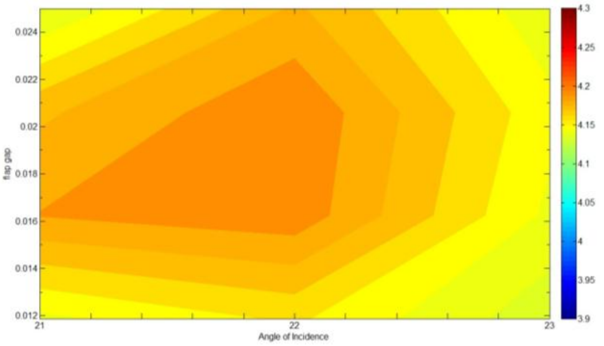


Figure 14. Design Space at 1000 iterations

Using the prediction method to predict the values of C_L around the identified C_{Lmax} at 1000 iterations produces the contour plot shown in Fig. 15. It can be seen from this plot that although the design space looks similar to the result at 1000 iterations, the prediction method has identified a C_{Lmax} value that is closer in magnitude and location to C_{Lmax} at 4000 iterations than can be achieved at 1000 iterations. The following summarizes this:

$$\mathbf{F(0.00,\ 0.01625,\ 22) = 4.243}$$

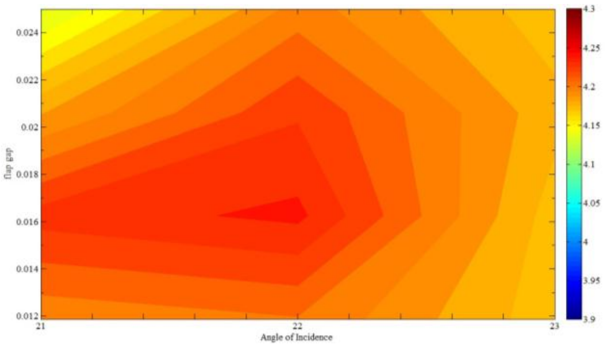


Figure 15. Predicted Design Space

4.2 Predictions for fixed flapLap

This work involves a different area of the design space, where the values of flapLap and alpha are fixed at 0.01125 and 21° respectively. Figure 16 illustrates what the design space looks like when flapGap is varied for 4000 and 1000 iterations, as well as the results of the prediction model. Similar to the previous results, the absolute magnitudes are not correct, but the trend of the design space is better when using the prediction method, compared to the CFD simulation being stopped at 1000 iterations.

Figures 17 and 18 are examples of two of the convergence histories that make up points in Fig. 16. Figure 17 is the convergence history and ensemble prediction for flapGap = 0.01625 and Fig. 18 is the convergence history and ensemble prediction for flapGap = 0.0206.

It can be seen from Fig. 17 the reason for under prediction of the C_L value, is that although the ensemble has predicted the correct trend of the convergence history, it has settled to a lower stable value. Similarly, it can be seen why the ensemble has predicted an overestimation of the final value of C_L for flapGap = 0.0206 in Fig. 18, with the ensemble predicting a damped oscillation for the convergence history, when in reality it is a decaying profile.

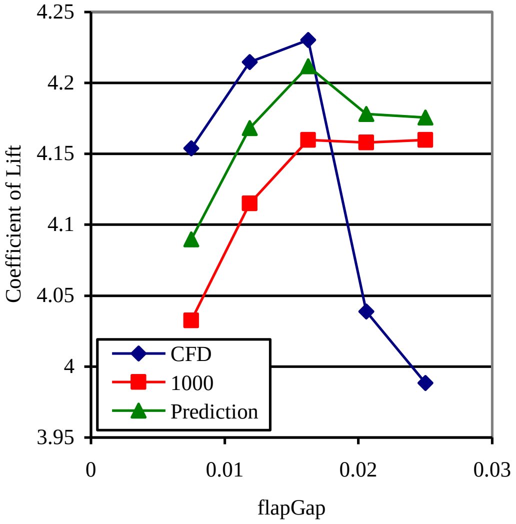


Figure 16. Design Space for flapLap = 0.01125 and alpha = 21^o

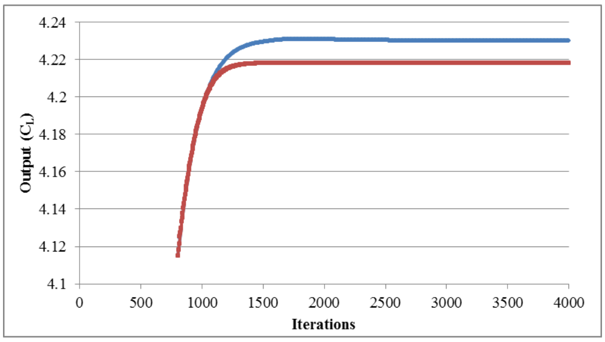


Figure 17. flapGap = 0.01625

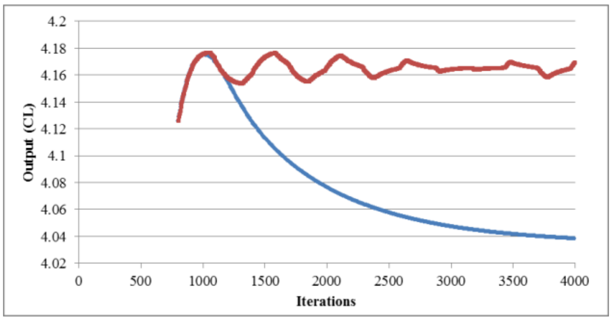


Figure 18. flapGap = 0.0206

However, because the correct trend in the design space can be achieved using the prediction models, this discrepancy in the actual values is far less important, because if the reconstructed design space were to be used during an optimization process, the correct maximum would still be found. This cannot be said for the design space created using the CFD data up to 1000 iterations, where the location of the maximum value is far less clear.

4.3 Predictions of a Wing Polar

The final analysis involved the prediction of a wing polar, with flapLap and flapGap kept constant at 0.00 and 0.025 respectively. The shape of the wing polar is shown in Fig. 19 and shows C_L reaches its maximum value at alpha=23^o, with C_L then reducing at higher values of alpha, probably as a result of flow separation. Figure 19 also provides the results of the predictions and it can be seen that the predictions are very similar to those of the CFD data stopped at 1000 iterations.

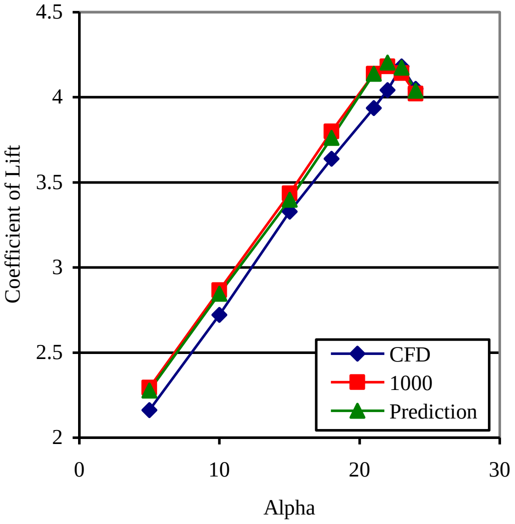


Figure 19. Wing Polar

Similar to the previous analysis, the shape of the convergence profiles and the amount of data used for training influences the performance of the prediction method. Convergence is a decaying profile for alpha = 21^{o} and 22^{o}, but there is a short climb to the top of the profile before it decays. This affects the prediction performance, as the model needs to learn these bumps. In contrast, the profiles for lower alphas are past the maximum values in their convergence profiles and are only decaying from 801 iterations onwards. The profiles for 23^{o} and 24^{o} degrees exhibit monotonic convergence, so are easier to predict.

Figure 20 is an example of the prediction performance of the ensemble at alpha = 23^{o}. This example is under predicting the converged value, although has learnt that the history is of monotonic convergence. On this scale, it can be seen that the CFD profile is exhibiting some low amplitude, high-frequency oscillations, that were not apparent during the initial analysis of the convergence histories in Section 2. However, the prediction is reasonable, based on the fact that only data from iterations 801 – 950 were used to train the model.

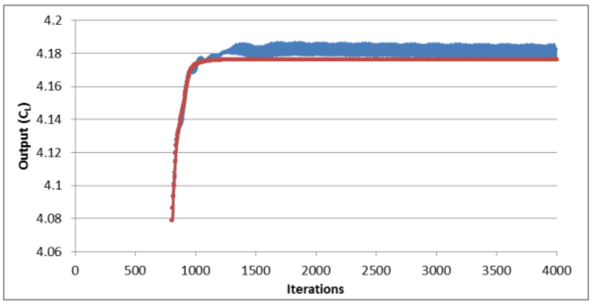


Figure 20. Wing Polar Prediction (Alpha = 23^{o})

In this case, the prediction model does not provide a benefit over simply using less well converged CFD simulations. Figure 19 showed the magnitudes and trends in the design space are approximately the same as simply using CFD simulations run for only 1000 iterations. However, it is worth remembering that the forward prediction process could potentially introduce errors. Hence, it is useful to confirm that the prediction process is at least no worse than simply using less well converged CFD output.

5. Discussion

All of the results presented have shown that the trends of the design space can be better predicted than the absolute magnitudes of the C_L convergence histories. It is clear that the convergence profile has a large impact on prediction performance. For example, if the training data is stopped before a bump in the convergence profile, then it may not be possible for the correct profile to be learnt and that more training data would be the only way of improving this. However, the data sampling may need to change so there are not too many data points that need to be recursively predicted during training. Predicting the trends is important in the context of design space exploration for novel concepts, as finding promising regions, rather than absolute prediction accuracy can still be useful and a search can still be guided in the right direction.

For the cases presented here, the training and prediction of a single simulation takes approximately 16 seconds. When this is added to the time taken for 1000 CFD iterations, the approximate time saving for a single prediction could be 73%, compared to running 4000 CFD iterations. Ideally this amount of time could be saved for each of the 100s or 1000s of CFD simulations required as part of a high-lift design optimization study.

Overall, the prediction method presented is no worse than stopping the CFD simulations after 1000 iterations and, in many of the cases studied, it can offer substantial benefits. The open questions that may still need to be answered is whether the variations in the design space are less than the noise of the predictions and, if the level of accuracy achieved by the predictions is suitable for directing a design search?

6. Conclusions

In this work we have analyzed the convergence histories of a high-lift system. This analysis revealed insights into the flow physics of the setup used and trends could be seen when the parameters of flapLap, flapGap and alpha were varied. Analysis of the CFD performance was investigated for different numbers of iterations and it was found that the effect of flapLap and alpha is resolved quickly, whereas longer convergence is needed to resolve the C_{Lmax} location for flapGap.

A convergence based prediction method has also been used to predict values of C_L using the intermediate data of the high-lift CFD convergence histories. Three different studies were carried out and it was found that the prediction method is no worse than stopping the CFD simulation early and that in certain areas of the design space the prediction method can offer significant benefits by establishing the correct trends of the space.

Convergence histories that exhibit monotonic convergence are generally easier to predict and because these profiles are associated with the areas of C_{Lmax} and therefore the most promising areas of the design space, this is an encouraging conclusion.

A main limitation is that the convergence profile type impacts the performance of the prediction method. In particular, prediction of some profile types would benefit from additional training data. Hence, perhaps an initial classification of an evolving CFD profile could be derived to indicate the number of data points to be used to better train the prediction models. This approach may help to solve the consistent over and under predictions and will be investigated in the future. There are also a number of other improvements that can be made to the model. These include the fact that a number of connections in some models do not have weight values associated with them. This does not affect the models being used, but does mean that some of the individuals are not being correctly represented during the search.

It would also be useful to test the prediction method on other data sets and ultimately integrate it into an optimization loop, so that the true benefits of the time savings can be realized.

7. Acknowledgments

The research is funded by an EPSRC DTC studentship.

Particular thanks to Jason Green and Chris Newbold, within the Flight Physics group at QinetiQ Farnborough.

8. References

[1] 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.

[2] 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.

[3] 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.

[4] 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.

[5] 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.

[6] C. Smith, J. Doherty, and Y. Jin, “Multi-objective Evolutionary Recurrent Neural Network Ensemble for Prediction of Computational Fluid Dynamic Simulations” in *Congress on Evolutionary Computation (CEC), 2014 IEEE WCCI*, July 2014.

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

[8] 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.

[9] 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.

[10] S.Haykin, “Neural Networks and Learning Machines”, 3rd Edition, Prentice Hall, 2008.

[11] B.A.Pearlmutter, “Gradient calculations for dynamic recurrent neural networks: a survey”, *IEEE Transactions on Neural Networks*, vol. 6, no. 5, pp. 1212–1228, 1995.

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

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

[14] 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.

[15] 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.

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

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

[18] 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.

[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, pp. 1–8, 2004.

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

[21] G. Jones, “Genetic and evolutionary algorithms,” 1990.

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

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

[24] 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.

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

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

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

[28] 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.

[29] 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*, pp. 617– 621, 1997.

[30] 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.