This article can be cited as N. E. Karkalos, J. Kundrak and A. P. Markopoulos, Assessment of the Performance of Neural Networks Models for the Prediction of Surface Roughness After Grinding of Steels, International Journal of Artificial Intelligence, vol. 15, no. 1, pp. 55-75, 2017.

Assessment of the Performance of Neural Networks Models for the Prediction of Surface Roughness After Grinding of Steels

Nikolaos E. Karkalos[1], János Kundrák[2] and **Angelos P. Markopoulos[1]**

[1]School of Mechanical Engineering, National Technical University of Athens
Heroon Polytechniou 9, 15780, Athens, Greece
nkark@mail.ntua.gr, amark@mail.ntua.gr

[2]Institute of Manufacturing Science, University of Miskolc
Miskolc-Egyetemváros H-3515, Hungary
kundrak@uni-miskolc.hu

ABSTRACT

Attaining acceptable levels of surface roughness is one of the primary goals of grinding. In order to achieve this goal, several approaches can be employed by varying the system parameters and attempting to find the conditions which ameliorate the outcome of the process. One way that can be rather assistive to the analysis of manufacturing processes is the use of soft computing methods such as artificial neural networks. In the current work, artificial neural network models of various categories, namely multi-layer perceptron and radial basis function neural networks, with various model parameters are developed and compared in order to determine the best performing model for the prediction of surface roughness during peripheral grinding of steel components. The results indicate that radial basis function networks outperform classical multi-layer perceptrons and constitute a promising alternative for the modeling of manufacturing processes.

Keywords: surface roughness, grinding, soft computing, artificial neural networks, multi-layer perceptron, radial basis functions.

Mathematics Subject Classification: 92B20, 68T05

Computing Classification System: F.1.1, I.5.1

1. INTRODUCTION

Grinding constitutes one of the most widely employed manufacturing processes in industrial practice. This process belong to the category of abrasive manufacturing processes and despite the fact that it is mostly utilized in finishing applications, it is often also applied for bulk material removal. Grinding process is conducted using a grinding machine where the workpiece is being processed by a grinding wheel, which acts as a cutting tool. Grinding is a very popular finishing process, as it can render surfaces with very high quality, whereas it can be very effective for bulk material removal, especially in the case of hard materials.

A grinding wheel contains numerous abrasive grains on its surface, held by a bonding medium and functioning as small single-point cutting edges acting on the workpiece. It is generally believed that these microscopic cutting edges function similar to negative rake angle cutting tools. As in other machining processes, grinding

wheel is subject to wear and regeneration of grains using a diamond indenter is often performed when tool performance is degrading. Some of the most often used conventional variants of grinding are surface grinding, cylindrical grinding, centerless grinding and internal grinding, depending on the relative position of grinding tool and workpiece, and also creep-feed grinding and high speed grinding, depending on cutting speed and feed. Thus, some of the most important parameters of grinding are process parameters such as feed rate, cutting speed, depth of cut, the use of cutting fluid, grinding tool properties such as hardness or abrasive grain geometry and workpiece material-related parameters. Finally, it is worth noting that grinding process is associated to production of considerable amount of heat even at wet grinding conditions.

As for other manufacturing processes, it is considered necessary to conduct analysis of grinding in order to determine optimum set of parameters, e.g. when power or lubricant consumption minimization along with surface quality amelioration is desired. Several techniques are available for this analysis, such as purely experimental techniques, which can be applied on-line, e.g. cutting forces and temperature monitoring, or after the process is finished, such as residual stresses analysis, microstructure characterization, numerical and soft computing techniques such as the Finite Element Method (FEM) (Brinksmeier et al., 2006; Doman et al., 2009; Parente et al. 2012; Kundrák et al, 2016), Artificial Intelligence (Gajate et al., 2010), regression techniques and other statistical methods (Habrat, 2016; Markopoulos et al., 2016b). Optimization is another field of interest in manufacturing processes, especially in the case of multi-objective problems. The latter require some state-of-the-art algorithms in order to obtain the desired results (Precup et al., 2012; Ramírez-Ortegón et al., 2013; Ali et al., 2016). A reduction of both cost and time can be achieved when using numerical or soft computing methods and so their popularity is increasing. More specifically, soft computing methods tend to be more versatile and fast methods as they can be easily customized to offer higher level of accuracy with minimum computational cost, given that they are supplied with carefully collected data.

A considerable amount of scientific work concerning application of Artificial Neural Networks (ANN) in simulation of machining processes is reported in the relevant literature. Although the most often utilized methods are long established as generic soft computing tools, the appropriateness of such methods for machining processes simulation and the determination of most preferable methods is still a subject of research. Some demonstrative examples of ANN applications are described hereafter. Özel and Karpat (2005) used an ANN model to predict surface roughness and tool wear in hard turning. Ezugwu et al. (2005) employed ANN models for high speed machining of Inconel 718 alloy. Zuperl et al. (2006), Adesta et al. (2012) and Al Hazza and Adesta (2013) employed also ANN model for the analysis of end-milling. From the relevant literature on ANN, it can be deduced that studies using radial basis function (RBF) networks in manufacturing process simulations are considerably fewer than these using multi-layer perceptron (MLP) networks, according to Pontes et al. (2010a). Gong et al. (2012) used an RBF model with various spread factor values to predict cutting consumption. When compared to results produced with MLP networks, it was found that network error and fit values were considerably improved in cases conducted with RBF models. Parikh and Lam (2009) used back-propagation and RBF networks, as well as regression models in their investigation of optimal parameters during abrasive water jet machining. From their study, it was noted that RBF networks outperformed other ANN models and that this type of ANN is very promising. Furthermore, Prahdan et al. (2009) conducted comparison of several ANN models used to predict surface roughness in Electro-discharge machining and observed

significant superiority of RBF models in terms of simulation speed. Raja Dhas et al. (2013) applied RBF neural networks models for the prediction of machining quality during turning and found that RBF networks were superior to MLP networks both in terms of computational speed and accuracy. Pontes et al. (2010b) employed a DOE-based approach to determine details about RBF model training in order to test a minimum number of different networks developed to predict surface roughness during turning of AISI 52100 hardened steel. Moreover, Pontes et al. (2012) employed a DOE-based approach using Taguchi orthogonal arrays in the case of hard turning investigating the effect of several parameters of the RBF model. While most studies indicate that RBF networks are more capable than MLP networks, Dashtbayazi and Ghanbarian (2015) conducted comparison between RBF networks and MLP networks with a single hidden layer and found that MLP networks outperformed RBF ones. He et al. (2015) used RBF networks to model surface roughness in diamond turning using both certain and uncertain input parameters and concluded that the developed networks exhibited high level of accuracy.

The use of numerical modelling for the simulation of abrasive processes can be found in the relevant literature. The authors have employed in the past the FEM approach for the thermal modelling (Mamalis et al., 2003) and thermo-mechanical modelling (Kundrák et al., 2016) of surface grinding. Furthermore, the Molecular Dynamics method was used for the simulation of grinding at grain scale (Markopoulos et al., 2015). Although these methods can produce useful results, computationally, they are very intensive. The same conclusion can be drawn in the case of hybrid FEM and ANN models (Markopoulos and Kundrák, 2016). The authors have employed MLP networks for the prediction of grinding performance with quite good results (Karkalos et al., 2015); these models are very flexible and fast. A comparison between MLP and RBF networks was introduced in the modelling of milling (Markopoulos et al., 2016a), indicating good results for both methods.

In this study, ANN models with various training algorithms, architectures and types are developed to predict surface roughness in cases of grinding of various steel components. A multiple step approach is developed, comprising the creation of MLP networks, determination of their basic characteristics and in case that their performance is not adequate, several types of RBF networks are developed and their performance is compared to that of MLP models in order to find out the optimal one. The performance of developed models is evaluated in terms of prediction error values and coefficient of correlation and conclusions are extracted, regarding the efficiency and applicability of these ANN models. More specifically, this approach is employed in order to evaluate the efficiency of MLP networks to that of RBF networks and determine the applicability of RBF networks as an advantageous alternative to the well-established MLP networks in the field of abrasive processes simulation. Furthermore, a useful comparison of the performance of RBF network with different activation networks is conducted, to determine if Gaussian type radial basis function, which is employed in the vast majority of similar studies, is indeed the best performing method for manufacturing processes simulation cases.

2. ARTIFICIAL NEURAL NETWORKS

Artificial neural networks are long considered as an efficient soft computing technique, capable of providing results to a wide range of engineering problems spanning from weather prediction, robot manipulation and

speech recognition to process simulation. ANN development was primarily based on observations on the biological neural networks function, which are extremely complex systems that transfer and process large amounts of data in extremely fast speed using a network of nodes and connections.

Likewise, ANN employ a structure similar to that of a biological neural network in order to process and analyze experimental data with a view to determine the relationship between inputs and outputs without the need of using theoretic models. In this study, feed-forward MLP models with back-propagation of error, along with RBF neural networks are employed. Thus, before presenting the methodology of soft computing experiments, it is considered worthwhile to conduct a brief description of these models.

2.1. Multi-layer perceptron models

MLP neural networks consist of neurons ordered in vertical rows called layers, which are interconnected with linking elements resembling the synapses of biological neural networks. As it becomes obvious from the schematic shown in Figure 1, a feed-forward neural network consists of several types of layers, namely the input layer, the output layer and internal layers called hidden layers.

The hidden layers are the most important for the processing of data and determination of correlation between input and output data. The number of neurons in these layers, as well as the number of these layers, are not fixed in advance and vary according to each case. The architecture of a neural network is defined as the number of neurons in each layer, starting from the number of neurons in the input layer. As the number of neurons is directly connected to an increase in the computational cost of the network, it is necessary to determine the minimum number of hidden layers and neurons that can fit the input-output data pairs with sufficient accuracy.

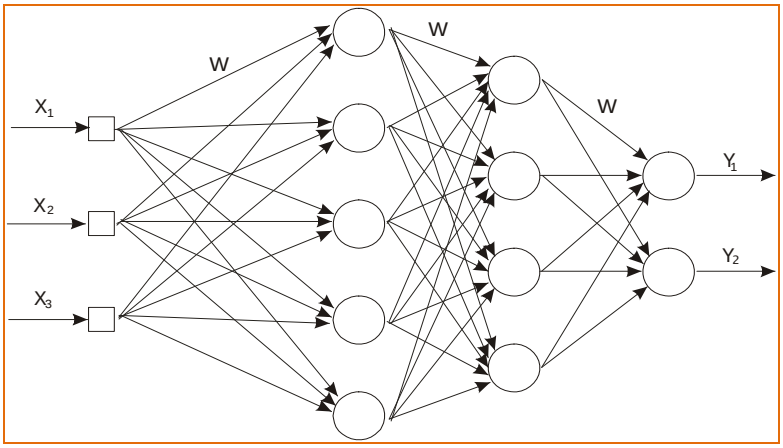


Figure 1. Architecture of an MLP network.

Each neuron in all but the input layer receives data from neurons of previous layers via the synapses. A weight coefficient is assigned to each neuron and each one of the incoming data is multiplied by the weight coefficient and then summed with all other data. Then, after a bias parameter is added to the sum, an activation function is acting on this value in order to produce the output of the neuron. Activation functions can be step functions, linear functions or sigmoid functions which give results in the range 0-1. Sigmoid functions are often preferred as they allow for a smooth transition for the range of input values employed.

The training process is essential in order for the ANN to gain predictive ability and model a specific case. The weight coefficients associated with weight coefficients are adjusted until the error is minimized. The error can be determined with various performance functions but usually the mean squared error (MSE) between predicted and actual outputs is preferred. An iterative process, during which the error is propagated from the output layer to the input layer, named error-backpropagation, is performed during training stage and at every step the weight coefficient values are updated. Two types of training can usually be employed, for different types of problems: i) supervised learning, in which each input vector is associated to an output vector and ii) unsupervised learning, where only input vectors are used in the training process. The goal of the process is to repeat the procedure until MSE becomes zero. Each time that the program passes through all pairs of training vectors an epoch is completed and training usually ends after reaching a great number of epochs.

Some of the most important types of learning algorithms for MLP neural networks are, among others, steepest descent, Newton and quasi- Newton methods, e.g. Broyden-Fletcher-Goldfarb-Shanno, Conjugate Gradient methods and Levenberg-Marquardt method. In particular, for the present work, three variants of conjugate gradient methods and Levenberg-Marquardt algorithms are employed and it is considered rather essential to discuss these four learning algorithms with more detail.

Conjugate Gradient method evolved initially as a method for solving systems of linear equations such as the systems which are employed in FEM, especially in cases with large sparse systems. Moreover, this method can be employed for optimization purposes and so it is often employed for the minimization of learning error in MLP models.

In the steepest descent methods, at the first iteration, the search starts in the direction (p_0) of the negative of the gradient (where g_0 represents the initial gradient value):

$$p_0 = -g_0 \tag{1}$$

After the first step, the optimal distance of movement along the search direction is computed by:

$$x_{k+1} = x_k \alpha_k p_k \tag{2}$$

where p_k is related to the search direction at iteration k, a_k is the step length and x_k, x_{k+1} represent the current and the next optimal location, respectively.

Then, the next direction of search is determined in such a way that it is conjugate to the previous search direction

$$p_k = -g_k + \beta_k p_{k-1} \tag{3}$$

where g_k represents gradient at iteration k and β_k is a coefficient used to ensure that new direction will be conjugate to all previous directions.

Conjugate gradient methods often employ reset methods when a certain condition exists, e.g. when the number of iterations performed is equal to the number of weights and biases of the system. Powell-Beale restarts method is one of the reset methods and it is activated when there is almost no orthogonality between the gradient at the current iteration and the gradient at the previous iteration, as defined by the following inequality:

$$\left| g_{k-1}^T g_k \right| \geq 0.2 \left\| g_k \right\|^2 \tag{4}$$

A reset occurs when this inequality is satisfied, resulting in a potential improvement of training efficiency. Compared to Polak-Ribiére updates variant, Powell-Beale algorithm has somewhat larger computational cost.
In Fletcher-Reeves updates variant of conjugate gradient algorithm, the coefficient β_k is derived from the following formula:

$$\beta_k = \frac{g_k^T g_k}{g_{k-1}^T g_{k-1}} \tag{5}$$

In Polak-Ribiére updates variant of conjugate gradient algorithm, the coefficient β_k is derived from the following formula:

$$\beta_k = \frac{\Delta g_{k-1}^T g_k}{g_{k-1}^T g_{k-1}} \tag{6}$$

where Δg_{k-1} term refers to the change in the gradient of the previous iteration (g_k - g_{k-1}) .
In comparison to Fletcher-Reeves algorithm, Polak-Ribiére algorithm is slightly more demanding in terms of computational cost.

The Levenberg-Marquardt (LM) method is used generally for the solution of non-linear least squares problems. This method was intended to be employed as an “intermediate method” between Gauss-Newton method and gradient descend algorithm, by unifying the strength of each method while trying to confront the drawbacks of each of them (Kermani et al., 2005). More specifically, the gradient descent algorithm is shown to be able to perform more independently regarding the choice of initial values but its convergence is rather slow. On the contrary, the Gauss-Newton algorithm has significantly faster convergence but depends on initial values to a greater extent than gradient-descent methods.

In general, the LM method is used to approximate the Hessian matrix, thus retaining the second-order convergence speed but without having to compute the exact Hessian matrix (H). In the case of performance function with the form of sum of squares, the approximation is conducted as follows:

$$H = J^T J \tag{7}$$

and then, the gradient is derived from the following formula:

$$g = J^T e \tag{8}$$

where J represents a Jacobian matrix, containing the first derivatives of errors with respect to weights (denoted often as w) and biases and e represents a vector of network errors, calculated as the difference between the vector of desired outputs and the vector of actual network outputs. Jacobian matrix is defined as $J_{ij} = de_i / dw_j$ in component-wise form.

As it was already mentioned, LM method lies between Gradient Descent and Gauss-Newton methods. Thus it is anticipated that the Hessian matrix approximation conducted using LM algorithm uses terms related to both of these methods, as it can be seen in the following update formula:

$$x_{k+1} = x_k - \left[J^T J + \mu I\right]^{-1} J^T e \tag{9}$$

where μ is a coefficient which will be discussed afterwards and I is the identity or unit matrix.

From the above formula, it can be seen that when μ takes large values, this expression is similar to gradient descent whereas for μ values close to zero, this expression is similar to Gauss-Newton. In order to profit from the advantages of both methods, after steps with decrease in performance function, which follow an appropriate initial guess and efficient use of gradient descent method, the value of μ is decreased to achieve faster convergence, shifting gradually to Gauss-Newton method and it is again increased when performance function values are increasing. Although this method has advantages in many cases, it is more complex than several other training algorithms.

2.2. Radial basis function models

RBF neural networks are considered as a special category of neural networks, consisting of three layers. This type of neural networks was first introduced by Broomhead and Lowe (1988). One particularity of RBF neural networks is that a single hidden layer is employed, as it can be seen in Figure 2, and so their architecture is simpler and generally lower computational cost is observed. However, their predictive ability is at least comparable to that of other neural network types.

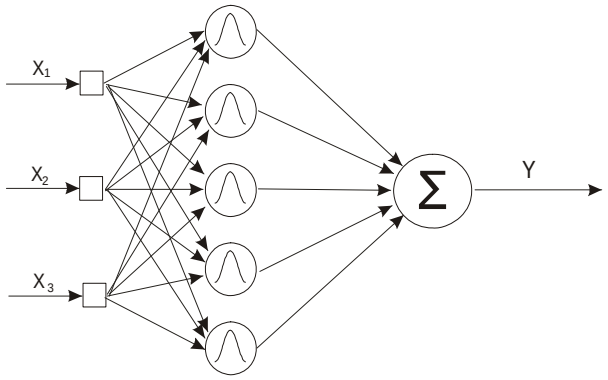


Figure 2. RBF neural network model.

As RBF neural networks are also a feed-forward network, the fundamental features of MLP networks apply to RBF networks and the only particularity is the application of a radial basis function to the input of neurons after summation process is performed, which is a special type of activation function. More specifically, radial basis functions, such as the Gaussian function, act as the activation function of the network and the output of the network is calculated as follows:

$$y(x) = \sum_{i=1}^{N} w_i \varphi\left(\left\| x - x_i \right\|\right) \tag{10}$$

where w_i are the weights of the network, φ the radial basis function, N the number of neurons in the hidden layer and x_i the center vector for neuron i.

The fact that functions dependent on the distance from a center vector are only used for this type of networks is the reason of the name radial basis neural networks. The norm in the last equation is supposed to be Euclidean but other formulations can be also employed. The weights of this network can be adjusted with similar methods as in MLP neural networks. In fact, the training procedure for RBF networks usually consists of two steps. At first, the center vectors for the RBF functions are determined and then the given input-output data pairs are

fitted to the network by adjusting network weight coefficients w_i. During this first step, the determination of center vectors (denoted as x_i in equation 10) is conducted by a k-means clustering algorithm whereas during the second step, backpropagation is often performed to determine the other parameters, by using a pseudoinverse solution method, a gradient descent method or minimum squares method (Markopoulos, 2016a), as in the current study. Furthermore, the majority of radial basis functions contain an additional factor, which requires to be also optimized, the so-called spread factor, ε. Spread parameter choice is an important element for the performance of these networks and it is often included in numerical investigations before the final neural network model is created.

Apart from Gaussian type radial basis function, several other - mostly non-linear - radial basis functions exist such as: multiquadric, inverse quadratic, inverse multiquadric and thin plate spline. In the current work, three of the aforementioned radial basis function will be employed, namely Gaussian, multiquadric and inverse quadratic. The three radial basis functions are:
Gaussian:

$$\varphi(r) = e^{-(\varepsilon r)^2} \tag{11}$$

Multiquadric:

$$\varphi(r) = \sqrt{1 + (\varepsilon r)^2} \tag{12}$$

Inverse quadratic:

$$\varphi(r) = \frac{1}{\sqrt{1 + (\varepsilon r)^2}} \tag{13}$$

3. METHODOLOGY

In the current study, MLP and RBF neural network models are created for the prediction of surface roughness during grinding. All ANN models have three input factors, namely workpiece material type, grinding wheel type and depth of cut as well as single output factor, surface roughness. Surface roughness can be considered as a key factor in machining. This quantity is often employed to evaluate and determine the quality of products for both engineering and general purposes. More specifically, surface roughness influences several attributes of a component, such as fatigue behavior, wear, corrosion, lubrication and surface friction.

Surface roughness corresponds to deviations from the nominal surface shape of the third up to the sixth order. More specifically, first and second order deviations refer to form and waviness, respectively. Furthermore, third and fourth order deviations refer to periodic grooves, cracks and dilapidations, which are related to the shape and condition of the cutting edges, chip formation and process kinematics. Fifth and sixth order deviations refer to workpiece material structure, which is connected to physical and chemical mechanisms, acting on a grain and at lattice scale. In general, surface roughness can be described as the inherent irregularities of workpiece surface after being processed. The easiest and more usual way to describe surface roughness is the average surface roughness, which is often denoted as Ra.

Surface roughness is considered to be affected by several controlled factors, such as feed rate, grinding wheel speed, depth of cut, as well as by some non-controlled factors, such as non-homogeneity of workpiece and tool material, tool wear, machine motion errors, formation of chips and other unpredictable random disturbances. Average surface roughness Ra is defined as "the arithmetic value of the deviation of profile from centerline along a sampling length". If l represents the sampling length and y represents the ordinate of the profile curve, then it is calculated as:

$$Ra = \frac{1}{l}\int_0^l |y(x)|dx \tag{14}$$

Surface roughness values are obtained for 72 cases of grinding, from a previous study (Markopoulos, 2011) and are used as the output variable vector for the developed of the proposed ANN models. The range of input variables for these cases is presented in Table 1. For two of the input variables, namely workpiece material type and grinding wheel type, their values are coded as 1-3 and 1-6, respectively for reasons of simplicity. That approach is shown not to affect ANN results (Markopoulos, 2011), as a normalization process takes place for each input and output variable before the ANN models are trained and their values lie in the range 0-1. All the other parameters of grinding process remained unchanged for the experiments. The values of these parameters are: workpiece feed v_w of 8 m/min, cutting speed v_c of 28 m/s and wet machining conditions. The Al_2O_3 grinding wheels have a diameter d_s of 250mm and width b_s of 20 mm, while they have different bonding material.

Table 1: Description of input parameters employed in ANN models.

Workpiece material	100Cr6 C45 X210Cr12
Grinding wheel type	6 types with variable bonding material
Depth of cut	0.01 mm 0.02 mm 0.03 mm 0.05 mm

Simulations are conducted with 4 different training algorithms, as well as several network architectures. A short description of the range of the models is given in Table 2. Several training algorithms pertinent to conjugate gradient (CG) class of methods, namely CG with Powell-Beale restarts (denoted hereafter as CG(B)), CG with Fletcher-Reeves updates (denoted hereafter as CG (F)), CG with Polak-Ribiére updates (denoted hereafter as CG (P)) are employed along with the Levenberg-Marquardt method. All these training algorithms operate in batch model. Hyperbolic tangent function is employed as activation function and MSE minimization is conducted. MSE is defined as follows: if $\hat{Y}$ represents the output values of the network and Y represents the actual output data fed to the network, the MSE of prediction can be formulated as:

$$MSE = \frac{1}{n}\sum_{i=1}^{n}\left(\hat{Y}_i - Y_i\right)^2 \tag{15}$$

Table 2: MLP models parameters

Training algorithms	LM CG(B) CG(F) CG(P)
Hidden layers	1 and 2
Number of neurons	4 to 10
Activation function	Hyperbolic tangent
Performance evaluation	Mean Squared Error

In the case of RBF simulations, three different activation functions, a variable number of neurons in the hidden layers and several spread factor values are employed, as it is presented in Table 3.

Table 3: RBF models parameters

Maximum number of neurons	5-60 60 60
Spread factor	0.25-2.00
Activation function	Gausian Multiquadric Inverse quadratic
Performance evaluation	Mean Squared Error

In the current study a multi-step approach is used for the determination of the optimal neural network, as it is depicted graphically in Figure 3. The initial stages involve the determination of the optimal MLP neural networks. If the results of MLP networks are not satisfactory, RBF network models will be developed and the optimal models of each type will be compared to determine the overall best performing model. As it can be seen in Fig.3, the results of ANN and RBF networks will be assessed in terms of correlation coefficient R, apart from MSE. Correlation coefficient R is defined as follows:

$$R = \frac{\sum_{i=1}^{n}(x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum_{i=1}^{n}(x_i - \bar{x})^2}\sqrt{\sum_{i=1}^{n}(y_i - \bar{y})^2}} \tag{16}$$

where $\bar{x}$ and $\bar{y}$ represent the average of x and y vectors, respectively.

Finally, as it is considered necessary to eliminate the effect that the initialization of weight coefficient may have on the results, each model is simulated for five times with different initialization values and the results of the best cases are retained.

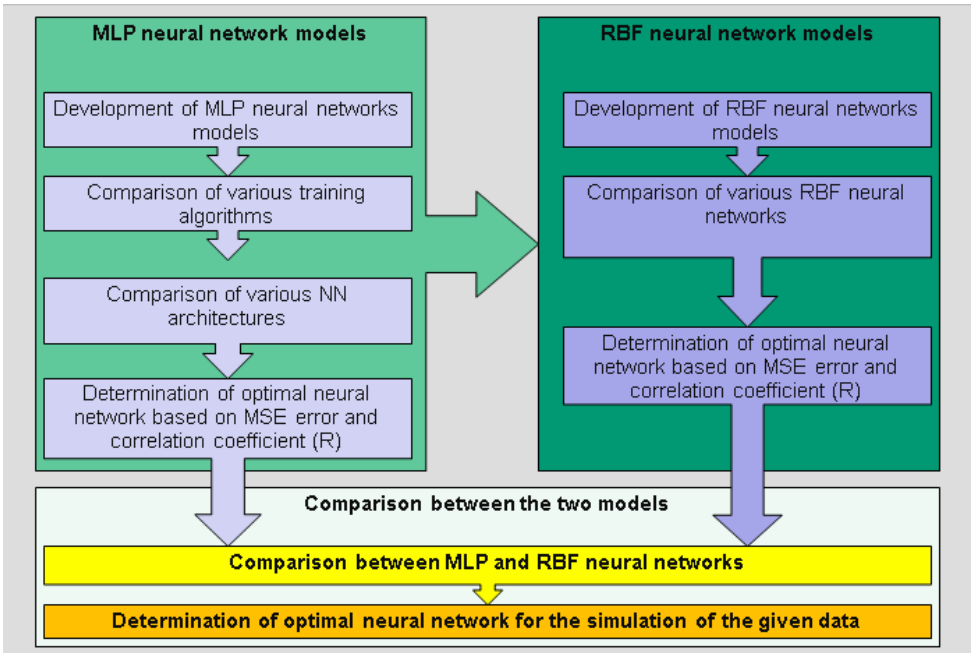


Figure 3. Schematic of the general approach towards the determination of optimal model

4. RESULTS AND DISCUSSION

4.1. Choice of training algorithm for MLP models

The first stage of the methodology proposed in the current work, consists of the determination of the optimal training algorithm of the MLP models. This step is rather crucial as the choice of an appropriate training algorithm can lead to a reduction of computational cost and the development of more accurate models. For the better choice of the optimum training algorithm, several single and two-layer algorithms are chosen, and the characteristics of all cases are presented in Table 4.

Table 4: Characteristics of studied cases

Case	1st hidden layer neurons	2nd hidden layer neurons
1	4	-
2	9	-
3	4	7
4	6	10
5	9	5

The performance of ANN models will be evaluated in terms of MSE and correlation coefficient R with specific emphasis put on the test stage of training process. At this step, cases using 5 different network architectures and the four different training algorithms, as presented in Table 4, are tested.

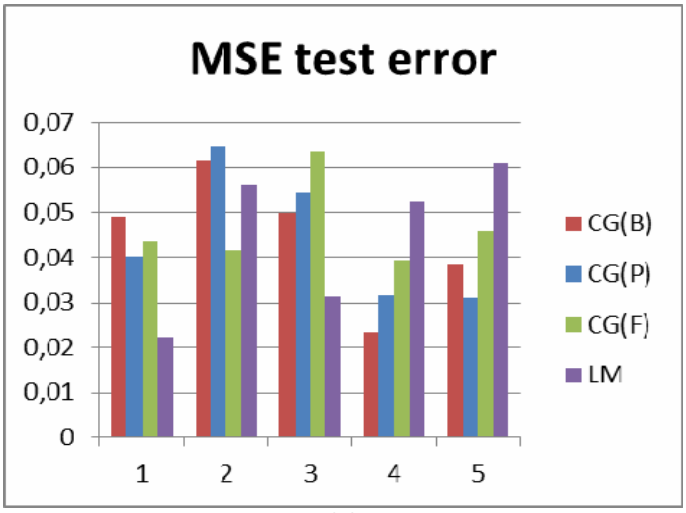
The results are presented in Table 5 and from the findings it is clearly noted that in the majority of the cases the performance of the developed model is considered relatively low, especially in terms of correlation coefficient

R, as these values are mainly below 0.5 (a perfect fit is indicated by values close to 1). MSE values in all simulations lies within an acceptable range, namely 0.02 to 0.06 during testing.

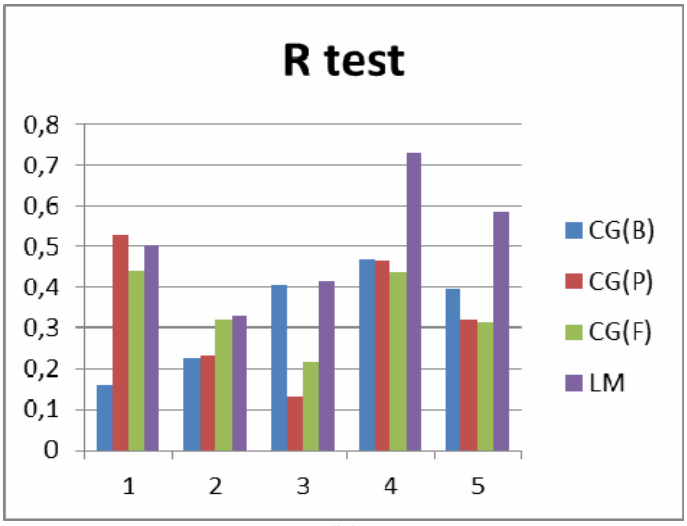
Table 5: Results concerning the comparison of performance of training algorithms

Case	Training algorithm	MSE train/test	R train/test	Time (s)
1.1	CG(B)	0.0324/0.0492	0.47425/0.16093	0.808
1.2	CG(B)	0.0227/0.0617	0.67123/0.2273	0.833
1.3	CG(B)	0.0369/0.0499	0.18486/0.40662	0.811
1.4	CG(B)	0.0353/0.0236	0.46038/0.47035	0.858
1.5	CG(B)	0.0388/0.0382	0.14644/0.39677	0.843
2.1	CG(P)	0.0391/0.0401	0.26506/0.52955	0.764
2.2	CG(P)	0.0321/0.0644	0.45527/0.23327	0.734
2.3	CG(P)	0.0351/0.0545	0.36685/0.12937	0.749
2.4	CG(P)	0.0442/0.0315	0.11048/0.46647	0.765
2.5	CG(P)	0.0381/0.0308	0.44366/0.31982	0.766
3.1	CG(F)	0.0338/0.0437	0.42805/0.43824	1.192
3.2	CG(F)	0.0336/0.0415	0.47679/0.31941	0.798
3.3	CG(F)	0.0279/0.0636	0.4875/0.21812	1.078
3.4	CG(F)	0.0151/0.0391	0.777/0.43787	1.243
3.5	CG(F)	0.0373/0.0458	0.38307/0.31177	1.554
4.1	LM	0.0329/0.0222	0.57221/0.50326	0.805
4.2	LM	0.0344/0.0561	0.38381/0.33037	0.889
4.3	LM	0.0347/0.0312	0.55297/0.41561	1.169
4.4	LM	0.0021/0.0523	0.97239/0.72789	0.892
4.5	LM	0.0377/0.0610	0.50676/0.58519	0.872

Nevertheless, the results produced from different algorithms are different both in terms of correlation coefficient and MSE. As it can be seen from Table 5 and Figure 4, it is obvious that results obtained with MLP networks trained with LM algorithm outperform the models trained with other algorithms at almost every case. Thus, although cases run with LM are not the fastest, they are performing better in terms of predictive ability. Based on the above, LM is determined as the best training algorithm and it is employed for the cases that will be used for the investigation of optimal network architecture.



(a)



(b)

Figure 4. Comparison of (a) MSE test error and (b) correlation coefficient values, for each algorithm

4.2. Determination of optimum network architecture

Although empirical relationships and suggestions from literature can indicate preferable parameters for ANN models it is essential to investigate these values separately for each problem studied as a general methodology is not easily applicable to every case.

A new set of MLP models were developed, after the LM algorithm was shown to be the most preferable training algorithm, with a view to determine the optimum architecture of the MLP model, which means the determination of optimum number of hidden layers and hidden neurons, within the range that was chosen. From the findings presented in Table 2, it can be seen that a variety of combinations of number of neurons and number of hidden layers are tested. The results for networks with one hidden layer are presented in Table 6 whereas the results for networks with two hidden layers are presented in Table 7.

Table 6: Results of cases conducted with a single hidden layer

Number of hidden layer neurons	MSE train/test	R train/test
4	0.0405/0.0496	0.26/0.471
5	0.0251/0.0764	0.62/0.379
6	0.0143/0.0571	0.812/0.546
7	0.0314/0.0596	0.528/0.238
8	0.0230/0.117	0.500/0.521
9	0.0219/0.0508	0.348/0.408
10	0.0255/0.0471	0.653/0.207

Table 7: Results of cases conducted with two hidden layers

Number of neurons in $1^{st}/2^{nd}$ hidden layer	MSE train/test	R train/test
4/4	0.0394/0.0582	0.23/0.416
4/5	0.0405/0.0570	0.369/0.201
5/4	0.0373/0.0389	0.411/0.452
4/6	0.0459/0.0251	0.392/0.306
5/5	0.0156/0.0678	0.713/0.627
6/4	0.0291/0.0670	0.481/0.437
5/6	0.0329/0.0761	0.211/0.475
6/5	0.0210/0.0471	0.489/0.612
5/7	0.0270/0.0378	0.543/0.162
6/6	0.0119/0.134	0.708/0.251
7/5	0.0380/0.110	0.296/0.379
6/7	0.0169/0.0288	0.143/0.367
7/6	0.0192/0.0739	0.488/0.262
6/8	0.0150/0.0482	0.403/0.287
7/7	0.0300/0.0378	0.483/0.565
8/6	0.0354/0.0530	0.126/0.643
7/8	0.0405/0.0563	0.150/0.447
8/7	0.000715/0.0994	0.963/0.353
7/9	0.0223/0.0354	0.629/0.537
8/8	0.0276/0.0752	0.530/0.385
9/7	0.00625/0.0244	0.754/0.764
8/9	0.0274/0.118	0.610/0.582
9/8	0.00280/0.0584	0.623/0.340
8/10	0.0175/0.0684	0.597/0.627
9/9	0.0235/0.0414	0.547/0.396
10/8	0.0213/0.0619	0.312/0.105
9/10	0.0201/0.0422	0.580/0.684
10/9	0.000379/0.117	0.856/0.211
10/10	0.0102/0.0558	0.0844/0.413

Initially, it is observed that the best performing single layer MLP is the network with 6 neurons in the hidden layer whereas the best performing two-layer network is the network with 9 neurons in the first and 7 neurons in the second hidden layer.

Between these two networks, the latter is significantly better performing both in terms of MSE and correlation coefficient and thus the architecture of this network is considered the optimum. Furthermore, the difference in training time between the two networks is not significantly large and this cannot affect the choice of optimum network.

It becomes obvious from the findings in Table 7, that for all MLP models rather moderate values of correlation coefficient are obtained, with R test values being under 0.76. Given that observation, as it is intended to obtain a better fit for the experimental results, the next step of the proposed procedure will take place and RBF networks will be developed afterwards in order to investigate their potential.

4.3. RBF neural networks models

After the evaluation of MLP model showed that these models cannot sufficiently model the experimental data, various RBF models with Gaussian activation function are developed in order to determine the capabilities of RBF networks. As it can be seen in Table 3, the maximum number of neurons and spread parameter values are varied within specific ranges. From all the results that were obtained, i.e. 45 cases in total, some indicative results are presented in Table 8.

Table 8: Results of cases conducted with RBF models (Gaussian activation function)

Neurons/ spread	MSE train	MSE test	R train	R test
50/0.250	0.0033	0.0131	0.9602	0.8802
50/0.500	0.0036	0.0142	0.9588	0.8768
60/0.250	0.0021	0.0084	0.9747	0.8942
60/0.500	0.0033	0.0068	0.9735	0.8932
60/1.00	0.0020	0.0081	0.9767	0.8960
60/2.00	0.0047	0.0187	0.9426	0.8648

As it can be observed from Table 8, the best performing RBF model is shown to be a network with 60 neurons in the hidden layer and spread parameter value equal to 1.00. From these findings it can be also noted that, in cases with over 40 neurons, better results were obtained from networks with spread parameter values lower or near 1.00 from networks with larger spread parameters values.

After the simulations with Gaussian activation function were conducted, it was considered important to compare the performance of RBF models with Gaussian activation function with the results obtained from simulations with two other activation functions, namely multiquadric and inverse quadratic functions. In these simulations, the same maximum number of neurons and range of spread parameter, which were used for Gaussian activation function cases is employed and the results are presented in Tables 9 and 10.

As it can be seen from the results presented in Table 9, the best results with multiquadric activation function are obtained with spread parameter equal to 2.000, as this model exhibits acceptable MSE, both during training and test procedures, as well as correlation coefficient during training, while it exhibits the higher correlation coefficient during test procedure. Consequently, it is deduced from these results that the model with 60 neurons and $\varepsilon = 2$ can reproduce the experimental results and generalize relatively better than the other networks. However the value of $R = 0.734$ for test procedure is not high enough to be considered satisfactory. Furthermore, the large difference in R values between training and test procedure in the most cases indicates that there are severe overfitting problems with this type of RBF networks, which are compatively larger than in the case of networks with the other two activation functions. Finally, it is also observed that the best results concerning R test are obtained for a case where R train as well as MSE train and MSE test have their lower values, which occurs also in the case of Gaussian and inverse quadratic activation functions, indicating that the improvement in generalization ability of the model coincides with a slight deterioration of predictive ability of the network.

Table 9: Results of cases conducted with RBF models (Multiquadric activation function)

spread	MSE train	MSE test	R train	R test
0.250	8.65×10^{-5}	0.1012	0.986	0.285
0.500	4.16×10^{-3}	0.1808	0.975	0.378
1.000	1.84×10^{-3}	0.1839	0.977	0.553
2.000	3.20×10^{-2}	0.5856	0.935	0.734

Table 10: Results of cases conducted with RBF models (Inverse quadratic activation function)

spread	MSE train	MSE test	R train	R test
0.250	3.05×10^{-3}	1.0100	0.947	0.886
0.500	3.98×10^{-3}	0.4474	0.989	0.856
1.000	3.88×10^{-4}	0.1974	0.988	0.424
2.000	2.68×10^{-4}	0.1313	0.985	0.349

In the case of inverse quadratic activation function, it can be observed that superior results regarding R test are obtained for spread parameter value equal to 0.25. Comparing this result with the result of multiquadric and Gaussian activation functions, as presented also in Table 11, it is observed that results using the inverse quadratic activation function outperform the results obtain with multiquadric activation function and are slightly inferior to the results obtained by Gaussian activation function as both R train and R test values are very close between the two models but MSE values are considerably lower in the case of Gaussian activation function. Consequently, the case with 60 neurons, spread parameter 1.00 and Gaussian activation function is considered the best performing RBF neural network.

Table 11: Comparison of the best cases for each activation function

network	Spread parameter	MSE train/test	R train/test
RBF-Gaussian	1.000	0.0020/0.0081	0.9767/0.8960
RBF- multiquadric	2.000	0.0320/0.5856	0.935/0.734
RBF-inverse quadratic	0.250	0.00305/1.0100	0.947/0.886

4.4. Comparison of MLP and RBF models

The final step of the approach employed in the current work consists of the comparison of best performing MLP and RBF models, with a view to figure out which of these models can be chosen as the overall best performing. In Table 12, the parameters and performance indexes of two models are presented.

Table 12: Comparison between the best performing MLP and RBF models

	Optimum MLP	Optimum RBF
MSE train error	0.00623	0.002
MSE test error	0.0244	0.0081
R train	0.754	0.977
R test	0.764	0.896

As it can be seen from Table 12, the best RBF model exhibits significantly better results both in terms of MSE and correlation coefficient values and thus, it is evaluated as the optimal network. For the sake of comparison in terms of computational time it was observed that the optimum RBF network was trained in 2.44 s, whereas the training procedure of the optimum MLP network was completed in 1.73 s. Nevertheless, as the RBF network is found to be able to produce results closer to experimental ones than the MLP network, this difference in training speed is not considered more important than the difference in the other indices, namely the superiority of RBF best case concerning MSE train and test error, as well as correlation coefficient values, so the RBF is selected as the more preferable model type.

To the authors' knowledge, there are not many results reported on the comparison of MLP and RBF networks for grinding. This study indicated that RBF networks can outperform MLP ones, a conclusion consistent with findings of other researchers. In the relevant literature (Finan et al., 1996; Xie et al., 2011), it is generally stated that RBF networks are more robust, especially when input data set contains noise, and their results indicate higher levels of accuracy and lower errors in several problems, such as an automatic fault location system, developed by Zayanderhroodi et al. (2010) or a classification problem studied by Mak et al. (1993). Especially, they are preferable for small and medium size training sets, e.g. up to 500 samples (Sug, 2009). For larger input sets, their performance is not always better than that of MLP networks, e.g. for a classification problem (Xie et al., 2011) or for a leak detection problem (Santos et al., 2013), where RBF networks exhibited higher absolute errors; however it is also advisable that attention should be paid in order to determine the optimum model parameters in each case, as only then the RBF network can be fully exploited and outperform MLP both with noisy and normal training sets (Finan et al., 1996). Furthermore, sometimes storage requirements are reported to be larger than those of MLP networks (Mak et al., 1993). Finally, although research is still ongoing in this field of neural networks, the promising results observed in the current study and those reported in the aforementioned literature indicate that it is worth studying this type of neural networks in order to further enhance its capabilities.

5. CONCLUSIONS

In the current work, ANN and RBFNN models were developed for the prediction of surface roughness during grinding of steel components. Results obtained from several ANN and RBFNN models with various characteristics were compared and finally, several crucial conclusions were deduced:

- Four different training algorithms and several network architectures with one and two hidden layers and up to 10 neurons in each layer were tested, in order to determine optimum parameters for the ANN models. It was found that the Levenberg-Marquardt algorithm was the best performing training algorithm and 3-9-7-1 was selected as the optimum network architecture.
- In the case of MLP ANN models, it is found that they can produce rather not very high levels of accuracy, as R test values do not exceed 76% and MSE test values do not exceed 2.44x10-2 for the best performing model.
- In the case of RBF ANN models, comparison between models with three types of activation functions, namely Gaussian, multiquadric and inverse quadratic, various numbers of hidden neurons and spread

parameter values was performed. It was finally observed that results obtained from models using the Gaussian activation function are superior to results obtained with the other activation functions and the optimum model parameters were 60 hidden neurons and spread parameter 1.0.

- Compared to the results obtain with ANN models, results obtained with RBF models are relatively better, since the optimum RBF model exhibited high levels of predictive and generalization ability. These results indicated that RBF networks can be considered as an advantageous soft computing method for manufacturing process simulation. However it is advised that further investigation of the applications of this method in manufacturing should be conducted in order to extend the capabilities of this method.

6. REFERENCES

Adesta, E.Y.T., Al Hazza, M.H.F., Suprianto, M.Y., Riza, M., 2012, Prediction of cutting temperatures by using back propagation neural network modelling when cutting hardened H3 steel in CNC end milling, *Advanced Materials Research* **576**, 91-94.

Al Hazza, M.H.F, Adesta, E.Y.T., 2013, Investigation of the effect of cutting speed on the surface roughness parameters in CNC end milling using artificial neural network, *IOP Conference Series: Materials Science and Engineering* **53**, 012089.

Ali, M.Z., Awad, N.H., Duwairi, R.M., 2016, Multi-objective differential evolution algorithm with a new improved mutation strategy, *International Journal of Artificial Intelligence* **14 (2)**, 23-41.

Broomhead, D.S., Lowe, D., 1988, Multivariable functional interpolation and adaptive networks, *Complex Systems* **2**, 321-355.

Brinksmeier, E., Aurich, J.C., Govekar, E., Heinzel, C., Hoffmeister, H.-W., Klocke, F., Peters, J., Rentsch, R., Stephenson, D.J., Uhlmann, E., Weinert, K., Wittmann, M., 2006, Advances in modeling and simulation of grinding processes, *CIRP Annals - Manufacturing Technology* **55(2)**, 667–696.

Dashtbayazi, M.R., Ghanbarian, M., 2015, Comparison of artificial neural networks in modelling of polymer matrix composite turning, *Amir Kabir Journal of Science & Research (Mechanical Engineering) (ASJR-ME)* **47(2)**, 33-36.

Doman, D.A., Warkentin, A., Bauer, R., 2009, Finite element modeling approaches in grinding, *International Journal of Machine Tools and Manufacture* **49(2)**, 109–116.

Ezugwu, E.O., Fadare, D.A., Bonney, J., Da Silva, R.B., Sales, W.F., 2005, Modelling the correlation between cutting and process parameters in high speed machining of Inconel 718 alloy using an artificial neural network, *International Journal of Machine Tools and Manufacture* **45(12-13)**, 1375-1385.

Finan, R.A., Sapeluk, A.T., Damper, R.I., 1996, Comparison of multilayer and radial basis function neural networks for text-dependent speaker recognition, *1996 IEEE International Conference on Neural Networks* **4**, 1992-1997.

Gajate, A., Haber, R.E., Vega, P.I., Alique, J.R., 2010, A transductive neuro-fuzzy controller: Application to a drilling process, *IEEE Transactions on Neural Networks* **21(7)**, 1158-1167.

Gong, M.Q., Sun, M.W., Huang, M., Xiang, S.W., 2012, Prediction of cutting consumption based on optimization-making RBF artificial neural network, *Advanced Materials Research* **430-432**, 659-663.

Habrat, W.F., 2016, Effect of Bond Type and Process Parameters on Grinding Force Components in Grinding of Cemented Carbide, *Procedia Engineering* **149**, 122-129.

He, C.L., Zong, W.J., Cao, Z.M., Sun, T., 2015, Theoretical and empirical coupled modeling on the surface roughness in diamond turning, *Materials & Design* **82**, 216-222.

Karkalos, N.E., Markopoulos, A.P., Dossis, M.F., 2015, Application of statistical and soft computing techniques for the prediction of grinding performance, *Journal of Robotics and Mechanical Engineering Research* **1(2)**, 1-11.

Kermani, B.G., Schiffmann, S.S., Nagle, H.T., 2005, Performance of the Levenberg-Marquardt neural network training method in electronic nose applications, *Sensors and Actuators B: Chemical* **110(1)**, 13-22.

Kundrák, J., Fedorovich, V., Markopoulos, A.P., Pyzhov, I., Kryukova, N., 2016, Diamond grinding wheels production study with the use of the finite element method, *Journal of Advanced Research* **7(6)**, 1057-1064.

Mak, M.W., Allen, W.G., Sexton, G.G., 1993, Comparing Multi-layer perceptrons and radial basis functions networks in speaker recognition, *Journal of Microcomputer Applications* **16**, 147-159.

Mamalis, A.G., Kundrák, J., Manolakos, D.E., Gyáni, K., Markopoulos, A., Horvath, M., 2003, Effect of the workpiece material on the heat affected zones during grinding: a numerical simulation, *International Journal of Advanced Manufacturing Technology* **22(11-12)**, 761-767.

Markopoulos, A.P., 2011, Simulation of grinding by means of the finite element method and artificial neural networks, J.P. Davim (ed), *Computational Methods for Optimizing Manufacturing Technology*, IGI Global, 193-218.

Markopoulos, A.P., Georgiopoulos, S., Manolakos, D.E., 2016a, On the use of back propagation and radial basis function neural networks in surface roughness prediction, *Journal of Industrial Engineering International* **12(3)**, 389-400.

Markopoulos, A.P., Habrat, W., Galanis, N.I., Karkalos, N.E., 2016b, Modeling and optimization of machining with the use of statistical methods and soft computing, J.P. Davim (ed), *Design of Experiments in production engineering*, Springer, 39-88.

Markopoulos, A.P., Kundrak, J., 2016, FEM/AI models for the simulation of precision grinding, *Manufacturing Technology* **16(2)**, 384-390.

Markopoulos, A.P., Savvopoulos, I.K., Karkalos, N.E., Manolakos, D.E., 2015, Molecular dynamics modeling of a single diamond abrasive grain in grinding, *Frontiers of Mechanical Engineering* **10(2)**, 168-175.

Özel, T., Karpat, Y., 2005, Predictive modeling of surface roughness and tool wear in hard turning using regression and neural networks, *International Journal of Machine Tools and Manufacture* **45(4-5)**, 467-479.

Parente, M.P.L., Natal Jorge, R.M., Aguiar Vieira, A., Monteiro Baptista, A., 2012, Experimental and numerical study of the temperature field during creep feed grinding, *The International Journal of Advanced Manufacturing Technology* **61(1)**, 127–134.

Parikh, P.J., Lam, S.S., 2009, Parameter estimation for abrasive water jet machining process using neural networks, *International Journal of Advanced Manufacturing Technology* **40(5)**, 497-502.

Pontes, F.J., da Paiva, A.P., Balestrassi, P.P., Ferreira, J.R., da Silva, M.B., 2012, Optimization of radial basis function neural network employed for prediction of surface roughness in hard turning process using Taguchi's orthogonal arrays, *Expert Systems with Applications* **39(9)**, 7776-7787.

Pontes, F.J., Ferreira, J.R., Silva M.B., Paiva, A.P., Balestrassi, P.P., 2010a, Artificial neural networks for machining processes surface roughness modeling, *International Journal of Advanced Manufacturing Technology* **49(9)**, 879-902.

Pontes, F.J., Silva, M.B., Ferreira, J.R., de Paiva, A.P., Balestrassi, P.P., Schönhorst, G.B., 2010b, A DOE based approach for the design of RBF artificial neural networks applied to prediction of surface roughness in

AISI 52100 hardened steel turning, *Journal of the Brazilian Society of Mechanical Sciences and Engineering* **32(5)**, 503-510.

Pradhan, M.K., Das, R., Biswas, C.K., 2009, Comparisons of neural network models on surface roughness in electrical discharge machining, *Proceedings of the Institution of Mechanical Engineers, Part B: Journal of Engineering Manufacture* **23**, 801-808.

Precup, R.E., Dragos, C.A., Preitl, S., Radac, M.B., Petriu, E.M., 2012, Novel Tensor Product Models for Automatic Transmission System Control, *IEEE Systems Journal* **6(3)**, 488-498.

Raja Dhas, E., Stalin, R.S., Rajeesh, J., 2013, RBF neural network model for machining quality prediction in CNC turning process, *International Journal of Modelling Identification and Control* **20(2)**, 174-180.

Ramirez- Ortegón, M.A., Märgner, V., Cuevas, E., Rojas, R., 2013, An optimization for binarization methods by removing binary artifacts, *Pattern Recognition Letter* **34(11)**, 1299-1306.

Santos, R.B., Rupp, M., Bonzi, S.J., Fileti, A.M.F., 2013, Comparison Between Multilayer Feedforward Neural networks and a radial basis function network to detect and locate leaks in pipelines transporting gas, *Chemical Engineering Transactions* **32**, 1375-1380.

Sug, H., 2009, Performance Comparison of RBF networks and MLPs for classification, *Proceedings of 9th International Conference on Applied Informatics and Communications*, 450-454.

Xie, T., Yu, H., Wilamowski, B., 2011, Comparison between traditional neural networks and radial basis function networks, *2011 IEEE International Symposium on Industrial Electronics*, 1194-1199.

Zayandehroodi, H., Mohamed, A., Shareef, H., Mohammadjafari, M., 2010, Performance comparison of MLP and RBF neural networks for fault location in distribution networks with DGs, *Proceedings of 2010 IEEE International Conference on Power and Energy*, 341-345.

Zuperl, U., Cus, F., Mursec, B., Ploj, T., 2006, A generalized neural network model of ball end milling force system, *Journal of Materials Processing Technology* **175(1-3)**, 90-108.