ČESKÉ VYSOKÉ UČENÍ TECHNICKÉ V PRAZE
FAKULTA STROJNÍ

CZECH TECHNICAL UNIVERSITY IN PRAGUE
FACULTY OF MECHANICAL ENGINEERING

Ing. Ivo Bukovský, Ph.D.

Nekonvenční neuronové architektury a jejich výhody pro technické aplikace

Nonconventional Neural Architectures and their Advantages for Technical Applications

Summary

The work briefly reviews history of artificial neural networks from publishing the first mathematical model of neuron in 1943 toward the specific area of nonconventional neural architectures. The core of the non-conventionality of the discussed neural architectures consists in distinct conceptions of nonlinearity of neuronal models, which are different from conventional neuronal models with linear synaptic neural operation. The particular contribution of the author to the field is the introduction of novel conception of nonconventional neural units (HONUs) and their classification according to three fundamental principals for modeling of dynamical systems; these are the customable nonlinearity via polynomial order of the synaptic operation, the order of dynamics of state space representation of a neuron, and the implementation of adaptable time delays that further enhances approximation capability for systems with high order of dynamics. This novel conception for neural units shall be understood as a natural merge of the early and more recent developments in artificial neural networks together with the conservative and well established approaches in the field of modeling and control engineering. Such a unifying classification of neuronal models as of standalone neural units while still mimicking the biological analogy had not been found in literature before. During applications to real world problems, the introduced neural units, esp. the static and discrete-time dynamic QNU, proved to be attractive from the point of tradeoff between the nonlinear strength of artificial neural networks and the practical usability of NN optimization and correct functionality with avoidance of local minima problem because of the linear nature of optimization problem of HONU for a given training data.

When using non-heuristic learning algorithms, which are considered suitable for small scale neural networks, standalone HONU substantially outoperforms perceptron type networks in approximation of nonlinear dynamical systems.

Further theoretical research on HONU can consist in investigation of networking of HONU. For applied research, HONUs are perspective tool for new and very fast hardware solutions of control, such as using the field programmable arrays (FPGA), chipset on board and embedded board solutions.

Souhrn

Tato práce stručně shrnuje historii umělých neuronových sítí od prvního modelu umělého neuron v roce 1943 směrem ke specifické kategorii nekonvenčních neuronových sítí. Podstata nekonvenčnosti diskutovaných neuronových architektur spočívá v odlišné koncepci nelinearity neuronových modelů, které jsou odlišné od konvenčních umělých neuronů s lineární synaptickou operací. Konkrétním příspěním autora do oblasti je představení nové koncepce nekonvenčních neuronových jednotek (HONU) a jejich klasifikace podle tří fundamentálních principů pro modelování dynamických systémů; jsou to uživatelsky nastavitelná kvalita nelineární aproximace stupněm polynomu synaptické operace, řád dynamiky stavové formulace neuronu, a typ implementce adaptovatelného dopravního zpoždění buďto na vstupu neuronu a/nebo v jeho stavové zpětné vazbě, což podstatně zvyšuje kvalitu aproximace systémů s vyším řádem dynamiky nebo s dopravními zpožděními. Tato nová koncepce by měla být chápána jako přirozené sjednocení původních i posledních trendů neuronových sítí spolu s konzervativními a dobře zavedenými koncepty modelování a řízení dynamických systémů. Podobná kompletní jednoticí koncepce klasifikace samostatných neuronových jednotek při snaze zachovat vztah k analogii biologického neuronu nebyla v literatuře ještě nalezena.

Při aplikacích na reálných datech a systémech se nekovenční neuronové jednotky, zejména statická a dynamická HONU, ukázaly jako atraktivní z hlediska vyváženosti mezi kvalitou nelineární aproximace, praktičností optimalizačního algoritmu, a správnou funkčností sítě a to při prakticky minimálním efektu lokálních minim, který je v případě HONU teoreticky eliminován lineární podstatou optimalizační úlohy HONU pro příslušnou trénovací množinu dat. Při použití neheuristických algoritmů učení, které jsou obecně považovány za vhodné pro neuronové sítě menších rozměrů, samostatné jednotky výrazně překonávaly neuronové sítě typu perceptron v aproximaci nelineárních dynamických systémů.

Případný následující základní výzkum HONU může přirozeně spočívat ve zkoumání nových neuronových sítí s jednotkami HONU.

Pro aplikovaný výzkum jsou jednotky HONU velmi perspektivní například i pro velmi rychlé úlohy řízení s využitím programovatelných hradlových polí (FPGA) a příbuzných hardwarových aplikací.

Klíčová slova

umělá inteligence, umělý neuron, gradientové metody adaptace, dynamický backpropagation, nekonvenční neuronové sítě, polynomiální neuronové sítě, neuronové sítě vyšších řádů, neuronové jednotky vyšších řádů, kvadratická neuronová jednotka, neuronová jednotka s dopravním zpožděním, časové řady, nelineární dynamické systémy, predikce, řízení, reálné systémy

Keywords

artificial intelligence, computational intelligence, artificial neuron, real time recurrent learning, backpropagation through time, nonconventional neural networks, polynomial neural networks, higher order neural networks, higher order neural unit, quadratic neural unit, time delay neural unit, time series, nonlinear dynamic systems, prediction, control, real systems

Contents

1 INTRODUCTION

It is known that human brain can have more than 100 billion neurons, which have complicated interconnections, and these neurons constitute a large-scale signal processing and memory network. Even though the computational power of computers is still exponentially increasing, it is still a long multidisciplinary race of sciences to obtain a cognitive computing machine that would approach the cognitive capabilities of human brain. However, the mathematical study of a single neural model and its various extensions is the first step in the design of a complex neural network for solving a variety of problems in the fields of signal processing, pattern recognition, control of complex processes, neurovision systems, and other decision making processes.

In the first half of 20^{th} century, the cross-fertilization of physiology and mathematics resulted in publication of the famous mathematical model of a neuron by McCulloch and Pitts in 1943 [1] that initiated research of novel computational disciplines of science. Together with the boom of computers since 1980', we have arrived to a very extensive area of artificial neural networks (ANN) and to the area of optimization methods devoted to learning techniques of ANN such as genetic algorithms (GA) and much more of other more or less heuristic optimization methods (i.e. differential evolution and memetic computing or adaptive resonance theory).

It may be mentioned that techniques of ANN and their learning techniques have grown for the necessity to model and control complex systems that are not possible to be analyzed via mathematical-physical analysis. In other words, it is not possible to analyze many real systems by comprehensive decomposition into simpler interconnected subsystems – this conforms to one of generally accepted engineering definition of complexity of real world systems [2][3]. The real systems are naturally complex and not closed ones, and they are further nonstationary that is caused by immeasurable perturbations or due to nonlinearity that may also be a reason why real systems are difficult to be modeled; the sensitivity to small perturbations of nonlinear system may cause large variation of system behavior as it is known form the theory of deterministic chaos. On the other hand, the theory of deterministic chaos shows that very complicated behavior can be found in relatively simple (rather complicated than complex) mathematical models such as logistic equation, Duffing oscillator or Lorenz system.

ANN are, in their nature, rather a deterministic tool that relies on availability of properly processed and configured training data because uncertainty in training data has negative impact to optimization and learning capabilities of ANN. Therefore, it is important to mention the foundation of fuzzy logic by Lotfi Zadeh in 1960', which has grown to another branch of CI that primarily focus uncertainty and that today includes also areas as neuro-fuzzy systems, type-2 fuzzy sets, or granular computing approaches.

In 1980', world has thought of artificial neural networks as of part of artificial intelligence (AI) that involved also other substantial and by-nature inspired computation techniques, i.e., genetic algorithms (GA) and fuzzy systems. Later in 1990', one can notice that it was more common to talk about NN, GA, or FS as of disciplines of Computer Science (CS, e.g., *IEEE Computer Society*) while AI appeared to be understood more broadly.

Today, we can notice that NN, GA, FS and major frameworks of computational algorithms that are covered by umbrella of Computational Intelligence (CI, e,g, *IEEE Computational Intelligence Society*), while the Computer Science may be seen as rather more inclining toward HW and SW issues of computers, their networking and performance including e.g., cloud computing, mobile computing, bioinformatics, and many others. Even though the fields of CI and CS are still not clearly separable, we may notice that CI inclines more toward research of computational and cognitive algorithms of artificial intelligence while CS inclines toward the hadware-like solutions that can be used for implementations of CI. Except general areas of NN, GA, FS, we may mention another additional disciplines of CI such as self-organizing maps, Boolean networks, support vector machines, spiking neural networks, particle swarm optimization, differential evolution, memetic computing, adaptive resonance theory, affective computing, quantum computing, type-2 fuzzy sets, granular computing,

For engineering tasks of modeling and control of dynamical systems, the most common tools are supervised neural networks of smaller scales (about with number of inputs <1000).

2 FROM CONVENTIONAL TO POLYNOMIAL NN

Returning back to the middle of 20th century, the first bio-inspired mathematical model of artificial neuron [1] had linearly aggregated neural inputs, i.e. linear synaptic neural, which was followed by a hard-limited threshold activation function, i.e. somatic operation, that allows neural network for classification purposes. The later introduced Perceptron model of a neuron [4] had also linear somatic operation; however, the somatic operation changed to a sigmoidal type of functions.

The next milestone in neural network area was the upgrade of static neuronal models to dynamic (recurrent) versions (Hopfield in 1982 [5]) and proposal of recurrently interconnected neural networks with Perceptron type of neurons. Therefore, the conventional neural networks can be basically classified into categories of static feedfoward networks and dynamical (recurrent) networks with various feedback configurations (hidden recurrent layers, tapped delayed feedbacks of neural outputs). Further, the neural networks can be classified according to if they are computed in continous-time domain or discrete time domain [6].

From the above mentioned foundations, ANN has grown to a vast field of theoretical science today, while real implementations of ANN is a multidisciplinary task including proper data acquisition and signal processing. As an aside, a practically useful technique for neural networks is the technique of so called PCA neural networks that can reduce the dimensionality of vector or external inputs [7][8]. Other interesting earlier-appearing neural network architectures are product neural units by Durbin & Rumelhart [29] and later logarithmic neural networks by Hines [30]. Another nonlinearly powerful approach in neural networks is support vector machines (SVM) founded by Vapnik and Lerner in [18]; they are dependent on proper selection of kernel functions that increase dimensionality of feature vectors to allow separation of data clusters by linear discriminants that is impossible in lower dimensional spaces.

Since 1970', there appeared novel direction in neural network field, i.e., so called polynomial neural networks (PNNs) or higher-order neural networks (HONNs). Basically, both the PNNs and HONNs represent the same style of computation in artificial neural networks where neurons involve polynomials, or the neurons are polynomials themselves, or where synaptic connections between neurons involve higher-order terms (i.e. higher-order polynomials). Nevertheless, the use of HONN notation seems to take over in popularity over PNN in most recent publications. For the early and important work on polynomial approximation in area of neural networks, we refer to work of Ivakhnenko [11] and his method known as Group Method of Data Handling (GMDH). Heuristically and with consideration of least square polynomial fitting to training data, GMDH technique builds very high-order polynomial (i.e. Ivakhnenko polynomial) to approximate a system model. Importantly, the technique considers also the optimal complexity polynomial evaluation that validates the Ivakhnenko polynomial against later data than just against the merely training ones. More recent and significant publications devoted to PNN that follows the Ivakhnenko concepts are the works of Nikolaev and Iba [27][28] that provides an extensive studies of optimization algorithms for PNN from direct least square computation of weights through backpropagation techniques to the use of genetic algorithms. The particular concepts and works that shall be referenced as framed within HONNs (rather than within PNN) can be found in works of Shin & Ghosh [20], Softky & Kammen [21], Taylor & Commbes [22], Chen & Manry [23], Schmidt & Davis [24], Kosmatopoulos, Polycarpou, Christodoulou and Ioannou [25], and Heywood & Noakes [26].

3 HIGHER ORDER NEURAL UNITS

Up to 2003, literature had not clearly nor systematically distinguished between polynomial neural networks, higher order neural networks or a single neural unit with polynomial synaptic operation. Also, there was no such unified approach to artificial neurons while it was apparent that conventional concept of single artificial neurons resulting from works of McCulloch and Pitts, Rosenblatt or Hopfield could be naturally enhanced in their quality for approximation of nonlinear systems and dynamic systems and while not significantly diverging from neuronal biological analogy.

On the other hand and up to 2003, there were emerging suitable fundamental mathematical approaches such as the Tailor polynomial, gradient optimization methods (backpropagation technique, gradient descent rule, and Levenberg-Marquardt algorithm), and well established engineering approach of time delay systems (e.g. works authored or co-authored by Zítek, Kučera, Víteček) that are of fundamental importance to modeling of dynamical systems and control engineering problems.

The following subsections includes the author's contribution to the field of neural networks, which is a merge of the contemporary neural network approaches with more or less traditional engineering approaches that results in novel classification, and thus new concepts of nonconventional neural units. Moreover, published results of on real systems are shown to support the theoretical achievements.

3.1 New Classification of Artificial Neurons

The novel proposed classification of artificial neural units, as published in [40] [51], is based on three very fundamental mathematical and engineering attributes:

a) the nonlinearity of the aggregating operation ν of neural inputs, i.e. $f_{HONNU}(\,)$ as in Fig. 2,
b) the order of dynamics of state space representation of a neuron (i.e., the number of time integrations of aggregated variable ν, applies to continuous dynamic HONU),
c) and the type of implementation of adaptable time delays within a neural unit.

The principal of classification is sketched in Fig. 1 and Fig. 2, where only the most important types are shown for brevity.

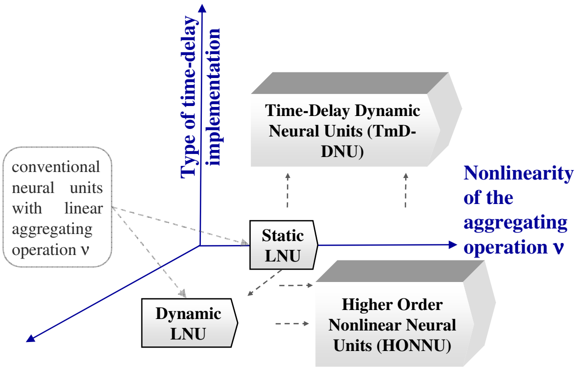


Fig. 1: Novel classification of basic artificial neural units according to: a) aggregating nonlinearity ν, b) its time integrations (i.e. the dynamic order), c) and adaptable time-delay implementation;

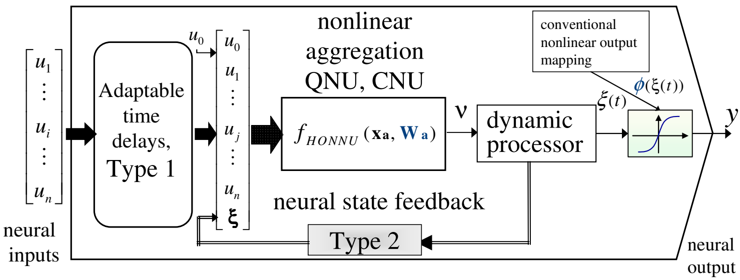


Fig. 2: Classification of Nonconventional Neural Units according to a) polynomial order of $f_{HONNU}(\ )$,b) dynamic processor, c) Type 1 or Type 2 implementation of adaptable time delays.

Systematic classification according to nonlinear aggregation and neural dynamics is sketched in Fig. 3 while the classification by Time Delays and Nonlinear Aggregation is sketched in Fig. 4.

3.2 Static HONU

This subsection focuses on static HONU, and on enhancements of gradient learning rules that are the part of derivations introduced in [33] and that are also discussed in [34]. Let r denote a polynomial order of HONU and consider somatic neural $\phi(\nu)=\nu$ (without loss of generality for standalone HONU), then neural output y of static HONU at time k is alternatively given as

$$y=\sum_{i=0}^{n}\sum_{j=i}^{n}\cdots\sum_{\dots}^{n}\left(x_i x_j \cdots x_{\dots}\right)\cdot w_{i,j,\dots}, \text{ where neural bias } x_0=1, \quad \textbf{(1)}$$

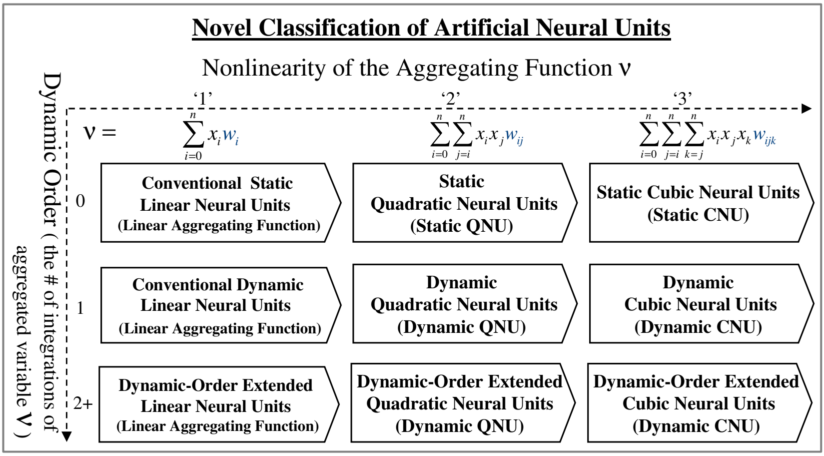


Fig. 3: Classification of nonconventional continuous artificial neural units according to a) aggregating nonlinearity $f_{HONNU}=\nu$,and b) the order of dynamics.

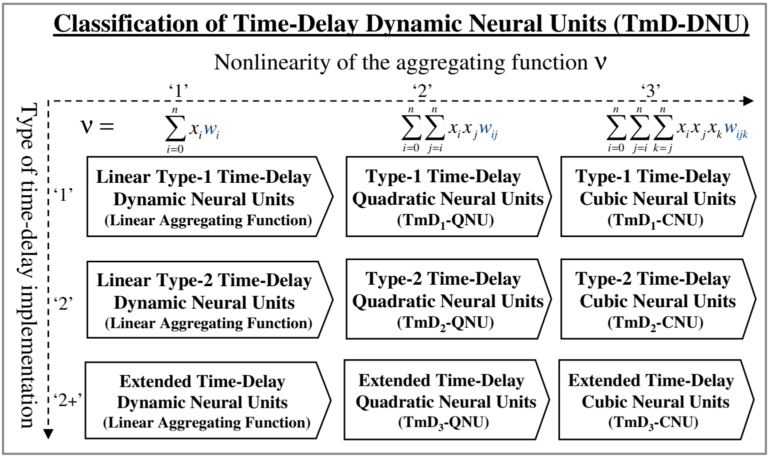


Fig. 4: Classification of basic time-delay dynamic neural units according to a) aggregating nonlinearity ν and c) the type of time delay implementation.

where also n is the number of neural inputs x_i and w denotes individual neural weights that are for r^{th} order of HONU represented by an r-dimensional array $\mathbf{W}$ of neural weights with position indexes $i,j,\ldots$

When adopting the original formulation of quadratic neural unit (QNU), i.e. HONU of r=2, by Gupta et al (2003), this second order neural unit (as recently also called in [5]), is given in matrix representation as follows

$$\mathbf{x}(k)=\begin{bmatrix} x_0=1 \\ x_1(k) \\ \vdots \\ x_n(k) \end{bmatrix},\ \mathbf{W}=\begin{bmatrix} w_{0,0} & w_{0,1} & \cdots & w_{0,n} \\ 0 & w_{1,1} & \cdots & w_{1,n} \\ \vdots & \ddots & \ddots & \vdots \\ 0 & \cdots & 0 & w_{n,n} \end{bmatrix},\ y(k)=\mathbf{x}^T\mathbf{W}\mathbf{x}, \qquad \textbf{(2)}$$

where T stands for matrix or vector transposition.

Recently in [33][34], flattening operation has been introduced for general r^{th}-order HONU so a general r^{th} order HONU can be expressed in a long vector representation. The novel flattening approach for HONU can be demonstrated on QNU from (2) as follows

$$y=\mathbf{rowx}\cdot\mathbf{colW}=\begin{bmatrix} x_0 \ x_1 \ \dots \ x_i x_j \ \dots \ x_n x_n \end{bmatrix}\cdot\begin{bmatrix} w_{0,0} \ w_{0,1} \ \dots \ w_{i,j} \ \dots \ w_{n,n} \end{bmatrix}^T \qquad \textbf{(3)}$$

where **rowx** is the long row vector representation of polynomial terms of synaptic neural operation (including lower orders due to bias x_0=1), and **colW** is the long column vector representation of otherwise r-dimensional weight array **W**. Without the loss of generality, the long vector representation in (3) is generally applicable to arbitrary polynomial order r and highlights the advantage of HONU, i.e., that the optimization of HONU is a linear problem when training data are substituted in y and **rowx**. Benefiting from the in-parameter linearity of HONU and considering (3), neural weights of static HONU can be directly calculated by least square method leading to a variation of the Wiener-Hopf equation for HONU and for an arbitrarily polynomial order r as follows

$$\mathbf{colW}=\left(\mathbf{colX}\cdot\mathbf{rowX}\right)^{-1}\cdot\mathbf{colX}\cdot\mathbf{y_p}\,, \qquad \textbf{(4)}$$

where **colX** represents a corresponding matrix of polynomially correlated input patterns and $\mathbf{y_p}$ is corresponding vector of targets. In case of real data where matrix **colX** may be not well conditioned, it is practical to train static HONU by Levenberg-Marquardt (L-M) algorithm as

$$\Delta\mathbf{colW}=\left(\mathbf{colX}\cdot\mathbf{rowX}+\frac{1}{\mu}\cdot\mathbf{I}\right)^{-1}\cdot\mathbf{colX}\cdot\mathbf{e} \qquad \textbf{(5)}$$

where **e** is the standard vector (or matrix) of neural output errors, μ is learning rate , and $\mathbf{rowX}=\mathbf{colX}^T$ already represents the Jacobian matrix. It is apparent from (5) for HONU, that the Jacobian matrix becomes only once calculated constant depending merely on neural inputs, so the whole training can rapidly accelerate in comparison to other neural networks where Jacobian matrix and the inverse matrix has to be calculated each training epoch.

As regards the gradient descent adaptation of static HONU, e.g., as for nonlinear adaptive filtering, the above introduced notation allows HONU to directly implement gradient descent regularization techniques such as the normalized gradient descent [12] and Benveniste's updates [13] as follows

$$\eta(k)=\frac{\mu}{\left\|\mathbf{rowx}(k)\right\|_2^2+\varepsilon},\ \eta(k)=\frac{\mu}{\mathbf{rowx}(k)\cdot\mathbf{colx}(k)+\varepsilon}, \tag{6}$$

and further also an algorithm by Farhang and Ang [14], Mathews' algorithm [16], or the generalized normalized gradient descent algorithm (GNGD) of Mandic [15] as follows

$$\varepsilon(k+1)=\varepsilon(k)-\gamma\cdot\mu\frac{e(k)\cdot e(k-1)\cdot\mathbf{rowx}(k)\cdot\mathbf{colx}(k-1)}{(\left\|\mathbf{colx}(k-1)\right\|_2^2+\varepsilon(k-1))^2}, \tag{7}$$

where the minimum value of the regulization constant ε is derivable from the Hurwitz stability of a priori error $e(k)$ to a posteriori error $\tilde{e}(k)$ mapping (as in [19] e.g.), which is is for static HONU as follows

$$\tilde{e}(k)=y_{real}(k)-(\mathbf{rowW}(k)+\Delta\mathbf{rowW})\cdot\mathbf{colx}(k)=e(k)-\eta(k)\cdot e(k)\cdot\mathbf{rowx}(k)\cdot\mathbf{colx}(k)$$

$$\varepsilon(k)>\mathbf{rowx}(k)\cdot\mathbf{colx}(k)\cdot\left(\frac{\mu}{2}-1\right) \tag{8}$$

and robust GD variations according to [17] [19] are also feasible.

3.3 Time-Delay Dynamic Neural Units (TmD-DNU)

This subsection demonstrates the conception of continuous-time dynamic neural units TmD-DNU with linear synaptic operation ((9) as published in [44] [52] [53]) and the modification of gradient descent adaptation approach also for adaptable time delay neural parameters w_4 and w_{3j}.

$$\frac{dx(t)}{dt}(w_1{}^2+\tau_{\min})+x(t-w_4{}^2)=\sum_{j=1}^{n}w_{2j}\cdot u_j(t-w_{3j}{}^2),\ \ y(t)=\phi(x(t)) \tag{9}$$

The feasibility of gradient descent adaptation of adaptable time delays of linear TmDNUs results from possibility to evaluate the partial derivative of Laplace quasipolynomial type transfer function of the above TmdDNU (9). Then the state feedback time delay w_4 , e.g., can be adapted according to the gradient descent rule that is derived using the backpropagation chain rule for partial derivatives according to following scheme

$$\Delta w_i{}^{(k)}=-2\overline{\mu}\frac{\partial(e(t))^2}{\partial w_i}=\mu\, e(t)\cdot\left[\frac{\partial\phi(t)}{\partial x(t)}L^{-1}\left\{\frac{\partial G_{TmDNU}(s)}{\partial w_i}U(s)\right\}\right] \tag{10}$$

where s stands for the Laplace operator, and L^{-1} denotes inverse Laplace transform, and G_{TmDNU} denotes the Laplace transfer function of the internal dynamics of a unit with linear synaptic operation.

Even though the above concept of TmD-DNU is highly potential for approximation of systems with high order of dynamics or systems with time delays via rigorously derived gradient descent updates (10), this continuous time concept is still of much academic nature in comparison to discrete time HONUs; it is sensitive to initial setup of weights, and it appeared more practical to combine gradient descent and genetic algorithm to identify time delay parameters of HONU as it was done for approximation of double tube heat exchanger in [50] or later in the contemporary research of our PhD student.

3.4 Recurrent HONU

Recurrent HONUs, esp. discrete time ones, are the most practically useful concept of the presented nonconventional neural units. The performance comparison of standalone recurrent HONUs to linear models and to MLP

Tab. 1: Fundamental implementation examples of static and dynamic QNU and supervised learning rules for updating individual weights [34][48].

QNU	Mathematical Structure	Learning Rule
Static	y_n... neural output $x_1, x_2, ..., x_n$ ... external neural inputs W... upper triangular weight matrix $\mathbf{x} = \begin{bmatrix} 1 \\ x_1 \\ \vdots \\ x_n \end{bmatrix}, \quad \mathbf{W} = \begin{bmatrix} w_{00} & w_{01} & \cdots & w_{0n} \\ 0 & w_{11} & \cdots & w_{1n} \\ \vdots & \ddots & \ddots & \vdots \\ 0 & \cdots & 0 & w_{nn} \end{bmatrix}$ $y_n = \sum_{i=0}^{n} \sum_{j=i}^{n} x_i x_j w_{ij} = \mathbf{x}^{\mathbf{T}} \mathbf{W} \mathbf{x}$	<u>Levenberg-Marquardt (global optimization)</u> N... number of samples (data length) $\Delta w_{ij} = -(\mathbf{j}_{ij}{}^T \cdot \mathbf{j}_{ij} + \frac{1}{\mu})^{-1} \cdot \mathbf{j}_{ij}{}^T \cdot \mathbf{e} \qquad \mathbf{j_{ij}} = \frac{\partial \mathbf{y_n}}{\partial w_{ij}} = \begin{bmatrix} x_i(1)\, x_j(1) \\ x_i(2)\, x_j(2) \\ \vdots \\ x_i(N)\, x_j(N) \end{bmatrix}$ $\mathbf{e} = \left[e(1) \quad e(2) \quad \cdots \quad e(N) \right]^T$ <u>Gradient Descent (local optimization)</u> k ... sample number $w_{ij}(k+1) = w_{ij}(k) + \Delta w_{ij}(k)$ $\Delta w_{ij}(k) = -\frac{1}{2} \mu\, e(k)^2 = \mu\, e(k)\, x_i(k)\, x_j(k)$
Discrete Dynamic	$\mathbf{x} = \begin{bmatrix} 1 \\ y_n(k+n_s-1) \\ y_n(k+n_s-2) \\ \vdots \\ y_n(k+1) \\ x_1(k) \\ \vdots \\ x_m(k) \end{bmatrix}, \quad y_n(k+n_s) = \mathbf{x}^{\mathbf{T}} \mathbf{W} \mathbf{x}$ y_r... real value typically for prediction: $x_1(k) = y_r(k)$ $x_m(k) =$ $= y_r(k-m+1)$	<u>RTRL (local optimization)</u> $\Delta w_{ij}(k) = \mu\, e(k) \frac{\partial y_n(k+n_s)}{\partial w_{ij}}$ $\frac{\partial y_n(k+n_s)}{\partial w_{ij}} = \frac{\partial(\mathbf{x}^T \mathbf{W} \mathbf{x})}{\partial w_{ij}} = \mathbf{j_{ij}}^T \mathbf{W}\, \mathbf{x} + x_i x_j + \mathbf{x}^T \mathbf{W}\, \mathbf{j_{ij}}$ $\mathbf{j_{ij}} = \frac{\partial \mathbf{x}}{\partial w_{ij}} = \left[0 \;\; \frac{\partial y_n(k+n_s-1)}{\partial w_{ij}} \;\; \frac{\partial y_n(k+n_s-2)}{\partial w_{ij}} \cdots \frac{\partial y_n(k+1)}{\partial w_{ij}} \;\; 0 \cdots 0 \right]^T$
Continuous Dynamic (example of 2nd order dynamics)	$\mathbf{x} = \begin{bmatrix} 1 \\ y_n(t) \\ y_n'(t) \\ x_1(t) \\ \vdots \\ x_m(t) \end{bmatrix}, \qquad y_n''(t) = \mathbf{x}^{\mathbf{T}} \mathbf{W} \mathbf{x}$	<u>RTRL (local optimization)</u> $\Delta w_{ij}(k) = \mu\, e(t) \frac{\partial y_n(t)}{\partial w_{ij}}$ $\frac{\partial y_n(t)}{\partial w_{ij}} = \frac{\partial}{\partial w_{ij}} \iint (\mathbf{x}^T \mathbf{W} \mathbf{x}) dt^2 = \iint \left(\mathbf{j_{ij}}^T \mathbf{W}\, \mathbf{x} + x_i x_j + \mathbf{x}^T \mathbf{W}\, \mathbf{j_{ij}} \right) dt^2$ $\mathbf{j_{ij}} = \left[0 \;\; \frac{\partial y_n(t)}{\partial w_{ij}} \;\; \frac{\partial y_n'(t)}{\partial w_{ij}} \;\; 0 \cdots 0 \right]^T, \quad \frac{\partial y_n''(t)}{\partial w_{ij}} = \mathbf{j_{ij}}^T \mathbf{W}\, \mathbf{x} + x_i x_j + \mathbf{x}^T \mathbf{W}\, \mathbf{j_{ij}}$

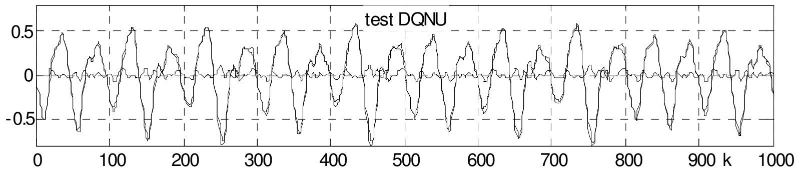


Fig. 5: Prediction of chaotic MacKey-Glass equation (normalized data) by the discrete recurrent QNU (Tab. 1); the 11 samples (sampling 1 second) prediction is superimposed on original data and the middle line is the error between original and predicted values; dynamic QNU has surprising prediction accuracy.

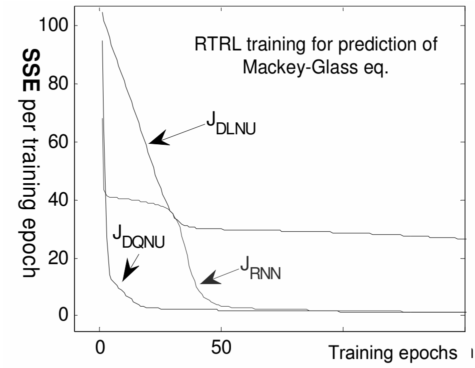


Fig. 6: The typical situation when QNU is almost always trained to the minimum of high accuracy and without getting stacked in some less accurate local minima (SSE…sum of square errors).

types of static and recurrent neural networks can be found in publications [31][34][42][48][49], where the superiority of standalone HONUs over the conventional MLP networks is apparent.

There are two supervised training techniques that are widely considered as useful for small scale recurrent neural networks; these are the real time recurrent learning (RTRL introduced by Williams and Zipser in 1989 [9]) and backpropagation through time (BPTT introduced by Werbos in 1990 [10]). The RTRL technique is a dynamic version of gradient descent adaptation rule that does not neglect recurrent calculations of Jacobian matrix. RTRL, e.g., for QNU is shown in Tab. 1. The BPTT technique is a batch training technique suitable for small scale dynamical neural networks, and it is presented as a technique of unfolding a recurrently computed neural architecture in time. However and unfortunately, it is not usually comprehensibly mentioned in literature that BPTT can be implemented as a combination of RTRL technique and the Levenberg-Marquardt algorithm that is a very efficient training technique for recurrent HONU [34]. Fig. 6 compares performance of dynamical QNU (HONU of r=2) with dynamical linear neural unit (LNU, i.e., HONU of r=1) and with a multilayer perceptron (MLP) neural network with a single hidden recurrent layer when adapted by RTRL to Mackey-Glass chaotic time series. However, Fig. 6 sketches a typical situation when a linear model can not reach acceptable accuracy because of its linearity, MLP neural network (RNN) may reach acceptable precision; however, for the risk of getting stuck in a local minima, and QNU reaches high accuracy and converges fast without any indication of stacking in local minima and it appeared also in real time implementations

3.5 Weight-Update Stability of Static HONU

This subsection presents a novel approach to evaluation of gradient descent weight-update stability of HONU that has been recently introduced in [33]. The long vector operator approach as introduced in subsection 3.2 (and in [33]) is used further. The weight-update system by gradient descent learning rule for update of the weights of HONU at sampling time *k* may be given as

$$\mathbf{colW}(k+1)=\mathbf{colW}(k)+\mu\cdot\left(y_p-y\right)\cdot\frac{\partial y}{\partial\mathbf{colW}}\ , \tag{11}$$

where y_p is the target, y is neural output, μ is the learning rate (scalar), and

$$\frac{\partial y}{\partial\mathbf{colW}}=\begin{bmatrix}\frac{\partial y}{\partial w_{0,0}} & \frac{\partial y}{\partial w_{0,1}} & \cdots & \frac{\partial y}{\partial w_{n,n}}\end{bmatrix}^T . \tag{12}$$

For static QNU, e.g., the derivative of neural output with respect to a single general weight of QNU is as follows

$$\frac{\partial y}{\partial w_{ij}}=\frac{\partial\left(\mathbf{rowx}\cdot\mathbf{colW}\right)}{\partial w_{ij}}=\mathbf{rowx}\cdot\frac{\partial\mathbf{colW}}{\partial w_{ij}}=x_i\cdot x_j\ , \tag{13}$$

then the neural weight-update system for a weight of static HONU is as

$$w_{ij}(k+1)=w_{ij}(k)+\mu\cdot\left(y_p-\mathbf{rowx}\cdot\mathbf{colW}(k)\right)\cdot x_i\cdot x_j, \tag{14}$$

and the column weight update formula for all weights can be expressed as follows

$$\mathbf{colW}(k+1)=\mathbf{colW}(k)+\mu\cdot\left(y_p-\mathbf{rowx}\cdot\mathbf{colW}(k)\right)\cdot\mathbf{colx}. \tag{15}$$

Let's denote the long vector multiplication term as follows

$$\mathbf{S}=\mathbf{colx}\cdot\mathbf{rowx}\ . \tag{16}$$

Considering that $\mathbf{colx}=\mathbf{rowx}^{\mathrm{T}}$ contain external inputs and that y_p is training target, we clearly see from (15) that the stability condition of the weight-update system of static HONU at each time k is classically resulting as follows

$$\rho\left(\mathbf{I}-\mu\cdot\mathbf{S}\right)\leq 1\,. \tag{17}$$

where $\rho(.)$ is spectral radius, and $\mathbf{I}$ is an identity matrix of diagonal length equal to the number of neural weights.

It can be also concluded that because of the in-parameter linearity that allows HONU to be expressed in long vector operation, and because of the resulting fact that stability of weight update system of static HONU depends merely on magnitudes of neural inputs (16) (17), the gradient descent adaptation of HONU can also adopt the efficient techniques that are applicable to linear filters. These techniques that are recognized as efficient to maintain stability during gradient descent adaptation are known as the normalized least mean square algorithm (NLMS), or the Benveniste's learning rate updates, algorithm by Farhang and Ang, Mathews' algorithm, and generalized normalized gradient descent algorithm (GNGD) (Mandic et al.2009) .

3.6 Weight-Update Stability of Recurrent HONU

Recurrent HONU feeds its step delayed neural output back to its input. The individual weight update of recurrent HONU by fundamental gradient descent (RTRL) can then be given using the above introduced operators and for any polynomial order as follows

$$\begin{aligned} &w_{i,j,\ldots}(k+1)=w_{i,j,\ldots}(k)+\\ &+\mu\cdot\left(y_p(k+n_s)-\mathbf{rowx}(k)\cdot\mathbf{colW}(k)\right)\cdot\frac{\partial y(k+n_s)}{\partial w_{i,j,\ldots}}, \end{aligned} \tag{18}$$

where n_s is the discrete prediction interval, and the individual derivatives of neural output are for recurrent HONU as follows

$$\frac{\partial y(k+n_s)}{\partial w_{i,j}}=\left(\frac{\partial\mathbf{rowx}(k)}{\partial w_{i,j}}\mathbf{colW}(k)+\mathbf{rowx}(k)\frac{\partial\mathbf{colW}(k)}{\partial w_{i,j}}\right), \tag{19}$$

where weight indexing is shown as if for QNU, and here $\partial\mathbf{rowx}(k)/\partial w_{i,j}\neq\mathbf{0}$ because the neural input $\mathbf{x}$ of recurrent architecture is concatenated with delayed neural outputs, and it can be expressed for all derivatives of neural output in a long-column vector as

$$\frac{\partial y(k+n_s)}{\partial\mathbf{colW}}=\left(\frac{\partial\mathbf{rowx}(k)}{\partial\mathbf{colW}}\mathbf{colW}(k)+\mathbf{colx}(k)\right), \tag{20}$$

where for the example of QNU

$$\mathbf{J}=\begin{bmatrix}\frac{\partial \mathbf{rowx}}{\partial w_{0,0}}\\ \frac{\partial \mathbf{rowx}}{\partial w_{0,1}}\\ \vdots\\ \frac{\partial \mathbf{rowx}}{\partial w_{n,n}}\end{bmatrix}=\begin{bmatrix}\frac{\partial x_0{}^2}{\partial w_{0,0}} & \frac{\partial (x_0x_1)}{\partial w_{0,0}} & \cdots & \frac{\partial x_n{}^2}{\partial w_{0,0}}\\ \frac{\partial x_0{}^2}{\partial w_{0,1}} & \frac{\partial (x_0x_1)}{\partial w_{0,1}} & \cdots & \frac{\partial x_n{}^2}{\partial w_{0,1}}\\ & \vdots & & \\ \frac{\partial x_0{}^2}{\partial w_{n,n}} & \frac{\partial (x_0x_1)}{\partial w_{n,n}} & \cdots & \frac{\partial x_n{}^2}{\partial w_{n,n}}\end{bmatrix}, \tag{21}$$

where $\mathbf{J}$ represents the recurrently calculated Jacobian matrix with dimensions $n_w \times n_w$, where n_w is the total number of weights, which is also equal to the number of elements of **rowx** or **colW**.

$$\mathbf{R}=\mathbf{J}\cdot y_p(k+n_s)-\mathbf{J}\cdot\mathbf{colW}(k)\cdot\mathbf{rowx}(k)\,. \tag{22}$$

If we introduce a diagonal matrix of learning rates $\mathbf{M}$ instead of a single μ and separate the parts of the update recurrent system as follows

$$\mathbf{colW}(k+1)=\left(\mathbf{I}+\mathbf{M}\cdot\left(\mathbf{R}-\mathbf{S}\right)\right)\cdot\mathbf{colW}(k)+\mathbf{M}\cdot\mathbf{colx}(k)\cdot y_p(k+n_s)\,, \tag{23}$$

then the stability condition for adaptation of recurrent HONU by RTRL technique via evaluation of spectral radius $\rho()$ is as follows

$$\rho\left(\mathbf{I}+\mathbf{M}(k)\cdot\left(\mathbf{R}-\mathbf{S}\right)\right)\leq 1 \tag{24}$$

where $\mathbf{S}$ is defined in (17) and the time indexing of the learning rate matrix $\mathbf{M}(k)$ indicates the time variability of individual learning rates. More detailed derivations for weight-update stability of both static and recurrent HONU was published recently in [33]. For a single learning rate to all weights, Fig. 7 demonstrates the use of the weight update stability criteria during adaptation to Mackey-Glass time series. The first violations of the stability condition (24) is detected at around k=650, while the onset of unstable oscillations of neural output can be seen late after k>670.

It can be concluded here that another advantage of HONU as of nonlinear adaptive model is that their weight update stability can be evaluated via monitoring of spectral radius of matrix of weight update dynamics, that is analogical to linear systems, which is again resulting from the in-parameter linearity of HONU.

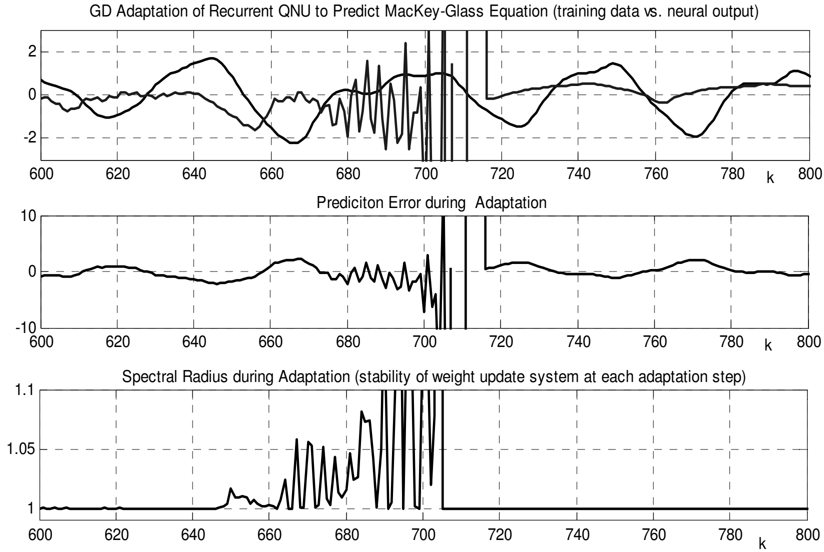


Fig. 7: Unstable adaptation –significantly violated stability condition (24) well before unusually large oscillations and divergence of neural output can be seen.

4 HONUS AND REAL SYSTEM IMPLEMENTATIONS

4.1 Static HONU for Process Data Reconciliation

Conventional MLP networks and HONU were studied and applied to neural network inspired data reconciliation [35][45][58] of hot steam turbine loop data and energetic boiler data (thermal plant Komořany, I. & C. energo). At every sampling moment, there were available 18 process variables such as steam pressures, temperatures, flows, and actual turbine power. The provided data had been averaged to consider steady state models of process data. Using multiple instances of static QNU and multiple instance of various configurations of feedforward MLP network and Levenberg-Marquardt algorithm, each of the 18 variables was individually and redundantly modeled (inspired by bootstrapping) and always the remaining 17 variables fed the neural network inputs. Because of complexity of this real system, it was not possible to identify the only relevant inputs to a model and all process variables were used on inputs, excluding the modeled one.

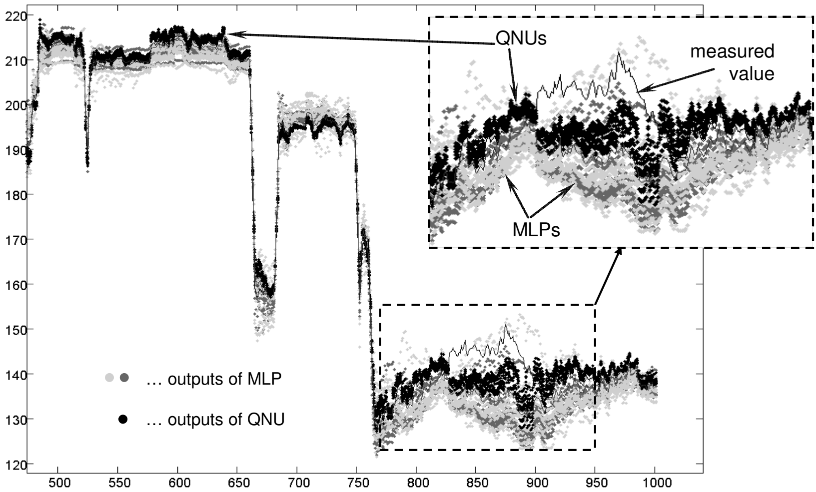


Fig. 8: Redundant modeling of hot steam flow by instances of single QNU and feedforward MLP neural networks for data reconciliation [45], QNU (darkest points) have smaller variance of neural outputs than MLPs.

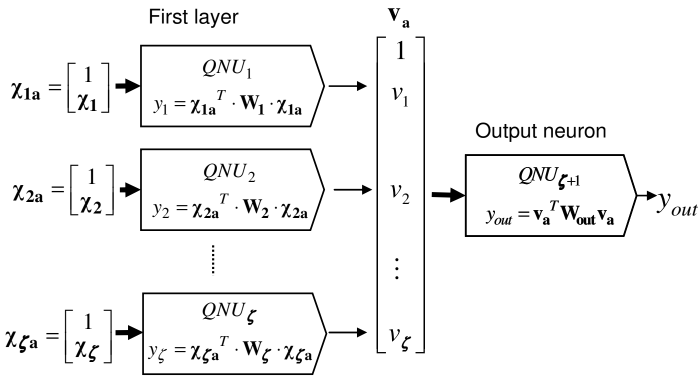


Fig. 9: The feed forward quadratic neural network (QNN).

Fig. 8 shows redundant modeling of hot steam flow steady states using other process variables, the solid line is the measured variable (steam flow) running in time (via horizontal axes) and the darkest points are redundantly modeled values by QNU; the lighter points are redundant values by MLP networks, and it shows that MLP had much larger variance of outputs than outputs of QNU that confirms the advantages of HONU over MLP regarding the testing accuracy.

To minimize to computational burden of static QNU that exponentially increases with the number of inputs, the feed-forward quadratic neural network (QNN) has been developed as a tool for data reconciliation of hot steam turbine loop process data in Fig. 9 and in (25) as follows

$$\mathbf{x}=\begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix}=\begin{bmatrix} \begin{bmatrix} x_1 & \cdots & x_m \end{bmatrix}^T \\ \begin{bmatrix} x_{m+1} & \cdots & x_{2m} \end{bmatrix}^T \\ \vdots \\ \begin{bmatrix} x_{\zeta \cdot m+1} & \cdots & x_n \end{bmatrix}^T \end{bmatrix}=\begin{bmatrix} \boldsymbol{\chi_1} \\ \boldsymbol{\chi_2} \\ \vdots \\ \boldsymbol{\chi_\zeta} \end{bmatrix},\ m<n \tag{25}$$

$$\boldsymbol{\chi_{1 \cdots \zeta-1}} \in \mathrm{R}^{m \times 1},\ \boldsymbol{\chi_\zeta}=\mathrm{R}^{(n-(\zeta-1)m)\times 1},\ \zeta=\mathrm{ceil}\left(\frac{n}{m}\right),$$

where proper details on the architecture and the training technique can be found in [46] (please notice that QNN becomes QNU for $m=n$).

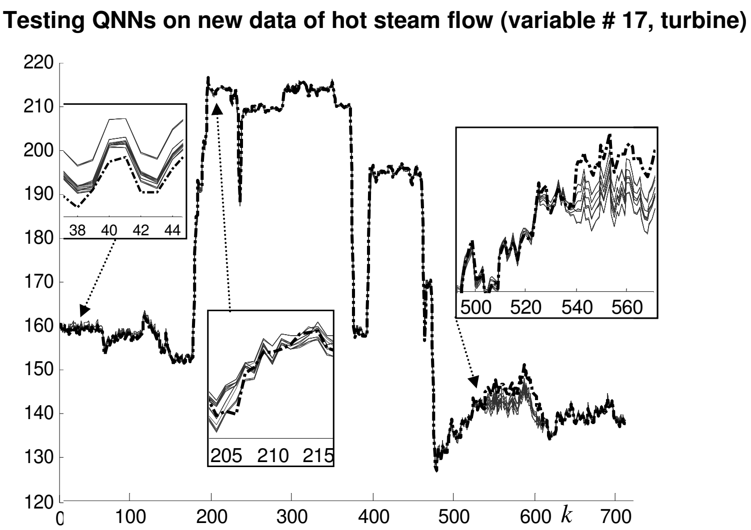
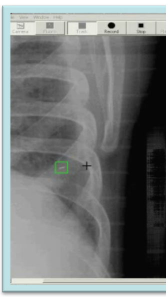


Fig. 10: Testing redundant static feedforward QNN models of the hot steam flow in a turbine loop on a new (testing) data by, m=3,4,…8, each 5x, total 30xQNN.

The proposed QNN architecture allows us to customize the number of total neural weights. In [46], it was shown how parameter m given in (25) affects the total numer of neural weights of QNN and that directly affects the accuracy of training, i.e., parameter m directly affects the duality of nonlinear approximation of QNU. The above introduced concept is straightforwardly applicable to HONUs of arbitrary order.

4.2 Lung Tumor Motion Prediction by HONU

During continuing cooperation with biomedical-engineering specialists from Tohoku University since 2009 [59][60], also static and dynamic HONUs have been studied for real time prediction of lung tumor motion along three body axes during a patient respiration and with the objective to reach 3-D mean absolute error (MAE) <1 mm for 1 second prediction horizon. So far (2012), neither recurrent neural networks, nor sophisticated autoregressive models, nor support vector machine techniques have been found performing with better accuracy of online prediction than the static QNU with Levenberg-Marquardt algorithm in a sliding window retraining approach [37][47][49][56][59][60]. Also the Jacobian matrix of QNU is a constant (contrary to MLP networks) and it depends merely on neural inputs significantly, and that accelerates real time computation where the sliding window retraining is carried out at every sample. Further, when Levenberg-Marquardt is implemented for neural weights individually, we may completely avoid matrix inversion in the update rule (there is more research needed on this topic, which is attractive and suitable even for undergraduate students).



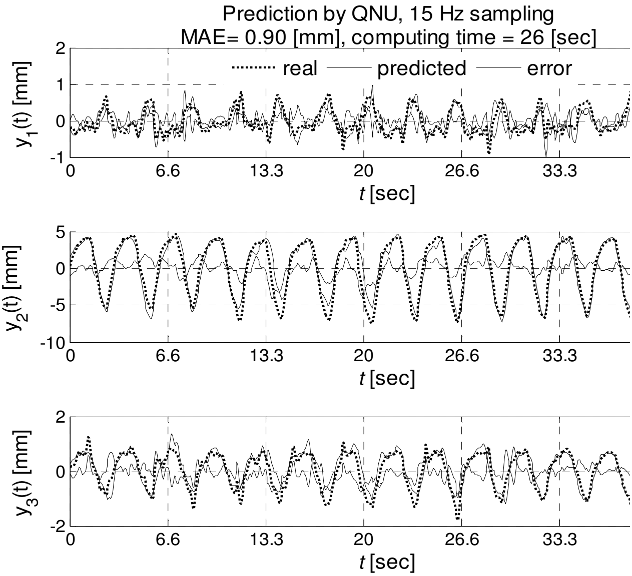


Fig. 11: The real time 1second prediction of lung tumor motion by QNU with real-time retraining in a sliding window; the left picture shows located lung tumor, and the axes show the online prediction of tumor position during respiration (error is the middle line at each axis) .

Next two figures show the typical comparison of estimated distribution of

2-D MAE of online prediction by MLP network vs. QNU and there is apparent that QNU has a smaller spread of error than the MLP network.

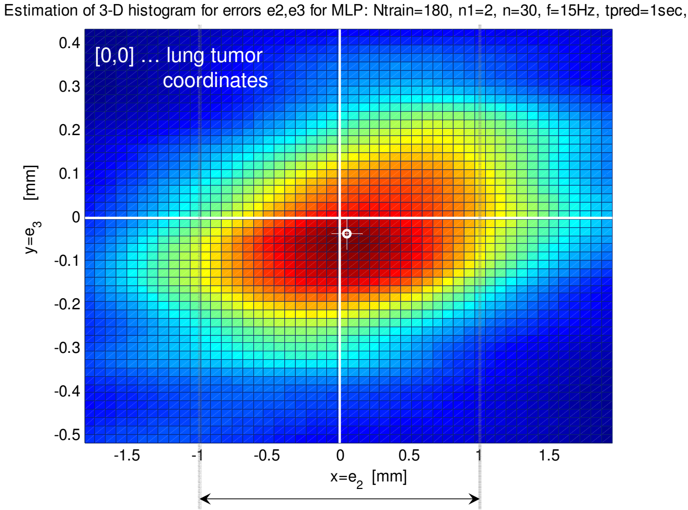


Fig. 12: Estimated distribution of 2-D mean absolute error by MLP network for data in Fig. 11.

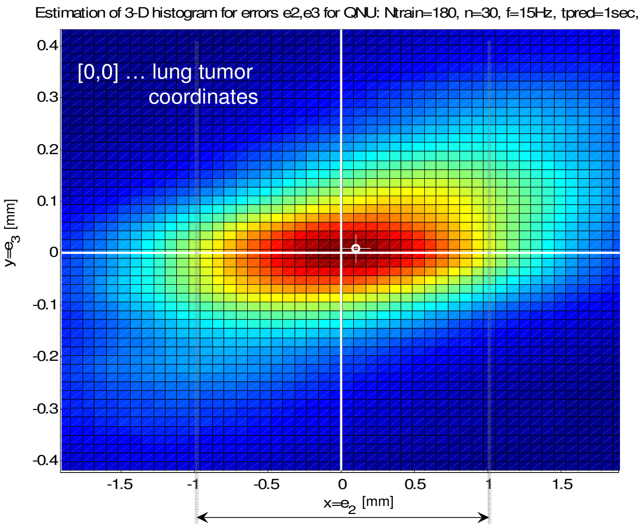


Fig. 13: Estimated distribution of 2-D MAE by QNU for data in Fig. 11.

4.3 Adaptive Control of Laboratory System

As another result of a supervised student work, dynamic QNU was implemented for an adaptive control of a real laboratory system in Master's thesis [55] with consequent publication of results in [31] and [48]. This student work was the practical implementation of the early introduced concept of QNU for adaptive control by Bukovsky et al [54] in 2003. Except the superiority of the control performance of adaptive QNU over conservative and linearly limited PID controller (Fig. 14), the student's work [55] demonstrates that the concept of HONUs is attractive and comprehensible to students.

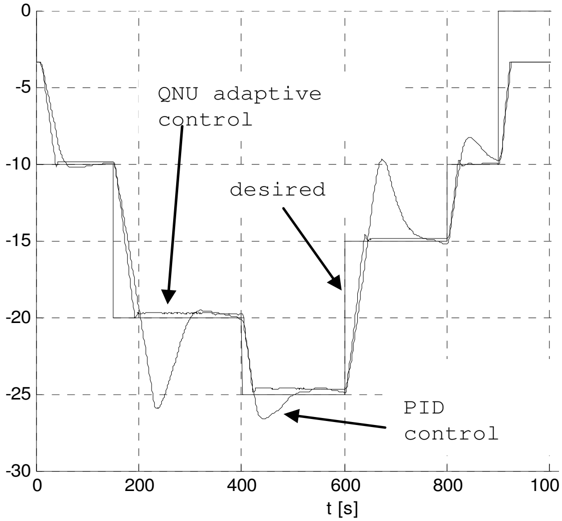


Fig. 14: Comparison of the state feedback QNU control with classical PID control of a real (laboratory) system (PID control has the overshoots)

4.4 NO_x Prediction of Pulverized Firing Boiler at EMĚ I

Within cooperation with I. & C. Energo, a.s.. [57], the neural networks also for prediction of NOx emissions was studied for process data validation and for possible control of the emissions of the pulverized firing boiler at Elektrárna Mělník 1 (Fig. 15). The requirement was to achieve an accurate model that would not involve measured O2, NOx, or CO as inputs; this was a unique requirement for which no solution had been known nor had it been found in literature. The pulverized boiler is highly nonstationary because of varying technical conditions inside the combustion chamber, because of varying and unknown quality of coal powder, and also because of the measurement outages that happen quite often on hourly basis. Therefore it was not principally likely to obtain neural network model that would reliably predict emissions of NOx on a long term basis. For those reasons, it was concluded that a possible use of even very sophisticated optimization algorithms would not help to obtain reliable long term predictive model without online retraining.

Pulverized firing boiler **K6 EMĚ1**

Parameters: 230 t/h, 540℃, 9,4 MPa, retyped to 250 t/h

Fig. 15: The photo and parameters of the studied boiler at powerplant Mělník 1 (an ideally clean combustion chamber is on the right picture)

After implementations and studies of performance of various predictive models from adaptive linear filters via static and dynamic MLP network, and static and dynamic HONUs on this boiler [38], the need for customable neural architecture was concluded while it was preferable to maintain real time computational efficiency for frequent retraining about every 30 minutes or less on a machine with common performance parameters (the input vector length >200; one minute sampling).

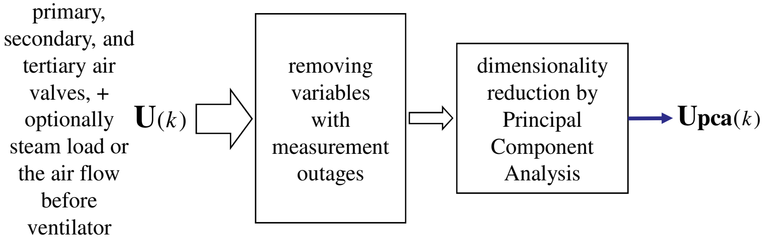


Fig. 16: The data preprocessing before each reconfiguration and retraining of neural network [32].

To handle the nonstationary nature of the studied boiler in EMĚ1 and after certain understanding to mutual dependencies of the measured input variables, we arrived to the data preprocessing technique that is briefly sketched in Fig. 16.

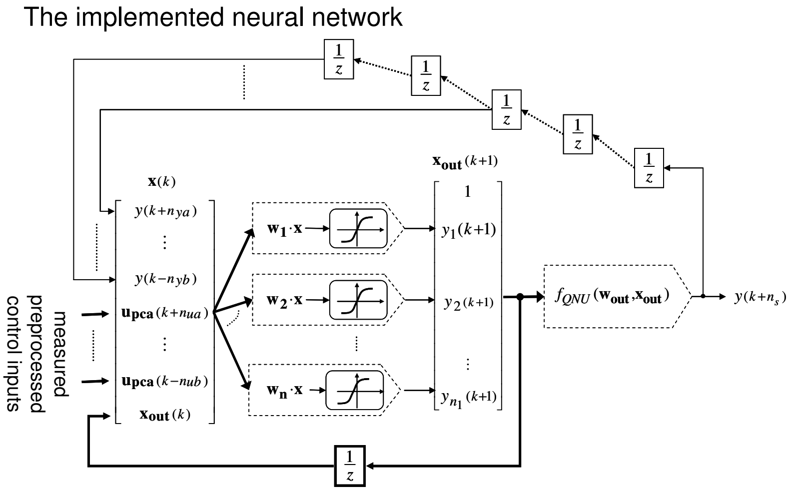


Fig. 17: The discrete-time recurrent neural network that was implemented for NOx prediction without any of measured O2, NOx, or CO on its input.

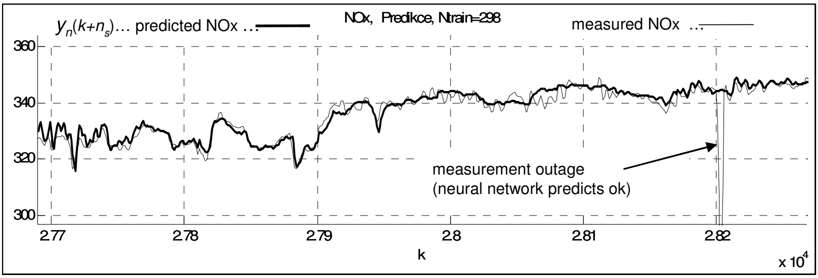


Fig. 18: Detail of NOx prediction; temporary NOx measurement outage occurred after sample k=2.82E+4; the output of the dynamic neural network substituted the measurement outage, the good neural network prediction is conditioned by a good retraining data prior the measurement outage of NOx.; sampling=1min, prediction horizon 3 minutes.

Then the nonconventional neural network in Fig. 17 was found acceptable for 3-minute NOx prediction (Fig. 18). The network has a hidden recurrent layer of perceptron neurons; the recurrent hidden layer introduces a filtering character of input measurement outages, it decreases the dimensionality of inputs to consequent QNU, and it limits and thus assures stability of the hidden layer (for the discrete-time networks). QNU was chosen as the output neuron for its nonlinear quality of approximation, and there are also tapped delay lines of predicted NOx to neural input because the measured NOx, when introduced as an external input, appeared to disqualify the network from proper prediction (then the network learnt just to blindly follow the previously measured NOx, that is common problem of improper use of ANN).The network was retrained every 30 minutes by backpropagation through time algorithm, which was implemented as a combination of RTRL technique and the Levenberg-Marquardt batch training [34] with recent measured data history of 298 samples (5 hours). Each retraining took less than 3 minutes of real computation time in Matlab on a PC (Win7, i7) and that is well suitable for real time implementations on a commonly available HW.

5 CONCLUSIONS

The paper briefly reviews history of artificial neural networks with focus on nonconventional neural architectures. The core of the non-conventionality of the discussed neural architectures consists in distinct conceptions of nonlinearity of neuronal models, which are different from conventional neuronal models of linear synaptic neural operation. The most commonly referenced works of authors that sketches the story of nonconventional neuronal models are in [1]–[30] and they are discussed in first sections of this paper. The author's contributions to the discussed topics can be found in [31]–[54] plus one supervised defended Master's thesis [55] (2008) and plus one co-supervised submitted PhD thesis [56] (09/2012); the works are discussed and referenced throughout later sections of this paper as well. The particular contribution of the author is the introduction of nonconventional neural units and their classification according to three fundamental aspects of modeling dynamical systems; these are the customable nonlinearity via polynomial order of the synaptic operation, the order of dynamics by the number of integrations of the synaptic output, and the implementation of adaptable time delays that further enhances approximation capability for systems with high order of dynamics. This introduction of neural units can be understood as a natural merge of early and more recent developments in artificial neural networks together with the conservative and well established approaches in the field of modeling and control engineering. Such a unifying classification of neuronal models as of standalone neural units while still mimicking the biological analogy has not been found in literature before. In the discussed applications to real world problems, the introduced neural units, esp. discrete-time static and dynamic HONU and their networks, appeared as practically useful. HONU can be used as standalone units or can be used in networked architectures. Even as standalone neural units, HONUs significantly outperformed conventional static and dynamic MLP neural networks when trained by non-heuristic adaptive algorithms that are suitable for small scale neural networks (such as real time recurrent learning or backpropagation through time) and when non heavy computational optimization techniques are needed. Fig. 19 and Fig. 20 demonstrate important properties of HONU that have been approved through many simulation and real data experiments and implementations.

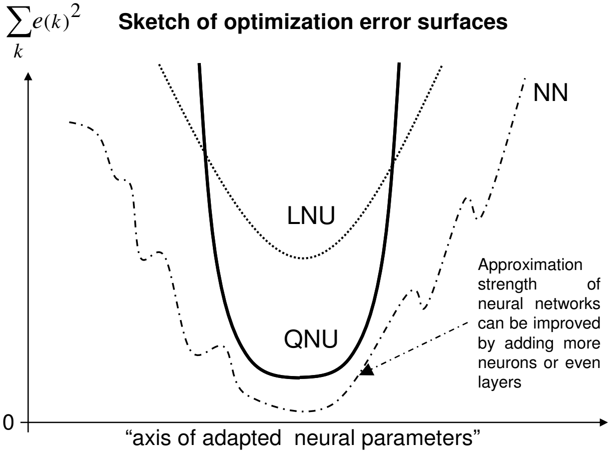


Fig. 19: QNU can be trained fast, significantly more precisely than linear models, and does not suffer from local minima issue for a given training data as nonlinear neural networks do when simple and feasible learning algorithms are used [31] [41][48].

An intuitive sketch of the overfitting problem with conventional neural networks

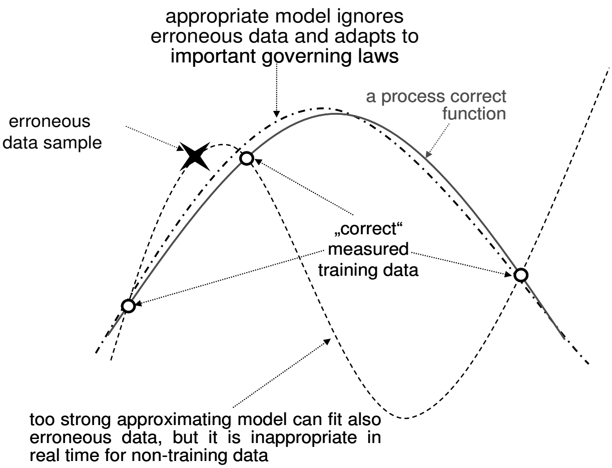


Fig. 20: HONUs are reasonable from the point of tradeoff between nonlinearity and the ability to learn only the prevailing governing laws in data.

It shall be recalled that the introduced concept of HONU is the concept of standalone neural units. Therefore, the theoretical research of novel neural networks built of HONU and with utilization of their all customable power is one interesting area for further research.

For applied research, HONUs are also a perspective tool for new and very fast hardware solutions of control, such as using the field programmable arrays (FPGA), chipset on board and embedded board solutions.

Of course, the performance of neural networks including HONUs on real systems is still substantially affected by availability of reasonable training data, which practically implies that a proper solution by neural networks turns in the end into a complex and customized approach, where certain level of understanding to the particular process and to data and to their measurement is inevitable.

At this very end, I would like to thank kind readers for their time and to beg them for a pardon for a fairly limited scope of the presented reviews, as it is impossible to concentrate here all topics that would deserve to be mentioned.

REFERENCES

[1] McCulloch, W. and Pitts, W. (1943). "A logical calculus of the ideas immanent in nervous activity". *Bulletin of Mathematical Biophysics*, 7:115 - 133.

[2] Warren Weaver, "Science and complexity," *American Scientist*, vol. 36, no. 948, pp. 536-544, 1968., pp. 449-456.

[3] Kinsner, W.: "System Complexity and Its Measures: How Complex Is Complex", *Advances in Cognitive Informatics and Cognitive Computing, Studies in Computational Intelligence*, Volume 323/2010, Springer-Verlag, 2010, pp.265-295.

[4] F. Rosenblatt: "The Perceptron: a probabilistic model for information storage and organization in the brain", *Psychol. Rev.*, 65 (1958), pp. 386–408

[5] Hopfield, John (1982). "Neural networks and physical systems with emergent collective computational abilities." *Proceedings of the National Academy of Sciences of the USA*, vol. 79, no. 8 (April 1982), pp. 2554-2558.

[6] M. M. Gupta, J. Liang, and N. Homma, *Static and Dynamic Neural Networks: From Fundamentals to Advanced Theory*, IEEE Press and Wiley-Interscience, pub. John Wiley & Sons, Inc., 2003.

[7] Konstantinos I. Diamantaras, Sun Yuan Kung: *Principal component neural networks:theory and application*, Wiley, ISBN: 978-0-471-05436-8, 1996

[8] Aapo Hyvärinen, Juha Karhunen, Erkki Oja :*Independent Component Analysis*, Wiley , ISBN: 978-0-471-40540-5,2001

[9] R. J. Williams and D. Zipser, "A learning algorithm for continually running fully recurrent neural networks," *Neural Comput*., vol. 1, pp. 270–280, 1989.

[10] P. J.Werbos, "Backpropagation through time: What it is and how to do it," Proc. IEEE, vol. 78, no. 10, pp. 1550–1560, Oct. 1990.

[11] A. G. Ivakhnenko, "Polynomial Theory of Complex Systems," *IEEE Tran. on Systems. Man, and Cybernetics. Vol. SMC-1* (4). pp. 364-378, 1971.

[12] B. Widrow and S. D. Stearns, *Adaptive Signal Processing*. Englewood Cliffs, NJ: Prentice-Hall, 1985.

[13] A. Benveniste, M. Metivier, and P. Priouret, *Adaptive Algorithms and Stochastic Approximation*. New York: Springer-Verlag, 1990.
[14] W.-P. Ang and B. Farhang-Boroujeny, "A new class of gradient adaptive step-size LMS algorithms," *IEEE Trans. Signal Process*., vol. 49,no. 4, pp. 05–810, Apr. 2001.
[15] D. P. Mandic, "A Generalised Normalised Gradient Descent Algorithm", *IEEE Signal Processing Letters,* Vol. 11, No. 2, pp. 115-118, 2004.
[16] V. Mathews and Z. Xie, "A stochastic gradient adaptive filter with gradient adaptive step size*," IEEE Trans. Signal Process*., vol. 41, no. 6, pp. 2075–2087, Jun. 1993.
[17] Q. Song, Y. Wu, Y. Ch. Soh: "Robust Adaptive Gradient-Descent Training Algorithm for Recurrent Neural Networks in Discrete Time Domain", *IEEE Transactions on Neural Networks*, Vol.19, No.11, pp. 1841-53, 2008.
[18] V. Vapnik and A. Lerner: *Pattern Recognition Using Generalized Portrait Method,* Automat. Remote Contr., 24, 774–780, 1963.
[19] Young-Seok Choi, Hyun-Chool Shin, Woo-Jin Song: "Robust Regularization for Normalized LMS Algorithms", *IEEE Transactions on Circuits and Systems—II:Express Briefs*, Vol. 53, No.8, 2006.
[20] Y. Shin and J. Ghosh, "The Pi-sigma Network: An Efficient Higher-order Neural Network for Pattern Classification and Function Approximation," *Proc. Int. Joint Conf. on Neural Networks* (pp. 13-18), 1991.
[21] R. W. Softky and D. M. Kammen, "Correlations in high dimensional or asymmetrical data sets: Hebbian neuronal processing," *Neural Networks*, 4, pp. 337-347, 1991.
[22] J. G. Taylor and S. Commbes, "Learning higher order correlations," *Neural Networks*, 6, pp. 423-428, 1993.
[23] M. S. Chen and M. T. Manry, "Conventional modeling of the multilayer perceptron using polynomial basis functions," *IEEE Trans. Neural Networks*, v.4,pp.164–166, 1993.
[24] W. Schmidt, and J. Davis, "Pattern recognition properties of various feature spaces for higher order neural networks," *IEEE Trans. Pattern Analysis and Machine Intelligence*, 15, pp. 795-801, 1993.
[25] E. Kosmatopoulos, M. Polycarpou, M. Christodoulou, and P. Ioannou, "High-Order Neural Network Structures for Identification of Dynamical Systems," *IEEE Trans. on Neural Networks*, vol. 6, no. 2, pp. 422-431, March 1995.
[26] M. Heywood and P. Noakes, "A framework for improved training of Sigma-Pi networks," *IEEE Trans. on Neural Networks,* vol. 6, pp. 893–903, July 1996.
[27] N. Y. Nikolaev and H. Iba, "Learning Polynomial Feedforward Neural Network by Genetic Programming and Backpropagation": *IEEE Trans. on Neural Networks*, vol. 14, no. 2, pp.337-350, March, 2003.
[28] N. Y. Nikolaev and H. Iba, *Adaptive Learning of Polynomial Networks: Genetic Programming, Backpropagation and Bayesian Methods, Series: Genetic and Evolutionary Computation Springer*, New York, XIV, ISBN: 0-387-31239-0, 2006.
[29] R. Durbin and R. Rumelhart, "Product units: A computationally powerful and biologically plausible extension to backpropagation networks," *Neural Comput*., vol. 1, pp. 133–142, 1989.
[30] J. Wesley Hines, "A logarithmic neural network architecture for unbounded nonlinear function approximation" in *Proc. Int. Conf. Neural Networks* 1996, 2, pp. 1245–1250.
[31] Rodriguez , R., Bukovsky, I., Homma, N.: "Potentials of Quadratic Neural Unit for Applications", in *International Journal of Software Science and Computational Intelligence (IJSSCI)* ,vol 3, issue 3, IGI Global, Publishing, Hershey PA, USA ISSN 1942-9045, DOI: 10.4018/jssci.2011070101, pp.1-12, July-September 2011.
[32] Bukovsky, I., Kolovratnik M.: „A Neural Network Model for Predicting NOx at the Mělník 1 Coal-powder Power Plant", *Acta Polytechnica* ,Vol. 52 No. 3/2012,pp.17-22, ISSN 1805-2363, 2012.

[33] Bukovsky, I., Rodriguez, R., Bila, J., "Prospects of Gradient Methods for Nonlinear Control", *Strojárstvo Extra*, MEDIA/ST, s.r.o. publ. house, 2012, ISSN 1335-2938.
[34] Gupta, M., M., Bukovsky, I., Homma, N. , Solo M. G. A., Hou Z.-G.: "Fundamentals of Higher Order Neural Networks for Modeling and Simulation", in *Artificial Higher Order Neural Networks for Modeling and Simulation*, ed. M. Zhang, IGI Global, 2012, ISBN 978-1-4666-2175-6, pp.102-132, (in print 09/2012).
[35] Bai SH, McLean DD, Thibault J: "Impact of model structure on the performance of dynamic data reconciliation", *Comp. & Chem. Engineering* 31 (3): 127-135, 2007
[36] Bukovsky, I., Kolovratnik, M.: "Neural Network Model for Prediction of NOx at Coal-Powder Powerplant Mělník 1", *ERIN 2012*, CTU in Prague, 25–27 , 2012, Czech Rep.
[37] Ichiji, K., Homma, N., Bukovsky, I., Yoshizawa, M..: "Intelligent Sensing of Biomedical Signals - Lung Tumor Motion Prediction for Accurate Radiotherapy ", in proceedings of *2011 IEEE Symposium Series on Computational inteligence (SSCI), IEEE Workshop CompSens 2011*: ISBN 978-1-4577-0470-3, Paris 2011, pp. 65-72.
[38] Bukovský, I.: Křehlík, K.: *Testy neuronového modelu kotle elektrárny Mělník I,* Výzkumná zpráva č. 8-ZI00069/ E06) pro I. & C. Energo, a.s. U12110, Fakulta strojní, ČVUT v Praze, 61 stran. 2011,
[39] Gupta, M., M, Homma, N., Hou, Z-G., Solo, M., G., Bukovsky, I.: "Higher Order Neural Networks: Fundamental Theory and Applications", in *Artificial Higher Order Neural Networks for Computer Science and Engineering: Trends for Emerging Applications*", ed. M. Zhang, IGI Global,2010.
[40] Bukovsky, I., Bila. J,.: "Foundation and Classification of Nonconventional Neural Units and Paradigm of Nonsynaptic Neural Interaction" in *Discoveries and Breakthroughs in Cognitive Informatics and Natural Intelligence* within the series of the *Advances in Cognitive Informatics and Natural Intelligence* (*ACINI*), (invited chapter), ed. Y. Wang, IGI Publishing, Hershey PA, USA, Nov.. 2010. ISBN: 978-1-60566-902-1, p.508-523.
[41] Bukovsky, I., Homma, N.: "Dynamic Backpropagation and Prediction (Dynamický backpropagation a predikce)" (in Czech), In: *Automatizace*, Vol. 53, No. 1-2, Prague, Czech Republic, Jan-Feb 2010, p.61-66, ISSN 0005-125X.
[42] Bukovsky, I., Homma, N.: "Dynamic Backpropagation (Dynamický backpropagation)" (in Czech), In: *Automatizace*, Vol. 52, No. 10, Prague, Czech Republic, Oct 2009, p.586-590, ISSN 0005-125X.
[43] Bukovsky, I., Hou, Z-G., Bila, J., Gupta, M., M.: "Foundation of Nonconventional Neural Units and their Classification", *International Journal of Cognitive Informatics and Natural Intelligence (IJCiNi)*, 2(4), October-December 2008, IGI Publishing, Hershey PA, USA, pp.29-43, ISSN 1557-3958.
[44] Bukovsky, I., Bila, J., Gupta, M., M.: "Linear Dynamic Neural Units with Time Delay for Identification and Control (in Czech)", In: *Automatizace*, Vol. 48, No. 10, Prague, Czech Republic, Oct 2005, p. 628-635. ISSN 0005-125X.
[45] Bukovsky, I., Bila. J: Program system for advanced reconciliation of process data: Program System I. & C. NEURECON (In Czech, *Programový systém* pro *pokročilé validování provozních dat*: *Programový systém* I. & C. NEURECON), závěrečná zpráva pro I. & C. Energo, a.s. U12110, Fakulta strojní, ČVUT v Praze, 2008, 107 stran.
[46] Bukovsky, I., Lepold, M., Bila J.: "Quadratic Neural Unit and its Network in Validation of Process Data of Steam Turbine Loop and Energetic Boiler",*WCCI 2010*, *IEEE Int. Joint. Conf. on Neural Networks IJCNN*, Barcelona, Spain, 2010.
[47] Bukovsky, I., Ichiji, K., Homma, N., Yoshizawa, M.: "Testing Potentials of Dynamic Quadratic Neural Unit for Prediction of Lung Motion during Respiration for Tracking Radiation Therapy", *WCCI 2010, IEEE Int. Joint. Conf. on Neural Networks IJCNN*, Barcelona, Spain, 2010.

[48] Bukovsky, I., Homma, N., Smetana, L., Rodriguez, R., Mironovova M., Vrana S.,: “Quadratic Neural Unit is a Good Compromise between Linear Models and Neural Networks for Industrial Applications”, *ICCI 2010 The 9th IEEE International Conference on Cognitive Informatics*, Tsinghua University, Beijing, China, July 7-9, 2010.
[49] Rodriguez R., Ichiji K., Bukovsky I., Bila J., Homma N.: “Lung Motion Prediction by Static Neural Networks“, *4th International Symposium on Measurement, Analysis and Modelling of Human Functions,IMEKO,* Prague, 2010
[50] Simeunovic , G., Bukovsky, I.: The Implementation of the Dynamic-Order-Extended Time-Delay Dynamic Neural Units to Heat Transfer System Modelling, *16th International Conference on Nuclear Engineering (ICONE 16)*, Orlando, Florida, USA, May 11-15, 2008, ISBN 0-7918-3820-X.
[51] Bukovsky, I., Hou, Z-G., Gupta, M., M., Bila, J.: “Foundation of Notation and Classification of Nonconventional Static and Dynamic Neural Units”, *ICCI 2007 The 6th IEEE International Conference on Cognitive Informatics*, California , USA, 2007, ISBN: 978-1-4244-1328-7.
[52] Bukovsky, I., Simeunovic, G.: “Dynamic-Order-Extended Time-Delay Dynamic Neural Units*”, 8th Seminar on Neural Network Applications in Electrical Engineering, NEUREL-2006*, IEEE (SCG) CAS-SP, Belgrade, 2006
[53] Bukovsky, I., Bíla, J., Gupta, M., M.: “Stable Neural Architecture of Dynamic Neural Units with Adaptive Time Delays*”, 7th International FLINS Conference on Applied Artificial Intelligence*, 2006, p. 215-222, ISBN 981-256-690-2
[54] Bukovsky I., S. Redlapalli and M. M. Gupta : “Quadratic and Cubic Neural Units for Identification and Fast State Feedback Control of Unknown Non-Linear Dynamic Systems”, proceedings of *ISUMA 2003*, IEEE Computer Soc., 2003, USA, pp.330-334.
[55] Ladislav Smetana: *Nonlinear Neuro¬Controller for Automatic Control Laboratory System*, Master’s Thesis (in Czech), Czech Tech. Univ. in Prague, 2008. (supervisor Ivo Bukovsky)
[56] Rodriguez, R: *Lung Tumor Motion Prediction by Neural Networks,* a submitted PhD thesis, FME, CTU in Prague, sup. Jiri Bila, co-sup: Ivo Bukovsky, submitted 09/2012
[57] FR-TI1/538 - Technologie pro zvýšení účinnosti spalování a pro omezení emisí kotlů na fosilní paliva ,(2009-2012, projekt MPO)
[58] FI-IM4/062 - Vývoj metod a SW nástrojů pro zajištění spolehlivosti dat pro řízení provozu energetických a průmyslových zařízení. (2007-2008, projekt MPO/FI)
[59] The Matsumae International Foundation fellowship to Tohoku University , 2010.
[60] Tohoku-Hirosaki-Varian joint research project, fund.Varian Medical Systems, Inc., 2011.

Ing. Ivo Bukovský, Ph.D.

Birth date March 4, 1975 ***Place*** Prague

Nationality Czech ***Citizenship*** Czech Republic

Education

7.9.2007, Ph.D., Graduated in the field of Technical Cybernetics from the Department of Instrumentation and Control Engineering U12110, Faculty of Mechanical Engineering, Czech Technical University in Prague

2002, Ing., Graduated in the field of Automatic Control and Engineering Informatics from the Dpt. of Inst. and Cont. Eng., FME, CTU in Prague

Job positions

2010–today Head of the Division of Automatic Control and Engineering Informatics U12110.3, CTU in Prague, FME

2005–today Assistant Professor at the CTU in Prague, Faculty of Mechanical Engineering, Department of Instrumentation and Control Engineering U12110

2005–2007 Professional instructor of MS Office including MS Access at professional computer school Gopas

Fellowships and visiting research

2011, Visiting research at the Tohoku University, Sendai, Japan, (2 months)

2010, Visiting research at the University of Manitoba, (1 month)

2009, The Matsumae International Fellowships for visiting research at the Tohoku University, Japan, (6 months)

2003, NATO Science Fellowships to University of Saskatchewan, Canada (7 months)

Memberships

IEEE Computational Intelligence Society; INSTICC Institute for Systems and Technologies of Information, Control and Communication

Activity in International Scientific Community

Associated Editor of IEEE Transactions on Neural Networks and Learning Systems (2012) and of IEEE Transactions on Neural Networks (2011)

Co-supervised PhD thesis: 1 PhD thesis was recently submitted in 09/2012

Defended Master Theses: 3x

Defended Bachelor Theses: 5x

Publication Activity:

WoS or Scopus indexed journal papers: 2x

Book chapters: 4x (1x Springer, 3x IGI Global)

Int. rev. journals: 4 (local journals:4)

WoS or IEEE conference papers: 16x (total int. conferences: 25)

Citations:

Total number of independent publications on WoS or Scopus that cites: 5

Recent citations currently not found on WoS or Scopus: 1x IEEExplore, 1x InTech,