International Conference on Control, Automation and Systems 2008
Oct. 14-17, 2008 in COEX, Seoul, Korea

Prediction of Dynamical Properties of Flow Over a Three-element Airfoil via Computationally Intelligent Architectures

Coşku Kasnakoğlu[1,2] and Mehmet Önder Efe[1]

[1]Department of Electrical & Electronics Engineering, TOBB University of Economics and Technology, Ankara, Turkey.
[2]Corresponding author. Email: kasnakoglu@etu.edu.tr

Abstract: In this paper we study various computationally intelligent architectures for prediction of pressure values and velocity components of flow past a three-element airfoil. Six sensor locations are selected around the airfoil and the goal is to predict the flow behavior at the rear of the airfoil using pressure readings from the remaining five sensors. To make the problem more interesting we require the predictor to estimate the flow twenty time steps ahead of current time. Data is collected from CFD simulations of the flow and predictors are built using four different computationally intelligent architectures: Multilayer Perceptron (MLP), Adaptive Neuro Fuzzy Inference System (ANFIS), Radial Basis Function Neural Network (RBFNN), and Least Squares Support Vector Machine (LS-SVM). Levenberg-Marquardt optimization technique is utilized for parameter tuning purposes. In addition, a simple linear predictor is built as a benchmark for comparing the MLP, ANFIS, RBFNN, and LS-SVM based predictors. It is observed that MLP and ANFIS based predictors achieve the best prediction, and the performace of all predictors are superior to that of the simple linear predictor.

Keywords: computational intelligence, artificial neural networks, air flow, Navier-Stokes, NS, airfoil, pressure prediction, velocity prediction, multilayer perceptron, MLP, adaptive neuro fuzzy inference system, ANFIS, radial Basis function neural network, RBFNN, least squares support vector machine, LS-SVM

1. INTRODUCTION

Due to their ability to learn from observed data, Artificial Neural Networks (ANNs) have found extensive fields of application, especially where the data contains hidden implications which are impossible to model by hand [1]. Several types of ANNs exist, each with its own advantages and drawbacks peculiar to the chosen structure. Among the most common type of ANNs one can find the Multi Layer Perceptron (MLP) [2], the Radial Basis Function Neural Network (RBFNN) [3–5], the Support Vector Machine (SVM) [7–9] and the Adaptive Neuro Fuzzy Inference System (ANFIS) [10–13].

In this paper, we apply the methods mentioned above to a problem regarding the prediction of pressure and velocity fields on a three element airfoil. Airfoil analysis and design is undoubtedly a major facet of aerodynamics. The particular problem of predicting certain characteristics of an airfoil from measurements at hand is also an important problem to which considerable research effort has been devoted [14–18]. In this paper we attempt to build a predictor based on MLP, RBFNN, LS-SVM and ANFIS, which predicts the pressure and velocity at a given point on the airfoil, from measurements taken from certain other locations. Pressure is often the only quantity that is feasible to measure and moreover, pressure sensors can only be placed at a limited number of locations on the airfoil. Therefore to obtain information on flow variables other than pressure and at locations other than where the sensors are installed, such a predictor is of potential benefit.

2. PROBLEM DESCRIPTION

The problem we consider regards the airflow over a three element airfoil, and our goal is to predict the pressure and velocities at the rear of the airfoil using pressure measurements at five other locations on the airfoil. We study a cross section of the airfoil over the domain $\Omega := \Omega_1 \setminus \Omega_2$, where $\Omega_1 \subseteq \mathbb{R}^2$ is the rectangle given by $[a,b] \times [c,d]$, and $\Omega_2 \subseteq \Omega_1$ is the domain corresponding to the interior of the airfoil. The flow over the airfoil is governed by the 2D incompressible Navier-Stokes equations

$$\frac{\partial \mathbf{q}}{\partial t} + \mathbf{q} \cdot \nabla \mathbf{q} = -\frac{\nabla p}{\rho} + \nu \nabla^2 \mathbf{q}$$
$$\nabla \cdot \mathbf{q} = 0 \qquad (1)$$

where $t \in \mathbb{R}_+$ is the temporal variable, $(x,y) \in \Omega$ are the coordinates of a point in the flow domain, $\nu \in \mathbb{R}_+$ is the kinematic viscosity, $\mathbf{q} : \Omega \times \mathbb{R}_+ \to \mathbb{R}^2$ is velocity of the flow, $p : \Omega \times \mathbb{R}_+ \to \mathbb{R}$ is the pressure, and $\rho \in \mathbb{R}_+$ is air density. Let u denote the horizontal component of $\mathbf{q}$, and let v denote the vertical component. The flow is subject to the initial conditions

$$u(x,y,0) = v(x,y,0) = 0 \text{ for } (x,y) \in \Omega$$

and boundary conditions

$$u(a,y,t) = u_{\text{in}},\ v(a,y,t) = 0 \text{ for } y \in [c,d],\ t \in \mathbb{R}_+$$
$$u(x,c,t) = u_{\text{in}},\ v(x,c,t) = 0 \text{ for } x \in [a,b],\ t \in \mathbb{R}_+$$
$$u(x,d,t) = u_{\text{in}},\ v(x,d,t) = 0 \text{ for } x \in [a,b],\ t \in \mathbb{R}_+$$
$$\frac{\partial p}{\partial x}(b,y,t) = 0 \text{ for } y \in [c,d],\ t \in \mathbb{R}_+$$
$$u(x,y,t) = v(x,y,t) = 0 \text{ for } (x,y) \in \partial\Omega_2,\ t \in \mathbb{R}_+$$

978-89-93215-01-4-98560/08/$15

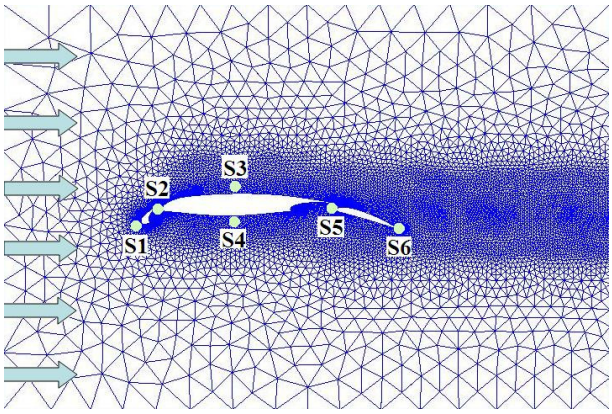

Fig. 1 The airfoil geometry, sensor locations and the mesh used for numerical simulations.

where $\partial\Omega_2$ is the surface of the airfoil. The initial and boundary conditions represent a situation where the wing is stationary in a wind tunnel, after which the air starts flowing from the left to right at speed u_{in}. Fig. 1 shows the airfoil geometry and sensor locations, together with the mesh used for numerical simulations. The numerical simulations are performed using the *Navier2d* solver for MATLAB [19]. The pressure sensors are located at the front (S1) and rear (S2) of the first element, top (S3) and bottom (S4) of the second element, and at the front (S5) of the third element. The sensor measurements are takes every Δt seconds, where Δt is the time step. The goal is to predict the flow behavior at the rear of the airfoil (S6), using measurements from sensors S1-S5. To make the problem more interesting and challenging, we seek for a predictor that will predict the flow behavior D time steps ahead, where $D \in \mathbb{N}$. More formally, we would like to find functions $f_p,\ f_u,\ f_v : \mathbb{R}^5 \to \mathbb{R}$ such that

$$
\begin{aligned}
p_6(k+D) &= f_p(p_1(k), p_2(k), p_3(k), p_4(k), p_5(k)) \\
u_6(k+D) &= f_u(p_1(k), p_2(k), p_3(k), p_4(k), p_5(k)) \\
v_6(k+D) &= f_v(p_1(k), p_2(k), p_3(k), p_4(k), p_5(k))
\end{aligned}
$$

for $k \in \mathbb{N}$ where $p_1(k)$ is defined as $p_1(k) := p(x_{\text{S1}}, y_{\text{S1}}, t_k)$, with $t_k = k\Delta t$ and $(x_{\text{S1}}, y_{\text{S1}}) \in \partial\Omega_2$ is the location of sensor S1. The definitions for $p_2(k)$, $p_3(k)$ and so on follow similarly. As mentioned in the introduction, the functions f_p, f_u and f_v will be sought using four different ANN architectures, which will be described briefly in the section following.

3. MODELING STRATEGIES

This section introduces the MLP, RBFNN, ANFIS and LS-SVM structures briefly.

3.1 Multilayer Perceptron

The structure of MLP with two hidden layers, m inputs and single output is illustrated in Fig. 2. The motivation of utilizing MLP structures for applications involved with possibly large numerical data sets is the ability of discovering hidden relations within a massively interconnected network structure. Possibility of changing the configuration and the availability of very efficient learning mechanisms make the MLP structure a good alternative to follow in applications like the one considered in this paper. For learning with the MLP structure, consider the regression problem over the pairs

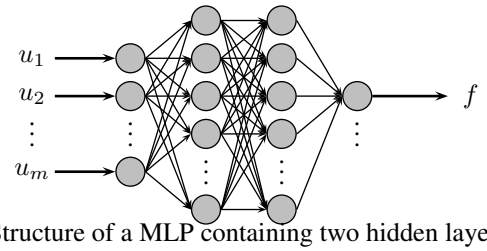

Fig. 2 Structure of a MLP containing two hidden layers and a single output

$$
\mathcal{T} = \{(\mathbf{u}_1, \mathbf{d}_1), \ldots, (\mathbf{u}_{N_\mathcal{T}}, \mathbf{d}_{N_\mathcal{T}}, \mathbf{d}_{N_\mathcal{T}})\}, \quad \mathbf{u}_i \in \mathbb{R}^m,\ \mathbf{d}_i \in \mathbb{R}^n \tag{2}
$$

Compactly, denote the number of layers excluding the input layer by H and the ith layer output vector by $\mathbf{h}_i$, where $\mathbf{h}_i = \Phi(\mathbf{s}_i)$ and $\mathbf{s}_i$ is the vector of net sums computed as

$$
\mathbf{s}_i = \mathbf{w}_i \mathbf{h}_{i-1} + \mathbf{B}_i, \quad i = 1, 2, \ldots, H \tag{3}
$$

where $\mathbf{w}_i$ and $\mathbf{B}_i$ correspond to the weight and bias terms of the ith layer. It is clear that for a NN structure with two hidden layers containing hyperbolic tangent type activation functions, and a linear output layer, the successive computations through the network (one forward pass to compute the output) would be

$$
\begin{aligned}
&\mathbf{h}_0 = \mathbf{u} && \text{(4a)} \\
&\mathbf{s}_1 = \mathbf{w}_1 \mathbf{h}_0 + \mathbf{B}_1 \ \text{ and } \ \mathbf{h}_1 = \tanh(\mathbf{s}_1) && \text{(4b)} \\
&\mathbf{s}_2 = \mathbf{w}_2 \mathbf{h}_1 + \mathbf{B}_2 \ \text{ and } \ \mathbf{h}_2 = \tanh(\mathbf{s}_2) && \text{(4c)} \\
&\mathbf{s}_3 = \mathbf{w}_3 \mathbf{h}_2 + \mathbf{B}_3 \ \text{ and } \ \mathbf{f} = \mathbf{s}_3; && \text{(4d)}
\end{aligned}
$$

where the weight and bias terms seen above have appropriate dimensions and the input-output relation would simply be $\mathbf{f} = \mathbf{w}_3 \tanh(\mathbf{w}_2 \tanh(\mathbf{w}_1 \mathbf{u} + \mathbf{B}_1) + \mathbf{B}_2) + \mathbf{B}_3$. Once a structure is built, the next issue is to adopt a suitable learning strategy. Although there are numerous alternatives for tuning the MLP weights and biases, Levenberg-Marquardt (LM) optimization technique is the one that is frequently used for its rapid convergence. The LM algorithm is an approximation to the Newton's method, and both of these methods have been designed to solve the nonlinear least squares problem, (See [20, 21]). Since the problem considered here is involved with offline training of a MLP structure, LM algorithm fits our problem setting well. We vectorize the set of all adjustable parameters and denote this vector by ω, which is a $P \times 1$ vector. At time k, a cost function, which is an empirical risk function, qualifying the performance of

the interpolation can be given as

$$E(\omega_k) = \frac{1}{2}\sum_{i=1}^{N_{\mathcal{T}}} ||\mathbf{d}_i - \mathbf{f}(\mathbf{u}_i, \omega_k)||^2 \tag{5}$$

and the LM update is formulated as

$$\omega_{k+1} = \omega_k - \left(\alpha\mathbf{I} + \nabla^2_{\omega_k} E(\omega_k)\right)^{-1} \nabla_{\omega_k} E(\omega_k) \tag{6}$$

where $\alpha > 0$ is a user-defined scalar design parameter and $\mathbf{I}$ is an identity matrix of appropriate dimensions. It is important to note that, for small α, (6) becomes the standard Gauss-Newton method, and for large α, the tuning law becomes the standard Error Backpropagation (EBP) algorithm. Therefore, LM method establishes a good balance between EBP and Gauss-Newton strategies.

3.2 Radial Basis Function Neural Networks

RBFN structure is a special class of ANN that operate on the basis of features. The regions of the input space are partitioned by the basis functions that become active on some subset of the input space and the input-output interpolation is carried out based on this locality based mechanism. Simply, the output of a RBFNN structure can be computed via weighted sum of all activation degrees of neurons containing the feature vectors. In summary, the output of a neuron is given by

$$f = \sum_{i=1}^{H} \mathbf{y}_i \mathbf{h}_i \tag{7}$$

where

$$\mathbf{h}_i = \prod_{j=1}^{m} \mu_{ij}(u_j, c_{ij}, \sigma_i, \phi_{ij}) \; . \tag{8}$$

The adjustable parameters of the RBFNN structure are the centers (c_{ij}), variances (σ_i) and other relevant shape parameters (ϕ_{ij}) for the basis functions in (8), and weight parameters ($\mathbf{y}_i$) in (7). As the tuning strategy, we choose the LM algorithm presented in the previous subsection.

3.3 Adaptive Neuro-Fuzzy Inference Systems

Fuzzy logic offers one natural way for representing knowledge that is similar to human reasoning. Partitioning the input space by the use of fuzzy membership functions, determining the local conclusions through rules and utilizing a flexible method of combining the localized information result in a highly interpretable and useful model that acts globally. ANFIS, in this respect, is one of the widely known architectures exploiting the power of connectionist structures while maintaining the verbal nature through membership functions and inference mechanisms, [10]. In the ANFIS structure illustrated in Fig. 3, the crisp inputs are fuzzified through the computation of membership functions. This practically maps the input space to a feature space characterized by fuzzy sets. In the inference engine, computed membership values for each rule are converted into a firing strength that indicates the activation level of the rule. The parameters of

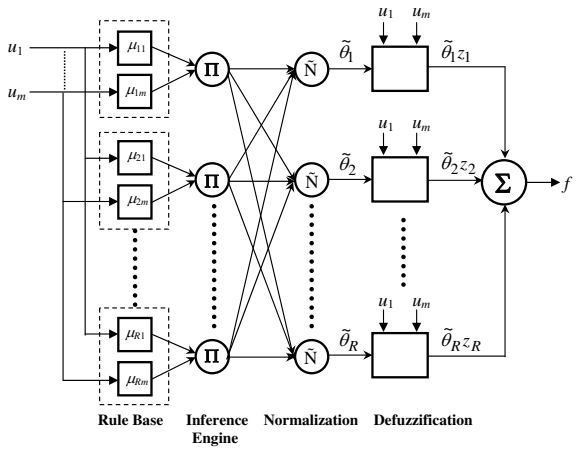

Fig. 3 Internal connectivity of ANFIS structure

the membership functions and auxiliary parameters are stored in the knowledge base, and a defuzzifier maps the output of the inference engine to a scalar output value, which is crisp. As shown also on the figure, defining θ_i as the firing strength and $\tilde{\theta}_i$ as the normalized firing strength of ith rule, the input output relation of the ANFIS structure with the rulebase structure containing R rules having the structure IF u_1 is $\mathbb{U}_{r,1}$ AND u_2 is $\mathbb{U}_{r,2}$ AND ... AND u_m is $\mathbb{U}_{r,m}$ THEN $y_r = z_r$, product inference and first order Sugeno type defuzzifier is as given in (9a)-(9d) (See [22]). Note that $\mathbb{U}_{r,i}$ stands for the fuzzy set characterized by the membership functions and y_r in the rth rule is the local conclusion suggested by the rule.

$$\theta_i = \prod_{j=1}^{m} \mu_{ij}(u_j) \tag{9a}$$

$$\tilde{\theta}_i = \frac{\theta_i}{\sum_{k=1}^{R} \theta_k} \tag{9b}$$

$$z_i = \zeta_i + \sum_{j=1}^{m} \phi_{ij} u_j \tag{9c}$$

$$f = \sum_{i=1}^{R} \tilde{\theta}_i z_i \tag{9d}$$

In (9a) and (9c) u_j corresponds to the jth entry of the input vector $\mathbf{u}$. According to (9d), it is seen that the ANFIS structure has single output. The training is achieved by adopting a hybrid tuning mechanism. Specifically, ζ_i and ϕ_{ij} are adjusted by Least Mean Squares (LMS) algorithm, while the other parameters are tuned by EBP method. It is emphasized in [10] that such a tuning scheme reduces the dimensionality of the search space of EBP algorithm and speeds up convergence.

3.4 Least Squares Support Vector Machines

Support vector machine technique is an alternative approach that is based on the minimization of a structural risk function instead of the empirical risk function. The methods introduced above are based on the latter and therefore, the performance of SVMs are much higher

than those structures, especially in classification problems. However, regression problems have a different nature, so requiring the best fit to a set of data might result in a situation recommending one of the strategies considered above. Least squares support vector machines, as discussed in detail in [9], is a special class of SVMs changing the inequality constraints to equality constraints and resulting in the loss of sparseness properties. In this paper we consider LS-SVM setting as our goal is to approach the target values as much as possible. Consider the regression problem over the pairs

$$\mathcal{T} = \{(\mathbf{u}_1, d_1), \ldots, (\mathbf{u}_{N_\mathcal{T}}, d_{N_\mathcal{T}})\}, \quad \mathbf{u}_i \in \mathbb{R}^m, \;\; d_i \in \mathbb{R} \tag{10}$$

with a function

$$f(\mathbf{u}) = \mathbf{w}^\mathrm{T}\varphi(\mathbf{u}) + \delta \tag{11}$$

where $\mathbf{w}$ and δ denote the weight vector and the bias value, respectively. φ stands for an implicitly defined, possibly a nonlinear map allowing the application of kernel trick wherever necessary. Defining the error $e_i := d_i - f(\mathbf{u}_i)$ and minimizing the structural risk given by

$$R = \frac{1}{2}||\mathbf{w}||^2 + \frac{C}{2}\sum_{i=1}^{N_\mathcal{T}} e_i^2 \tag{12}$$

lets us obtain the best values of $\mathbf{w}_i$s causing least complexity represented by $||\mathbf{w}||^2$, where C is the regularization constant determining the relative importance of the terms contributing to R, [23]. According to (12), large C results in better fit to the given data. The primal form of the optimization problem can be expressed compactly as

$$\min_{\mathbf{w},\delta,e} \;\; \frac{1}{2}||\mathbf{w}||^2 + \frac{C}{2}\sum_{i=1}^{N_\mathcal{T}} e_i^2 \tag{13}$$
$$\text{such that} \quad d_i = \mathbf{w}^\mathrm{T}\varphi(\mathbf{u}_i) + \delta + e_i, \quad i = 1, 2, \ldots, N_\mathcal{T}$$

The problem described above can be converted into an optimization problem by exploiting the dual representation. Denoting the Lagrange multipliers by λ, the Lagrangian can be constructed as in

$$\mathcal{L}(\mathbf{w}, \delta, e, \lambda) = \frac{1}{2}||\mathbf{w}||^2 + \frac{C}{2}\sum_{i=1}^{N_\mathcal{T}} e_i^2 - \sum_{i=1}^{N_\mathcal{T}} \lambda_i \left(\mathbf{w}^\mathrm{T}\varphi(\mathbf{u}_i) + \delta + e_i - d_i\right) \tag{14}$$

and the solution to this optimization problem is obtained at the saddle point of the Lagrangian, i.e. $\max_{\lambda} \min_{\mathbf{w},\delta,e} \mathcal{L}(\mathbf{w}, \delta, e, \lambda)$. The conditions of optimality are given as

$$\frac{\partial \mathcal{L}}{\partial \mathbf{w}} = 0 \Longrightarrow \mathbf{w} = \sum_{i=1}^{N_\mathcal{T}} \lambda_i \varphi(\mathbf{u}_i) \tag{15}$$

$$\frac{\partial \mathcal{L}}{\partial \delta} = 0 \Longrightarrow \sum_{i=1}^{N_\mathcal{T}} \lambda_i = 0 \tag{16}$$

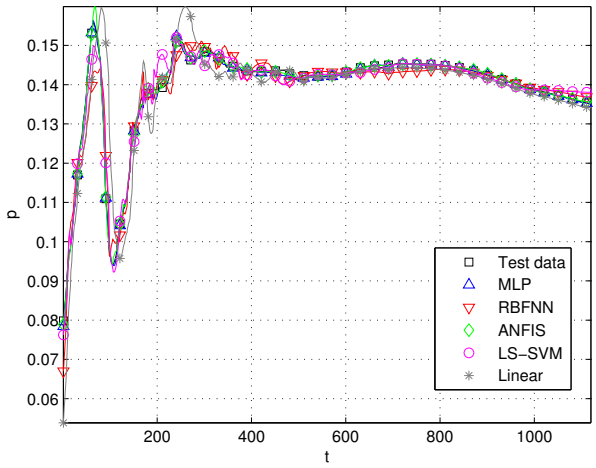

Fig. 4 Pressure prediction results

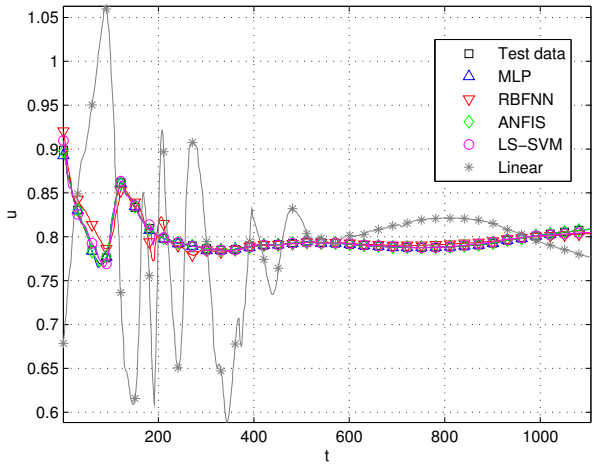

Fig. 5 Velocity (u component) prediction results

$$\frac{\partial \mathcal{L}}{\partial e_i} = 0 \Longrightarrow \lambda_i = Ce_i, \;\; i = 1, 2, \ldots, N_\mathcal{T} \tag{17}$$

$$\frac{\partial \mathcal{L}}{\partial \lambda_i} = 0 \Longrightarrow \mathbf{w}^\mathrm{T}\varphi(\mathbf{u}_i) + \delta + e_i - d_i = 0, \;\; , \;\; i = 1, 2, \ldots, N_\mathcal{T} \tag{18}$$

The solution can be obtained by solving the $N_\mathcal{T}+1$ equations simultaneously. These equations are

$$\delta + \frac{\lambda_k}{C} - d_k + \sum_{i=1}^{N_\mathcal{T}} \lambda_i \varphi(\mathbf{u}_i)^\mathrm{T}\varphi(\mathbf{u}_k) = 0, \quad k = 1, 2, \ldots, N_\mathcal{T} \tag{19}$$

4. SIMULATION RESULTS

In this section, the comparison of the obtained results are discussed. The tests start with the generation of the training, checking and testing data sets. The system described in Section 2. and shown in Fig. 1 has been simulated using Navier2d solver in MATLAB for $25.2 \times 10^{-3} s$ of time with $\Delta t = 1.5112 \times 10^{-5}$ s corresponding to 1666 instants of time. The simulation parameters, described in Section 2have been selected as follows: $a = -0.75$ m, $b = 2.25$ m, $c = -1$ m, $d = 1$ m,

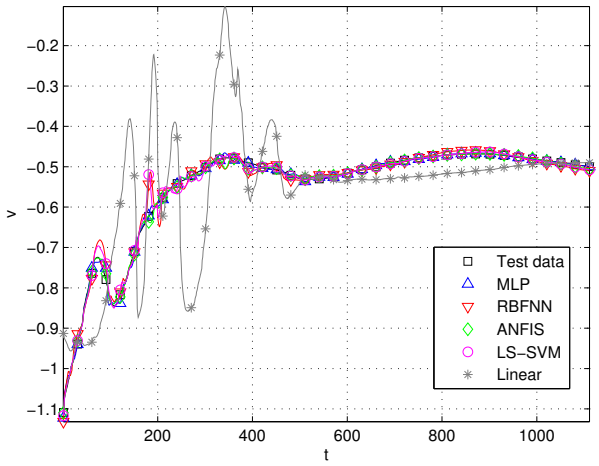

Fig. 6 Velocity (v component) prediction results

$\rho = 1\ kg/m^3$, $\nu = 10^{-5}\ m^2/s$, $u_{\text{in}} = 68\ m/s (\approx$ Mach 0.2). In generating the training data, we first scale the simulation data for p by 1000, u by 65 and v by 30 so as bring the data into a range that is appropriate for the computationally intelligent methods. We set the number of training data $N_{\mathcal{T}} = 400$ and randomly select these rows. The information coming from the sensors S1-S5 are the inputs to the predictor, whereas the desired output is $D = 20$ steps ahead value of the information read at S6. This is particularly important as the prediction horizon is prolonged, the prediction performance of simpler models degrade significantly. The pairs chosen for training set are marked and are excluded in further data selection. Regarding the checking data, which enables the designer to stop training at the best generalization level, we set $N_{\mathcal{C}} = 100$ and consider the remaining data for validating the models.

In MLP approach, the configuration of the model is $H_1 = 15$ and $H_2 = 8$ for the number of hidden layer neurons, with hyperbolic tangent type neurons, and a linear output layer having a single neuron. The MLP structure has 227 adjustable parameters. ANFIS approach considers two linguistic labels for each input leading to a total of 32 rules pointing 222 adjustable parameters in total. In the case of RBFNN, we choose Gaussian functions as basis functions and consider $H = 32$ hidden neurons. The number of adjustable parameters in this approach is 225. Regarding the LS-SVM approach, bspline kernel is utilized with $C = 10$. In this approach, due to the loss of sparseness properties, [9], all of the training pairs are contained as support vectors. We have chosen the settings for the approaches to result in a comparable number of adjustable parameters so as to render the complexities similar and thus making it fair to perform a comparison.

Three sets of experiments have been carried out. In all cases, the predictions are based on the pressure values read from sensors S1-S5, but in the first case the value to be predicted is the pressure p at S6, in the second case is it the u component of the flow velocity at S6, and in the third case, it is v component of the flow velocity at S6.

Method	p Prediction	u Prediction	v Prediction
MLP	1.4194e-007	1.6056e-007	1.928e-005
RBFNN	8.8575e-006	6.1744e-005	2.3624e-004
ANFIS	2.5051e-009	4.4698e-011	3.8114e-009
LS-SVM	1.2413e-005	1.5837e-005	3.9192e-004
Linear	4.0742e-005	6.9796e-003	1.5509e-002

Table 1 MSE Values for Training Data

Method	p Prediction	u Prediction	v Prediction
MLP	3.1378e-007	7.7433e-007	3.2086e-005
RBFNN	8.8575e-006	6.1744e-005	2.3624e-004
ANFIS	3.5955e-007	6.1744e-005	3.1844e-005
LS-SVM	4.4334e-006	5.7278e-006	1.4110e-004
Linear	4.0742e-005	7.0000e-003	1.5500e-002

Table 2 MSE Values for Unseen Test Data

For every case, a separate training and testing has been carried out. Based on these, in Table 1, the mean squared error (MSE) levels obtained during the training phases are tabulated. According to the table, smallest values are obtained with the ANFIS based predictor yielding the S6(k+20) based upon the observations at time k. A closer look would choose MLP structure as the second method that performs well. LS-SVM and RBFNN approaches are seen to perform poorer than these two methods. A simple linear estimator of the form $\text{S6}(k+20) = \sum_{i=1}^{5} \omega_i \text{S}i(k)$ is also built for benchmark purposes. The error levels seen in Table 1 are very small as these quantities are for the data manipulated by the training algorithm. The performances of the methods for unseen test data are given in Table 2, where it is seen that the MLP and ANFIS based predictors are the best performing two approaches that yield accurate predictions for the target data. Clearly, the linear predictor fails totally in providing the velocity predictions. RBFNN and LS-SVM approaches yield poorer results that ANFIS and MLP. In Fig. 4, we illustrate the pressure predictions based on pressure measurements, whereas in Fig. 5 and in Fig. 6, the velocity predictions along u and v directions based on pressure measurements are illustrated. It is clear that all methods except the linear predictor achieve the goal roughly, yet, precise predictions are obtained from MLP and ANFIS based structures.

At this stage, one can question whether these results could be improved by altering the configuration parameters. Based on the numerous experiments we have carried out, the answer to this question is yes; however, maintaining the number of adjustable parameters close to each other, the order of best-performers will not change.

5. CONCLUSIONS

This paper considers the prediction performances of computationally intelligent structures for predicting the pressure and velocity components of a flow past a three-element airfoil. We believe the problem to be of interest because pressure is often the only flow variable that can be measured and sensors can only be placed at a limited number of locations on the airfoil. Hence there is poten-

tial benefit in being able to predict the flow information other than pressure and at locations other than where one has sensors. The challenging nature of the problem considered is the spatially distributed nature of the problem and the unavailability of the past sensory information at the point of interest. Towards the prediction goal, MLP, ANFIS, LS-SVM, RBFNN approaches, as well as a simple linear estimator for benchmark are considered. It is seen that the ANFIS structure results in a very small MSE value between the training data and predictions. Yet, in the test phase, MLP approach performs as well as the the ANFIS structure. Regarding the LS-SVM and RBFNN approaches, in spite of the similar complexities, the prediction accuracies are seen to be poorer than the former two, but still superior to that of the simple linear predictor.

We believe that a main contribution of this paper is unfolding the fact that distant predictions can be made by utilizing computationally intelligent architectures, and showing that ANFIS and MLP approaches are capable of yielding the best predictions for velocity components based on pressure measurements. One practical value having this sort of a predictor is the possibility of devising feedback control systems to meet a predefined set of performance criteria described over the spatial domain of the process. The future work of the authors aim to realize these structures in a predictive control application.

6. ACKNOWLEDGMENTS

The authors gratefully acknowledge the facilities of TOBB ETÜ Library. The authors would also like to thank Darren Engwirda for the *Navier2d* solver.

REFERENCES

[1] S. Haykin. *Neural Networks: A Comprehensive Foundation*. Prentice Hall PTR, Upper Saddle River, NJ, USA, 1994.

[2] K. Hornik, M. Stinchcombe, and H. White. Multilayer feedforward networks are universal approximators. *Neural Networks*, 2(5):359–366, 1989.

[3] J. Park and I. W. Sandberg. Universal approximation using radial-basis-function networks. *Neural Computing*, 3(2):246–257, 1991.

[4] M. J. D. Powell. *Radial basis functions for multivariable interpolation: a review*, pages 143–167. Clarendon Press, New York, NY, USA, 1987.

[5] M. T. Musavi, W. Ahmed, K. H. Chan, K. B. Faris, and D. M. Hummels. On the training of radial basis function classifiers. *Neural Netw.*, 5(4):595–603, 1992.

[6] B. Scholkopf and A.J. Smola. *Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond*. MIT Press, Cambridge, MA, USA, 2001.

[7] N. Cristianini and J. Shawe-Taylor. *An Introduction to Support Vector Machines*. Cambridge University Press, Cambridge, UK, 2000.

[8] B. Schölkopf, C.J. C . Burges, and A.J. Smola. *Advances in Kernel Methods: Support Vector Learning*. MIT Press, 1999.

[9] J.A.K. Suykens, T. Van Gestel, J. De Brabanter, B. De Moor, and J. Vandewalle. *Least Squares Support Vector Machines*. World Scientific, Singapore, 2002.

[10] J.S.R. Jang. ANFIS: Adaptive-network-based fuzzy inference system. *IEEE Transactions on Systems, Man, and Cybernetics*, 23(3):665–685, 1993.

[11] J.S.R. Jang and C.T. Sun. Functional equivalence between radial basis function networks and fuzzy inference systems. *IEEE Transactions on Neural Networks*, 4(1):156–159, 1993.

[12] J.S.R. Jang and C.T. Sun. Neuro-fuzzy modeling and control. *Proceedings of the IEEE*, 83(3):378–406, 2002.

[13] S.-P. Lo. The application of an ANFIS and grey system method in turning tool-failure detection. *The International Journal of Advanced Manufacturing Technology*, 19(8):564–572, 2002.

[14] J.A. Ekaterinaris and M.F. Platzer. Computational prediction of airfoil dynamic stall. *Progress in aerospace sciences*, 33(11-12):759–846, 1997.

[15] L. Davidson. Prediction of the flow around an airfoil using a reynolds stress transport model. *Journal of fluids engineering*, 117(1):50–57, 1995.

[16] Eric Manoha, Catherine Herrero, Pierre Sagau, and Stephane Redonne. Numerical prediction of airfoil aerodynamic noise. In *8th AIAA/CEAS Aeroacoustics Conference and Exhibit*, Breckenridge, CO, 2002.

[17] C. Hah and B. Lakshminarayana. Measurement and prediction of mean velocity and turbulence structure in the near wake of an airfoil. *Journal of Fluid Mechanics*, 115:251–282, 1982.

[18] Jiunn-Chi Wu, L. N. Sankar, and K. R. V. Kaza. A technique for the prediction of airfoil flutter characteristics in separated flow. In *28th Structures, Structural Dynamics and Materials Conference*, pages 664–673, Monterey, CA, 1987.

[19] Darren Engwirda. Navier-Stokes solver (Navier2d). MATLAB Central File Exchange, 2006.

[20] R. Battiti. First- and second-order methods for learning: between steepest descent and newton's method. *Neural Computation*, 4(2):141–166, 1992.

[21] M.T. Hagan and M.B. Menhaj. Training feedforward networks with the marquardt algorithm. *IEEE Transactions on Neural Networks*, 5:989–993, 1994.

[22] T. Takagi and M. Sugeno. Fuzzy identification of systems and its applications to modeling and control. *IEEE Transactions on Systems, Man, and Cybernetics*, SMC-15:116–132, 1985.

[23] S.R. Gunn. Support vector machines for classification and regression. Technical report, ISIS Technical Report, University of Southampton, United Kingdom, 1998.