

COMPRESSION PRESSURES APPROXIMATION USING FEEDFORWARD NEURAL NETWORKS

АПРОКСИМАЦИЯ ДАВЛЕНИЯ СЖАТИЯ ПРИ ИСПОЛЬЗОВАНИИ НЕЙРОННЫХ СЕТЕЙ ПРЯМОГО РАСПРОСТРАНЕНИЯ

Eng. Hejmal Z., CSc.

Faculty of Military Technology, University of Defence, Czech Republic

Abstract: Radial basis function (RBF) network proved a very good predictive performance of absolute compression pressures inside individual cylinders based on the crankshaft angular velocity. RBF networks offer a means of avoiding difficulties with training process because the learning algorithms for RBF networks can overcome the problem of training that normally occurs in multilayer networks. The prediction quality of the designed RBF artificial neural network depends predominantly on the density of the applied knowledge base (training set) and on RBF network parameters particularly on the spread constant and on the number of neurons in the hidden layer.

KEYWORDS: COMPRESSION PRESSURE, RBF NETWORK, PREDICTION

1. Introduction

Compression pressure of the CI engine presents important diagnostic parameter which significantly influences the CI engine operational characteristics. Especially cold starts are troublesome when compression pressure is low. Decrease of power and increase of harmful emissions are specified as other consequences. Mostly excessive wear down of cylinder (lining), piston rings or obstructed air supply are reasons of low compression pressures. There are procedures based on the instantaneous crankshaft angular velocity for relative compression pressures assessment inside of individual cylinders. Absolute pressure assessment methods based on the course of crankshaft angular velocity have difficulty in the reconstruction algorithms. A possible approach, how to overcome this difficulty, is introduction of pattern recognition techniques. The relation between measured crankshaft angular velocity and the values of compression pressures inside of individual cylinders can be established by suitable artificial neural network.

Figure 1 presents the course of the crankshaft angular velocity during the cranking process of the CI engine T-924.

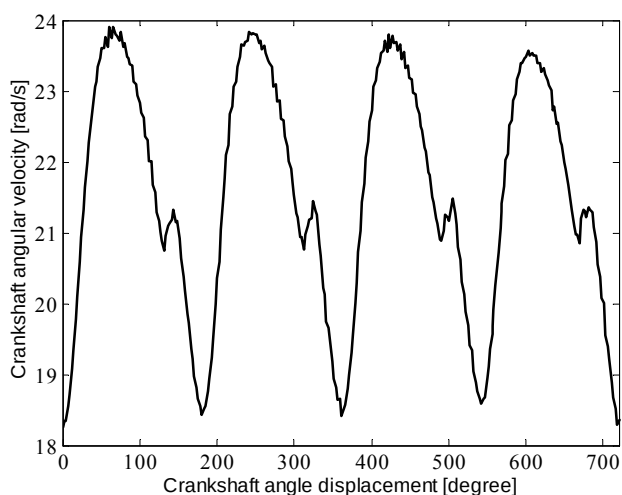


Figure 1: The course of the crankshaft angular velocity during the cranking process

Application of artificial neural network requires huge amount of data (training and testing sets of input and output vectors), which can be obtained using an appropriate model of the whole system (cranking motor and CI engine) or by measurement of real object. The second approach is far more time-consuming but more reliable because verification of the model is not necessary. In the case of the new artificial neural network application the first approach is recommended to obtain the needed information about

the type and topology of the suitable network and about the structure and required extent of the training set.

2. Architecture and working principle of RBF neural network

While there has been much success in developing nonlinear neural network models, in particular multilayer networks trained by backpropagation, researchers, and practitioners have found that, in some cases, it can be difficult to train a multilayer network and obtain good performance without resorting to sophisticated algorithms. The main reason it is difficult to train a multilayer network is that such networks are nonlinear in the parameters. This means that learning algorithms typically use gradient descent approaches as a means of solving the nonlinear optimization problem. It is desirable, therefore, to obtain models which can overcome this problem of training. RBF networks do offer a means of avoiding exactly this problem.

RBF neural networks have the universal approximation ability and from mathematical point of view represent a multidimensional nonlinear function mapping that depends on the distance between the input vector and the centre vector, see [1]. The appropriate RBF network is able to provide the requested mapping of n -dimensional inputs (e.g. the courses of the crankshaft angular velocities during the whole engine cycle) into a set of corresponding compression pressures. The term *radial basis function* derives from the fact that transfer function of individual neuron is radially symmetric. It means that each node produces an identical output for inputs that lie at the same radial distance from the centre. For example the following classes of radial basis functions can be used:

Gaussian radial basis function:

$$f(x) = e^{-\frac{(x-c)^2}{r^2}} \quad (1)$$

Multiquadratic radial basis function:

$$f(x) = \left((x-c)^2 + r^2 \right)^{1/2} \quad (2)$$

Inverse multiquadratic radial basis function:

$$f(x) = \frac{\left((x-c)^2 + r^2 \right)^{-1/2}}{r} \quad (3)$$

Cauchy radial basis function:

$$f(x) = \frac{\left((x-c)^2 + r^2 \right)^{-1}}{r} \quad (4)$$

Radial basis functions mentioned above are depicted in Figure 2.

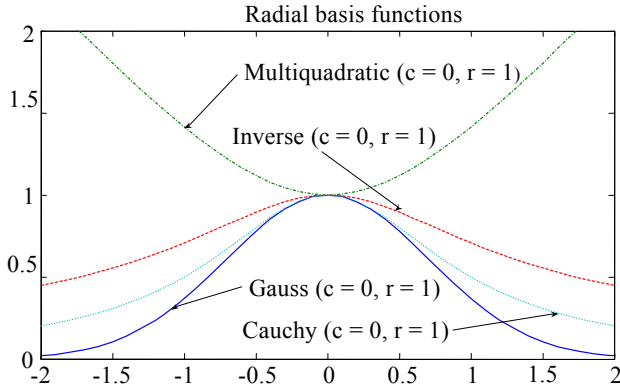


Figure 2: Frequently used radial basis functions

Radial basis function networks may require more neurons than standard feedforward backpropagation networks, but often they can be designed in a fraction of the time it takes to train standard feedforward networks. They work best when many training vectors are available.

RBF networks consist of two layers. The first layer is a hidden radial basis layer of S^1 neurons and the second layer is an output linear layer of S^2 neurons. The structure of RBF neural network is presented in Figure 3.

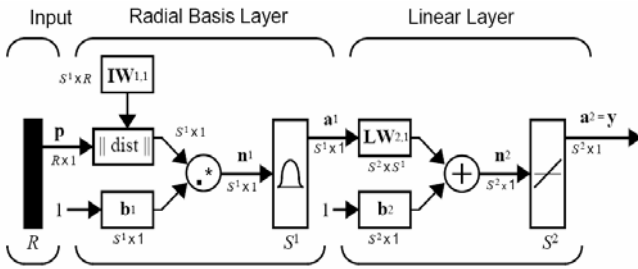


Figure 3: The structure of RBF network

The first layer consists of neurons with radial basis transfer function (in our case Gaussian function (1)). Each neuron's weighted input is the distance between the input vector and its weight vector. Each neuron's net input is the element-by-element product of its weighted input with its bias b^1 . Each neuron's output is its net input passed through radial basis transfer function. If a neuron's weight vector is equal to the input vector (transposed), its weighted input is 0, its net input is 0, and its output is 1. If a neuron's weight vector has the distance of *spread* (optional constant) from the input vector, the net input of the neuron is 0.8326, because each bias b^1 in the first layer is set to $0.8326/\text{spread}$. This gives radial basis functions that cross 0.5 at weighted inputs of $\pm \text{spread}$ and the output of the neuron is then 0.5. The bias b^1 (through the *spread* constant) allows the sensitivity of the RBF neuron to be adjusted. For example, if a neuron had a bias of 0.1 (*spread* = 8.326) it would output 0.5 for any input vector at distance of 8.326 from its weight vector.

The only condition we have to meet is to make sure that *spread* is large enough so that the active input regions of the RBF neurons overlap enough so that several neurons have always fairly large outputs at any given moment. This makes the network function smoother and results in better generalization for new input vectors not occurring between input vectors used in the design. However, *spread* should not be so large that each neuron is effectively responding in the same, large area of the input space, see [2]. The second layer weights and biases are found by simulating the first layer outputs and then solving respective linear equation system. The generalized regression network presents a special

type of RBF network. It is similar to the standard RBF network, but has a slightly different second layer.

3. Design of RBF neural network and obtained results

Radial basis networks can be designed with zero error on training vectors. In this case as many RBF neurons are created as there are input vectors in training set. Other possibility to design the network is to create a radial basis network iteratively one neuron at a time. Neurons are added to the network until the sum-squared error (SSE) falls beneath an error goal or a maximum number of neurons has been reached. At each iteration the input vector that results in lowering the network error the most, is used to create a new RBF neuron with weight vector which equals the given input vector. The iterative training process is more time-consuming but usually gives better results. The design with zero error on the training set provides network with poor ability to predict correct answers for input vectors not presented in the training set. It holds generally that with increasing number of neurons and decreasing value of *spread* the SSE goes down but the ability of network to generalize is reduced.

Because there is no general methodology available for neural network parameters design and these parameters are nearly always application dependant, complete enumeration approach was used to determine optimum values of searched parameters (possible combinations of *spread* parameter and numbers of neurons in the first layer were taken into account). According to the extent of the training set (28 input/target vectors) and number of components in each input vector (296) there are totally 8288 possible combinations. All these networks were designed and maximum error on the testing set was evaluated for each network. The best result were obtained for 21 neurons in the first layer and constant *spread* = 290. Weight vectors of the 21 RBF neurons are depicted in Figure 4.

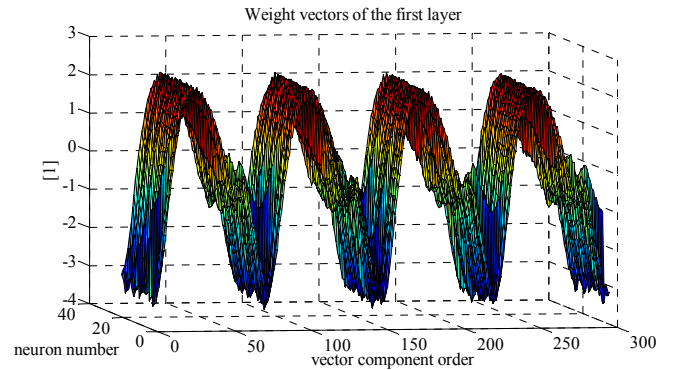


Figure 4: First layer weight vectors of designed network

Increasing number of neurons and decreasing value of *spread* constant decreases the maximum error through the training process.

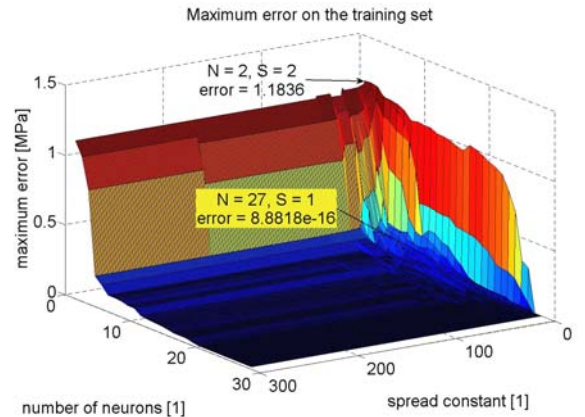


Figure 5: Maximum error of designed networks on the training set

The minimum error on the training set has the network with 27 neurons in the hidden layer and with constant $spread = 1$, see Figure 5. On the other hand increasing number of neurons and decreasing value of $spread$ causes growth of maximum error on the testing set, see Figure 6.

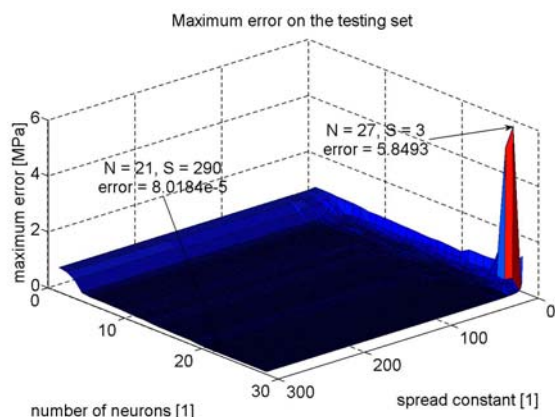


Figure 6: Maximum error of designed networks on the testing set

The maximum error on the testing set has the network with $N = 27$ and $S = 3$.

Results achieved on the testing set using the network with the best performance ($N = 21$ and $S = 290$) are in the following table.

Table 1

1		2		3	
Target	Prediction	Target	Prediction	Target	Prediction
3,15	3,149981	2,9	2,900004	2,8	2,800025
1,98	1,980080	3,15	3,149965	2,9	2,900002
2,9	2,899966	2,8	2,799972	1,98	1,980046
2,8	2,799962	1,98	1,980046	3,15	3,149929
4		5		6	
Target	Prediction	Target	Prediction	Target	Prediction
1,98	1,980088	3,15	3,149998	2,9	2,900002
2,8	2,800010	2,5	2,500047	3,15	3,149976
3,15	3,149922	2,9	2,899940	2,8	2,799964
2,9	2,899949	2,8	2,799950	2,5	2,499996

4. Conclusion

Radial basis function networks are able to approximate functional dependence between the course of the crankshaft angular velocity and the values of compression pressures inside of individual cylinders with requested accuracy. Training process of RBF neural network proved that the maximum error goes down with increasing number of neurons and decreasing value of $spread$ constant. However, the design of the network with zero error on the training set and with maximum possible number of neurons in the hidden layer, which is equal to the number of input vectors in the training set, is of little use for operational conditions. Behaviour of such a network proves lack of generalization for the input vectors which were not used during the training process (e.g. vectors from the testing set) and the answers for these inputs may be very inaccurate, see Figure 6. The way how to overcome this difficulty is to design the network iteratively and to keep track of error on the testing set during the training process and as soon as the error on the testing set proves the tendency to grow the training process is terminated. Another approach how to design the optimum RBF network for a given application is the complete enumeration method. But this procedure can be effectively used when the size of the training set and the extent of spread constant are small enough. The designed RBF network using complete enumeration method provides very precise answers to all input vectors from the testing set, see Table 1. Achieved performance is much higher than necessary accuracy requested for operational diagnostics of combustion engines. It is important to point out that all input data were measured on the engine at the temperature of 25 °C approximately. Further investigation should be done to find out the temperature impact on prediction quality of designed network.

5. References

- [1] GUPTA, M. – JIN, L. – HOMMA, N.: Static and Dynamic Neural Networks. John Wiley & Sons, Inc., Hoboken, New Jersey, 2003. ISBN 0-471-21948-7.
- [2] Hejmal, Z.: Parameters Tuning of Radial Basis Function Network for CI Engine Compression Pressures Approximation. In: 4th International Scientific Conference „Challenges in Transport and Communication“. Pardubice, University of Pardubice, Jan Perner Transport Faculty, 2006. ISBN 80-7194-880-2, 1057-1062.