Applied Soft Computing 36 (2015) 114–124

Contents lists available at ScienceDirect

Applied Soft Computing

journal homepage: www.elsevier.com/locate/asoc

Correcting geometric deviations of CNC Machine-Tools: An approach with Artificial Neural Networks

Wanderson de Oliveira Leite [a], Juan Carlos Campos Rubio [b,*], Jaime Gilberto Duduch [c], Paulo Eduardo Maciel de Almeida [d]

[a] Universidade Federal de Minas Gerais, Programa de Pós-Graduação em Engenharia de Produção, Av. Antonio Carlos, 6627 Pampulha, Belo Horizonte, MG, Brazil
[b] Universidade Federal de Minas Gerais, Departamento de Engenharia Mecânica, Av. Antônio Carlos, 6627 Pampulha, Belo Horizonte, MG, Brazil
[c] Universidade de São Paulo, Escola de Engenharia de São Carlos, Departamento de Engenharia Mecânica, Av. Trabalhador São Carlense, 400 Centro, São Carlos, SP, Brazil
[d] Centro Federal de Educação Tecnológica de Minas Gerais, Laboratório de Sistemas Inteligentes, Av. Amazonas, 7675 Nova Gameleira, Belo Horizonte, MG, Brazil

ARTICLE INFO

Article history:
Received 22 November 2013
Received in revised form 17 July 2015
Accepted 18 July 2015
Available online 26 July 2015

Keywords:
Artificial Neural Networks
CNC Machine-Tools
Error compensation
Design for Manufacturing
Precision technology

ABSTRACT

This paper presents an experimental methodology of Design for Manufacturing (DFM) used for survey and analysis of geometric deviations of CNC Machine-Tools, through their final product. These deviations generate direct costs that can be avoided through the use of Intelligent Manufacturing Systems (IMS), by the application of Artificial Neural Networks (ANNs) to predict the fabrication parameters. Finally, after the experiments, it was possible to evaluate the experimental methodology used, the equations, the variables of data adjustment and thus enable the validation of the methodology used as a tool for DFM with high potential return on product quality, development time and reliability of the process with wide application in various CNC Machines.

1. Introduction

Along with machine-tools (MT) evolution, there have been great advances in machining processes. However, obtaining complex surfaces with tolerance in the micrometric range has become extremely difficult. At the same time, the machining manufacturing process control has been evolving to attend the technical challenges imposed by complex requirements of form, by narrow specification limits and by the frequent introduction of new materials, tools and operational variables that originate new interactions in the processes with non-linear and non-standardized characteristics [1–4].

In the current manufacturing environment, indirect manufacturing operations generate direct costs that can be avoided or reduced by using control systems [5,6]. The use of Intelligent Manufacture Systems (IMS) has been researched through the application of Artificial Neural Networks (ANNs) since 1980 [7].

In this context, an important methodology of predictive engineering is the Design for Manufacturing (DFM). It incorporates, in the project processes, information referring to the manufacture, also allowing for the project to be adapted during each stage of production [8].

This paper presents a study where, at first the geometric deviations of a machine-tool are analyzed through the unfinished product. Afterwards, ANNs are used to develop models to minimize error prediction of the milled surface. A DFM approach is proposed to implement in the Computer-aided design (CAD) software, through the existing Application Programming Interface (API), the parameter soft correction/control of the geometric deviations compensation, before the manufacturing of the products. Finally, some practical experiments are discussed to evaluate the proposed approach.

2. Literature review

2.1. Machine-tools

Machine-tools (MT), also called tooling machines, were initially defined as machines used in the manufacturing of pieces of

* Corresponding author.
E-mail addresses: juancarlos681@gmail.com, juan@ufmg.br (J. Carlos Campos Rubio).

http://dx.doi.org/10.1016/j.asoc.2015.07.014
1568-4946/

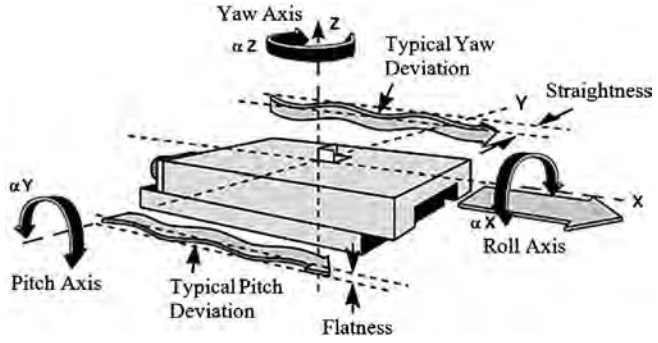

Fig. 1. Geometrical errors on the trajectory of a positioner.

various materials through the mechanical movement of a tool kit. As early as the 1970s, with the development of microprocessors and precision technology, the computerized numerical command (CNC) made possible the so-called CNC manufacturing [9].

A manufacturing center is a CNC machine-tool equipped with tool magazines that allow the execution of a group of sequential manufacturing operations. These machines have automatic systems of tool changes and a set of indexed axes. The displacements of these axes are based on the definition of the coordinates of the points to be covered. This axes system is defined by DIN 66217 standard, according to ISO R841 recommendation. However, CNC programming starts by the geometric elements definition (dimensions, solids, 3D, etc.) in a Computer-aided manufacturing (CAM) system. Based on this information, the elaboration program takes place. The designer can opt for the following methods: (a) manual programming; (b) interactive graphic programming; (c) CAD/CAM systems [10].

2.2. Geometric errors or beat deviations (runouts) in machine-tools

According to Schwenke et al. [11] and Tian et al. [12] there are three main geometric errors sources for machine-tools, namely: (1) errors due to geometric deviations, (2) errors due to heat effects and (3) errors due to forces.

Adding to the above, geometrical errors are a linear part (versus angular) of the non-axial errors. That is, geometrical errors can be characterized by two orthogonal straight components, one measured on the deviation plan (straightness) and another measured out of the deviation plan, (flatness) [13,14], as shown Fig. 1.

Geometrical errors are originated from mechanical imperfections in the machine-tools structure and the misalignment of their components [11]. As a result, we have a lack of parallelism and perpendicularity in the movements of many components, causing errors of form and position [12]. Inclination and oscillation errors (Tilt, Wobble) correspond to the angular part of geometrical errors. They represent the deviations between the ideal straightness movement and the real movement in a translation. The inclination and oscillation problem has three orthogonal components called Roll, Pitch and Yaw, also shown in Fig. 1. These terms are generally used to name the global geometrical error of the positioner, since it contemplates all deviations of this kind [11].

Gangwei et al. [14] and Ramesh et al. [15] cite other sources of possible errors, among them: radial error, tilt error around an-axis, errors due to heat effects, errors due to forces, kinematic errors, errors due to material instability, instrumentation errors, cut tool wear, fixation errors and servo-position errors (movement and interpolation errors). In Okafor and Yalcin [16], Chana and Parnichkun [17] and Chen et al. [18], 21 different geometrical errors are cited, being possible to occur in a 3-axis vertical machining.

2.3. Design for Manufacture

According to Bogue [19], Design for Manufacture (DFM) originated in 1960 in the General Electric Corporation. In the 1970s, Boothroyd and Dewhurst presented studies which brought many productive factors to the product development using the assembling time as a parameter for the product analysis. For Bralha [20] and Boothroyd et al. [8] DFM today tries to conciliate simultaneously the project goals with the manufacturing limitations, emphasizes manufacturing and processes aspects avoiding manufacturing problems in future stages. DMF is therefore a systematic approach that allows the engineers to foresee production problems at the beginning of the conception process.

According to Hoque et al. [21] the difficulties found in the implementation of this technique are directly related to the need to work in a Computer-Integrated Manufacturing (CIM) environment and the availability in the market of few CAD, CAM and computer-aided process planning (CAPP) systems that offer the possibility of estimating manufacturing costs and processes optimization.

2.4. ANN and machining processes

The use of Manufacturing Intelligent Systems (MIS) by applying artificial neural networks (ANNs) has been researched since the 1980s. Several researchers have proposed the use of machining models as a solution to many problems using Computational Intelligence (CI) techniques [1–7].

In this context, ANNs are been employed and applied with the objective of controlling and optimizing of cutting conditions [22]. Tandon and El-Mounayri [23] studied the cutting force for developed a model to predict parameters for end milling. On the other hand, Panda et al. [24] and Yang et al. [25] used ANN for prediction of drills wear out through data capturing by sensors, measuring and controlling of machining forces. Briceno et al. [26] developed an ANN with *back-propagation* (BP) training to estimate milling parameters, Zuperl and Cus [27] applied a similar network to predict the three cutting forces (*Fx*, *Fy*, and *Fz*) in the machining of molds. Similarly, Zuperl et al. [28] developed two supervised network models to control the three components of forces using many parameters of entries, such as, cutting fluids, hardness, material, etc. Leite et al. [29] studied and developed ANN models for evaluating the influence in the geometry and surface finish of different milling strategies, thus generating a set of maps/answers of analysis and prediction.

In his work on machinability of reinforced polyamide composite, Karnik et al. [30] studied the drilling of composite plates through ANN to develop a prediction model for the drilling effects under different conditions. For its part, Mishra et al. [31] developed a predictive model of the residual resistance to traction to perforate plastic reinforced fibers (FRP) using ANN.

Finally, the surface quality, mainly characterized by the surface roughness, is an area for which the machinability models using CI techniques have also been investigated, for example, in the prediction of surface roughness in high-torque face milling operations, combining multilayer perceptrons (MPL) with genetic algorithms [32–35]. However, ANN uses for control of roughness statistical are characterized by a great variability in the adopted techniques, as described both in Correa et al. [36] as Çaydas and Hasçalik [37].

3. Methods

This paper uses the DFM methodology described by Gupta and Nau [38] to evaluate machining capability during the initial phase of a work-piece's project, based upon its geometry and on the tolerance of each operation involved in its fabrication to generate

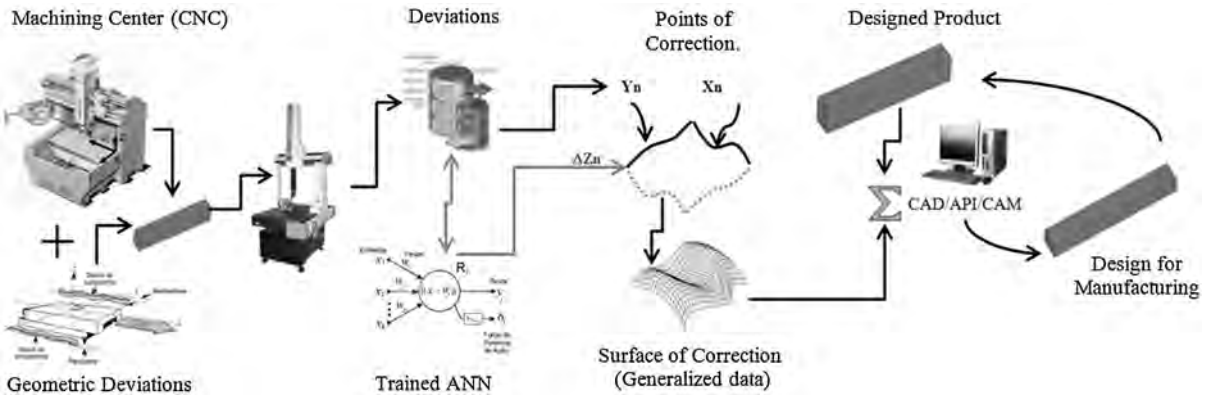

Fig. 2. Flow chart of the idealized compensation process.

an operational plan. The implementation of such a plan includes the manufacturing equipment, the auxiliary equipment and accessories used in the project, CAD, CAM, Computer-aided inspection (CAI), DOE (Minitab^{TM}), and the ANN (Matlab^{TM}). Fig. 2 shows the flow chart of the idealized compensation process.

3.1. Multi-layers ANN models

The multi-layer artificial neural networks, referred to as Multilayer Perception (MLP), will be implemented with *Back-Propagation* (BP) and *Levenberg–Marquardt* (LM) training algorithms [29]. These models for the modeling of the milling process are shown in Fig. 3a.

Basically, MLP consists of neurons divided into entrance layer, exit layer and one or two other hidden layers. The neurons between the layers are connected by links with synapses (weights) that store the knowledge about the related entrance and exit [39].

Fig. 3b represents the MLP architecture employed, which consists of two (02) neurons in the entrance layer (*X* and *Y* axes), "*j*" neurons in the hidden layers and one (01) neuron in the exit layer (deviation in *Z*-axis).

The activation of the "*net*" entrance for the *j* neurons in t he main, hidden and exit layers, is given by Eq. (1).

$$net_j^l = \sum_{i=1}^{n} \{W_{j,i(p)}^l * X_i + \theta_j\} \tag{1}$$

where "*W*" is the weight vector of the links of the *j* neuron and "*X*" *is* the vector or matrix of the entrance and "*n*" is the number of patterns shown to the net (entrances of this neuron).

Among the functions of possible activation, if we assume the unipolar *sigmoid*, the exit of this neuron is given by Eq. (2):

$$Y_{j(p)}^l = \frac{1}{1 + e^{-\lambda net_j}} \tag{2}$$

where, λ is a scalar factor.

The algorithm trying to minimize the SSE for the *j*th neurons in the layer of each time, may be calculated by Eq. (3):

$$\mathrm{SSE}_{j(p)} = \frac{1}{2} \sum_{i=1}^{n(l)} (e_{j,i(p)}^l)^2 \tag{3}$$

Therefore, the net is trained for all training patterns and the *Mean Square Error* (MSE) is calculated for all patterns, Eq. (4):

$$\mathrm{MSE}_{(p)} = \frac{1}{K} \sum_{j=1}^{K} \frac{1}{n} \sum_{i=1}^{n(l)} (e_{j,i(p)}^l)^2 \tag{4}$$

The training process is over when the MSE reaches an error equal or inferior to the one specified or when the number of epochs reaches its maximum determined value.

In all interactions, the updating of link weights is determined for the *j*th neurons, by Eq. (5):

$$W_{j,i(p+1)}^l = W_{j,i(p)}^l + (mu) * (\delta_{j,j(p)}) * (Y_{j,i(p)}^l) + \alpha * (\mathrm{SSE}_{j(p)} * Y_{j,i(p)}^l) \tag{5}$$

where *p* is the training interactions, α is the learning rate and *mu* is the *momentum constant*.

The error propagation term ($\delta_{j,(p)}$) or error gradient is defined by Eq. (6), for the exit layer and by Eq. (7) for the hidden layer [30].

$$\delta_{j,k(p)} = (D_i - Y_{j,i(p)}^l) * (1 - Y_{j,i(p)}^l); \quad j = 1, \ldots, K \tag{6}$$

$$\delta_{j,i(p)} = Y_{j,i(p)}^l * (1 - Y_{j,i(p)}^l) * (\delta_{j,k(p)}) * (W_{j,i(p)}^l); \quad i = 1, \ldots, n \tag{7}$$

On the other hand, the *Levenberg–Marquardt* (LM) algorithm adopted by Hagan and Menhaj [41] for the neural nets training, and later implemented by Demuth and Beale [42] for the MATLAB^{TM}, provides a numerical solution for the problem of minimizing the non-linear function through the calculation of the *Jacobian* Matrices, based on the *Gauss–Newton* method and the *Gradient Descent Algorithm* [43].

So, mathematically can be conclude that the entrance activation "*net*" for the *j* neurons on the first layer ($1^{\underline{a}}$) is:

$$net_j^1 = \sum_{i=1}^{n} \{W_{ji}^1 * X_i + \theta_{j0}^1\} \tag{8}$$

where, X_i are the *i*th entrances of the *j* neurons pondered by the respective weights W_{ji}^1, from 1 to*n*, *n* being the entrance and weight number of patterns. The over-index "1" means the first layer (entrance layer), and the *j* indicates which neuron of the layer we are working on and θ_{j0}^1 is the bias of *j* neuron of the first layer.

The processing of entries X_i and exit Y_j of *j* neuron in the interaction "*p*" is given by equation

$$Y_{j(p)}^1 = \varphi_j^1 \{net_j^1\} \tag{9}$$

where $Y_{j(p)}$ is the exit sign or value of *j* neuron and φ_j the function of activating this neuron.

The calculation of the derivatives of the activation function in the *p* interaction for the *j* neurons of the first layer is defined as:

$$S_{j(p)}^1 = \frac{\partial \varphi_j^1}{\partial net_j^1} \tag{10}$$

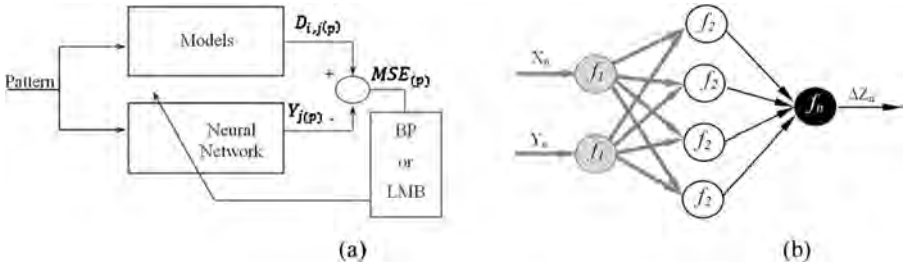

Fig. 3. (a) Neuronal Network MLP with BP or LMB algorithms; (b) architecture MLP used.

Admitting the neurons exit from the current layer (1st) as an entrance to the *l*th subsequent layers of the net, Eqs. (8)–(10) are adopted for the net_j^l, $Y_{jp}^l eS_{jp}^l$ calculation of all layers of the net. Therefore, we have all the "*Y*" and "*S*" *matrices'* points (cells) for the *j* neurons of the net necessary for the error, gradient and weight nets.

For the last layer neurons, we have:

$$e_{j(p)} = (D_{i,j(p)} - Y_{j(p)}) \tag{11}$$

$$\delta_{j(p)}^l = S_{j(p)}^l \tag{12}$$

where, $e_{j(p)}$ is the error calculated for the *j* neuron in time *p*, by the respective *i*th item desired from the objective vector ($\delta_{j(p)}^l$) and by the neuron exit value $Y_{j(p)}$. Furthermore, the error auto-propagation term ($\delta_{j(p)}^l$) is defined by Eq. (12), only for the exit layer.

For the hidden layers until the entrance ones we have:

$$\delta_{j(p)}^{l-1} = \sum_{i=1}^{n(l)} w_{j,i(p)}^l * \delta_{j,i(p)}^l \tag{13}$$

$$\delta_{j(p)}^l = \delta_{j(p)}^l * S_{j(p)}^l \tag{14}$$

where $\delta_{j(p)}^{l-1}$ is the entrances' error gradient of the current layer (1) for the outputs of the previous layer ($1-1$) and $\delta_{j(p)}^l$ is the output error propagation term (self-propagation).

In all epochs, the updating of the weight links is determined for the *j*th neuron of each layer by Eq. (15):

$$W_{j(p+1)}^l = W_{j(p)}^l - (H_{(p)} + \mu * I)^{-1} * G_{(p)} \tag{15}$$

where

$$H_{(p)} = J_{(p)}^T * J_{(p)} \tag{16}$$

$$G_p = J_{(p)}^T * SSE_{(p)} \tag{17}$$

where $H_{(p)}$ is the *Hessian* matrix approximation and G_p is the *Descent Algorithm* solution in time *p* for the *n* net entrance patterns. Where, $H_{(p)}$ is a $n \times n$ matrix, G_p is a vector of the *n* elements column, the term *I* is the identity matrix $n \times n$ and μ is the combination coefficient [43].

The *Hessian* matrix and the error gradient approximation are calculated by the numerical solutions of the system formed by the *Jacobian* matrix *J* of a $m \times n$ size, where *m* is the amount of data for a pattern X_n of the *X* matrix and J^T its transpose. $J_{(p)}$ is defined by Eq. (18):

$$J_{(p)} = \begin{bmatrix} \frac{\partial e_{1,1}}{\partial w_1} & \cdots & \frac{\partial e_{1,1}}{\partial w_n} \\ \cdots & \cdots & \cdots \\ \frac{\partial e_{1,m}}{\partial w_{1(p)}} & \cdots & \frac{\partial e_{1,m(p)}}{\partial w_{n(p)}} \end{bmatrix} \tag{18}$$

Finally, for both cases, this research adopted the *Mean Absolute Error (MAE)* with the performance of the studied nets parameters to best represent the real deviations (absolutes) among the collected and trained data. Eq. (19) describes the MAE calculations:

$$MAE_{(p)} = \frac{1}{K}\sum_{j=1}^{K}\frac{1}{n}\sum_{i=1}^{n(l)}|e_{j,i(p)}^l| \tag{19}$$

Therefore, the error (1 μm) will be considered on the same precision range of the CAI machining equipment.

4. Experimentation

4.1. Computational equipment and tools used

A *Discovery 560* ROMI™ Machining Center and a *Micro-Hite 3D* TESA™ 3D Coordinate Measuring Machine (CMM 3D) were used.

A commercial *Personal Computer (PC)* to integrate the *net* system, software and equipment was used. Software were chosen so that information could be shared, the main ones used were *SolidWorks*™ *2008*, GibbsCAM by SolidWorks™, *Volcomps*™, *MiniTab16*™ and *MATLAB*™ *R2011a*, on the *MS Windows XP Professional*™ *SP3* PT-BR.

4.2. Experimental work

Initially, the zero position of the machine ($x = 0$, $y = 0$ and $z = 0$) and the directions of the displacements of the axes are set up. Also, the coordinates which would be machined in the samples were determined (conversion of the measured data versus real data). After this phase, the position to set the samples with the table marking was planned. To have this accomplished, guide blocks were used to position the samples being aligned using a comparative clock.

The cutting tool was a MITSUBISHI milling cutter with 50 mm diameter and three teeth. Two cemented carbide inserts had geometry of SEGT13T3AGFN-JP (with chip breaker) and one special wiper cemented carbide insert WEEW13T3AGER8 C, which were clamped in a milling cutter code ASX445-050A03R (see Fig. 4a). The geometric combination of the insert and the tool holder resulted in 45° cutting edge angle and 20° rake angle (see Fig. 4b). In these end milling experiments, only one set insert was used, which would eliminate the influence of the tool tip run out on tool wear. The experiments were conducted at a constant cutting speed of 400 m/min, which would eliminate the influence of the built-up edge.

Once fixed the work piece was machined (milled) with the following cutting parameters: cutting speed (V_c) = 400 m/min, spindle speed (S) = 2546 rpm, feed rate (f_z) = 0.1 mm/rev/tooth, depth of cut (a_p) = 0.5 mm, width of cut (a_e) = 17 mm, up-milling with cutting fluid.

Work piece samples (51 mm × 51 mm × 460 mm) were prepared in aluminum (ASTM 6351 T6) to be submitted to milling tests along the *X*-axis and *Y*-axis in order to check for errors in both directions according to Fig. 1. Their two upper faces of the samples were

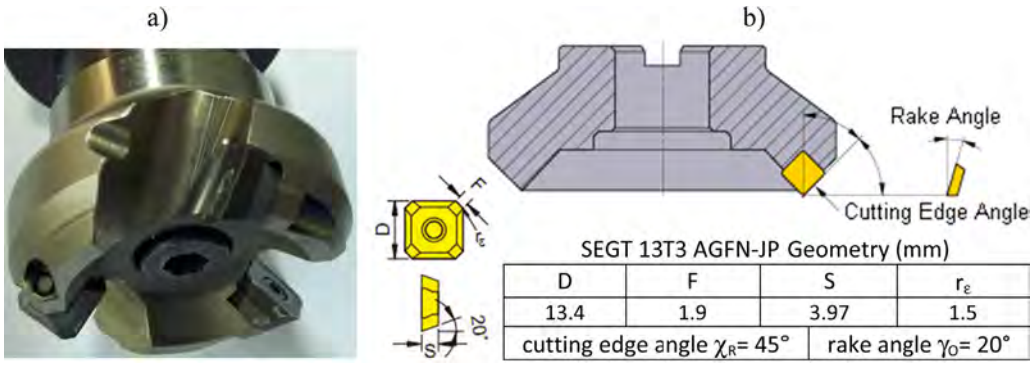

Fig. 4. (a) Cutting tool details; (b) milling cutter and insert geometric details.

Fig. 5. (a) Milling along the *X*-axis; (b) milling along the *Y*-axis; (c) MMC 3D measurement.

milled: one along the *X*-axis and another along the *Y*-axis, see Fig. 5a and b for details.

After milling, the work pieces were measured (CAI) according to the principles presented by Nakazawa [44]. The *Z* measures were taken through touches in the direction *Z* (compensation) along the *X*-axis and keeping *Y* constant (MMC 3D coordinate system) minimizing parallax errors (the *Abbe's* principle) (see Fig. 5c).

Ten points were randomly probed in each surface. Each point was measured three times and the average value *Z* was used. Fig. 6 shows the errors (deviations in *Z*) of these points collected in a tridimensional space (3D scatter plot) along the *X* (Fig. 6a) axis and *Y* (Fig. 6b), with respect to the work-piece's nominal dimensions.

A set of *Z* versus *X* and *Z* versus *Y* displacements points for both machined samples were acquired. Table 1 shows the variations found on *Z*-axis along the *X*-axis. It is possible to observe that, as the mill moves toward a negative direction of *X* (*X*→−), (see Fig. 5a), *Z* values increase (*Z*→+). Likewise, in Table 2 it is possible to observe the *Z*-axis variation along the *Y*-axis. Displacements in the negative direction of *Y* (*Y*→−) cause *Z* values to decrease (*Z*←−), (see Fig. 5b). Ideally, *Z* should be constant along *X*- and *Y*-axes.

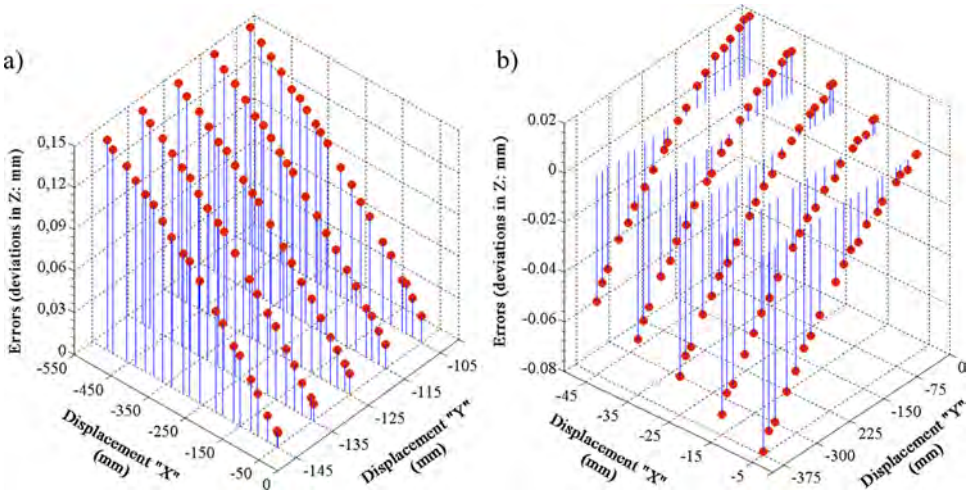

Fig. 6. (a) deviations in *Z* (mm) versus *X* (mm) 3-D scatter plot, acquired data; (b) deviations in *Z* (mm) versus *Y* (mm) 3-D scatter plot, acquired data.

Table 1
Measured data: displacement "Z" versus surface "X" axis.

Values[a]															
No.	(1st) sequence points			(2st) sequence points			(3st) sequence points			(4st) sequence points			(4st) sequence points		
	X	*Y*	*Z*	*X*	*Y*	*Z*	*X*	*Y*	*Z*	*X*	*Y*	*Z*	*X*	*Y*	*Z*
1	−50.000	−144.850	0.010	−50.000	−134.913	0.012	−50.000	−124.975	0.015	−50.000	−115.038	0.017	−50.000	−105.100	0.019
2	−52.801	−144.850	0.011	−58.004	−134.913	0.015	−63.206	−124.975	0.020	−68.409	−115.038	0.024	−73.611	−105.100	0.028
3	−78.317	−144.850	0.018	−81.387	−134.913	0.022	−84.458	−124.975	0.027	−87.528	−115.038	0.032	−90.598	−105.100	0.036
4	−103.397	−144.850	0.030	−103.015	−134.913	0.032	−102.634	−124.975	0.033	−102.252	−115.038	0.035	−101.870	−105.100	0.037
5	−123.175	−144.850	0.037	−125.347	−134.913	0.040	−127.518	−124.975	0.043	−129.690	−115.038	0.047	−131.861	−105.100	0.050
6	−151.684	−144.850	0.050	−152.485	−134.913	0.052	−153.286	−124.975	0.053	−154.087	−115.038	0.055	−154.888	−105.100	0.056
7	−169.020	−144.850	0.054	−174.300	−134.913	0.058	−179.581	−124.975	0.061	−184.861	−115.038	0.065	−190.142	−105.100	0.069
8	−198.687	−144.850	0.066	−202.220	−134.913	0.068	−205.752	−124.975	0.071	−209.285	−115.038	0.073	−212.817	−105.100	0.076
9	−218.582	−144.850	0.072	−223.584	−134.913	0.075	−228.586	−124.975	0.079	−233.588	−115.038	0.082	−238.590	−105.100	0.085
10	−260.685	−144.850	0.087	−262.480	−134.913	0.088	−264.275	−124.975	0.090	−266.069	−115.038	0.091	−267.864	−105.100	0.093
11	−289.079	−144.850	0.097	−292.660	−134.913	0.099	−296.240	−124.975	0.101	−299.821	−115.038	0.103	−303.401	−105.100	0.105
12	−304.695	−144.850	0.100	−308.784	−134.913	0.103	−312.873	−124.975	0.105	−316.962	−115.038	0.107	−321.051	−105.100	0.110
13	−337.192	−144.850	0.107	−336.504	−134.913	0.109	−335.816	−124.975	0.111	−335.128	−115.038	0.112	−334.440	−105.100	0.114
14	−360.496	−144.850	0.115	−360.198	−134.913	0.116	−359.899	−124.975	0.118	−359.601	−115.038	0.119	−359.302	−105.100	0.120
15	−385.216	−144.850	0.123	−383.426	−134.913	0.124	−381.637	−124.975	0.125	−379.847	−115.038	0.125	−378.057	−105.100	0.126
16	−407.300	−144.850	0.127	−405.538	−134.913	0.128	−403.777	−124.975	0.128	−402.015	−115.038	0.128	−400.254	−105.100	0.129
17	−433.779	−144.850	0.133	−432.081	−134.913	0.134	−430.383	−124.975	0.135	−428.684	−115.038	0.136	−426.986	−105.100	0.137
18	−457.408	−144.850	0.140	−456.855	−134.913	0.141	−456.303	−124.975	0.142	−455.750	−115.038	0.143	−455.197	−105.100	0.144
19	−494.795	−144.850	0.146	−491.680	−134.913	0.147	−488.566	−124.975	0.148	−485.452	−115.038	0.149	−482.338	−105.100	0.150
20	−510.781	−144.850	0.151	−510.794	−134.913	0.153	−510.807	−124.975	0.154	−510.819	−115.038	0.156	−510.832	−105.100	0.157

[a] Points with coordinates (*X*, *Y* and *Z*) converted to actual placement of the Machining Center.

Table 2
Measured data: displacement "Z" versus surface "Y" axis.

Values[a]															
No.	(1st) sequence points			(2nd) sequence points			(3rd) sequence points			(4th) sequence points			(4th) sequence points		
	X	*Y*	*Z*	*X*	*Y*	*Z*	*X*	*Y*	*Z*	*X*	*Y*	*Z*	*X*	*Y*	*Z*
1	−5.100	−28.000	0.000	−15.138	−28.000	0.006	−25.175	−28.000	0.012	−35.213	−28.000	0.017	−45.250	−28.000	0.023
2	−5.100	−29.691	0.000	−15.138	−32.567	0.006	−25.175	−35.443	0.012	−35.213	−38.319	0.017	−45.250	−41.195	0.023
3	−5.100	−47.816	−0.004	−15.138	−48.524	0.002	−25.175	−49.231	0.008	−35.213	−49.939	0.015	−45.250	−50.646	0.021
4	−5.100	−65.056	−0.004	−15.138	−66.807	0.002	−25.175	−68.558	0.007	−35.213	−70.308	0.013	−45.250	−72.059	0.019
5	−5.100	−74.318	−0.006	−15.138	−77.163	0.000	−25.175	−80.008	0.006	−35.213	−82.853	0.011	−45.250	−85.698	0.017
6	−5.100	−104.702	−0.010	−15.138	−104.844	−0.003	−25.175	−104.986	0.003	−35.213	−105.127	0.010	−45.250	−105.269	0.016
7	−5.100	−115.301	−0.013	−15.138	−118.439	−0.007	−25.175	−121.578	0.000	−35.213	−124.716	0.006	−45.250	−127.854	0.012
8	−5.100	−140.073	−0.015	−15.138	−141.512	−0.009	−25.175	−142.951	−0.003	−35.213	−144.389	0.003	−45.250	−145.828	0.009
9	−5.100	−158.697	−0.020	−15.138	−161.843	−0.014	−25.175	−164.989	−0.008	−35.213	−168.134	−0.003	−45.250	−171.280	0.003
10	−5.100	−176.435	−0.021	−15.138	−179.783	−0.016	−25.175	−183.131	−0.011	−35.213	−186.478	−0.005	−45.250	−189.826	0.000
11	−5.100	−191.141	−0.025	−15.138	−196.458	−0.020	−25.175	−201.775	−0.015	−35.213	−207.091	−0.011	−45.250	−212.408	−0.006
12	−5.100	−210.854	−0.030	−15.138	−213.281	−0.024	−25.175	−215.708	−0.019	−35.213	−218.135	−0.013	−45.250	−220.562	−0.008
13	−5.100	−246.774	−0.039	−15.138	−246.485	−0.032	−25.175	−246.195	−0.026	−35.213	−245.906	−0.019	−45.250	−245.616	−0.013
14	−5.100	−266.579	−0.045	−15.138	−265.973	−0.038	−25.175	−265.367	−0.032	−35.213	−264.761	−0.025	−45.250	−264.155	−0.018
15	−5.100	−282.395	−0.048	−15.138	−283.527	−0.042	−25.175	−284.660	−0.035	−35.213	−285.792	−0.029	−45.250	−286.924	−0.023
16	−5.100	−300.844	−0.055	−15.138	−300.942	−0.048	−25.175	−301.040	−0.041	−35.213	−301.137	−0.035	−45.250	−301.235	−0.028
17	−5.100	−320.828	−0.061	−15.138	−321.208	−0.054	−25.175	−321.588	−0.046	−35.213	−321.968	−0.039	−45.250	−322.348	−0.032
18	−5.100	−347.136	−0.070	−15.138	−347.407	−0.063	−25.175	−347.678	−0.056	−35.213	−347.949	−0.048	−45.250	−348.220	−0.041
19	−5.100	−361.121	−0.072	−15.138	−360.995	−0.065	−25.175	−360.869	−0.058	−35.213	−360.743	−0.052	−45.250	−360.617	−0.045
20	−5.100	−372.901	−0.079	−15.138	−372.863	−0.072	−25.175	−372.824	−0.065	−35.213	−372.786	−0.058	−45.250	−372.747	−0.051

[a] Points with coordinates (*X*, *Y* and *Z*) converted to actual placement of the Machining Center.

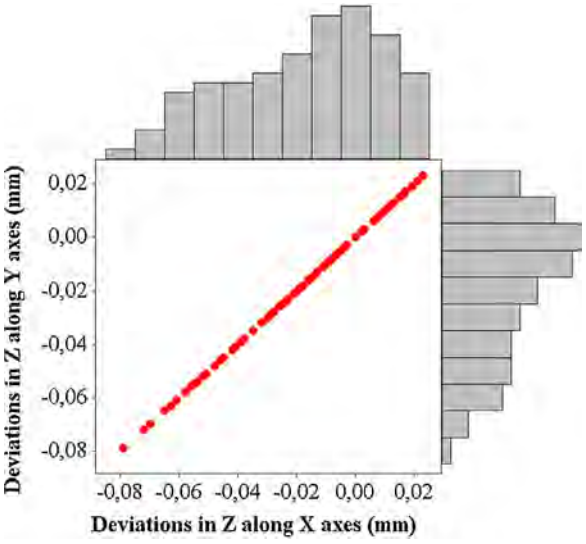

Fig. 7. Joint distribution plot of the deviations value in Z (mm) on the X- and Y-axis.

5. Mathematical modeling of flatness deviation

5.1. Data analysis

A statistical analysis of the collected data was conducted (see Tables 1 and 2) to check for the existence of any outliers and to eliminate them. Fig. 7 illustrates the deviations found during the measurement process after the elimination or adjustment of the outliers for X and Y surfaces, presenting the joint distribution of data. This is composed of a 2D scatter plot with two histograms in the margins. Each histogram shows how the deviations in Z on the X- and Y-axis are distributed and the scatter plot the dependencies of points with error value in Z (mm).

5.2. Studied models and results

At first, we studied MLP models with *Back-Propagation* training algorithm with only one hidden layer, one neuron and 1–8 neurons on the main layer. The simulation general parameters used were: test values for X = [−560], test values for Y = [−420], *Learning rate* = 0.10, *network performance* = "MAE" and *error maximum increment* = 0.08. Joined with the stop parameters formed: *Maximum number of epochs to train* = 10000, *Performance goal* = 0.001, *Minimum performance gradient* = $10e^{-26}$, *Maximum Number of Validation Increases* = 200, *Maximum time to train in seconds* = 120 and *Maximum mu* = $1e^{30}$.

For each proposed model, three replicates were generated considering the average value of the comparison model results. Among the studied variations, we can highlight the standard model using three neurons on the main layer with MAE = 0.00233, whose results are shown in Fig. 8a.

However, the generalization test for this model showed surfaces with irregular or distorted answers as could be seen in Fig. 8a. Therefore, the variations of the Activation Functions *Tangent-Sigmoid* (*tansig*), *Logarithmic-Sigmoid* (*logsig*) and *Linear* (*purelin*) were studied for the main and hidden layers in the model with three neurons, using a smaller number of neurons and consequently a simpler architecture. In Fig. 8b, we have the model with *tansig* function in both layers. This showed the best performance with MAE = 0.00447.

Subsequently, keeping the same simulation parameters and criteria of the previous model (MLP + BP), the models with the *Levenberg–Marquardt* training algorithm with one hidden layer and from 1 to 3 neurons in the main layer were studied.

In this context, Fig. 9 a and b shows, respectively, the surfaces generated by the models with three neurons on the main layer and with *tansig* and *logsig* activation functions in this layer. So, both models converged to the desired final error.

Thus, in Fig. 9a we have the model generated by the average of five trained networks. A generalization test was conducted with discretization of 10 mm in X and 10 mm in Y, and achieved MAE was $9.4343e^{-4}$. Fig. 9b shows results for the same process but now using *logsig* function, which obtained MAE = $9.5525e^{-4}$.

Complementing these results, Fig. 10 shows value of residues for the points collected/training versus the models developed, with *tansig* and *logsig* activation functions, respectively.

5.3. Analysis and discussion

Table 3 shows data of the main MLP models with *BP* training algorithm (tests 1–11) that were employed. The data refer to the average obtained for each model with three replicates. The test number 3, as shown in Fig. 7a and b, presented the best performance using this algorithm with MAE = 0.00233. We can conclude that the change in the activation models with three neurons (tests 7–11) did not produce any important improvement in the model performance. Table 3 also presents data of the developed LM models (tests 12–15), whose results are averages of five replicates obtained for each model. Test number 14, according to what has been shown in Fig. 11, showed the best performance, with $MAE = 9.4343e^{-4}$ and $MSE = 1.5425e^{-6}$.

Fig. 11a shows the performance of models 1–6 versus training epochs of the models with the BP algorithm. There, we can confirm that the increase in the number of neurons did not result in a performance improvement. All models show no more performance improvement at 10,000 epochs. Also, some variations reached the desired performance very close to the epoch limit (at 10,000), what is expected when a great quantity of patterns is presented to the models for training and each time step is small [43].

At last, for the tests carried out, BP showed to be incapable of developing generalization properties to solve similar problems, according to what is presented in Fig. 8a and b. There, some distorted responses generated by incoherent generalizations of the developed models can be seen. Due to these facts, we can infer that the processed BP models got stuck at a local minimum and the generated models were not able to solve the problem in hand. BP algorithm limitations have already been discussed by a number authors as Schalkoff [40], Jian et al. [45], Özel and Davim [46] and Yang et al. [25].

On the other hand, models using the LM algorithm (Fig. 11b) had a better response with a significant reduction in the processing time and epoch numbers, converting to the desired error with a relatively small neuron number. This has already been shown in previous researches comparing the BP with the LM algorithms in the resolution of problems that have a significant number of entrance patterns [4,35,43,47].

Fig. 11b shows the performance of the four models tested with 1, 2 and 3 neurons versus epoch evolution. We can see that the model trains fast, stabilizing its performance at 55 epochs and also showing a significant improvement when adding new neurons to the network architecture.

Tests 11a–b with three neurons on the main layer and the *tansing* and *logsing* activation functions, respectively, are also shown in Fig. 11b. We can conclude that the *logsing* function shows a better performance on the first 10 epochs, but after this, both have a similar behavior and furthermore, model 14 (*tansig*) has a better performance in the processing (epochs/time). At last, in Fig. 10, we can see that residues of MLP present themselves totally dispersed

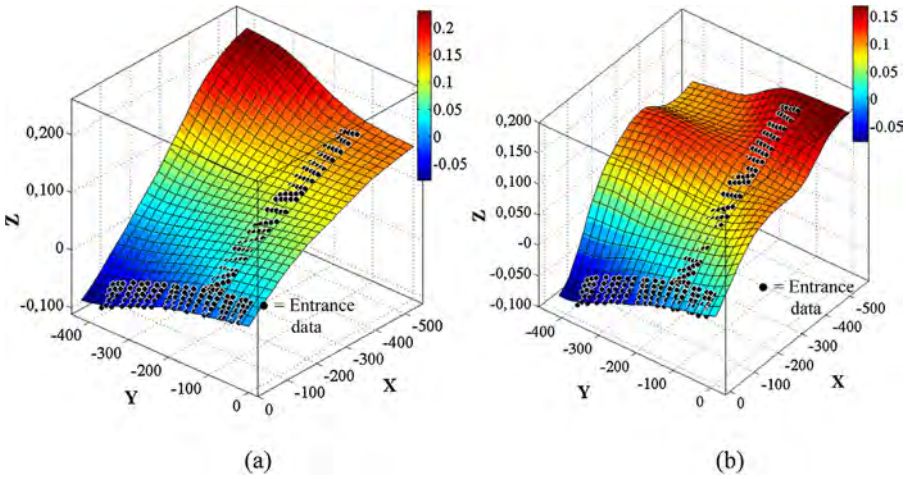

Fig. 8. (a) Answer plan of deviations in *Z* and entrance data, net with 3 neurons in the main layer; (b) answer plan of deviations in *Z* and entrance data, net with 8 neurons on the main layer.

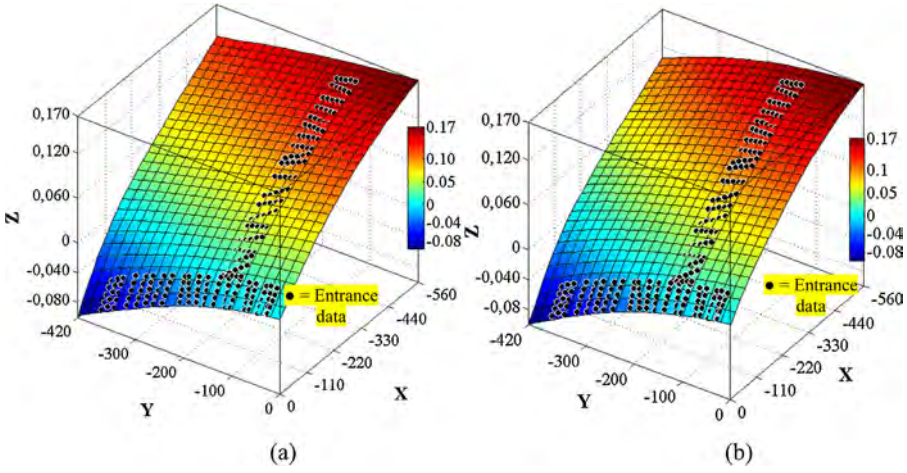

Fig. 9. (a) Answer plan of deviations in *Z* and entrance data, LMB net with *tansig* transference functions; (b) answer plan of deviations in *Z* and entrance data, LMB net with *logsig* transference functions.

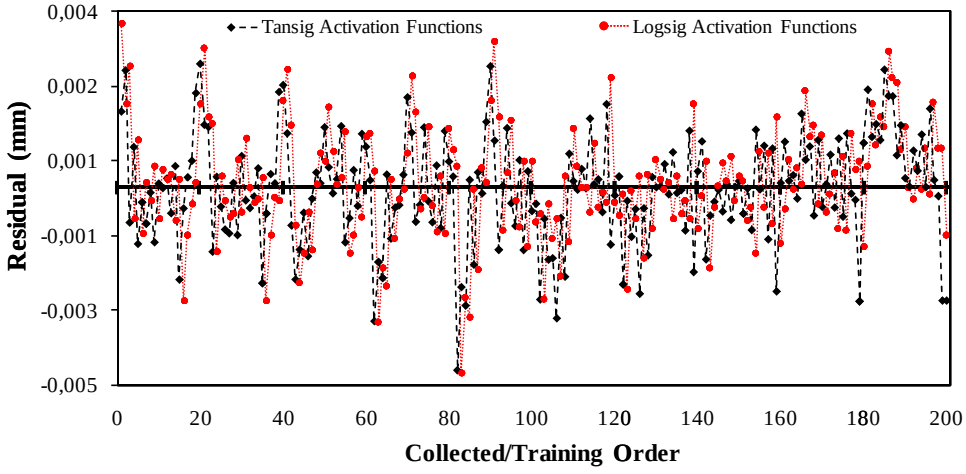

Fig. 10. Residues value for the LMB net with functions *tansig* and *logsig*.

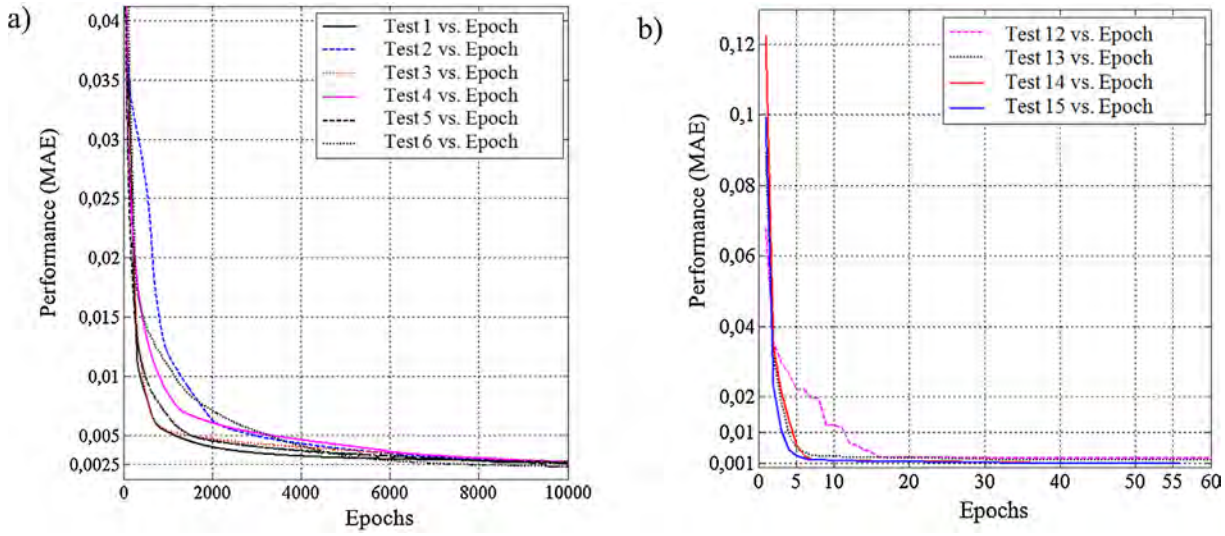

Fig. 11. (a) BP researched performance versus epoch numbers; (b) LM researched performance versus epoch numbers.

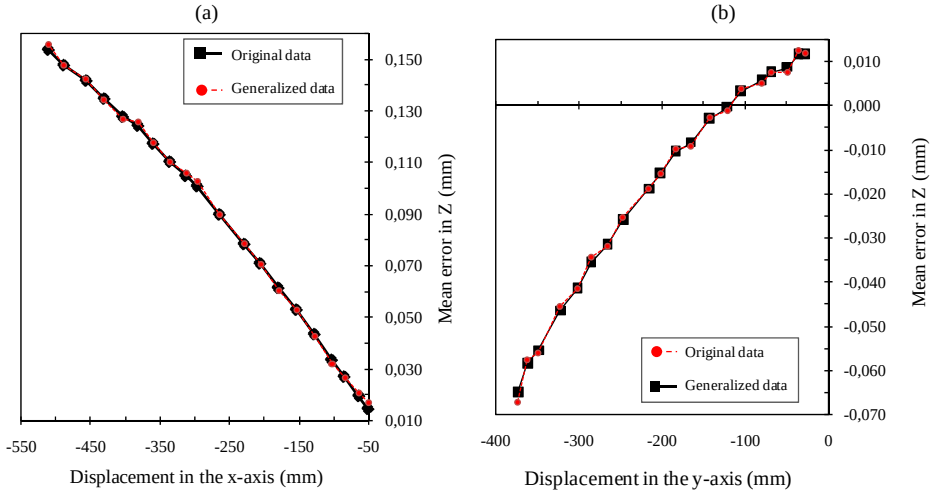

Fig. 12. (a) *Z* (mm) versos *x*-axis (mm) displacement, original and generalized data; (b) *Z* (mm) versus *y*-axis (mm) displacements, original and generalized data.

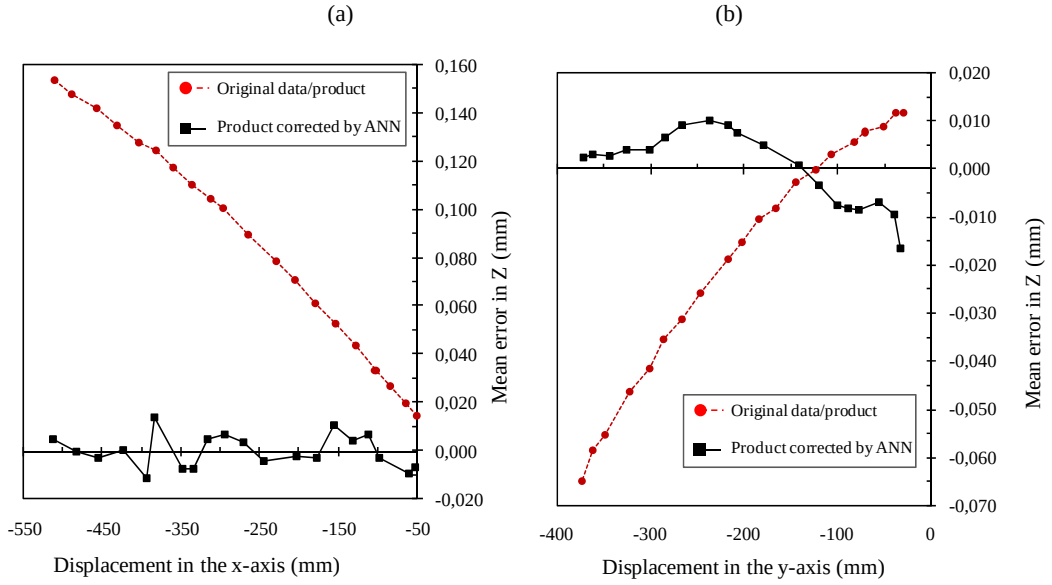

Fig. 13. (a) Graphic of *Z* deviations along *Y*-axis; (b) graphic of *Z* deviations a long *X*-axis.

Table 3
Test parameters for MPL with BP and LM algorithms.

Tests	Number of neuron (1st layer)	Number of neuron (layer hidden)	Network training function	Network's adaption function	Transfer function (1st layer)	Transfer function (layer hidden)	learning function	Error MAE	Error MSE	Best Epoch	Processing time (s)
1	1	1	traingd	adaptwb	*tansig*	*purelin*	learngdm	0.003300	0.0000200	7.230	59.615
2	2	1	traingd	adaptwb	*tansig*	*purelin*	learngdm	0.002800	0.0000100	9.987	83.552
3	3	1	traingd	adaptwb	*tansig*	*purelin*	learngdm	0.002330	0.0000100	9.998	57.563
4	4	1	traingd	adaptwb	*tansig*	*purelin*	learngdm	0.002900	0.0000200	9.994	84.239
5	6	1	traingd	adaptwb	*tansig*	*purelin*	learngdm	0.002800	0.0000100	9.994	85.109
6	8	1	traingd	adaptwb	*tansig*	*purelin*	learngdm	0.002630	0.0000100	9.977	58.484
7	3	1	traingd	adaptwb	*logsig*	*purelin*	learngdm	0.005130	0.0000500	10.000	58.901
8	3	1	traingd	adaptwb	*logsig*	*logsig*	learngdm	0.046430	0.0031300	10.000	62.897
9	3	1	traingd	adaptwb	*tansig*	*tansig*	learngdm	0.004470	0.0000400	9.995	59.937
10	3	1	traingd	adaptwb	*purelin*	*tansig*	learngdm	0.005300	0.0000300	220	1.306
11	3	1	traingd	adaptwb	*purelin*	*logsig*	learngdm	0.033500	0.0021300	3.265	17.874
12	1	1	trainlm	adaptwb	*tansig*	*purelin*	learngdm	0.016100	0.0000084	9.470	120.0
13	2	1	trainlm	adaptwb	*tansig*	*purelin*	learngdm	0.002066	0.0000068	4.969	120.0
14	3	1	trainlm	adaptwb	*tansig*	*purelin*	learngdm	0.0000943	0.00000154	53	0.548
15	3	1	trainlm	adaptwb	*logsig*	*purelin*	learngdm	0.0000956	0.00000167	56	0.554

and without any tendency of error accumulation due to the recursive model, numerically validating the adoption of this model along the research.

Therefore, a generalization test of the error surface in *Z* with the model 14 was carried out. Fig. 12 a and b shows the graphics of the original data average and average values generated by the generalization for values *X* versus *Z* and *Y* versus *Z* where we see an excellent fitting of the model to solve the problem.

5.4. Computational implementation

The computational implementation developed as shown in the flow chart presented at Fig. 2. It was done through the discretization of the values in points (matrix) of a correction surface on the CAD parametric software using the developed ANN model (as discussed in Section 5.2, see Fig. 9a). This cloud of points was mathematically converted in a surface *feature*, and using an API, it integrated the geometric forms and generated the corrected surface of the CAD/CAM model.

The main steps of this computational implementation were: (a) development of the mathematical model and calibration of the correction surface variables (*MatLab script* and database); (b) data integration by the CAD software (points cloud); (c) curve (*Spline)* creation through points; (d) development of curves on a parametric surface (*feature*); (e) posting the program to the CNC Machining Center.

6. Development of the applied experiment

Through the integration of the 3D solid in the CAD/CAM software, the machining reprogramming was done.

To simulate the real fabrication and validity process of the experiment, the same parameters of machining and tooling used in Section 4.2, were used. So, both sides of the dowel were machined again, one along the *X*-axis and one along the *Y*-axis, (seem Fig. 5a and b). After the fabrication of this piece, a MMC 3D CAI was carried out (see Fig. 5c), in order to verify which would be the new deviations in relation to the *Z*-axis (flatness) in the *X* and *Y* directions.

Therefore, through the CAI, the dimensional deviations (errors) of the *Z*-axis along the *X*- and *Y*-axes were collected again. These data were tabulated to verify the product deviations and compare them to the deviations presented previously. Fig. 13a and b shows this comparison of mean error (*X* versus *Z* and *Y* versus *Z*) along *X*- and *Y*-axes, respectively.

7. Conclusions

Concerning the ANN models studied, we could verify that MLP model has a simple architecture (3-1-1) and can exhibit excellent performance (Table 3). Still, it is observed that there is a connection with the work done by Zhang [39] and also described by Zain [33], suggesting some MLP network models as symmetrical formations and/or geometrical proportion. Then, they can be used as basis for future research to solve problems in manufacturing processes, for example: turning, electro-erosion, micro-machining, CNC oxygen cutting, and fast prototyping among others.

Concerning the practical experiments, we can perceive that the average deviations for *Z* along the *X* and *Y* axes showed to be significantly smaller (about 60%) than the reference values (see Fig. 13a and b), these results are better than those presented both by Shaowei et al. [13] as Gangwei et al. [14] and similar to those presented by Akafor and Yalcin [16], but with in this work a simpler methodology and ANN model were used. Also, can be inferred that the deviations existing in the product should come from other

sources, not approached in this research, such as: thermal effects, forces on the piece/tool, etc.

Furthermore, we can conclude that the experimental method used to obtain practical data allowed for excellent quality acquisition to carry out the statistical analysis and computational modeling, but in a median precision since the MMC 3D has a precision range similar to MF. Nevertheless, we could still validate the method proposed here because the activities flow provided the quantity and types needed for the research. This work also developed a new set of IMS techniques which can be used for this specific application and for others, according to each company's needs, or in the academic environment to carry out new studies and to improve systems, parameters, metrics or for future research associate other deviations existing in MT presented by other authors [11–17].

Finally, it is evident that the techniques proposed in this research allow for more competitiveness by insuring that the final product is in accordance with the respective tolerances and with smaller error. Thus, it contributes to provide better quality for the final product and the integration between the project and the manufacturing activities in a computational integrated environment.

Acknowledgements

The authors are grateful to CAPES Foundation, Ministry of Education (Brazil), for the graduate scholarship of W.O. Leite. Part of the work reported here is sponsored by CNPq and FAPEMIG Research Program.

References

[1] K. Patra, S.K. Pal, K. Bhattacharyya, Artificial neural network based prediction of drill flank wear from motor current signals, Appl. Soft Comput. 7 (2007) 929–935.
[2] Y. Karpat, T. Özel, Multi-objective optimization for turning processes using neural network modeling and dynamic neighborhood particle swarm optimization, Int. J. Adv. Manuf. Technol. 35 (2007) 234–247.
[3] Q. Zhang, M. Mahfouf, J.R. Yates, C. Pinna, Model fusion using fuzzy aggregation: Special applications to metal properties, Appl. Soft Comput. 12 (2012) 1678–1692.
[4] J.C. Campos Rubio, J.G. Duduch, Magnetostrictivemicropositioning device with fuzzy-neural-based controller, Int. J. Adv. Mechatron. Syst. 2 (3) (2010) 174–181.
[5] M. Rizal, J.A. Ghania, M.Z. Nuawia, C.H. Haron Che, Online tool wear prediction system in the turning process using an adaptive neuro-fuzzy inference system, Appl. Soft Comput. 13 (2013) 1960–1968.
[6] R.J. Lian, Intelligent control of a constant turning force system with fixed metal removal rate, Appl. Soft Comput. 12 (2012) 3099–3111.
[7] S.H. Huang, H.C. Zhang, Artificial neural networks in manufacturing: concepts, applications, and perspectives, IEEE Components, Packaging and Manufacturing Technology-Part I 17-2 (1994) 212–228.
[8] G. Boothroyd, P. Dewhurst, A.W. Knight, Product Design for Manufacture and Assembly, 2nd edition, CRC Press, Boca Raton, 2001, pp. 670.
[9] H.M. Arnold, The Machine Tool Industry and the Effects of Technological Change. Technology Shocks: Origins, Managerial Responses and Firm Performance, Springer-Verlag, New York, 2003, ISBN 3-7908-0051-1.
[10] J.P. Davim, A.E. Correia, Maquinagem a Alta Velocidade. Publindústria Edições Técnicas, Lisbon, 2006.
[11] H. Schwenke, W. Knapp, H. Haitjema, A. Weckenmann, R. Schmitt, F. Delbressine, Geometric error measurement and compensation of machines – an update, Manuf. Technol. 57 (2008) 660–675.
[12] W. Tian, B. He, T. Huang, Universal geometric error modeling of the CNC machine tools based on the screw theory, in: Fourth International Seminar on Modern Cutting and Measurement Engineering, Proc. SPIE 7997 (2011) 6.
[13] Z. Shaowei, D. Guofu, Q. Shengfeng, L. Jiang, Z. Li, Y. Kaiyin, Integrated geometric error modeling, identification and compensation of CNC machine tools, Int. J. Mach. Tools Manuf. 52 (2012) 24–29.
[14] C. Gangwei, L. Yong, L. Jianguang, G. Dong, Y. Yingxue, Geometric error compensation software system for CNC machine tools based on NC program reconstructing, Int. J. Adv. Manuf. Technol. 63 (2012) 169–180.
[15] R. Ramesh, M.A. Mannan, A.N. Poo, Error compensation in machine tools-a review. Part I: Geometric, cutting-force induced and fixture-dependent errors, Int. J. Mach. Tools Manuf. 40 (2000) 1235–1256.
[16] A.C. Okafor, M.E. Yalcin, Derivation of machine tool error models and error compensation procedure for three axes vertical machining center using rigid body kinematics, J. Emerald Integr. Manuf. Syst. 40 (2000) 1199–1213.
[17] R. Chana, P. Manukid, Geometric and force errors compensation in a 3-axis CNC milling machine, Int. J. Mach. Tools Manuf. 44 (2004) 1283–1291.
[18] G. Chen, J. Yuan, J. Ni, A displacement measurement approach for machine geometric error assessment, Int. J. Mach. Tools Manuf. 41 (2001) 149–161.
[19] R. Bogue, Design for manufacture and assembly: background, capabilities and applications, Assembly Autom. 32 (2) (2012) 112–118.
[20] J.G. Bralha, Design for Manufacturability Handbook, 2nd edition, McGraw-Hill, Boston, 1999.
[21] A.S.M. Hoque, P.K. Halder, M.S. Parvez, T. Szecsi, Integrated manufacturing features and design-for-manufacture guidelines for reducing product cost under CAD/CAM environment, Comput. Ind. Eng. 66 (2013) 988–1003.
[22] F. Cus, U. Zuperl, Approach to optimization of cutting conditions by using artificial neural networks, J. Mater. Process. Technol. 173 (2006) 281–290.
[23] V. Tandon, H. El-Mounayri, A novel artificial neural networks force model for end milling, Int. J. Adv. Manuf. Technol. 18 (10) (2001) 693–700.
[24] S.S. Panda, D. Chakraborty, S.K. Pal, Flank wear prediction in drilling using back propagation neural network and radial basis function network, Appl. Soft Comp. 8 (2008) 858–871.
[25] X. Yang, H. Kumehara, W. Zhang, Back-propagation wavelet neural network based prediction of drill wear from thrust force and cutting torque signals, Comp. Inf. Sci. 2 (3) (2009) 75–86.
[26] J.F. Briceno, H. El-Mounayri, S. Mukhopadhyay, Selecting an artificial neural network for efficient modeling and accurate simulation of the milling process, Int. J. Mach. Tools Manuf. 42 (6) (2002) 663–674.
[27] U. Zuperl, F. Cus, Tool cutting force modeling in ball end milling using multilevel perceptron, J. Mater. Process. Technol. 153–154 (2004) 268–275.
[28] U. Zuperl, F. Cus, B. Mursec, T. Ploj, A generalized neural network model of ball-end milling force system, J. Mater. Process. Technol. 175 (2006) 98–108.
[29] W.O. Leite, J.C. Campos Rubio, F. Mata Cabrera, J. Tejero Manzanares, I. Hanafi, Utilización de modelos de redes neuronales artificiales para predecir la influencia del tipo de fresado en la calidad del producto, DYNA-BILBAO 89 (2014) 457–466.
[30] S.R. Karnik, V.N. Gaitonde, J.C. Campos Rubio, A.E. Correia, A.M. Abrão, J.P. Davim, Delamination analysis in high speed drilling of carbon fiber reinforced plastics (CFRP) using artificial neural network model, Mater. Design 29 (2008) 1768–1776.
[31] R. Mishra, J. Malik, I. Singh, J.P. Davim, Neural network approach for estimating the residual tensile strength after drilling in uni-directional glass fiber reinforced plastic laminates, Mater. Design 31 (2009) 2790–2795.
[32] G. Maciej, B. Andres, The evolutionary development of roughness prediction models, Appl. Soft Comput. 13 (2013) 2913–2922.
[33] A.M. Zain, H. Haron, S. Sharif, Prediction of surface roughness in the end milling machining using Artificial Neural Network, J. Expert Syst. Appl. 37 (2010) 1755–1768.
[34] T. Erzurumlu, H. Oktem, Comparison of response surface model with neural network in determining the surface quality of moulded parts, Mater. Design 28 (2007) 459–465.
[35] C. Ahilan, S. Kumanan, N. Sivakumaran, J.E.Dhas. Raja, Modeling and prediction of machining quality in CNC turning process using intelligent hybrid decision making tools, Appl. Soft Comput. 13 (2013) 1543–1551.
[36] M. Correa, C. Bielza, J. Pamies, Comparison of Bayesian networks and artificial neural networks for quality detection in a machining process, Expert Syst. Appl. 36 (2009) 7270–7279.
[37] U. Çaydas, A. Hasçalik, A study on surface roughness in abrasive waterjet machining process using artificial neural networks and regression analysis method, J. Mater. Process. Technol. 202 (2008) 574–582.
[38] S.K. Gupta, D.S. Nau, A systematic approach for analysing the manufacturability of machined parts, Comp. Aided Design 27 (1995) 323–342.
[39] G. Zhang, B.E. Patuwo, M.Y. Hu, Forecasting with artificial neural networks: the state of the art, Int. J. Forecast. 14 (1998) 35–62.
[40] R.J. Schalkoff, Artificial Neural Networks, McGraw-Hill Companies, New York, 1997.
[41] M.T. Hagan, M.B. Menhaj, Training feed forward networks with the Marquardt algorithm, IEEE Trans. Neural Netw. 5 (6) (1994) 989–993.
[42] H. Demuth, M. Beale, Neural Network Toolbox, User's Guide, Version 4, The MathWorks, Inc., Natick, MA, 2004, revised for version 4.0.4 edition. http://www.mathworks.com
[43] H. Yu, B.M. Wilamowski, Levenberg–Marquardt training, in: Industrial Electronics Handbook-Intelligent Systems, 2nd edition, CRC Press, Boca Raton, 2001.
[44] H. Nakazawa, Principles of Precision Engineering, Oxford University Press, New York, 1994.
[45] J. Ye, J. Qiao, L. Ming-ai, J. Ruan, A Tabu based neural network learning algorithm, Neurocomputing 70 (2007) 875–882.
[46] T. Özel, J. Paulo, Davim, Intelligent Machining, John Wiley & Sons, London, 2009, ISBN 9781848211292.
[47] A. Kaur, Impact of training function based neural network on reusable software modules, Int. J. Comp. Sci. Inf. Technol. 3 (3) (2012) 4024–4027.