Available online at www.sciencedirect.com

ScienceDirect

IFAC Papers Online CONFERENCE PAPER ARCHIVE

IFAC PapersOnLine 56-2 (2023) 1057–1062

Echo-state-network-based iterative learning control of distributed systems ⋆

Krzysztof Patan * **Maciej Patan** *

* *Institute of Control and Computation Engineering, University of Zielona Góra ul. Szafrana 2, 65-246 Zielona Góra, Poland (e-mail: {k.patan,m.patan}@issi.uz.zgora.pl).*

Abstract: The paper proposes an effective modeling and control procedure for the distributed-parameter systems using the echo-state network. The main idea is to reconstruct the spatio-temporal dynamics defined in a given multi-dimensional domain. In the investigated problem positions of both sensors and actuators are fixed allowing to delegate the complex system dynamics to echo-state network. Imposing a proper partitioning of the spatial domain, a specific topology of a neural network is used to form a reservoir capable to follow not only temporal but also spatial dynamics of the system. Based on available historical data, neural network model is initially trained and then used to derive the control law in the framework of iterative learning control. The echo-state network can be retrained after a particular control iterate in order to reduce model uncertainty and to fit it to the current operating conditions as much as possible. The performance of the proposed approach is tested and evaluated on the example of the squared clamped plate control.

Keywords: Neural networks, modeling, iterative learning control, control system design, training, distributed-parameter system, adaptive system.

1. INTRODUCTION

Control of distributed-parameter systems (DPS) remains a challenging task, as its dynamics are infinite-dimensional. The state, inputs as well outputs may depend on spatial position. Thus, the natural way of representing DPSs is the partial differential equations (PDEs) which can provide a satisfactory accuracy (Ucinski, 2004; Polyanin, 2002). The control system design is even more complex in case of non-linear DPS. The commonly used approaches principally relies on linearization of DPS around some steady-state or lumping DPS to ordinary differential equation form, and then the application of the known lumped-parameter design methods. Unfortunately, linearization can be troublesome in case of non-homogeneous steady-state as coefficients become spatially distributed. In turn, lumping can lead to significant loss of information in the achieved model (Ray, 1981). To control DPS closed- as well open-loop methods are used (Haber and Bifano, 2021; Schmidt et al., 2020). However, it should be noted that most of the developed solutions uses PDE or ODE models of DPS. In this context the problem of state estimation of DPS from noisy process data arises in many fields, for example smart materials, thermal processes or air pollution (Tricaud and Chen, 2012; Cacuci et al., 2014; Patan et al., 2019).

In this context, classical approaches dedicated for lumped-parameter systems simply cannot provide the required level of performance and are not straightforward to apply. A finite difference methods often used due to their simplicity, cannot fully satisfy the accuracy constraints and guarantee stability and/or convergence. A finite element method (FEM) is an established method used to solve systems represented by PDEs Ames (2014). It provides accurate and stable solutions but usually at the cost of using a dense spatial-grid. As a direct consequence, this group of numerical prescriptions is associated with a relatively large computational burden. Additionally, these methods require the simultaneous analysis of both the system and input dynamics, thus by definition they are off-line procedures what makes their application in real-time not straightforward and troublesome. These deliberations lead to conclusion that there is still a strong necessity for alternative approaches to DPS modeling dedicated to these practical situations where on-line type estimators are preferred with a lower numerical cost.

Alternative solutions can be developed by means of neural network models due to its powerfull modeling capabilities. However, existing contributions on this topic are rather scarce and usually limited to stationary or one-dimensional equations with just a few successful attempts to more general classes of DPSs. In Aguilar-Leal et al. (2016) the authors imitate the meshing procedure according to idea of finite element method and build the differential neural network on such spatial structure. The resulting model may be very accurate but the network inherits the large scale of the initial spatial mesh. In this work for control purposes we adapt the approach proposed in Patan and Patan (2022b) where the complex system dynamics was represented by neural network reservoir

⋆ This research was funded in part by National Science Centre in Poland, grant No. 2020/39/B/ST7/01487. For the purpose of Open Access, the author has applied a CC-BY public copyright licence to any Author Accepted Manuscript (AAM) version arising from this submission.

10.1016/j.ifacol.2023.10.1704

with dedicated architecture consisting of a number of sparsely interconnected local units representing spatial subdomains.

In feedback control it can be observed that at each control iteration the system produce the same tracking error, oscillations and overshot. To overcome that problems a promising approach is to use Iterative Learning Control (ILC) invented in the late 70s (Bristow et al., 2006). Moreover, simple operating principles as well as possibility to combine ILC with feedback control constitute that ILC is very attractive control scheme for practitioners. In the context of distributed systems control, there is a number of interesting applications developed to date, e.g. control of clamped membrane (Cichy et al., 2021) control of cantilever beam (Patan et al., 2019), control of friction stir welding (Patan et al., 2022), control of deformable mirrors (Haber et al., 2013). However, all these approaches uses either lumped linearized representation of a system or FEM-based procedures for distributed system modeling. The contribution of this paper is to apply a echo-state neural network to state estimation of DPS and then to apply derived model to design ILC for DPSs. The accuracy of control is illustrated in the numerical example on the process of temporal vibrations of the two-dimensional clamped elastic plate.

2. DISTRIBUTED-PARAMETER SYSTEM REPRESENTATION

Let $y = y(x,t)$ denote the scalar system state at some point x of the spatial domain $\Omega \subset \mathbb{R}^2$ and let time t belong to the bounded observational interval $T = [0, t_f]$. Mathematically, we handle a dynamic system described by the following PDE:

$$\frac{\partial y}{\partial t} = \mathcal{F}\big(x, t, y, \nabla y, \nabla^2 y; u\big), \quad (x,t) \in \Omega \times T, \tag{1}$$

subject to the boundary and initial conditions

$$\mathcal{B}\big(x, t, y, \nabla y; u\big) = 0, \quad (x,t) \in \partial\Omega \times T, \tag{2}$$

$$y\big(x, 0\big) = y_0(x), \qquad x \in \Omega, \tag{3}$$

where $\mathcal{B}, \mathcal{F}$ and y_0 are some known (possibly non-linear) functions, ∇ and ∇^2 denote the gradient and Hessian, respectively, and u stands for the vector of system actuating inputs.

It is assumed that the state y is observed by N sensors taking the observations of the system continuously in time

$$z^j(t) = \int_\Omega g_j(x) y(x,t;u)\mathrm{d}x + \varepsilon^j(t),\ t \in T,\ j = 1, \ldots, N \tag{4}$$

where $z^j(t)$ is the measurement output, u is the control input vector and $\varepsilon^j(\,\cdot\,)$ denotes the zero-mean Gaussian and uncorrelated measurement noise. A non-negative measurement density function $g_j(x)$ is representing the spatial dynamics of j-th sensor. In practical situations a different choices for such function exist, but most popular approximations are: the uniform distribution (fully distributed measurement), Gaussian distribution and Dirac delta distribution (point-wise measurement). In the following, in order to keep things simple we focus on the last choice. Then, the state of the system $y(\,\cdot\,)$ has to be reconstructed using the observations (4).

Since the inherent feature of such system representation is its infinite dimensionality, therefore in practical engineering setting related to DPSs, most often we are forced to provide the accurate finite dimensional approximation of the system (1)-(3) to make it useful in the control design. There exist a number of dedicated numerical approaches to address this issue such as the method of finite differences, or modern techniques of boundary or finite element. Especially, the FEM become very popular with great number of numerical libraries and solvers due its flexibility, accuracy and stability, establishing a standard computational treatment of DPSs. However, to guarantee the proper accuracy we have to cope with large scale system. This constitutes the great disadvantage in the context of control systems as it is very hard to provide the online schemes due to heavy computational complexity. Therefore the alternative method is proposed based on the reservoir computing technique build on the echo-state network which imitates, to some extent, the concept of FEM with the potential of fast on-line reconstruction of the DPS state with a comparable accuracy.

3. ECHO-STATE NEURAL NETWORK

Reservoir computing constitutes a machine learning concept transforming inputs into a high dimensional space by means of the reservoir. The reservoir is formed using non-linear neurons connected recurrently. The Echo-State Network (ESN) proposed by Jaeger (2001) is well recognized representative of reservoir computing. The desirable advantage here is to provide dynamic characteristic of the model and at the same time avoiding the vanishing gradient problem during model training. In this work we consider the ESN model of the form:

$$\boldsymbol{x}(k+1) = \boldsymbol{f}_h\left(\boldsymbol{W}^x \boldsymbol{x}(k) + \boldsymbol{W}^u \boldsymbol{u}(k+1)\right), \tag{5}$$

$$\hat{\boldsymbol{y}}(k) = \boldsymbol{f}_o\left(\boldsymbol{W}^{out} \boldsymbol{x}(k)\right) \tag{6}$$

where $\boldsymbol{x}(k) \in \mathbb{R}^n$, $\boldsymbol{u}(k) \in \mathbb{R}^q$ and $\hat{\boldsymbol{y}}(k) \in \mathbb{R}^m$ represent the state, input and network output, respectively, $\boldsymbol{W}^x \in \mathbb{R}^{n\times n}$, $\boldsymbol{W}^u \in \mathbb{R}^{n\times q}$ are the reservoir and input weight matrices selected randomly, $\boldsymbol{W}^{out} \in \mathbb{R}^{m\times n}$ is the output weight matrix, $\boldsymbol{f}_h : \mathbb{R}^n \to \mathbb{R}^n$ and $\boldsymbol{f}_o : \mathbb{R}^m \to \mathbb{R}^m$ stand for the vector-valued activation functions of hidden and output neurons, respectively.

To achieve a sufficient level of approximation, the reservoir should ensure a diverse set of dynamic relations. Therefore, the reservoir is selected to include hundreds of processing units but connected sparsely and randomly. In a typical application, only a few percent of possible connections between neurons is implemented. The key concept of ESN is the echo-state property (ESP) relying on the idea that the effect of initial conditions should vanish as time passes. To guarantee ESP the state matrix should be selected in such a way as to satisfy the following condition (Jaeger, 2001; Yildiz et al., 2012):

$$\sigma_{\max}(\boldsymbol{W}^x) < 1, \tag{7}$$

where $\sigma_{\max}$ is the largest singular value of the state matrix $\boldsymbol{W}^x$.

The training procedure of ESN is very simple as only the output weights are subject of training. The only thing we

need to do is to collect input-output pairs and to generate the response of the reservoir on the input data. Then, the output weight matrix can be derived off-line using the psuedoinverse operation as follows:

$$\boldsymbol{W}^{out} = ((\boldsymbol{X} + \mu\mathbb{I})^{-}\boldsymbol{Y})^{\mathsf{T}}, \tag{8}$$

where $\boldsymbol{X} = [\boldsymbol{x}(0), \boldsymbol{x}(1), \ldots, \boldsymbol{x}(N)]$ is the state collection matrix , $\boldsymbol{Y} = \boldsymbol{f}_o^{-1}([\boldsymbol{y}(0), \boldsymbol{y}(1), \ldots, \boldsymbol{y}(N)])$, is the teacher output collection matrix, $\boldsymbol{f}_o^{-1}$ represents the inverse of the function $\boldsymbol{f}_o$, the operator $^{-}$ represents the matrix pseudoinverse and finally μ is a regularization parameter ($\mathbb{I}$ stands for the identity matrix). The regularization is introduces in order to improve the generalization abilities of the model (Patan and Patan, 2022b).

The considered kind of neural networks, due to a large number of processing units connected using recurrent links, is represented by the state vector of a large size. That is the reason, that ESN is suitable to represent DPS for which state-space is infinite dimensional. In our previous works we introduced the idea of spatial domain partitioning (Patan and Patan, 2022a,b). The main idea is to split the entire spatial domain into smaller regular areas and to arrange to each area a sensor measuring the system output (Fig. 1a). In the presented example the spatial domain was divided into four areas. One sensor is assigned to each area, where $S_i, \quad i = 1, \ldots, 4$ represents the sensor notation. To facilitate model design process, we assumed that each sensor measures the output of the system in the center of the sector assigned to it (x_{c_i}, $i = 1, \ldots, 4$) as portrayed in Fig. 1b. Moreover we get a point-wise measurement at the spatial point x_{c_i} and the output of the system observed by the i-th sensor can be represented in the following way:

$$z^i(k) = y(x_{c_i}, k). \tag{9}$$

The data set $\{z^i(k), i = 1, \ldots, 4, k = 0, \ldots, N\}$ constitutes the target training patterns. The excitation of the system has the form of a spatially distributed input. Taking into account an assumption that the distribution of actuation located at the spatial point x as well as its evolution in time are known, we are able to represent the spatial actuation using a properly selected number of point-wise actuations. The input spatial space is divided into sub-regions. To each of them a point-wise actuation is assigned located exactly in the center. As a result, the chosen number of point-wise actuations constitutes the size of the network input space and dimensionality of data.

However, it should be kept in mind that the neural network model has to properly learn spatial as well as time characteristics of DPS. Here, we applied the idea of reservoir partitioning proposed in the work Patan and Patan (2022b)

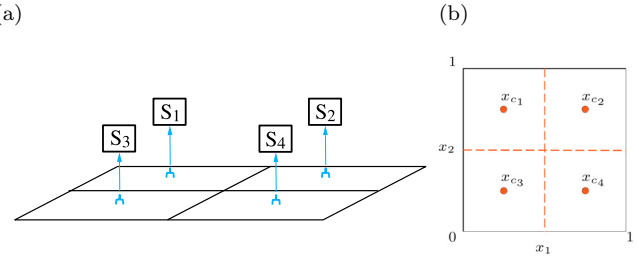

Fig. 1. Example of system spatial partitioning: sensing idea (a), output representation (b).

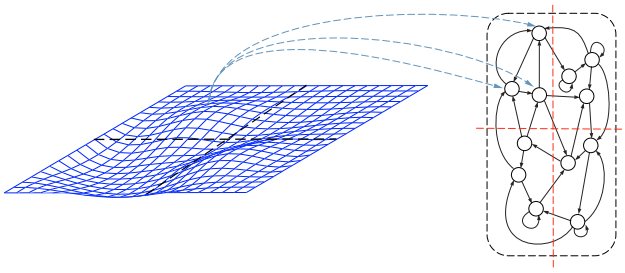

Fig. 2. Partitioning concept of input space (left) and spatial reservoir (right).

and portrayed in Fig. 2. Processing units inside the reservoir are divided into several partitions, which number is equal to the number of point-wise actuations. Each group of units receives excitation from a particular actuator as depicted in Fig. 2 by the blue-dashed lines. For the clarity of presentation, only connections from one actuator are marked there. Which is important, the reservoir is still sparsely connected, and units from different partitions are still interconnected via recurrent links as illustrated at the right hand side of Fig. 2.

The idea of reservoir partitioning can be easily applied to the classical ESN using block-diagonal input matrix $\boldsymbol{W}^u = \text{diag}(\boldsymbol{W}_1, \boldsymbol{W}_2, \ldots, \boldsymbol{W}_{R^2})$, where $\boldsymbol{W}_i, \quad i = 1, \ldots, R^2$ is the input matrix connecting the i-th actuation with the i-th reservoir partition, R^2 denotes the number of partitions. We need to guarantee that each partition will have the similar number of processing units. Assuming, that n_p is the number of neurons in each partition, the total number of neurons in the reservoir will be $n_p \cdot R^2$.

When solving DPS, e.g. using FEM, the derived outputs of the system are spatially interpolated in order to achieve smooth output space and to be able to determine the output of the system in any spatial point of the domain Ω. In the proposed approach we applied the similar procedure. After ESN training the estimated system outputs $\hat{y}_i$ provided by the neural network are properly arranged according to scheme presented in Fig. 1b and are subject of interpolation. Additionally, we need to take into account boundary conditions which are assumed to be zero. The presented procedure matches the level of system partitioning illustrated in Fig. 1b. The input grid used during interpolation is dependent on the number and locations of point-wise actuations. However, it should be stressed that nodes placed on the border of the system as well as on the corners are also used. For more details the interested reader is referred to Patan and Patan (2022a).

4. PROPOSED CONTROL SCHEME

In this study we applied a very simple iterative learning control scheme given as follows:

$$\boldsymbol{u}_{p+1}(k) = \boldsymbol{u}_p(k) + L\boldsymbol{e}_p(k), \tag{10}$$

where the index p stands for the trial (iteration), $\boldsymbol{e}_p(k)$ is the tracking error defined as $\boldsymbol{e}_p(k) = \boldsymbol{y}_{ref}(k) - \hat{\boldsymbol{y}}_p(k)$, and L is the learning gain. In spite of the fact that ILC (10) has standard linear P-type form it should be noted that in this paper we deal with the control of spatio-temporal systems where the tracking error $\boldsymbol{e}_p(k)$ has the multidimensional form. Moreover, contrary to existing approaches proposed

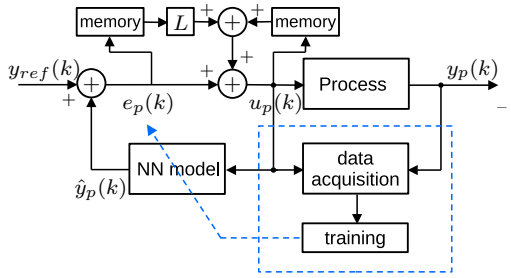

Fig. 3. General structure of iterative learning control based on neural networks.

to date to control such kind of systems, e.g. (Patan et al., 2018), here the excitation signal is developed by means of a novel approach based on distributed neural network model. The ultimate idea is shown in Fig. 3. The control signal is developed by means of plate displacement predictions provided by a neural network model. In order to catch spatio-temporal behavior of the plate a reservoir computing model described in Section 3 is used. The model can be preliminarily trained. However, in order to provide a high quality representation of the plate displacement, the proposed approach makes it possible to adapt neural model to actual working conditions of the control system. To realize this goal after each trial available data is recorded and stored in the memory and then used to adapt the neural network parameters. In order to prevent the so-called catastrophic forgetting of the previously remembered data, the neural network parameters are adapted according to exponential moving average as follows:

$$\boldsymbol{W}_{new}^{out} = \boldsymbol{W}_{old}^{out}(1-\lambda) + \boldsymbol{W}^{out}\lambda \tag{11}$$

where λ is a parameter taking a value from the interval $(0,1)$, and the matrix $\boldsymbol{W}^{out}$ is calculated using (8).

The crucial problem is to select the value of the gain L. We applied a very simple solution with the scalar gain L. Our first approach was to select L using a trial-and-error method. We found out that setting L as follows:

$$L(p) = -(17000e^{-0.02p} + 3000), \tag{12}$$

provides the convergence of the tracking error. It is obvious that selecting L in this way is troublesome and time-consuming. Moreover, there is no proof that using the learning gain (12), the monotonic convergence will be ensured. Therefore, we have tested also a second approach proposed in the book of Xu and Tan (2003):

$$L(p) = \frac{2}{\alpha_1 + \alpha_2}, \quad 0 < \alpha_1 < \frac{\partial \boldsymbol{f}_o}{\partial \boldsymbol{u}_p} \leq \alpha_1. \tag{13}$$

However, it should be noted that the learning gain L selected according to (13) guarantees the monotonic convergence of the tracking error for lumped-parameter systems with one output, under pretty common assumptions such as Lipschitz continuity, identical initialization conditions for each trial and existence of the unique control.

5. ILLUSTRATIVE EXAMPLE

As for illustrative example we consider the vibrations of the thin clamped elastic membrane made from aluminum. The considered distributed-parameter system is represented by the following hyperbolic partial differential equation with the biharmonic operator (Polyanin, 2002):

$$\rho\frac{\partial^2 y(x,t)}{\partial t^2} + \kappa\nabla^4 y(x,t) = u(x,t), \tag{14}$$

where $y(x,t)$ is the transverse displacement at a spatial point x and a time instant t, ρ stands for the mass density per unit area, κ represents the elasticity coefficient, $u(x,t)$ is a pressure field distributed over the domain Ω. The investigated system has the shape of square with the side of the length equal to 1m and the thickness of $d = 0.003$m, $\rho = 2700$, κ is represented by the formula:

$$\kappa = \frac{Ed^3}{12(1-\nu^2)}, \tag{15}$$

where $E = 7.11 \cdot 10^{10}$ stands for the elasticity modulus, $\nu = 0.3$ represents the Poisson's ratio. Since the plate is clamped, the following conditions at the boundary $\partial\Omega$ holds:

$$y(x,t) = 0 \quad x \in \partial\Omega. \tag{16}$$

The initial conditions are as follows:

$$y(x,0) = 0, \quad \dot{y}(x,0) = 0, \quad x \in \Omega. \tag{17}$$

In this study, our objective is to determine a temporal pressure field applied to the membrane at the consecutive trials providing the reference displacement given by the spatial profile of the elliptic paraboloid with the magnitude increasing linearly for 5 seconds, and then vanishing within the next 5 seconds given by the formula:

$$y_{ref}(t) = 10^{-3}\left(1 - \frac{|t-100|}{100}\right)e^{-20\left((x_1-0.4)^2+(x_2-0.6)^2\right)}. \tag{18}$$

The length of each trial was equal to 20 seconds. The sampling time was set to 0.1 second. Then, the reference profile consisted of 201 samples. All experiments were carried out using the Matlab 2018a software using a laptop equipped with Intel Core i7 of the 8th generation, 16GB RAM running under Windows 11. In this study we performed a series of experiments including control of the plate by means of both ESN and FEM as well as investigations regarding the selection of the learning gain. Each experiment was conducted for 300 trials and the control quality was expressed in the form of the norm of the tracking error.

5.1 Model design

The first step of the procedure is to configure both sensing and reservoir partitioning. Based on our knowledge acquired during conducting experiments on the similar topic (Patan and Patan, 2022b), we decided to divide each spatial variable into $R = 5$ subsections. In consequence, the entire spatial domain was partitioned into $R^2 = 25$ sectors. To each sector a point-wise sensor was assigned located at the sector center. Thus, the number of outputs was equal to 25. In turn, to each partition a point-wise actuation was allocated giving 25 input signals.

The ESN parameters were selected carrying out a series of experiments. The best modeling results were observed setting the number of neurons in the reservoir partition $n_p = 10$ giving 250 units in the entire reservoir; sparsity index of neurons connection equal to 20%; and the largest singular value of the state matrix $\sigma_{max} = 0.95$. Moreover, the function $\boldsymbol{f}_o$ was set as a linear one.

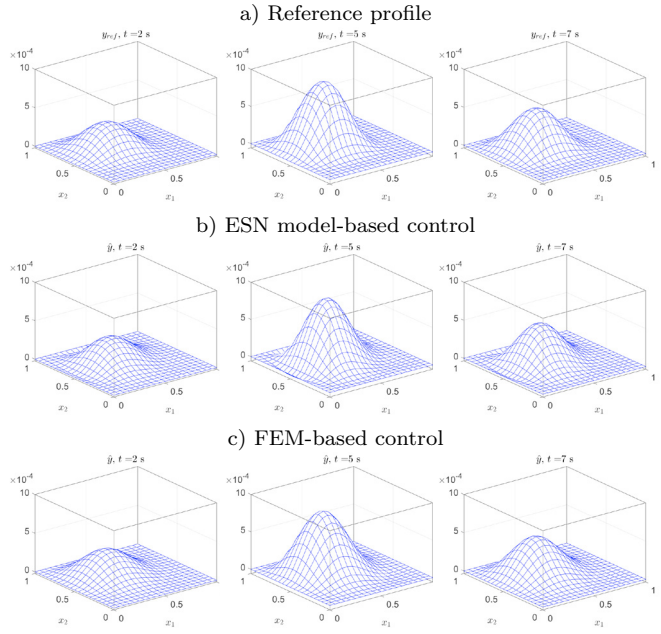

Fig. 4. Control for selected time instances: reference profile (a); ESN-based control (b); FEM-based control (c).

To properly identify the spatio-temporal characteristics of the clamped plate training set including time-varying excitations located at different locations was formed. Each excitation was applied to the plate and the plate displacement was recorded by sensors. Thus input-output data set was prepared for training purposes. The inputs and outputs were linearly scaled to fall into the interval $[-10, 10]$. Details concerning data generation can be found in Patan and Patan (2022b). The training was carried out using the rule (8) with the regularization parameter $\mu = 0.1$ which assured good generalization results of ESN.

5.2 Iterative learning control

The first experiment was carried out using ESN and the learning gain of the form (12). Designed ESN was used to realize ILC according to the scheme presented in Fig. 3. After each trial the output weights of the model were updated by the rule (11). The forgetting factor λ set to 0.05 provided an acceptable balance between already remembered data and new ones. The control results are presented in Fig. 4. The first row includes the reference profile in the selected time instances. The second row shows the outputs of ESN for the same time instances. The control system works well because we can observe that the predicted plate displacement follows the desired profile closely. The convergence of the tracking error norm for this case is shown in Fig. 5 using the blue–solid line. Clearly, we can observe some slight fluctuation in the error curve.

In the second experiment we used FEM to solve the system represented by PDE of the form (14) and on this basis to design the control system. As was mentioned in Introduction FEM is widely used to provide accurate approximation of the continuous distributed systems (Patan et al., 2018). To achieve accurate results, the system (14) was spatially discretized on the spatial grid of the size 21×21. As a result, a spatial mesh consisting of 441 nodes and 800 triangles was obtained. In this case it is assumed that the measurements can be taken on the nodes of spatial mesh. Thus, in this scenario the number of sensors is significantly greater (800) than in the previously investigated case (25). The system was controlled using ILC (10) but the learning gain was selected in a different way. Simply, using the formula (12) leaded to unstable work of FEM and numerical problems when solving the system (14). Then, we proposed a less varying gain as follows:

$$L(p) = -(5000e^{-0.012p} + 2500). \tag{19}$$

The achieved results are shown in the third row of Fig. 4. Clearly, results are comparable to achievements of ESN-based ILC. The convergence of the tracking error is portrayed in Fig. 5 using red–dashed line. The control performance is a little bit more accurate than in the case of ESN-based control. Moreover, one can see that it is possible to obtain smaller value of the tracking error norm carrying out more trials. However, it should be kept in mind that in order to avoid numerical problems when using FEM we need to develop more sophisticated rule for selecting the learning gain than the formula (19).

The objective of the third experiment was to check the control performance of ESN-based control when the learning gain (13) was used. The partial derivative of $\boldsymbol{f}_0$ with respect to the control signal can be easily derived using neural network model:

$$\frac{\partial \boldsymbol{f}_o}{\partial \boldsymbol{u}_p(k)} = \boldsymbol{W}^{out} \frac{\partial \boldsymbol{f}_h}{\partial \boldsymbol{u}_p(k)} = \boldsymbol{W}^{out} \boldsymbol{f}_h^{'} \boldsymbol{W}^{u}. \tag{20}$$

A derivative of hyperbolic tangent takes values from 0 to 1 ($0 < \boldsymbol{f}_h^{'} \leq 1$). Now, substituting $\boldsymbol{f}_h^{'} = 0$ into (20) it yields $\alpha_1 = 0$. In turn, putting $\boldsymbol{f}_h^{'} = 1$ in (20) one can derive α_2 as

$$\alpha_2 = \boldsymbol{W}^{out} \boldsymbol{W}^{u}. \tag{21}$$

In fact, achieved α_2 is a matrix of the size $R^2 \times R^2$. In this paper we simplified the procedure and determined the common learning gain for all actuations as follows:

$$L(p) = 0.1 \frac{2}{\alpha_2}, \tag{22}$$

where $\alpha_2 = \max\{\boldsymbol{W}^{out} \boldsymbol{W}^{u}\}$. In (22) we applied the scaling factor 0.1 in order to achieve the convergence of the tracking error at the very beginning. For this scenario, the convergence of the tracking error is marked in Fig. 5 by the black–dash-dot line. In this case we observe a superior convergence rate contrary to other scenarios. However, after approximately 150 trials the convergence rate begin to slow down, finally reached the tracking error norm similar like in the case of the first scenario investigated. Nevertheless, the advantage of this approach is that the learning gain is selected automatically at every trial analyzing influence of the control on the model output.

The total time of experiment for ESN-based control was approximately 47 seconds. However, if we exclude ESN output weight updating process, the time of simulation decreases to approximately 11 seconds. For the well trained model there is no need to retrain it after each trail but only on demand, let say every 10 trials. This is a serious advantage of the proposed approach contrary to the existing methods. For example, in the case of application of FEM to control system design, working on the grid 21×21, the 300 trials were performed with approximately

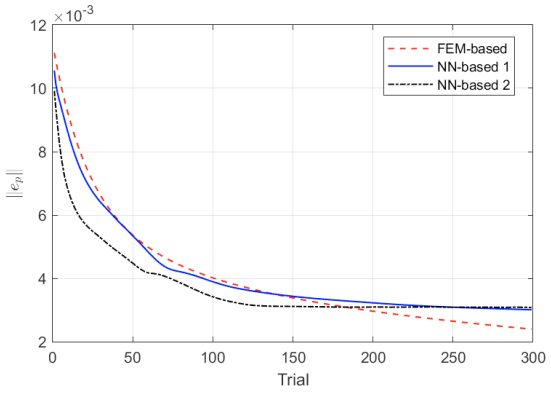

Fig. 5. Convergence of the considered ILC schemes.

132 seconds. Obviously, ESN-based ILC with a regular model parameter updating process is approximately three times faster than FEM-based ILC.

6. CONCLUDING REMARKS

The paper proposes the approach for the modeling and control of distributed-parameter systems using the echo-state network. The main outcome of the developed method is that it is less computationally expensive than frequently used finite element method and at the same time provides the comparable control quality. This is a serious advantage of the proposed method because in real control engineering practice, the distributed-parameter systems need to be processed in the real-time. Another interesting property of neural-network-based control is that the model can be retrained after some number of operation cycles keeping the high level of approximation of the distributed-parameter system all the time.

However, there is a room for improvements. As it is our first work about ILC of distributed systems represented via echo-state network, a very simple form of learning controller was used. Especially, more effort is needed in terms of the learning gain determining as well as providing convergence conditions of proposed control strategy.

REFERENCES

Aguilar-Leal, O., Fuentes-Aguilar, R., Chairez, I., García-González, A., and Huegel, J. (2016). Distributed parameter system identification using finite element differential neural networks. *Applied Soft Computing*, 43, 633–642. doi:https://doi.org/10.1016/j.asoc.2016.01.004.

Ames, W.F. (2014). *Numerical methods for partial differential equations*. Academic press.

Bristow, D.A., Tharayil, M., and Alleyne, A.G. (2006). A survey of iterative learning control: a learning-based method for high-performance tracking control. *IEEE Control Systems Magazine*, 26(3), 96–114.

Cacuci, D.G., Navon, I.M., and Ionescu-Bujor, M. (2014). *Computational Methods for Data Evaluation and Assimilation*. CRC, Boca Raton, FL.

Cichy, B., Augusta, P., Galkowski, K., and Rogers, E. (2021). Modeling and iterative learning control of spatially distributed parameter systems with sensing and actuation over a selected area of the domain. *Multidimensional Systems and Signal Processing*, 32, 1237–1258.

Haber, A. and Bifano, T. (2021). General approach to precise deformable mirrir control. *Optics express*, 29, 33741–33759.

Haber, A., Polo, A., Smith, C.S., Pereira, S.F., Urbach, P., and Verhaegen, M. (2013). Iterative learning control of a membrane deformable mirror for optimal wavefront correction. *Applied Optics*, 52(11), 2363–2373.

Jaeger, H. (2001). The echo state approach to analysing and training recurrent neural networks. Technical Report GMD Report 148, German National Research Center for Information Technology, Germany.

Patan, K. and Patan, M. (2022a). Neural-network-based models ensemble for identification in distributed-parameter systems with application to elastic materials modeling. In *2022 International Joint Conference on Neural Networks (IJCNN)*, 01–08. doi: 10.1109/IJCNN55064.2022.9892489.

Patan, K. and Patan, M. (2022b). Reservoir modeling of distributed-parameter systems. In *The 17th International Conference on Control, Automation, Robotics and Vision (ICARCV 2022)*, 198–203. doi: 10.1109/ICARCV57592.2022.10004225.

Patan, M., Klimkowicz, K., Maniarski, R., Patan, K., and Rogers, E. (2019). Iterative learning control of the displacements of a cantilever beam. In *58th IEEE Conference on Decision and Control, CDC 2019, Nice, France*, 5593–5598. doi:10.1109/CDC40024.2019.9030205.

Patan, M., Klimkowicz, K., and Patan, K. (2022). Optimal sensor selection for prediction-based iterative learning control of distributed parameter systems. In *The 17th International Conference on Control, Automation, Robotics and Vision (ICARCV 2022)*, 449–454. doi: 10.1109/ICARCV57592.2022.10004370.

Patan, M., Patan, K., Gałkowski, K., and Rogers, E. (2018). Iterative learning control of repetitive transverse loads in elastic materials. In *57th IEEE Conference on Decision and Control, CDC 2018, Miami, FL, USA, December 17-19, 2018*, 5270–5275. doi: 10.1109/CDC.2018.8619699.

Polyanin, A.D. (2002). *Linear Partial Differential Equations for Engineers and Scientists*. Chapman and Hall/CRC, New York.

Ray, W.H. (1981). *Advanced Process Control*. McGraw-Hill.

Schmidt, K., Beirow, F., Böhm, M., Graf, T., Abdou Ahmed, M., and Sawodny, O. (2020). Towards adaptive high-power lasers: Model-based control and disturbance compensation using moving horizon estimators. *Mechatronics*, 71, 102441.

Tricaud, C. and Chen, Y. (2012). *Optimal Mobile Sensing and Actuation Policies in Cyber-physical Systems*. Springer, London.

Ucinski, D. (2004). *Optimal measurement methods for distributed parameter system identification*. CRC Press.

Xu, J.X. and Tan, Y. (2003). *Linear and Nonlinear Iterative Learning Control*, volume 291 of *Lecture Notes in Control and Information Sciences*. Springer-Verlag, Berlin.

Yildiz, I.B., Jaeger, H., and Kiebel, S.J. (2012). Re-visiting the echo state property. *Neural Networks*, 35, 1–9. doi: https://doi.org/10.1016/j.neunet.2012.07.005.