Computer Assisted Mechanics and Engineering Sciences, **16**: 143–160, 2009.

Application concept of artificial neural networks for turbomachinery design

Mark Azzam[1], Jan-Christoph Haag[2], Peter Jeschke[2]
[1]*Institute of Flight Systems and Automatic Control*
Technische Universität Darmstadt
64287 Darmstadt, Germany
[2]*Institute of Jet Propulsion and Turbomachinery*
RWTH Aachen University
52062 Aachen, Germany

(Received in the final form August 19, 2009)

This paper presents the results of an extensive investigation evaluating and improving the development of artificial neural network (ANN) models for turbomachinery design purposes. A set of 1100 differing axial compressor geometries based on 5 single-stage compressor rigs was prepared. Computations with the mean line analysis tool $AXIAL^{TM}$ took place to determine the according compressor maps defined by 15 operating points each. The challenge of ANN model development in terms of dimensionality reduction (feature selection), data normalization, defining the networks necessary plasticity, and network training is discussed using the example of three different models. As a result, the first model is able to predict the total pressure loss of the rotor blade row with a mean magnitude of the relative error (MMRE) of 3.6%. The second model predicts the total pressure ratio with an average accuracy of 0.8%. The third and last model was trained to predict basic geometrical parameters by presenting the load level and the performance data as an input. The achieved MMRE varied between 2.4% and 5.6% in respect of the particular output variable. The results show that ANNs are applicable to develop efficient models for turbomachinery design and analysis purposes, respectively.

1. Introduction

The importance of findings and developments within the field of neuroinformatics for an application in the field of turbomachinery has increased over the last years. Several research activities were conducted determining a great potential to improve and accelerate the design process. However, publications focus on the application of ANNs for design purposes while neglecting the challenge of developing and optimizing the prediction model. In this paper the authors present a standard procedure for ANN model development for turbomachinery design. Thus, first a short overview on ANN research work in turbomachinery is given followed by theoretical considerations of model development from a mathematical and neuroinformatical perspective.

Benini and Toffolo (2002) [4] demonstrate ANNs' (which are universal function approximators) ability to predict the performance of compressor cascades. For that purpose, the inlet Mach number as well as the stagger angle and the incidence were used to predict the outlet flow angle, the outlet Mach number and the static pressure ratio separately with three self-contained ANNs, to increase accuracy. Uelschen (2000) [39] applied and successfully trained an ANN to design airfoils for axial compressors. Nine boundary conditions, taking geometry and aerodynamic parameters into account, acted as input variables to determine an optimum set of further nine geometric and aerodynamic parameters of interest. A different approach to optimize compressor geometry was investigated by Ahmed (2005) [1]. Instead of training an ANN to predict optimum geometry values directly, the ANN was trained to review the large number of geometry descendants generated by a genetic algorithm. In case the ANN determined that a specific descendant (varying compressor geometry)

had a reasonable chance of improved performance, an accurate CFD simulation was performed. In this manner, the number of descendants to be reviewed could be enlarged tremendously, due to the time saving computation by an ANN compared to a CFD simulation. A very similar approach had already been successfully pursued by Van den Braembusche et al. [32, 33] to optimize radial impeller geometry. In their aproach candidate geometries are proposed by a genetic algorithm and their performance is estimated with an ANN model. After completing the first iteration of fast optimization the optimized geometry is verified with a 3D Navier-Stokes (NS) solver. In case the results deviate, an ANN model based on the new set of pattern is trained and again the geometry is optimized. The iteration ends as soon as the predicted performance agrees with the NS solver results. Centrifugal compressor design with an ANN is also the focus of Pazzi et al. (2002) [31] and Japikse et al. (2004) [25]. While in [31] there is a focus on the interaction between the geometry of the return channel and the performance, [25] uses various foremost geometry parameters to determine variables of design calculation models ("two elements in series" and "two zone" model). Both publications have limited significance, while in [31] there is no information on the volume of applied data (a very critical factor when modeling with ANNs), some of the predicted variables presented in [25] have an error of up to 86%.

The great advantage of ANNs is their ability to find and represent mathematical dependencies within data. While conventional solvers compute a solution to a specific problem using a predetermined algorithm, an ANN is able to learn the solution to a pattern of input data by training with the according data. Hence, an ANN reaches its limits of performance capability in respect of the data applied. This circumstance also demands that the data presented for training contains the necessary information to determine the output parameter(s) of interest. One of the main issues of applying artificial neural networks for turbomachinery design, results from the great number of explanatory variables. Unfortunately, too many governing parameters decrease the ANNs' ability to approximate the function of interest, [5, 6]. This issue can easily be understood if e.g. it is considered that the necessary volume of data exponentially increases with the increase of the input dimension (defined by the number of employed entirely independent explanatory variables) if a steady data density within the mathematical space shall be maintained. Bellmann (1962) [3] was the first to entitle this issue as the "curse of dimensionality". For example, if for a 2-dimensional input vector every dimension can be sampled sufficiently with 3 evenly-spaced samples, a data volume of 9 patterns is needed for a sufficient coverage of the input space. However, if an extra dimension is added to this vector and the additional dimension can also be sampled sufficiently with 3 evenly-spaced samples, the necessary data pattern increases up to 27.

Fortunately, ANNs are not as susceptible to the "curse of dimensionality". That is because input parameters are usually not entirely independent and correlate among one another, respectively leading to the coverage of a subspace instead of the full mathematical input space and additionally the explained variable usually does not vary arbitrarily from one input to another, [5, 6]. Still, the data density and the coverage of the mathematical space are crucial to an adequate approximation of a function and the volume of the data available is generally finite. Therefore, the number of input variables needs to be minimized and the dimension of the input vector needs to be as low as possible, respectively. This problem calls for an adequate preprocessing of the respective data. Two general approaches with good prospects are conceivable, *feature/variable selection* and *feature extraction*. The first approach minimizes the dimension of the input vector by reducing the input variables to a crucial amount. The selection algorithms usually allow to order the input variables according to their importance for the dependent variable. Carried out as a stepwise regression (based on multivariate statistical analysis) this method allows the selection of explanatory variables according to their ability to account for the variability of the dependent variable, by means of an automatic procedure. Unlike feature selection, feature extraction methods reduce the dimensionality of the input vector by searching for the low-dimensional representation (LDR) of the high dimensional space, [22]. This approach may apply auto associative bottle-neck Multi Layer Perceptrons (MLP) (auto-encoder), to perform dimensionality reduction. Since the number of neurons within the hidden layer is less than the number of input neurons (and the output neurons, respectively) the application

of such an ANN leads to a data processing comparable to a method of multivariate statistics, the *principal component analysis*. Although both approaches induce a loss of information in the input vector, they lead to an improvement of the obtained model, due to the limited training data. An overview on methods for dimensionality reduction is given in [17].

If the crucial explanatory variables have been determined, the ANN model can be developed. In order to obtain a sufficient ANN model several criteria and procedures ought to be considered. To demonstrate the procedure in this paper the research activity employed a large set of performance data. The data was obtained by means of varying the rotors of five actual single-stage axial compressors (referred to as basic configuration in this work) and performance calculations computed with the mean line analysis tool $AXIAL^{\mathrm{TM}}$. The computations determined the respective compressor maps, defined by 15 operating points each, for a record of 1100 differing axial compressor geometries. In this research work three different ANN models were developed. The first predicts the total loss coefficient and was therefore applied to remodel the semi-empirical loss-models for mean line analysis, see Sec. 3.2. Furthermore, the method stepwise regression for dimensionality reduction was evaluated during the model development as well as the importance of evenly sampled data widely covering the input space. The second predicts the pressure ratio and accents the potential of ANNs, as it is able to remodel the mean line flow analysis. And the third MLP was trained to predict general geometric parameters in respect of the desired operating point and further aerodynamic boundary conditions as an investigation on requirements of ANNs for the application as a design tool.

2. The multi layer perceptron

ANNs are information processing models of diverse complexity and construction. They are composed of artificial neurons (nodes) interconnected in a certain way to allow communication. Each neuron operates as a simple processor. Its ability is limited to a rule for processing the output o_i, of other neurons i connected with the respective neuron j, to an own input net_j and a transfer function to process this input net_j into an output o_j. This output can be propagated to further nodes k by means of weighted connections (w_{jk}). In addition to the terms of data processing by the single neurons, it is the network structure that has a significant impact on the character of an ANN. The Multi Layer Perceptron (MLP) which was applied for this research work is a feed-forward ANN, stating that the information presented is propagated through the network in strictly one direction, from the input neurons to the output neurons, [35]. Furthermore, the network nodes are ordered in different layers. There is an input layer, one or two hidden layers and an output layer. The input layer has no processing features and merely operates as an interface for the data input. Thus the number of nodes for the input layer (N) is equal to the dimension of the input vector $\mathbb{R}^N$ and the number of applied explanatory variables, respectively (the number increases in case of categorical variables, this is described in more detail in Sec. 4.2). The hidden layer processes the input data according to the weighted connections and the predefined transfer function. The optimum number of hidden layers and neurons (H) within the hidden layers cannot be determined a priori and is generally determined empirically. The output layer also consists of processing neurons and presents the networks output. Thus the number of neurons for the output layer (M) is defined by the dimension of the output vector $\mathbb{R}^M$ and the number of dependent variables, respectively.

The design of MLP neurons is quite simple. Neurons beyond the input layer compute their input from the weighted outputs of the prior nodes, connected with the respective node, by summation. Subsequently, the output of the neuron is computed by the transfer function:

$$o_j = a_j = f_{act}(net_j) = f_{act}\left(\sum_{i=0}^{I} w_{ij} \cdot o_i\right). \tag{1}$$

Different transfer functions can be applied in MLPs. The neurons within the input layer should apply a linear transfer function with a constant of proportionality of one. This specific function

is termed identity, due to the processed information being equal to the input information. The hidden layer nodes apply sigmoid transfer functions that are continuously differentiable, such as the logistic function or the hyperbolic tangent, to introduce nonlinearity to the model and to meet the conditions for most training-algorithms, e.g. "Backpropagation". The output layer nodes generally apply either linear or non-linear transfer functions.

The MLP is an ANN that is trained by supervised learning with a data record that contains the input pattern and the according output pattern (teacher). Given that the neurons are designed prior to network training and in respect of the applied transfer functions, the networks weights are the only adaptive parameters involved in the training process. The Backpropagation algorithm is commonly applied for MLP training and therefore was chosen for this research work. This numerical method is based on gradient descent; hence the gradient of the error function is calculated in respect of the modifiable weights by means of backwards propagating the error. In this way the randomly initialized weights w_{ij} are being adapted,

$$\Delta^p w_{ik} = -\eta \cdot \frac{\partial^p E(w)}{\partial w_{ik}}, \tag{2}$$

to reduce the error E and to minimize the error function, respectively:

The error function $E(w)$ accounts for the deviation of the computed output y at the output node k from the according output $\widehat{y}$ (teacher) for a pattern "p" and is defined as:

$${}^pE(w) = \frac{1}{2} \cdot \sum_{k=1}^{K} ({}^p\widehat{y}_k - {}^p y_k)^2. \tag{3}$$

Figure 1 displays single neurons in a three layer MLP (input, hidden and output layer). Their characteristics are assigned the manner of propagating information and information processing, [34, 40].

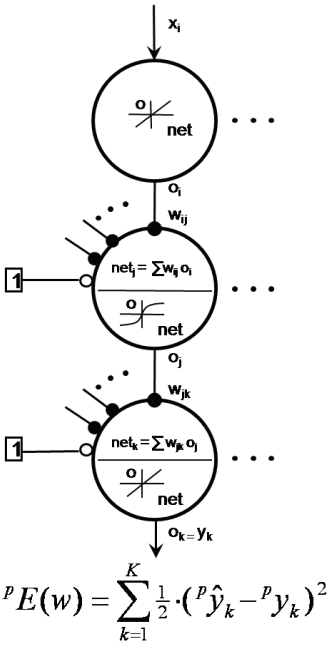

Fig. 1. MLP configuration

3. Generating the data

To generate a large database, performance calculations had to be executed for a wide range of compressor geometries. The applied software tool for this purpose was $AXIAL^{TM}$ by *ConceptsNREC* [20] which is based on a reduced-order through flow preliminary compressor design system [11]. This preliminary design system uses the equation of momentum in radial direction for the calculation of enthalpy, entropy and the velocities of the flow at the inlet and exit of the blade row (e.g. rotor inlet and rotor exit) by assuming inviscid flow in-between the bladed segment. Based on the simple radial equilibrium, only the influence by the circumferential velocity as the primary component of the centrifugal force is considered. The influence of the streamline curvature is neglected. With respect to a better representation of the compressor performance, modified loss models (for airfoils and for endwalls as well as for stall and for deviation) were used. The advantage of this calculation method is its ability to predict aerodynamic performance parameters in the first approximation and its marginal computational effort. Allowing to embed empirical information on a large scale, mean line analysis is sufficiently accurate to design main geometric parameters on a preliminary basis, [8]. For further information on the theory of axisymmetric annulus design for compressors please see [11, 24]. To approximate the efficiency of a compressor several semi-empirical loss models for different loss sources can be found in literature. The most commonly applied models are implemented in $AXIAL^{TM}$ and the choice for applying a specific model to sufficiently predict efficiency is dependent on the analogy between the compressors (and operating points, respectively) contained in the empirical set of data on which the model is based and the target compressor design. An overall evaluation of divers models for the design loss and off-design loss (as well as models for the determination of minimum loss incidence and deviation angles) and an improved off-design correlation is presented in [9]. The loss model configuration applied for performance computations in this research work was chosen consistently for all computations and independently of the compressor geometry to achieve a uniform set of data with equal governing parameters and equal associations between input and output for network training. The concrete selection was made in respect of the best response to the measured performance data for all basic configurations.

3.1. Axial compressor geometries

In this chapter the five single-stage axial compressors in their basic and varying configurations are discussed. The basic geometric configurations and the according compressor maps were taken from the literature. Due to missing information some of the parameters had to be assumed. The tip clearance was set to 0.3 mm uniformly for all rotors, the blade surface roughness was assumed $1.6 \cdot 10^{-3}$ mm. The compressors were: E/CO-3, [15]; E/CO-4, [12]; KOM-130, (compressor at the Institute of Jet Propulsion and Turbomachinery); NASA Rotor 37, e.g. [38] and [10]; first stage of BBC/Sulzer, [19].

From these basic configurations two data sets were created. The first data set contained 132 derived geometries for each of the compressors. Prior to varying the 43 different geometrical parameters, a minimum and maximum geometry configuration for each of the compressors was determined. Subsequently, every single parameter adopted the minimum, basic and maximum configuration value independently for all three configurations and five compressors. Hence, the first data set consisted of 660 varying compressor geometries. The parameters that were varied (some of them independently for the *hub-*, *mean-* and *tip position* as well as for the *leading-* (LE) and *trailing edge* (TE)) are airfoil type, chord length, max. blade thickness, blade angle, axial width, LE/TE thickness, hub to tip ratio, hub radius, and blade number. The second data set was created in much the same way, yet only two configurations were the basis for a further variation of geometrical parameters. These two configurations consisted of mean values between the minimum and basic configuration and between the basic and maximum configuration. The second data set was therefore made up of 440 varying compressor geometries. Since the compressor maps computed for every compressor

geometry contained 15 operating points each, the total number of the first set added up to 9900 tuples and the second data set contained 6600 tuples. The distinction between the two data sets was necessary to point out the importance of widely covering the input data space to allow the MLP to generalize. This issue will be discussed in detail in Sec. 4.3.

3.2. Applied loss models

The loss models with significant impact on the total loss coefficient (according to the results presented by the $AXIAL^{TM}$ performance calculations) which were applied for the computations are as follows.

"Blade Profile Loss" model by C. C. Koch and L. H. Smith, Jr. (1976) [26]: This model accounts for "[...] blade profile losses due to surface diffusion and trailing edge thickness [...]" and thus determines the total loss coefficient ω_T^* at minimum loss incidence i^*. It is based on compressible boundary layer theory and is an extension of the low-speed correlation presented by Lieblein (1956/1959) [27, 29] "[...] into the Mach number and Reynolds number regions that are of interest to compressor designers." The profile loss is predicted by application of the suction surface diffusion ratio which is associated with the cascade geometry, blade thickness, annulus contraction and compressibility effects as well as corrections taking the Mach number, Reynolds number and stream tube height ratio into account.

"End-Wall Loss" model by C. C. Koch and L. H. Smith, Jr. (1976) [26]: This model presents an approach to predict efficiency decrease resulting from end-wall effects. For that purpose two properties of an end-wall boundary layer, the displacement thickness and the tangential-force thickness are related to the according efficiency loss. In this manner the displacement thickness accounts for the resulting mass flow deficit and the tangential force thickness represents the reduction of the tangential component of the blade force, due to the presence of the end-wall boundary layer.

"Off-Design Loss" model (AGARD Report No. 745 (1987) [9]): The total loss coefficient for off-design loss is determined by application of simple polynomials plotted as an array of curves,

Table 1. Measures of the loss models

Variables (Loss Models)	Abbreviation
Maximum Blade Thickness	tmax
Chord Length	chord
Radius (in)	R.in
Radius (out)	R.out
Camber Angle (in)	Phi.in
Camber Angle (out)	Phi.out
Blade Angle (in)	Beta_B.in
Blage Angle (out)	Beta_B.out
Blade Height (in)	H.in
Blade Height (out)	H.out
Solidity	Solidity
Stagger Angle	StaggerAng
Annulus Area (in)	Aann.in
Annulus Area (out)	Aann.out
Blade Axial Gap	BladeAxGap_Basis
Foil Type	FoilType
Roughness	Roughness
Relative Flow Velocity	W
Reynolds Number	RE
Relative Mach Number (in)	Mrel.in
Incidence Angle	Inc
Deviation Angle	Dev

based on empirical data. While data shows that the airfoil geometry and the inlet Mach number have significant impact on the total loss coefficient at off-design incidence, the cascade geometrical parameters have no systematic influence. Hence, the polynomials present the difference between the total loss coefficient and the minimum total loss ($\omega_T - \omega_T^*$) against the difference of the incidence angle and the minimum loss incidence ($i - i^*$), in respect of the airfoil and the inlet Mach number.

The angles "incidence" at minimum loss i^* and "deviation" at minimum loss δ^* used in the loss models were calculated according to Lieblein (1960) [28]. They substitute the actual flow angles at the blade and allow a comparison between different airfoils. It is necessary to say that these variables were considered as possible input variables and not to be determined by the MLP model, to reduce the model's complexity.

A summary of measures applied in these loss-models to determine the total loss coefficient is displayed in Table 1. Hence, these measures complemented with all other geometrical parameters were considered possible input variables for the MLP model development.

After the two data sets were computed a plausibility check was performed removing all tuples that had a total pressure loss coefficient not within the boundaries indicated by the AGARD Advisory Report No.745 [9] ($0 < k_{\text{overall}} < 0.4$). This reduced the first data set to 6988 tuples (from 9900) and the second data set to 4868 (from 6600).

4. Procedural method for model development

After a sufficient volume of data is available for model development, the presented procedural method is performed, see Fig. 2. First, the dimension of the input vector is reduced by reduction

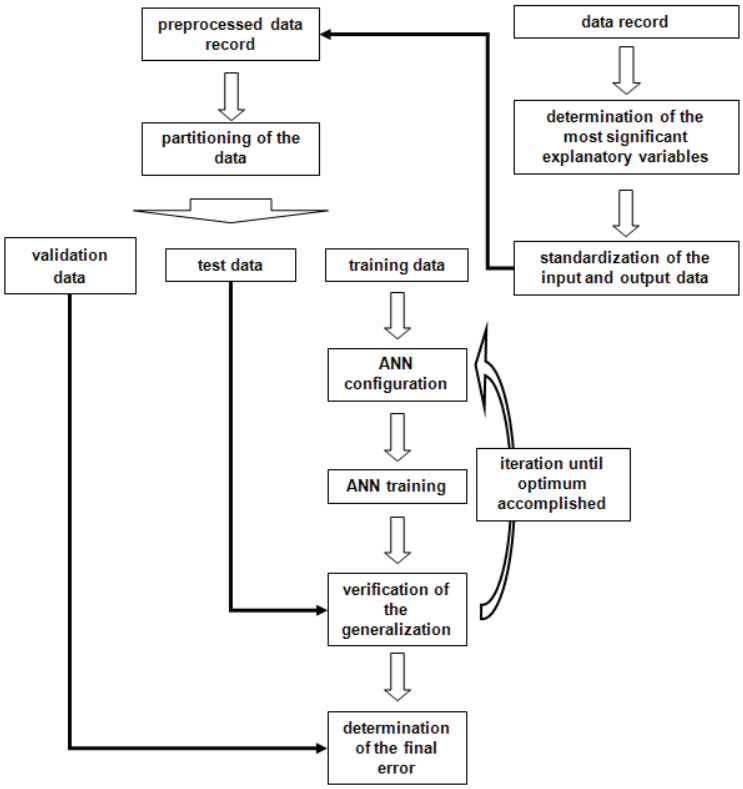

Fig. 2. Procedural method for model development

of the explanatory variables to a crucial amount by means of a stepwise forward regression. Thereafter, the data is standardized to a predefined interval, to improve the training process in respect of computing time and the achieved model quality. These first two steps are referred to as data preprocessing which is necessary to enable an efficient network development. The next step is to divide the data volume into training-, test- and validation data. This allows an evaluation of the training process, leading to the achievement of an optimum MLP model. In a final step a large number of network trainings need to be performed with varying initial weights, learning constants and neuron quantities to determine the optimum network topology and minimal achievable error and furthermore assure the ability of generalization.

The detailed theoretical background and the procedural approach of this method for model development will be presented in the following.

4.1. Data preprocessing

Stepwise regression

The following stepwise regression model (forward approach) presents a statistical approach to select explanatory variables by means of an automatic procedure, [18, 36, 37]. Based on the linear regression it is a classical method in statistics to model the relation between the independent variables $x_{0,1,\ldots,n}$ and the dependent variable $\widehat{y}$:

$$\widehat{y}_p = y_p + \epsilon_p = \beta_0 + \sum_{n=1}^{N} (\beta_n x_{n,p}) + \epsilon_p. \tag{4}$$

For this purpose the regression parameters $\vec{\beta} = \beta_{0,1,\ldots,n}$ are determined in the manner that the remaining error ϵ_p (residual) between the actual dependent variable $\widehat{y}_p$ (teacher) and the calculated dependent variable y_p is minimized, by partial differentiation, for every tuple p. This approach follows the method of least squares. To evaluate how well a regression model approximates the given dependent variable, the coefficient of determination can be applied. In consideration of the variance decomposition, the coefficient of determination accounts for the explained variance y of the model in respect of the total variance of $\widehat{y}$. However, since the coefficient of determination does not provide any information on the significance of the features selected or possible correlations between the various input variables it does not support the feature selection. Hence, the stepwise regression applies a sequence of F-tests to check statistical significance. This test allows selecting features by an automatic procedure. The procedure starts with no variables included in the model and determines the first variable to include under consideration of the coefficient of determination. In the following step a further variable is added and a test of significance is performed to prove the significance of the included variable. Furthermore, the remaining variables are tested in respect of the potential to exclude individual variables from the model, without increasing the residual sum of squares. In this manner the potential explanatory variables are considered one by one to verify their impact on the dependent variable. Restrictively, it has to be mentioned that stepwise regression cannot substitute knowledge in the accordant field of expertise. It can only support the user by reducing the dimensionality of the input vector. First of all the degree of the polynomial for the regression model needs to be predefined (in this research the degree was unity). Without a doubt this constraint leads to an insufficiency concerning the ability for automatic feature selection, due to simultaneously defining the relation between dependent and independent variables a priori. And secondly, the hypothesis created and the tests of significance have to be performed with the identical set of data if the data volume is too small to randomly partition it into two subsets. This aspect can possibly lead to relations found by chance instead of by evidence.

In this research work stepwise regression was applied by means of the *SPSS*$^{\text{TM}}$ data-mining tool *Clementine*$^{\text{TM}}$ (meanwhile *PASW Modeler*$^{\text{TM}}$) [21], to select the input parameters for the MLP models predicting the total loss coefficient and the total to total pressure ratio. The results

presented for determination of the total to total pressure ratio were adopted one-to-one as an input vector for the MLP model development. Yet, the results achieved to predict the total loss coefficient, were compared to the explanatory variables identified on the basis of the loss models which were applied for data generation. Hence, two different sets of explanatory variables were compiled; the first following the findings of the stepwise regression but excluding 4 of the 17 most significant parameters due to scientific considerations and the other excluding 9 parameters and adding 5 variables not determined by the stepwise regression although considered essential for an optimum model. MLP models with the two different input vectors were developed to evaluate the potential of stepwise regression for feature selection.

4.2. Normalizing the data

In terms of preprocessing, normalizing the data was proven to be helpful or even necessary, depending on the transfer function applied to the networks neurons. If sigmoid transfer functions are applied for the neurons of the output layer, normalizing is obviously necessary to assure that the predicted data lies within the co-domain of the transfer function. But Barron (1991) [2] and Jansen (1995) [23] could prove that normalizing the output data is recommendable by all means. Also regarding the input, normalizing the data improves the model and reduces computing time. This is not only due to removing the problem of initial weights dependence on the input scale to avoid saturation with sigmoid transfer functions. Normalizing also reduces the chance of disrupting the training process due to getting caught in local sub optima. For MLP networks no rules regarding the normalization can be put forth, yet indicators can be stated. The domain of a sigmoid transfer function is an indicator which, in consideration of the initial weights and the number of input values, helps to declare the range for normalized values to avoid saturation. That is to say that the values should not add up to an input net_j that lies within the saturation domain of the graph.

Next to the applied transfer function also the levels of measurement indicate how to normalize (or code) the data and how to present a feature to the MLP in training, respectively, [30].

Nominal measurement. The parameter values of categorical variables are presented to the network by a "one-of-n" coding or equilateral coding. For the first approach each parameter value activates a different neuron; the latter approach is similar, yet one parameter value does not activate any neurons.

Ordinal measurement. n-1 input nodes are necessary for n parameter values of an ordinal variable. According to the order of the attribute presented to the MLP, between zero and all neurons are activated.

Interval/ratio measurement. The parameter values are normalized to a defined interval and presented to the network via a single neuron. While there are several algorithms for normalizing the interval measured data, in some cases it might be suggestive to maintain the zero point for the ratio measured data.

The normalization performed in this research work was performed irrespective of the level of measurement (airfoil is a categorical variable), due to no noticeable improvement of the MLP models, and to reduce computing time. The normalization was executed by the following equation:

$$V^* = \frac{V^*_{\max} - V^*_{\min}}{V_{\max} - V_{\min}} (V - V_{\min}) + V^*_{\min}. \tag{5}$$

In this equation V is the variable that is normalized, and max and min indicate the two extreme values of the variable. The variables indicated with * determine the interval after normalizing

In this research work the * interval was set to $[-1.4; 1.4]$ for the input variables and $[-0.9; 0.9]$ for the output variables.

4.3. Partitioning the data

Two separate data sets were created (in the following termed approach I) for training and testing (+validation) purposes. Test data is necessary to verify the models capability of generalization (the models' ability to represent patterns not presented during training), [5, 7, 30]. The training data includes the performance computations for 660 differing compressor geometries yielding in 9900 (6988 after plausibility check) training patterns under consideration of 15 loading levels per geometry. The test data includes the performance computations for 440 differing compressor geometries adding up to 6600 (4868 after plausibility check) test patterns which were divided into test and validation data by a ratio of 9:1 subsequently. The validation data set is necessary, due to the test data slightly introducing to the MLP model by means of applying the graph of the validation error in the network development procedure (illustrated in the following chapter). Hence, the additional validation data is indispensable to assure a purely independent model validation. In a second approach (in the following termed approach II) the two data sets were combined and the training, testing and validating datasets were compiled randomly at a ratio of 67:30:3. A comparison of the models trained with these differently compiled data sets shows the impact of a wide coverage of the input space.

Please note: Bishop (2006) [6] suggests cross-validation for model validation if data volume is highly limited. This approach requires partitioning the patterns P into S fractions (with S equaling P at max (leave-one-out technique)). $S-1$ fractions are employed for training while the remaining fraction validates the trained model. This procedure is repeated S times to validate in respect of all fractions. Although this method does not assure a purely independent validation it presents a powerful method if the data volume is highly limited as for measured compressor data or CFD computed data.

5. Developing the mlp

To the course of developing an MLP model several properties need to be defined. The network structure has to be determined in respect of the applied transfer functions and the network topology (number of neurons for the hidden layer(s) as well as the number of layers). Additionally the most suitable learning constant and weight initials need to be identified.

Three different MLP models were developed using the *Java* based version of the *Stuttgart Neural Network Simulator* (*JavaNNS*) [13, 41]), to predict the total loss coefficient, the total to total pressure ratio and general geometrical parameters.

The first step when developing a MLP model is defining which transfer functions to apply for neurons of hidden and output layer. Although various functions can be applied, the following considerations will indicate potential advantages for particular functions. For the neurons of the hidden layer the hyperbolic tangent has a great advantage compared to the logistic transfer function when employing the Backpropagation training algorithm. Since the weight adaption responds to the output of a node (o_j) multiplicatively in training, even a highly negative activation leads to an adaption with the hyperbolical tangent, while the adaption converges to zero for the logistic transfer function. Hence the hyperbolical tangent can reduce computing time significantly when applied. For the neurons of the output layer a linear transfer function can additionally reduce computing time. If a teacher is presented with a parameter value close to saturation, large weight values are necessary to represent this output. Yet, when applying a sigmoid transfer function, values close to saturation lead to a very time-consuming weight adaption in the training process, due to the marginal slope of the curve progression in this area. Thus, a linear transfer function for the neurons of the output layer can reduce computing time additionally. The MLP models developed in this work uniformly employed the hyperbolic tangent for the neurons in the hidden layers and linear transfer functions for the neurons of the output layer.

The second step is to determine the quantity of neurons within the hidden layer(s) and the amount of weights, respectively. These define the plasticity of an MLP, which corresponds to the MLPs ability

to model the given function in respect of complexity and stored information, [34]. Hence, too many neurons lead to an overfitting of the model in training while too few nodes anticipate that the MLP is able to learn the function enclosed in the data. Since the complexity of the function is generally unknown the amount of nodes has to be estimated or determined empirically. Goerke (1997) [16] suggests a purely empirical approach starting with one hidden layer and 10-20 hidden neurons. Following, neurons are added stepwise until an optimum is found. However, Masters (1993) [30] suggests the "geometric pyramid rule" to determine the number of neurons (H) for a model in respect of the number of hidden layers and the dimension of the input (N) and output vector (M). For a single hidden layer:

$$H = \sqrt{N \cdot M}. \tag{6}$$

And for two hidden layers:

$$H_1 = M \cdot \left(\frac{N}{M}\right)^{\frac{2}{3}}, \tag{7}$$

$$H_2 = M \cdot \left(\frac{N}{M}\right)^{\frac{1}{3}}. \tag{8}$$

However, the authors point out that this approach can underestimate the necessary networks plasticity if the input or output vector is of low dimensionality. When increasing the plasticity of a network, the remaining initial weights are suggested to be held constant if single neurons are added to a network, to allow comparison. To determine if a network topology offers a sufficient plasticity without supporting overfitting, the error graph can be employed. Displaying the sum of square error against the epoch (single cycle in the teaching process in which all patterns are presented to the network) for the training and test data, a comparison of the two graphs allows to verify the networks generalization and its ability to model the function. In this research work the application of the "geometric pyramid rule" led to an underestimation of the necessary networks plasticity in this research. Therefore a steady capacity enhancement was performed, leading to the three optimum network topologies consisting of 10 to 19 hidden neurons divided onto two layers. MLP networks with two hidden layers generally led to a better performance in respect of the problems dealt with in this work. Furthermore, investigations showed that the optimum was not restricted to an explicit quantity but was rather ambiguous in a range of $\pm 2 - 3$ nodes. As a resume of the investigations into network plasticity, it can be stated that a combination of the geometric pyramid rule (for initial network capacity determination) and a stepwise capacity enhancement proved to be best.

Figures 3, 4 and 5 display the error graphs for three different network topologies and the according capacities. It clearly reveals that while the MLP displayed in Fig. 3, as a single hidden layer network, has a too low plasticity to sufficiently model the function. The network in Fig. 4 leads to overfitting which causes an increased error in the test data compared to training data. Figure 5 shows the error graph for the network with an optimum topology determined for the respective input and output vector. The training process was discontinued after 2400 epochs (early stopping) to avoid overfitting. Hence this topology also indicates a plasticity too large; still topologies with reduced neuron quantities lead to impairment rather than improvement. Early stopping is always an option if the optimum network plasticity cannot be determined precisely. It demands to discontinue the training process as soon as the testing error graph has reached its minimum.

Before a network can be trained with the according data, the weights need to be initialized randomly. The initial weights constitute the initial function modeled by the MLP. Hence, they are essentially responsible for the success of the training process, due to defining the initial network error and the adaption necessary to find an optimum. For reasons of saturation and the associated computation time, they should also be limited to a small interval (e.g., $[-0.3; 0.3]$). Numerous iterations with random weights are generally necessary to achieve an optimum model (for the network to find a sufficient minimum to the error function).

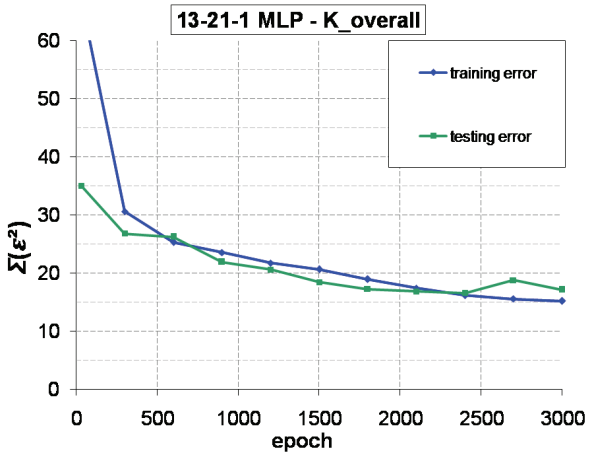

Fig. 3. Error graph 13-21-1 network topology

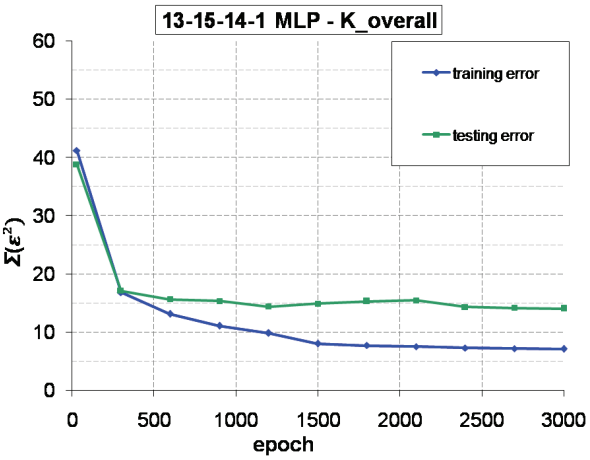

Fig. 4. Error graph 13-15-14-1 network topology

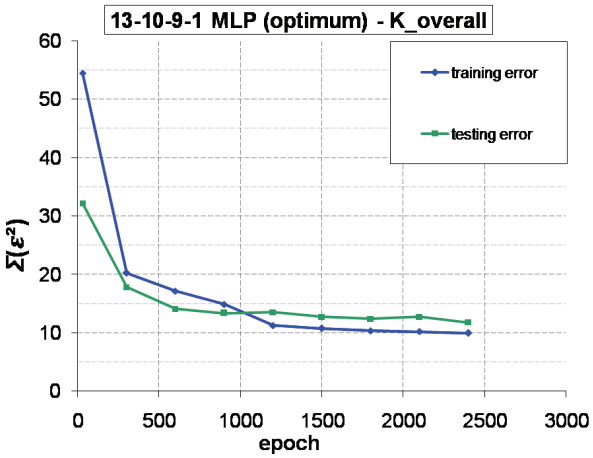

Fig. 5. Error graph 13-10-8-1 network topology

The subsequent training process can be performed either in the manner of batch learning or single step learning. While the weight adaption is carried out after presenting all training patterns to the network for batch learning, an adaption is carried out after the presentation of every single pattern for single step learning. The great advantage of single step learning is a significant computing time reduction, due to increasing the iterations of weight adaption by the factor P compared to batch learning. However, if single step learning is employed, the patterns need to be shuffled randomly before every epoch to ensure efficient learning, especially if the data volume is little, [16]. In this work the training process was performed by means of single step learning with random pattern shuffling.

Furthermore, the learning constant η needs to be defined for the training process. It defines the step length of a single iteration toward the negative direction of the gradient. If the constant is chosen to high an optimum might be missed during training, yet if the constant is set to low, the training process might end up in a sub-optimum or even be disrupted on a plateau of the error function. Investigations showed training improvement when decreasing the learning constant stepwise. For the work at hand the initial value was set to $0.03-0.04$ and was reduced stepwise to a value of approx. 0.01.

To avoid possible excessive oscillations in narrow minima of the error function or disruption of the teaching process in sub-optima, several sources suggest *Backpropagation with momentum term*. However, according to the findings of the present investigations, initializing the weights iteratively was more efficient than adjusting the momentum term and led to equal results.

6. Results

To evaluate the achieved models three statistical measures are employed for each of the three data sets: the mean magnitude of the relative error (MMRE as the measure of central tendency) and two measures of dispersion, the mean absolute deviation (MAD) of this error and the $\alpha = 99\%$-percentile (P_{99}) of the error distribution, [14]. In this manner statements can be made concerning the MLP's ability to model the according function, its ability of generalizing and the reliability of the model.

6.1. MLP model to predict the total loss coefficient

The model developed to predict the total loss coefficient is a 13-10-9-1 MLP. This network configuration was trained with patterns of two different input vectors (No. 1 and No. 2) as displayed in Table 2 and with two different compilations of training-, test- and validation data (approach I, approach II). The results displayed in Table 3 clearly reveal that the MLP trained with the independent variables according to feature selection No. 1 is less accurate in loss coefficient prediction than the model trained with feature selection No. 2. Hence the stepwise regression did find several of the most significant independent variables but yet not all. This comparison reveals that a statistical approach cannot substitute scientific expertise but can support feature selection. The reason for error values of the same order is made clear if correlations among the input parameters are considered. Although only 8 features were equal between the two input vectors (No. 1 and No. 2), several more were strongly correlated. With 3.6% the achieved MMRE was acceptable and the verification with the test and validation data proved reliability. Still, the high value of the percentile P_{99} (26%) reveals that several patterns could not be represented well by the MLP model. This irregular performance can be understood if it is kept in mind that about 30% of the training data was dismissed, due to the plausibility check. Therefore the density of the data space was very irregular and the MLP could not adapt to all patterns equally well in the training process, since some geometries were presented 15 times (once for every operating point) per epoch and others only once (14 patterns dismissed, due to plausibility). Also a comparison of the results achieved with the two different data compilations indicates the importance of training data covering the input space. The model

Table 2. MLP model to predict the total loss coefficient

Sequence of inclusion of input variable after multiple regression	**Exclusion of variable by insignificance**	**input variable No. 1 (K_overall)**	**input variable No. 2 (K_overall)**
(t_mx/Chrd)(m)∼E.1		(t_mx/Chrd)(m)∼E.1	(t_mx/Chrd)(m) ∼E.1
Inc(m)∼E.1		Inc(m)∼E.1	
Mrel(h).in∼E.1		Mrel(h).in∼E.1	Mrel(h).in∼E.1
R.out∼E.1		R.out∼E.1	R.out∼E.1
R.in∼E.1		R.in∼E.1	R.in∼E.1
inc-inc_opt		inc-inc_opt	inc-inc_opt
Solidity(m)∼E.1		Solidity(m)∼E.1	Solidity(m)∼E.1
Asp_Ratio∼E.1		Asp_Ratio∼E.1	Asp_Ratio∼E.1
Solidity(h)∼E.1			
Widt_Thrt(m)∼E.1		Widt_Thrt(m)∼E.1	
m.norm∼D		m.norm∼D	m.norm∼D
M-relativ			
N-relativ			
Hin/Hout_E.1		Hin/Hout_E.1	
Mrel.(t)∼E.1			
Athroat∼E.1		Athroat∼E.1	
	m.norm∼D		
BetaB(h).in∼E.1		BetaB(h).in∼E.1	
			Re(m).in∼E.1
			StaggerAng(m)∼E.1
			Profil
			Aann.out∼E.1
			Aann.in∼E.1

Table 3. Results of MLP model for the prediction of the total loss coefficient

model			**total loss coefficient (K_overall)**		
selected input variables			No. 2		No. 1
network topology			13-10-9-1	13-10-9-1	13-10-9-1
data partition			approach I	approach II	approach I
data set	**error**				
training data	**mean magnitude of the relative error**	[%]	**3.6435**	**3.7296**	**6.8583**
	mean absolute deviation of the error	[-]	3.0738	3.0742	5.4539
	alpha 99% - percentile P99 of the error	[%]	26.6217	21.5970	39.4445
test data	**mean magnitude of the relative error**	[%]	**3.4742**	**4.0031**	**6.3647**
	mean absolute deviation of the error	[-]	2.7287	3.3166	5.3556
	alpha 99% - percentile of the error	[%]	21.6575	23.9486	35.2057
validation data	**mean magnitude of the relative error**	[%]	**3.1491**	**4.4384**	**6.0244**
	mean absolute deviation of the error	[-]	2.4312	3.6529	4.8069
	alpha 99% - percentile of the error	[%]	18.6363	31.8824	35.3407

trained with data from approach I has a superior ability to generalize indicated by lower errors for test and validation data. Not only the MMRE but especially the MAD of this error and the P_{99} of the error distribution are far lower. That is because while in approach I all test and validation data was within the boundaries of the training data, the training data in approach II was picked randomly and hence some of the tuples in the test and validation data was out of the trained range. These findings are in agreement with the statements found in the literature that MLPs are able to interpolate very well while having difficulties to extrapolate.

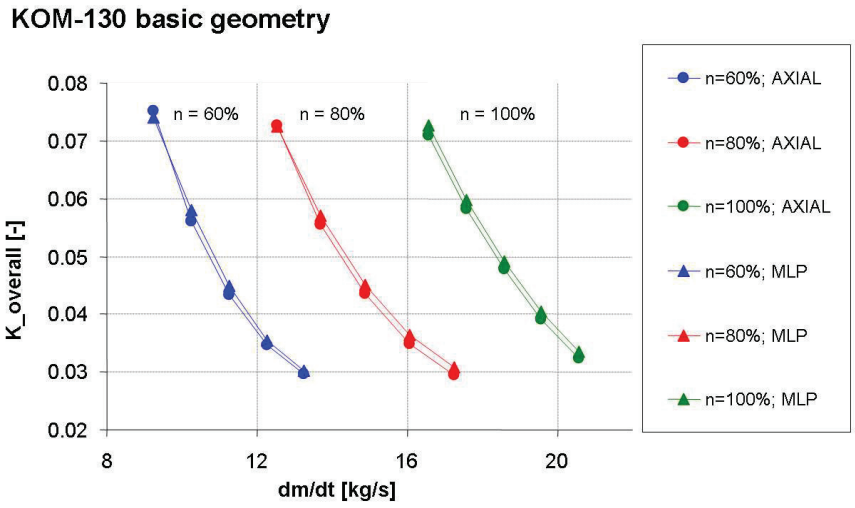

Fig. 6. Results of total loss coefficient by the $AXIAL^{TM}$ solver and the MLP model

To exemplify the achieved results the total loss coefficient determined by $AXIAL^{TM}$ and the coefficient predicted by the MLP model are displayed in a graph against the mass flow for three constant speed lines (60%, 80% and 100%) for the basic geometry of the Compressor KOM-130, see Fig. 6.

6.2. MLP model to predict the total to total pressure ratio

The model developed to predict the total to total pressure ratio is a 13-5-5-1 MLP. The results displayed in Table 4 show very high prediction accuracy. And again the test and validation data indicate reliable results. The fact that the ratio of the MMRE to the P_{99} percentile is comparable to the total loss coefficient model, confirms that irregular data density is the cause.

Table 4. Results of MLP model for the prediction of total to total pressure ratio

model	**total pressure ratio (PR_tt)**		
network topology			13-5-5-1
data partition			approach I
data set	**error**		
training data	**mean magnitude of the relative error**	[%]	**0.8216**
	mean absolute deviation of the error	[-]	0.6999
	alpha 99% - percentile P99 of the error	[%]	5.8005
test data	**mean magnitude of the relative error**	[%]	**0.9798**
	mean absolute deviation of the error	[-]	0.8870
	alpha 99% - percentile P99 of the error	[%]	7.4345
validation data	**mean magnitude of the relative error**	[%]	**0.9235**
	mean absolute deviation of the error	[-]	0.8193
	alpha 99% - percentile P99 of the error	[%]	7.6526

Figure 7 displays the compressor map of the measured data in comparison with the $AXIAL^{TM}$ computations and the MLP model predicted values for three speed lines (60%, 80% and 100%) for KOM-130. The authors would like to note that the agreement for the basic geometry of the KOM-130 is above average prediction model performance.

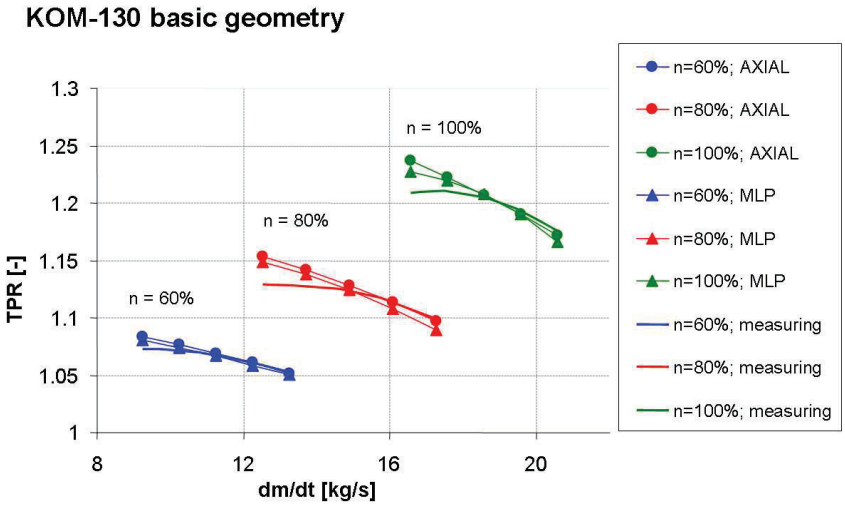

Fig. 7. Compressor map of KOM 130 comparing the $AXIAL^{TM}$ solver results and the values predicted by the MLP model

6.3. MLP model to predict basic geometrical parameters

The results achieved with the MLP model to predict basic geometrical parameters are not quite uniform. The results for the three data sets can be gathered from Table 5. While the radius can be predicted with high accuracy (MMRE = 2.4%), the error for blade height prediction is more than twice as high (MMRE = 5.6%). The explanation is reckoned to lie in the mathematical property of the modeled functional dependency. Since output and input of the actual functional dependency was exchanged to train the MLP, the performed approximation is of inverse kind. Yet, to allow the modeling of an inverse function, the actual function needs to be bijective. While it can be assumed that the function is surjective, it is quite likely that it is not injective within the whole domain for some input parameters, which would lead to an error increase. To respond to this assumption doubtlessly, an extensive investigation on the actual function is inevitable but no part of this research.

Table 5. Results of the three data sets (training, test and validation data)

model			**geometrical variables**				
data partition			approach I				
topology			6-9-9-5				
output variables			**R.In**	$\frac{t\_mx}{Chrd}$ **(m)**	**Solidity (m)**	**Chord (m)**	**Hblade.in**
data set	**error**						
training data	**mean magnitude of the relative error**	[%]	**2.4218**	**2.8529**	**4.5034**	**2.9525**	**5.6207**
	mean absolute deviation of the error	[-]	1.8053	2.3634	4.1521	2.4267	4.5178
	alpha 99% - percentile P99 of the error	[%]	11.9634	18.6227	27.2635	18.3575	39.7277
test data	**mean magnitude of the relative error**	[%]	**2.8035**	**2.1868**	**3.6518**	**2.5099**	**5.8016**
	mean absolute deviation of the error	[-]	1.7676	1.8435	2.8553	1.9173	4.1635
	alpha 99% - percentile P99 of the error	[%]	11.1914	12.1432	20.4341	13.0829	31.2635
validation data	**mean magnitude of the relative error**	[%]	**2.6854**	**2.1613**	**3.7071**	**2.5010**	**5.5536**
	mean absolute deviation of the error	[-]	1.6063	1.8888	2.8569	1.9413	3.7340
	alpha 99% - percentile P99 of the error	[%]	10.6027	11.8212	19.7331	12.2484	25.9722

7. Conclusion

The objective was to present a concept for ANN model development in respect of turbomachinery design purposes and under consideration of the high dimensional input space as well as computing time reduction. Hence, this paper contributes to assessing the potential of ANNs in the field of turbomachinery. The approach was to evaluate different procedures by means of a large data volume computed with the mean line analysis tool *AXIAL*$^{\mathrm{TM}}$ on the basis of five empirically tested single stage compressors. The results show that stepwise regression can support feature selection, yet cannot substitute knowledge in the respective field of expertise and application. Further research is needed to investigate the potential of feature selection and feature extraction methods. The MLP models developed show in average good performance with high accuracy. The suggested stringent procedure yields optimum generalization of the MLP models. The error distribution indicates the importance of high data density in an evenly covered mathematical space to achieve good overall performance with a narrow error distribution. The results presented in this paper applying MLPs to predict basic geometrical parameters indicate that further investigations into the application of ANNs as a design tool for turbomachinery need to be conducted under consideration of the inverse mathematical problem.

Acknowledgement

The Authors would like to thank *SPSS*$^{\mathrm{TM}}$ for providing the data-mining tool *Clementine*$^{\mathrm{TM}}$ (meanwhile *PASW Modeler*$^{\mathrm{TM}}$) for scientific use.

References

[1] R. Ahmed. *Auslegung vielstufiger Axialverdichter mit parallelen Evolutionsstrategien und neuronalen Netzen*. PhD thesis, VDI Verlag, Fortschritt-Berichte VDI Reihe 7, Strömungstechnik, NR.469, 2005.
[2] A. Barron. Approximation and estimation bounds for artificial neural networks. In *Workshop on Computational Learning Theory (COLT)*, 1991.
[3] R. Bellman. *Adaptive Control Processes: A Guided Tour*. Princeton University Press, 1961.
[4] E. Benini and A. Toffolo. Axial-flow compressor model based on a cascade stacking technique and neural networks. In *Proceedings of the ASME Turbo Expo*, June 2002.
[5] C. M. Bishop. *Neural Networks for Pattern Recognition*. Oxford University Press, 1995.
[6] C. M. Bishop. *Pattern Recognition and Machine Learning*. Springer Science+Business Media, LLC, 2006.
[7] R. Callan. *The Essence of Neural Networks*. Prentice Hall, 1999.
[8] M. V. Casey. *Computational Methods for Preliminary Design and Geometry Definition in Turbomachinery*. AGARD, LS-195, 1994.
[9] M. Cetin, A. S. Ücer, Ch. Hirsch Ch., G. K. Serovy. Application of modified loss and deviation correlations to transonic axial compressors. *AGARD Advisory Report No. 745*, 1987.
[10] R. V. Chima. *Calculation of Tip Clearance Effects in a Transonic Compressor Rotor*. NASA-TM-107216, 1996.
[11] O. Dubitsky, A. Wiedermann, T. Nakano, J. Perera. The reduced order through-flow modeling of axial turbomachinery. In *IGTC Tokyo TS-052*, 2003.
[12] R. Dunker. Test case e/co-4 single transonic compressor stage. *AGARD Advisory Report No. 275*, 1990.
[13] Zell et al. *SNNS: User Manual, Version 4.2*, 1998.
[14] L. Fahrmeir, R. Kuenstler, I. Pigeot, and G. Tutz. *Statistik*. Springer, Berlin Heidelberg New York, 6. edition, 2007.
[15] R. B. Ginder, D. Harris. Test case e/co-3 single subsonic compressor stage. *AGARD Advisory Report No. 275*, 1990.
[16] N. H.-R. Goerke. *Repräsentation und Generierung von Robotertrajektorien mit neuronalen Netzen durch Einbau von Vorwissen*. PhD thesis, Mathematisch-Naturwissenschaftliche Fakultät der Rheinischen Friedrich-Wilhelms-Universität Bonn, 1997.
[17] I. Guyon, A. Elisseeff. An introduction to variable and feature selection. *Journal of Machine Learning Research*, 3:1137–1182, 2003.
[18] J. Hartung, B. Elpelt. *Multivariate Statistik*. Oldenbourg Verlag, 1992.
[19] Ch. Hirsch, J. D. Denton. *Through Flow Calculations in Axial Turbomachines*. 1981.
[20] Concepts ETI Inc. *Axial User Guide, Software Version 7.10.4*. Concepts ETI Inc., 2007.

[21] META Group Inc. Data mining tools: Metaspectrum evaluation, September 2004.
[22] N. Intrator, S. Edelman. Learning low-dimensional representations via the usage of multi-class labels. *Comput. Neural Syst.*, 8:259–281, 1997.
[23] M. Jansen. *Globale Modellbildung und garantiert stabile Regelung von Robotern mit Strukturierten Neuronalen Netzen*. PhD thesis, VDI Verlag, Fortschritt-Berichte VDI, 1995.
[24] D. Japikse, N.C. Baines. *Introduction to Turbomachinery*. Concepts ETI, Inc. and Oxford University Press, 1st reprint edition, 1997.
[25] R. Japikse, D. Japikse, R. Pelton. An evaluation of neural networks for meanline model development. In *Proceedings of the ISROMAC*, number 159, Honolulu, March 2004.
[26] C. C. Koch, L. H. Smith. Loss sources and magnitudes in axial-flow compressors. *Trans. of ASME: Journal of Engineering for Power*, 98(3):411–424, July 1976.
[27] S. Lieblein. Loss and stall analysis of compressor cascades. In *Trans. of ASME: Journal of Basic Engineering*, volume 81, pages 387–400, September 1959.
[28] S. Lieblein. Incidence and deviation-angle correlations for compressor cascades. In *Trans. of ASME: Journal of Basic Engineering*, volume 82, pages 575–587, 1960.
[29] S. Lieblein, W.H. Roudebush. Theoretical loss relations for low-speed two-dimensional-cascade and isolated airfoil sections. Tn 3662, NACA, 1956.
[30] T. Masters. *Practical Neural Network Recipes in C++*. Academic Press, 1993.
[31] S. Pazzi, F. Mertelli, V. Michelassi, M. Giachi. The use of artificial neural networks for performance prediction of return channels for industrial centrifugal compressors. In *Proceedings of the ASME Turbo Expo*, 2002.
[32] S. Pierret, R.A. Van den Braembussche. Turbomachinery blade design using a navier-stokes solver and artificial neural network. *ASME Trans., Journal of Turbomachinery*, 121(2):326–332, 1999.
[33] P. Rini, A. Alsalihi, R. A. Van den Braembussche. Evaluation of a design method for radial impellers based on artificial neural network and genetic algorithm. In *Proceedings of the 5th ISAIF Conference*, pp.535–543, Gdansk, September 4-7 2001.
[34] R. Rojas. *Theorie der Neuronalen Netze*. Springer-Verlag, 1993.
[35] F. Rosenblatt. The perceptron: A probabilistic model for information storage and organization in the brain. *Psychological Review*, 65:386–408, 1958.
[36] S. Schach, Th. Schäfer. *Regressions- und Varianzanalyse*. Springer-Verlag, 1978.
[37] H. Schneeweiss. *Ökonometrie*. MIT Press, 1986.
[38] K. L. Suder. *Blockage Development in a Transonic Axial Compressor Rotor*. NASA-TM-113115, 1997.
[39] M. Uelschen. *Entwurf und Optimierung der zweidimensionalen Gitterströmung axialer Turbomschinenbeschaufelungen mit Neuronalen Netzen und Genetischen Algorithmen*. PhD thesis, Kassel University Press, 2000.
[40] A. Zell. *Simulation Neuronaler Netze*. Addison-Wesley, 1994.
[41] A. Zell, I. Fischer, F. Hennecke, C. Bannes. *JavaNNS: User Manual, Version 1.1*, 2001.

A. Variables abbreviations

$Aann$	Annulus Area	$m.norm$	Mass Flow
Asp_Ratio	Aspect Ratio	(m)	Medium Position
$Athroat$	Area Throat	$max; mx$	Maximum
$BetaB$	Blade Angle	min	Minimum
$Chrd$	Chord	N	Number of neurons in the input layer
Dev	Deviation		
H	Number of neurons in the hidden layer	n	Speed Line
		$opt; *$	Optimum
$H.in$	Blade Height (in)	out	Outlet
$H.out$	Blade Height (out)	Phi	Camber Angle
(h)	Hub Position	Re	Reynolds Number
in	Inlet	rel	Relative
$inc; i$	Incidence	$StaggerAng$	Stagger Angle
$K\_overall; \omega$	Total Loss Coefficient	t	Thickness
le	Leading Edge	(t)	Tip Position
M	Number of neurons in the output layer	te	Trailing Edge
		$Thrt$	Throat
$M.rel$	Relative Mach Number	W	Relative Flow Velocity

81 **Do Sanh, Do Dang Khoa**
Problems of the equilibrium of a rigid body and mechanical systems

101 **A. Kaveh, K. Koohestani**
Formation of graph models for regular finite element meshes

117 **S. Y. Fialko**
A block sparse shared-memory multifrontal finite element solver for problems of structural mechanics

133 **A. Dargar, A. Hasan, R. A. Khan**
A method of identification of kinematic chains and distinct mechanisms

143 **M. Azzam, J.-C. Haag, P. Jeschke**
Application concept of artificial neural networks for turbomachinery design