A preliminary attempt to use neural networks for turbulent eddy classification

By Ron F. Blackwelder

1. Motivation and objectives

This note describes an attempt to use standard neural network tools to fashion a means of detecting eddy patterns in the wall region of a turbulent flow. The research was motivated by the desire to formulate a means to use only flow parameters that can be sensed on the wall to describe the passing eddy structure. If a simple formulation can be obtained, it could conceivably be utilized to control actuators embedded in the wall. Such actuators have been developed by Jacobson and Reynolds(1993a), Blackwelder and Liu (1994), Tung *et al.* (1995), and others. These actuators have the common characteristics that they are small and are typically flush with the wall when not deployed. When they are activated, it is assumed that they will be able to interact constructively with the turbulent eddies near their location to either decrease the wall shear stress, enhance or reduce the mixing, etc. At present, there is only a nascent understanding of the interaction dynamics between the actuators and the eddies in the flow. Nevertheless, for such interaction to succeed, methods to couple the actuators to the oncoming flow must be obtained. General methods must be found that will detect the space and temporal location of the desired structure. In particular, it will be necessary to know when the eddies will arrive at the location of the actuator. This research attempted to use the shear stress measurements on the wall in the vicinity of an actuator location to predict when a particular eddy pattern would arrive and/or occur at the designated location. In this work the eddy pattern to be detected was identified by its velocity signature only.

2. Techniques

Artificial Neural Networks(ANN) have been used rather extensively in control theory for a variety of purposes. They consist of algorithms that, when properly configured, have the ability to "learn" a desired response. In fluid mechanics, Faller *et al.* (1994) utilized an ANN to predict separation pressure on an airfoil after training it with existing unsteady airfoil data obtained at different pitch rates. Jacobson and Reynolds (1993b) used two different ANN controllers to alter the shear stress on the wall of a modeled boundary layer and deduced a skin friction reduction of 8%. Fan, *et al.* (1993) utilized ANN in a transitional boundary layer to reduce the magnitude of the disturbances in the layer.

The approach used in this note is similar to that of Jacobson and Reynolds (1993b). However instead of using a model to generate data, well-resolved direct numerical simulation (DNS) data from a turbulent channel flow was used. The ANN was configured similar to the feed-forward network shown in Fig. 1 adapted

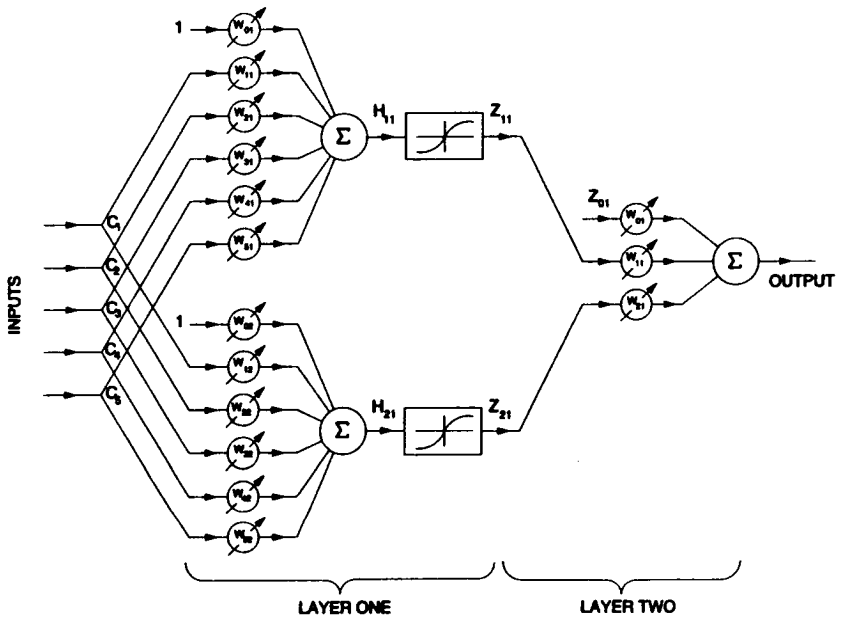

FIGURE 1. Schematic of a two layer ANN with five inputs and one output.

from Jacobson and Reynolds (1993b). This two-layer network consists of five inputs, two internal nodes, and a single output. It is designated as a 5-2-1 network which represents the number of inputs, nodes, and outputs. In a practical device, the inputs would correspond to signals obtained from a series of sensors located on the wall of the flow. Thus only data obtained in the wall region was used as input into the ANN. It was further assumed that for practical application the output from an ANN would be utilized to operate an actuator located at a point, **p**, on the wall.

For the work presented here, the input to the ANN utilized $i_{\max}$ inputs obtained from the two velocity components parallel to the wall. Typically this data was obtained at the first resolved calculation point lying above the wall and hence represented the wall shear, $\partial u/\partial y$ and $\partial w/\partial y$, at the various data points. The choice of these variables and their physical location with respect to the point, **p**, are crucial because this is one of the primary means by which the physics enter the problem. The number of inputs, $i_{\max}$, varied during the course of the investigation from 5 to 50. The inputs included data obtained from locations upstream, Δx, and spanwise, Δz, from the position **p**. Usually $\partial u/\partial y$ and $\partial w/\partial y$ were both used from a single spatial location; hence, the number of spatial locations providing data was always less than or equal to the number of inputs, $i_{\max}$.

Neural networks as shown in Fig. 1 are quite flexible and can consist of a large number of inputs and layers. Few rules exist for their design and it is left to the user to develop a network best suited to his application. One of the few guidelines available is that more than one layer must be utilized to adequately model nonlinearities in a problem. In addition, it behooves the user to keep the number of inputs, nodes, and layers to a minimum to reduce the computational effort.

The ANN used in this investigation consisted of two layers with two to five nodes in the second layer and a single output. The weights were designated as w_{ijk}, where the first subscript denotes the node in the previous layer, the second subscript is the output node for the present layer, and the third subscript is the layer number. A bias input is included in each layer and thus i varies from zero to $i_{\max}$. Likewise j and k have values between unity and $j_{\max}$ and $k_{\max}$ respectively. Thus a total of $(i_{\max}+1)j_{\max}+(j_{\max}+1)$ coefficients, w_{ijk}, were used in the ANN. Their initial values were chosen as random numbers and adjusted later by training.

Letting I_i be the i^{th} input, then the linear sum of the outputs from the first layer, $H_{jk}=I_i w_{ijk}$, was scaled to lie between ± 1 by the sigmoid function, F, which was taken to be the hyperbolic tangent function;

$$Z_{jk}=\tanh(H_{jk})$$

Z_{j1} are the outputs from the first layer and the input into the second layer. By convention, the output of the last layer, O, is not passed through the sigmoid; hence, a two layer ANN with a single output is simply $O=H_{12}$.

The value of the weights were found by training which used a back propagation algorithm described by Hertz *et al.* (1991). This requires *a priori* knowledge of a target vector, ϕ, which the ANN attempts to predict. Choi *et al.* (1994) have shown that a 25% drag reduction can be accomplished by using the normal velocity component at $y^+=10$ to prescribe suction and blowing at the wall directly below its location. Using this result, the target chosen for the present study was the scalar value of the normal velocity component located at $y^+=10$ above the point $\mathbf{p}$. The DNS data were used to extract $m_{\max}$ training sets; each consisted of the pattern of the u and w data near the wall in the neighborhood of $\mathbf{p}$ and the value of the target, $\phi=v(\mathrm{p}_x, y^+=10, \mathrm{p}_z)$. As each training set was presented to the ANN algorithm, the standard deviation was computed from the difference between the target and the ANN output over the m sets of data as

$$\epsilon^2=\frac{1}{2}\sum_{m=1}^{m_{\max}}[\phi^m-O^m(w_{ijk})]^2$$

To minimize the standard deviation, the gradient descent algorithm suggests changing w_{ijk} by an amount Δw_{ijk} proportional to the gradient of ϵ^2 given by

$$\Delta w_{ijk}=-\mu\frac{\partial\epsilon^2}{\partial w_{ijk}}$$

where μ is an arbitrary constant of order unity. Thus for the output stage of an $i_{\max}$-2-1 network, the changes for the weight coefficients are given by

$$\Delta w_{i12}=\mu\frac{\partial O^m}{\partial w_{i12}}(\phi^m-O^m)$$

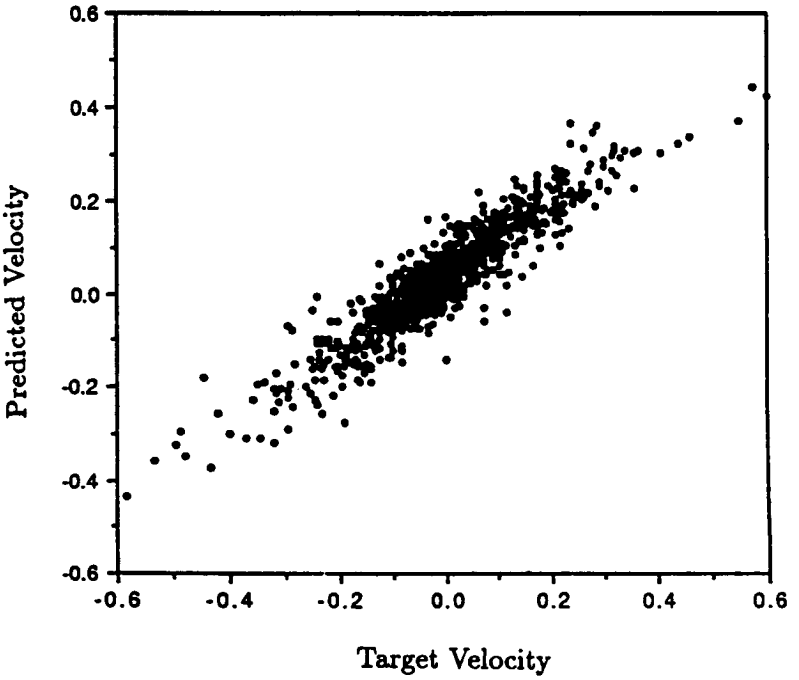

FIGURE 2. Predicted velocity versus the target velocity after one hundred iterations of the test patterns.

where the sum over m is implied. For such a network, the output variable is $O^m = H^m_{12} = w_{k12} Z^m_{k1}$ so that

$$\Delta w_{i12} = \mu Z^m_{k1} \frac{\partial w_{k12}}{\partial w_{i12}} \delta^m$$

or

$$\Delta w_{i12} = \mu \delta^m Z^m_{i1}$$

where $\delta^m = \phi^m - O^m$. This specifies the weights in the second layer. In a similar manner, the back propagation algorithm can determine the weights in the first layer.

3. Results

The main results of this study were obtained by examining the predicted output velocity as a function of the target velocity for the $m_{\max}$ patterns after training. Except where noted, the results are for a 10-4-1 neural network. Typically, 1024 test patterns were taken from one temporal set of data and used in the training. Figure 2 illustrates the output for ten values of $\partial u/\partial y$ and $\partial w/\partial y$ taken at $\Delta x = 0$ and at five spanwise locations, $\Delta z = 0, \pm 13$ and ± 26, with respect to **p**. The best results as determined by the standard deviation were obtained when $\Delta x = 0$; e.g. $\epsilon = 0.062$ for the data in Fig. 2 with $\Delta x = 0$. At $\Delta x = \pm 20$, ϵ increased to 0.10 and at $\Delta x = \pm 40$, $\epsilon = 0.14$.

The training algorithm attempted to calculate values of the weights that minimize the difference between the output and the desired target. Hence it is not

unreasonable to assume that the magnitude of the calculated weights would be an indication of the value of that particular input. If so, an examination of the weights could be used to prune those weights that are the least useful. This appeared to be a valid assumption as long as the input parameter was strongly correlated with the target variable. This was tested by letting one of the inputs have the value of the target velocity. The algorithm produced weights in the first layer, w_{ij1}, that were typically smaller than 0.1 except for the weight related to the input containing the target which was of order unity. However, as training continued and more targets were presented to the algorithm, the weights continued to change. This was true even though the targets presented were a repeat of those targets already analyzed by the algorithm. If the weight in the second layer became small, the values of the weights in the first layer were often large since their effect was not propagated through the second layer due to the smaller weight there. This relationship was a result of the non-linearity in the network and will probably be found in any ANN having more than one layer. It was found that the product of the weights along the propagation path was a better indicator of correlated inputs. That is, when the target value was used as an input on one channel, the product of the weights from the first and second layer for that data path was much larger than for the other channels.

On the other hand, when a random valuable was used as one of the inputs, the results were more consistent; namely the value of the weights associated with the random variable ultimately approached zero. However it often took more than 200 iterations through the set of pattern data before this result was achieved, which was deemed to be excessively long. As stated above, the products of the weights from the different layers through the propagation channel was a much better indicator of the lack of correlation with the target value. In general, an examination of the magnitude of the weights after a fixed number of iterations was of little help in choosing appropriate inputs. But if the weights approached zero and remained very small for a large number of iterations, this was considered a good indication that the input on that channel was indeed of little help in predicting the target and could be pruned. In general, it was found that physical insight was a better guide and indicator of appropriate input variables than the magnitude of the weights.

The time taken for the algorithm to converge to a good prediction of the target was of concern. It was found that the value of ϵ decreased rapidly to a nominal value of 0.06 after three to six passes through the test patterns. Further iterations provided very little decrease in the standard deviation. However the values of the weights were not constant and were often changing significantly after one hundred iterations through the set of patterns. In some cases, the values of the weights were not constant after ten thousand iterations. In a couple of cases no convergence was found at all but rather the weights oscillated. When the weights did converge to a constant value, that final value depended upon the initial random values of weights.

The convergence of the weights was studied by adding dither (i.e. random noise) to the weights at each iteration. A dither amplitude of approximately one per cent of the root mean square value of the weights eliminated the oscillatory nature of

the weights, but did not seem to speed their convergence. However, a slightly small value of the standard deviation was found with the dither.

During the iterations, the set of approximately 1024 pattern data and target values was usually presented sequentially; i.e. $m = 1, 2, 3 \ldots$. It was discovered that presenting the patterns in a random fashion had several advantages. First, the weights did not get caught into a cyclical pattern and oscillate. Secondly, the standard deviation decreased slightly to $\epsilon < 0.05$.

4. Conclusions

The artificial neural networks used in this exercise provided a reasonable prediction of the desired results. The standard deviation between the target values and the output value was typically 6% or less. However, the algorithms took a large number of iterations to converge, suggesting that more work needs to be devoted to improving their speed. Possible uses of the conjugate gradient or other tools could provide improvements in the algorithms. The use of temporal data, in addition to the spatial data use in this study, may also speed convergence.

Acknowledgments

The author wished to thank Stu Jacobson for the use of his ANN algorithm and Tom Bewley for his helpful discussions of control theory and his comments during the course of this research.

REFERENCES

BLACKWELDER, R. F. & LIU, D. 1994 Delay of break-down of streamwise vortices embedded in a boundary layer. ASME Fluids Engineering Division. FED-Vol. 193, *Turbulence Control*, 9, ASME.

CHOI, H., MOIN, P., & KIM. 1994 Active turbulence control for drag reduction in wall-bounded flows. *J. Fluid Mech.* **262,** 75–110.

FALLER, W. E., SCHRECK, S. J., & LUTTGES, M. W. Real-time prediction and control of three-dimensional unsteady separated flow fields using neural networks. AIAA Aerospace Sciences Meeting, Jan 10–13, 1994, AIAA-94-0532.

FAN, X., HOFMANN, L., & HERBERT, T. 1993 Active flow control with neural networks. AIAA Shear Flow Conference, July 6–9, 1993, AIAA-93-3273.

HERTZ, J., ANDERS, K., & PALMER, R. G. 1991 *Introduction to the Theory of Neural Computation.* Addison-Wesley Pub. Co., Redwood City, CA.

JACOBSON, S. A. & REYNOLDS, W. C. 1993a Active boundary layer control using flush-mounted surface actuators. *Bull. Am. Phy. Society.* **38,** 12, 2197.

JACOBSON, S. A. & REYNOLDS, W. C. 1993b Active control of boundary layer wall shear stress using self-learning neural networks. AIAA Shear Flow Conference, July 6–9, 1993, AIAA-93-3272.

TUNG, S., HONG, W., HUANG, J., HO, C.-H., LUI, C., & TAI, Y.-C. 1995 Control of a streamwise vortex by a mechanical actuator. Tenth Symposium on Turbulent Shear Flows, Penn. State University, August 14–16.