New learning options for polynomial neurons used as a controller

Martin Veselý[1*]

[1,2] *ČVUT v Praze, Fakulta strojní, Ústav přístrojové a řídící techniky, Technická 4, 166 07 Praha 6, Česká republika*

Abstract

The article presents a new method for improving learning rate of neural unit as a controller. Computations of static characteristic, static gain and offset of a neural unit and a closed loop are shown. This information is added to optimization function used for learning of the neural unit as a controller.

Keywords: HONU, LNU, QNU, Controller

1. Introduction

The article deals with using either linear or quadratic neural units as a discrete controller for continuous controlling [1],[2]. Using LNU and QNU as a controller is suitable for controlling the following types of systems:

- systems with partly nonlinear behavior
- systems with transport delay
- systems which are partly changed behavior during proses or their life
- systems which have unknown behavior or we don't want to know

The main advantages of this type of controller are:

- explicit expression $u_{(k)}$
- ability to learn in real-time [3]
- optimization function with only one local minimum [4]

This article shows new options of extending the optimization function for improving learning rate. The idea which was used is: If it's possible to express static characteristic of neural units and closed loops, then it is possible to use this information for extension of the optimization function.

This article also presents the basics of LNU and QNU used as a model and a controller; and their learning by using reference model; computations of static characteristics, offsets and static gains of LNU and QNU and closed loop which h consists of neural unit as a controller and neural unit as a model.

2. Basic of Neural Units as Controller and Model and Real-Time Learning

The basic principle of controlling with reference model is to force the behavior of the reference model onto the closed loop fig.1 [5].

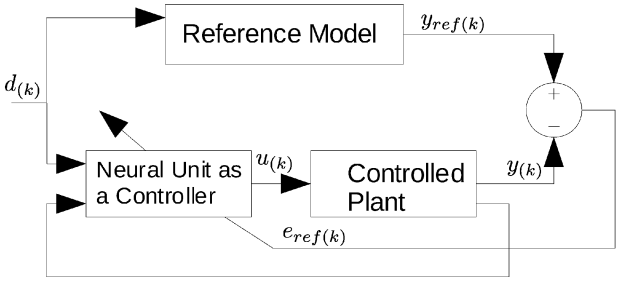

Fig. 1: Control Scheme of the Neuro-Controller with Reference Model

The input vector to the controller consists of previous samples of desired values and previous samples of process variables (1).

$$\xi_{(k)} = \begin{bmatrix} 1 \\ y_{(k-1)} \\ y_{(k-2)} \\ \vdots \\ y_{(k-my)} \\ d_{(k)} \\ d_{(k-1)} \\ \vdots \\ d_{(k-md+1)} \end{bmatrix} \tag{1}$$

The controller output is calculated as LNU (2) or as QNU (3).

$$u_{(k)} = \mathbf{V}_{(k)} \cdot \xi_{(k)} \tag{2}$$

$$u_{(k)} = \xi_{(k)}^{T} \cdot \mathbf{V}_{(k)} \cdot \xi_{(k)} \tag{3}$$

It's necessary to know the relation between controller output u and process variable y for learning of neural unit as controller. Therefore the model neural unit is built and taught similarly as a controller neural unit. The output of the controller neural unit (4) for LNU and (5) for QNU is calculated from the input vector (6).

$$y_{(k)} = \mathbf{W}_{(k)} \cdot \mathbf{X}_{(k)} \tag{4}$$

$$y_{n(k)} = \mathbf{X}_{(k)}^{T} \cdot \mathbf{W}_{(k)} \cdot \mathbf{X}_{(k)} \tag{5}$$

* Kontakt na autora: Martin.Vesely@fs.cvut.cz

$$X_{(k)} = \begin{bmatrix} 1 \\ y_{(k-1)} \\ y_{(k-2)} \\ \vdots \\ y_{(k-ny)} \\ u_{(k-1)} \\ u_{(k-2)} \\ \vdots \\ u_{(k-nu)} \end{bmatrix} \tag{6}$$

One of the methods which can be used for real-time learning of a neural unit as a controller is Gradient Descent [6]. The main advantages are simplicity and no need of using inverse matrix (controller must be able to update weights every sample and must be applicable in simple control units). Method Gradient Descent is shown in (7) for QNU for updating weights of neural unit as controller.

$$v_{ij(k+1)} = v_{ij(k)} - \mu_{(k)} \cdot \frac{\partial Q_{(k)}}{\partial v_{ij(k)}} \tag{7}$$

Where the optimization function $Q_{(k)}$ is calculated without a priori informavtion as follow (8).

$$Q_{(k)} = \frac{1}{2}(y_{ref(k)} - y_{n(k)})^2 \tag{8}$$

3. Statics characteristic of neural units as a model

A static characteristic of a dynamic system is a relation between its outputs and inputs in steady states. Steady states means that the inputs and the outputs are constant in time. Then the input vector (6) can be rewritten for static state (9).

$$X_{st} = \left[1, y_{st}, \ldots, y_{st}, u_{st}, \ldots, u_{st},\right]^T \tag{9}$$

3.1. Static Characteristic of LNU

Equation (4) can be rewritten to tensor form and divided by inputs (10).

$$y_{(k)} = \sum_{i=1}^{n} x_i w_i = w_1 + \sum_{i=2}^{ny+1} x_i w_i + \sum_{ny+2}^{n} x_i w_i \tag{10}$$

The static output can be calculated from static input (9) and equation (10) as followed (11).

$$\begin{aligned} y_{st} &= \sum_{i=1}^{n} x_{st_i} w_i \\ &= w_1 + y_{st} \cdot \sum_{i=2}^{ny+1} w_i + u_{st} \cdot \sum_{ny+2}^{n} w_i \end{aligned} \tag{11}$$

From (11) is possible to express static characteristic (14) of LNU (4), and static gain K_{st} (12)and $offset$(13).

$$K_{st} = \frac{\partial y_{st}}{\partial u_{st}} = \frac{\sum_{ny+2}^{n} w_i}{1 - \sum_{i=2}^{ny+1} w_i} \tag{12}$$

$$offset = \frac{w_1}{1 - \sum_{i=2}^{ny+1} w_i} \tag{13}$$

$$y_{st} = K_{st} \cdot u_{st} + offset \tag{14}$$

3.2. Static Characteristic of QNU

The static characteristic of QNU is compiled like a static characteristic of LNU. Equation (3) can be rewritten to tensor form and divided by input (15).

$$\begin{aligned} y(k) &= \sum_{i=1}^{n} \sum_{j=i}^{n} w_{ij} x_i x_j \\ &= w_{11} + \sum_{j=2}^{n_y+1} x_j w_{1j} + \sum_{j=n_y+2}^{n} x_j w_{1j} \\ &+ \sum_{i=1}^{n_y+1} \sum_{j=i}^{n_y+1} w_{ij} x_i x_j + \sum_{i=n_y+2}^{n} \sum_{j=i}^{n} w_{ij} x_i x_j \\ &+ \sum_{i=2}^{n_y+1} \sum_{j=n_y+2}^{n} w_{ij} x_i x_j \end{aligned} \tag{15}$$

The static output can be calculated from static input (9) and equation (15) as follow (16).

$$\begin{aligned} y_{st} &= w_{11} + y_{st} A + u_{st} B \\ &+ y_{st}^2 C + u_{st}^2 D + y_{st} u_{st} E \end{aligned} \tag{16}$$

Where constants A, B, C, D, E are following (17).

$$\begin{aligned} A &= \sum_{j=2}^{n_y+1} w_{1j} \\ B &= \sum_{j=n_y+2}^{n} w_{1j} \\ C &= \sum_{i=1}^{n_y+1} \sum_{j=i}^{n_y+1} w_{ij} \\ D &= \sum_{i=n_y+2}^{n} \sum_{j=i}^{n} w_{ij} \\ E &= \sum_{i=2}^{n_y+1} \sum_{j=n_y+2}^{n} w_{ij} \end{aligned} \tag{17}$$

From (16) it is possible to express static characteristic () of LNU (18)

$$\begin{aligned} y_{st1,2} &= \frac{1 - A - u_{st} E}{2C} \\ &\pm \frac{\sqrt{(A + u_{st} E - 1)^2 - 4C(w_{11} + u_{st} B + u_{st}^2 D)}}{2C} \end{aligned} \tag{18}$$

Then the static gain is following:

$$K_{st1,2} = \frac{\partial y_{st}}{\partial u_{st}} = \frac{E}{2C} \pm \frac{2E(A + u_{st}E - 1) - 4BC - 8CD}{4C\sqrt{(A + u_{st}E - 1)^2 - 4C(w_{11} + u_{st}B + u_{st}^2 D)}} \tag{19}$$

4. Static Characteristic of Closed Loop

The closed loop consists of a neural unit as a controller and a neural unit as a model of controlled plant. The static form of input vector to the neural unit as a controller is:

$$\xi_{st} = \left[1, y_{st}, \ldots, y_{st}, d_{st}, \ldots, d_{st}\right]^T \tag{20}$$

4.1. Static Characteristic Closed Loop LNU as a Controller and LNU as a Model

The static characteristic of LNU as a controller can be built analogously with LNU as a model:

$$u_{st} = w_1 + y_{st} \sum_{i=2}^{my+1} w_i + d_{st} \sum_{my+2}^{m} w_i \tag{21}$$

By integrating the (21) into (14) and simplifying we get static characteristic (24) static gain (22) and offset (23) of closed loop.

$$K^{CL} = \frac{\partial y_{st}}{\partial d_s t} = \frac{\sum_{my+2}^{m} v_i \cdot \sum_{ny+2}^{n} w_i}{1 - \sum_{i=2}^{ny+1} w_i - \sum_{i=2}^{my+1} v_i \cdot \sum_{ny+2}^{n} w_i} \tag{22}$$

$$offset^{CL} = w_1 + v_1 \cdot \sum_{ny+2}^{n} w_i \tag{23}$$

$$y_{st} = K_{st}^{CL} \cdot d_{st} + offset^{CL} \tag{24}$$

4.2. Static Charakteristic Closed Loop QNU as a Controler and QNU as a Model

The static characteristic of QNU as a controller can be built analogously with QNU as a model:

$$y_{st} = v_{11} + y_{st}A^r + d_{st}B^r + y_{st}^2 C^r + d_{st}^2 D^r + y_{st}d_{st}E^r \tag{25}$$

Where constants A^r, B^r, C^r, D^r, E^r are following:

$$\begin{aligned} A^r &= \sum_{j=2}^{m_y+1} v_{1j} \\ B^r &= \sum_{j=m_y+2}^{m} v_{1j} \\ C^r &= \sum_{i=1}^{m_y+1} \sum_{j=i}^{m_y+1} v_{ij} \\ D^r &= \sum_{i=m_y+2}^{m} \sum_{j=i}^{m} v_{ij} \\ E^r &= \sum_{i=2}^{m_y+1} \sum_{j=m_y+2}^{m} v_{ij} \end{aligned} \tag{26}$$

By integrating (25) into (16) and simplifying we get following equation:

$$\begin{aligned} 0 = {} & y_{st}(-1 + 2DA^r v_{11} + BA^r + Ev_{11} + A) \\ & + y_{st}^2(DAr^2 + EA^r + 2DC^r v_{11} + BC^r + C) \\ & + y_{st}^3(2 * DA^r C^r + EC^r) \\ & + y_{st}^4(DC^{r\,2}) \\ & + d_{st}(2DB^r v_{11} + BB^r) \\ & + d_{st}^2(2DD^r v_{11} + BD^r + DB^{r\,2}) \\ & + d_{st}^3(2DB^r D^r) \\ & + d_{st}^4(DD^{r\,2}) \\ & + y_{st}d_{st}(2DA^r B^r + EB^r + 2DE^r v_{11} + BE^r) \\ & + y_{st}^2 d_{st}(2DA^r E^r + DB^r C^r + EE^r) \\ & + y_{st}d_{st}^2(2DA^r D^r + 2DB^r E^r + ED^r) \\ & + y_{st}^2 d_{st}^2(2DC^r D^r + DE^{r\,2}) \\ & + y_{st}^3 d_{st}(2DC^r E^r) \\ & + d_{st}^3 y_{st}(2DD^r E^r) \\ & + Dv_{11}^2 + Bv_{11} + w_{11} \end{aligned} \tag{27}$$

5. Expansion Optimization Function

As was mentioned in paragraph 2, the basic principle of controlling with reference model is to force the behavior of the reference model onto the closed loop. Because the static characteristic, static gain and offset of reference model are known, it's possible to add this part. A reference model with a linear static characteristic without offset and with static gain equal to 1 is supposed. The optimization function is built as a function which minimum value is equal to 0 only when the behavior of the closed loop is optimal.

5.1. Optimization Function of Closed Loop LNU as a Controller and LNU as a Model

The optimization function for closed loop LNU as a controller and LNU as a model can build as following:

$$Q_{(k)} = \frac{\alpha_1}{2}(y_{ref(k)} - y_{n(k)})^2 + \frac{\alpha_2}{2}(1 - K_{(k)}^{CL})^2 + \frac{\alpha_3}{2}(offset_{(k)}^{CL})^2 \tag{28}$$

The optimization function (28) means, that the optimum is when output of reference model is equal output of neural unit as a plant (5) and static gain of closed loop is equal to 1 and offset of closed loop is 0 .

5.2. Optimization Function of Closed Loop QNU as a Controller and QNU as a Model

The static characteristic of QNU-QNU is nonlinear. The static characteristic is linear only when static gain is constant. Therefore the optimum will be, when components in (27) containing other variables than y_{st} or d_{st} are 0. Then the static gain can be calculated as the constant (29) and the optimization function can be expressed from (27) and (29) as (30).

$$K_{st}^{CL} = \frac{2DB^r v_{11} + BB^r}{(1 - 2DA^r v_{11} - BA^r + Ev_{11} - A)} \tag{29}$$

$$\begin{aligned} Q_{(k)} = & \frac{\alpha_1}{2}(y_{ref(k)} - y_{(k)})^2 \\ & + \frac{\alpha_2}{2}(1 - K_{st}^{CL})^2 \\ & + \frac{\alpha_3}{2}(DAr^2 + EA^r + 2DC^r v_{11} + BC^r + C)^2 \\ & + \frac{\alpha_4}{2}(2DA^r C^r + EC^r) \\ & + \frac{\alpha_5}{2}(DC^{r\,2})^2 \\ & + \frac{\alpha_6}{2}(2DD^r v_6 + BD^r + DB^{r\,2})^2 \\ & + \frac{\alpha_7}{2}(2DB^r D^r)^2 \\ & + \frac{\alpha_8}{2}(DD^{r\,2})^2 \\ & + \frac{\alpha_9}{2}(2DA^r B^r + EB^r + 2DE^r v_{11} + BE^r)^2 \\ & + \frac{\alpha_{10}}{2}(2DA^r E^r + DB^r C^r + EE^r)^2 \\ & + \frac{\alpha_{11}}{2}(2DA^r D^r + 2DB^r E^r + ED^r)^2 \\ & + \frac{\alpha_{12}}{2}(2DC^r D^r + DE^{r\,2})^2 \\ & + \frac{\alpha_{13}}{2}(2DC^r E^r)^2 \\ & + \frac{\alpha_{14}}{2}(2 * DD^r E^r)^2 \\ & + \frac{\alpha_{15}}{2}(Dv_{11}^2 + Bv_{11} + w_{11})^2 \end{aligned} \tag{30}$$

6. Experimental analysis

In this chapter, a comparison between learning rates of controller with and without a priori information in simulations is presented.

5.1. Linear Plant Control

The learning rate was tested on linear plant described in differential equation (31).

$$u_{(t)} = 0.14\ddot{y}_{(t)} + 0.12\dot{y}_{(t)} + 0.16y_{(t)} \tag{31}$$

Properties of controlled plant and desired behavior of closed loop (properties of reference model) are in tab.1

Table 1. *Properties of the Controlled Plant and the Reference Model*

	Controlled Plant	Reference Model
Static Gain	6.07	1
Roots	$-0.45 \pm i$	$-0.64; -1.58$
Transport Delay	$0.01s$	$0.01s$

The test was performed with relearning LNU as the model of the controlled plant. The simulation of real-time learning controller was with initial weighs $v_{\forall i(k=0)} = 0$. Results are shown in fig.. The sample time was $0.01s$.

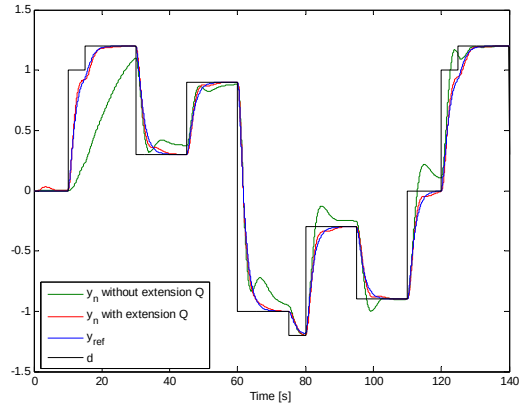

Fig. 2: Results of Learning QNU as a Controller with and without A Priori Information

5.2. Nonlinear Plant Control

The learning rate was tested on linear plant described in differential equation (32).

$$u_{(t)} = 0.14\ddot{y}_{(t)} + 0.12\dot{y}_{(t)} + 0.16sin(y_{(t)}) \tag{32}$$

Properties () depend on actual y and are shown in fig..

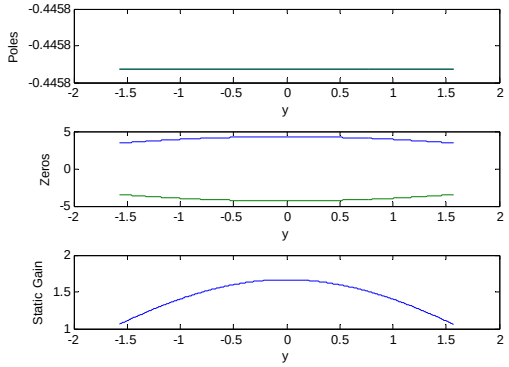

Fig. 3: Properties of the Controlled Plant

The Reference Model was same as in the previous case.

The results of the learning test are shown in fig.4.

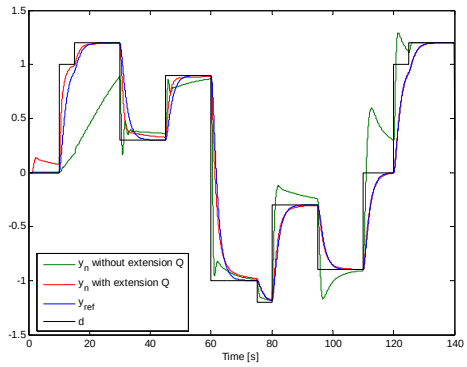

Fig. 4: Results of Learning QNU as a Controller with and without A Priory Information

6. Results

The article showed new options of learning for high neural units as controllers; static characteristic of a single neuron and neurons in closed loops. The information about static characteristic was used for building the optimization function that for learning neurons as controller. Results of simulations controlling linear and nonlinear plants shows increased learning rate.

Symbols

LNU	*Linear Neural Unit* $r = 1$
QNU	*Quadratic Neural Unit* $r = 2$
HONU	*Higher Order Neural Unit*
k	discrete time $k \in \mathbb{Z}$
$\mathbf{X}$	input vector into a neural unit as a model
y	process variable
y_n	output from a neural unit as a model
u	output of a controller
d	desired value
m_y	number of y in vector $\mathbf{X}$ ξ
m_d	output of d in vector $\mathbf{X}$ ξ
n_y	number of y in vector $\mathbf{x}$
n_u	number of u in vector $\mathbf{x}$
$v_{i,j}$	weights of a neuron as a controller
$w_{i,j}$	weights of a neuron as a model
r	order of HONU
K^{CF}	static gain of closed loop
$offset^{CF}$	offset closed loop
$\mathbf{W}$	weights vector or matrix of neural unit as a model
$\mathbf{V}$	weights vector or matrix of neural unit as a controller
μ	learning rate constant
α	proportional learning rate constant

References

[1] P. Benes and I. Bukovsky, "Neural network approach to hoist deceleration control," in *2014 International Joint Conference on Neural Networks (IJCNN)*, 2014, pp. 1864–1869.

[2] P. M. Benes, I. Bukovsky, M. Cejnek, and J. Kalivoda, "Neural Network Approach to Railway Stand Lateral Skew Control," *ArXiv14027136 Cs*, Feb. 2014.

[3] I. Bukovsky, Z.-G. Hou, J. Bila, and M. M. Gupta, "Foundation of Notation and Classification of Nonconventional Static and Dynamic Neural Units," 2007, pp. 401–407.

[4] I. Bukovsky, N. Homma, L. Smetana, R. Rodriguez, M. Mironovova, and S. Vrana, "Quadratic neural unit is a good compromise between linear models and neural networks for industrial applications," 2010, pp. 556–560.

[5] J. van Amerongen, "MRAS: Model Reference Adaptive Systems," 1981. [Online]. Available: https://www.ram.ewi.utwente.nl/aigaion/attachments/single/541. [Accessed: 13-Mar-2016].

[6] Ronald J. Williams and David Zipser, "Gradient-based learning algorithms for recurrent neural networks and their computational complexity," in *Backpropagation: Theory, Architectures, and Applications*, Psychology Press, 2013, pp. 433–386