MODELLING THE DYNAMICS OF THE LEAD BISMUTH EUTECTIC EXPERIMENTAL ACCELERATOR DRIVEN SYSTEM BY AN INFINITE IMPULSE RESPONSE LOCALLY RECURRENT NEURAL NETWORK

ENRICO ZIO*, NICOLA PEDRONI, MATTEO BROGGI and LUCIA ROXANA GOLEA
Energy Department, Polytechnic of Milan
Via Ponzio 34/3, 20133 Milan, Italy
*Corresponding author. E-mail : enrico.zio@polimi.it

Received February 11, 2009
Accepted for Publication August 2, 2009

In this paper, an infinite impulse response locally recurrent neural network (IIR-LRNN) is employed for modelling the dynamics of the Lead Bismuth Eutectic eXperimental Accelerator Driven System (LBE-XADS). The network is trained by recursive back-propagation (RBP) and its ability in estimating transients is tested under various conditions. The results demonstrate the robustness of the locally recurrent scheme in the reconstruction of complex nonlinear dynamic relationships.

KEYWORDS : Locally Recurrent Neural Network, Nuclear Reactor, Nonlinear Dynamics, Transient Recovery, Transient Interpolation, Transient Extrapolation

1. INTRODUCTION

System and process monitoring by empirical techniques is becoming very popular in various fields of engineering, physics, biology, and medicine, because it does not require a detailed physical understanding of the process nor knowledge of the material properties, geometry and other characteristics of the system and its components, which are often lacking in real, practical cases. The underlying process model is identified by fitting measured process data, a procedure which is often referred to as 'learning' or 'training'.

Artificial Neural Networks (ANNs) are among the most powerful algorithms for empirical modelling. In general terms, they are computing devices inspired by the function of the nerve cells in the brain. They are composed of many parallel, interconnected computing units; each of these performs a few simple operations and communicates the results to its neighbouring units. In contrast to conventional modelling, ANNs can learn the required input/output relationship, possibly nonlinear, by a process of training on many different input/output examples. They are also very effective in learning patterns in data that are noisy, incomplete and which may even contain contradictory examples.

Whereas feedforward neural networks can model static input/output mappings but do not have the capability of reproducing the behaviour of dynamic systems, dynamic Recurrent Neural Networks (RNNs) are recently attracting significant attention, because of their potential in temporal processing.

In particular, Infinite Impulse Response Locally Recurrent Neural Networks (IIR-LRNNs) have proven capable of providing accurate approximations of the dynamic behaviour of nonlinear systems [1-3]. The time dependencies on the previous input values and system states are accounted for by adding to the otherwise classical ANN architecture *tapped-delay-lines* (temporal buffers) to the inputs of each network layer and by inserting *internal recurrence*, i.e., feeding back the outputs of the neurons of a given layer in input to the neurons of the same layer.

In a previous paper [4], an IIR-LRNN application was considered, regarding the modelling of the nonlinear, time-dependent relationships between measurable, physical variables and safety-significant system state variables in a next-generation, sub-critical fast reactor, the Lead Bismuth eXperimental Accelerator Driven System (LBE-XADS) [5,6]. In particular, the focus of the paper was the comparison between batch and on-line recursive algorithms for IIR-LRNN training; the case study proposed in [4] was aimed at testing the capability of the IIR-LRNN to recover the transient behaviour of an evolving system, i.e., a system whose underlying physical parameters change in time, e.g., because of ageing, degradation or unexpected modification of the system. The present paper extends the work in [4] by considering a number of *additional* performance tests

of the IIR-LRNN under various *different* operating conditions: in particular, interpolation and extrapolation tasks, as well as fault tolerance due to incomplete or erroneous data and transient recovery from arbitrary initial conditions, are performed. To keep the presentation simple, only the batch mode of recursive training is considered here.

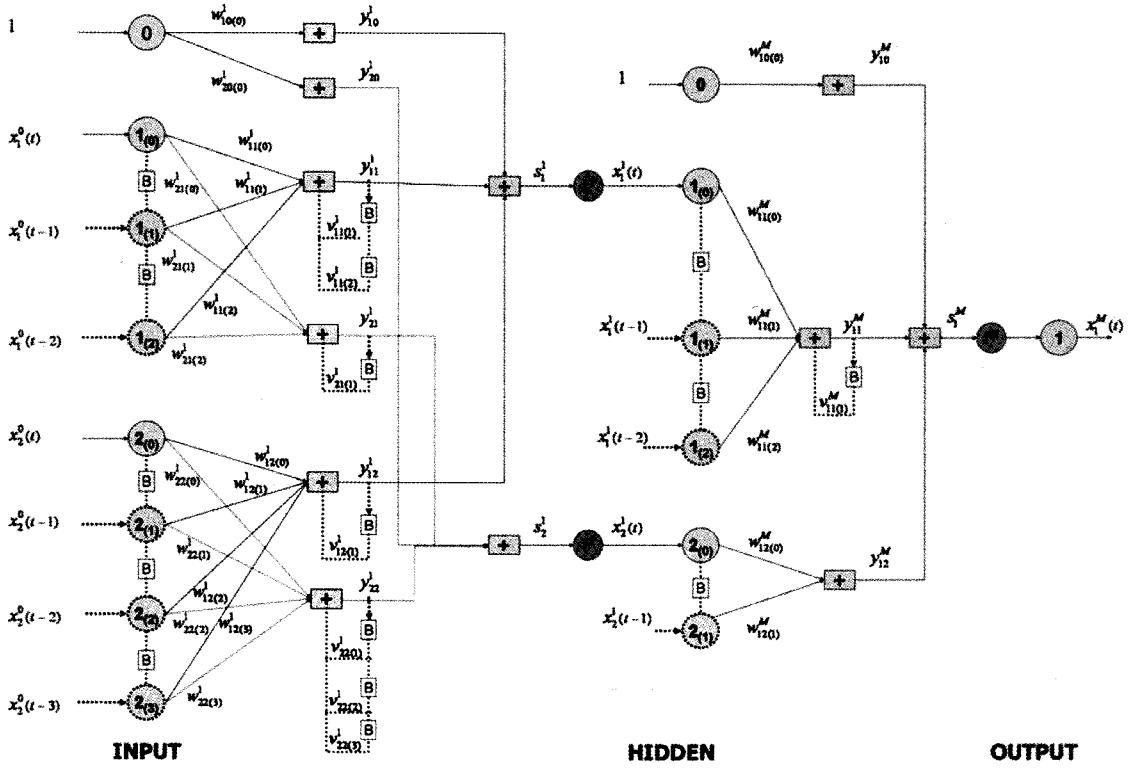
In Section 2, the IIR-LRNN architecture and the Recursive Back-Propagation (RBP) algorithm for the network training are presented. In Section 3, a mechanistic model of the LBE-XADS is introduced. This model is used to generate the transients for training and testing an IIR-LRNN under various challenging conditions of realistic operation. The results obtained by the trained network are analyzed in detail. Finally, some thoughts are expressed in Section 4 with regards to the future of these modelling techniques in safety-related applications.

2. LOCALLY RECURRENT NEURAL NETWORKS

2.1 IIR-LRNN Architecture and Forward Calculation

An IIR-LRNN is a time-discrete network consisting of a global feed-forward structure of nodes interconnected by synapses which link the nodes of the k-th layer to those of the successive $(k+1)$-th layers, $k=0, 1, \ldots, M$, layer 0 being the input and M the output. Different from the classical static feedforward neural networks, in an IIR-LRNN each synapse carries taps and feedback connections. Analogous to the classical static feedforward neural network, all data are normalized in a properly chosen range before being processed by the network.

For simplicity of illustration, and with no loss of generality, we start by considering a network constituted by only one hidden layer, i.e., $M=2$, like the one in Figure



INPUT ($k=0$)	HIDDEN ($k=1$)	OUTPUT ($k=2=M$)
$N^0=2$ (nodes)	$N^1=2$ (nodes)	$N^M=1$ (nodes)
	$L^1_{11}-1=2$, synapses $w^1_{11(p)}, w^1_{21(p)}$, p=1,2	$L^M_{11}-1=2$, synapses $w^M_{11(1)}$
	$L^1_{12}-1=3$, synapses $w^1_{12(p)}, w^1_{22(p)}$, p=1,3	$L^M_{12}-1=1$, synapses $w^M_{12(1)}$
	$I^1_{11}=2$, $I^1_{21}=1$ synapses $v^1_{11(1)}, v^1_{11(2)}, v^1_{21(1)}$	$I^M_{11}=1$, synapses $v^M_{11(1)}$
	$I^1_{12}=1$, $I^1_{22}=3$ synapses $v^1_{12(1)}, v^1_{22(1)}, v^1_{22(2)}, v^1_{22(3)}$	$I^M_{12}=0$, no recurrent synapses

Fig. 1. Scheme of an IIR-LRNN with One Hidden Layer

1. At the generic time t, the input to the IIR-LRNN consists of the pattern $x(t) \in \Re^{N^0}$, whose components feed the nodes of the input layer 0 which simply transmit in output the input received, i.e., $x_m^0(t) = x_m(t)$, $m = 1,2,\ldots,N^0$. A bias node is also typically inserted, with the index $m = 0$, such that $x_0^0(t)=1$ for all values of t. The output variable of the m-th input node at time t is tapped a number of delays $L_{nm}^1 - 1$ (except for the bias node output which is not tapped, i.e., $L_{n0}^1 - 1 = 0$) so that from each input node $m \neq 0$ actually L_{nm}^1 values, $x_m^0(t)$, $x_m^0(t-1)$, $x_m^0(t-2),\ldots$, $x_m^0(t-L_{nm}^1+1)$, are processed forward through the synapses connecting the input node m to the generic hidden node $n = 1, 2, \ldots N^1$. The L_{nm}^1 values sent from the input node m to the hidden node n are first multiplied by the respective synaptic weights $w_{nm(p)}^1$, $p = 0, 1, \ldots, L_{nm}^1 - 1$ being the index of the tap delay (the synaptic weight $w_{n0(0)}^1$ connecting the bias input node $m = 0$ is the bias value itself) and then processed by a summation operator. The weighed sum thereby obtained, y_{nm}^1, is fed back, for a given number of delays I_{nm}^1 ($I_{n0}^1 = 0$ for the bias node) and weighed by the coefficient $v_{nm(p)}^1$ to the summation operator itself to give the output quantity. Thus, the value transferred via the connection between the generic nodes m of the input layer and n of the hidden layer is

$$y_{nm}^1(t) = \sum_{p=0}^{L_{nm}^1-1} w_{nm(p)}^1 \cdot x_n^0(t-p) + \sum_{p=1}^{I_{nm}^1} v_{nm(p)}^1 \cdot y_{nm}^1(t-p) \quad (1)$$

Note that in a classical static ANN the connection between the two nodes would simply transfer the quantity $w_{nm}^1 \cdot x_n^0(t)$.

Also, as mentioned above, the index $m = 0$ usually represents the bias input node, such that $x_0^0(t)$ is equal to one for all values of t, $L_{n0}^1 - 1 = I_{n0}^1 = 0$ and thus $y_{n0}^1(t) = w_{n0(0)}^1$. The quantities $y_{nm}^1(t)$, $m = 0, 1, \ldots, N^0$, are summed to obtain the net input $s_n^1(t)$ to the nonlinear activation function $f^1(\cdot)$, typically a sigmoidal Fermi function, of the n-th hidden node, $n = 1, 2, \ldots, N^1$:

$$s_n^1(t) = \sum_{m=0}^{N^0} y_{nm}^1(t) \quad (2)$$

The output of the activation function gives the state of the n-th hidden neuron, $x_n^1(t)$:

$$x_n^1(t) = f^1[s_n^1(t)] \quad (3)$$

The output values of the nodes of the hidden layer 1, $x_n^1(t)$, $n = 1,2,\ldots,N^1$ are then processed forward along the synaptic connections linking the hidden and output nodes in a manner which is analogous to the processing between the input and hidden layers.

A bias node with index $m = 0$ is also typically inserted in the hidden layer 1, such that $x_0^1(t) = 1$ for all values of t, and as mentioned before $y_{r0}^M(t) = w_{r0(0)}^M$.

At the input to the output layer $M = 2$, the quantities $y_{rn}^M(t)$, $n = 0, 1,\ldots, N^1$, are summed to obtain the net input $s_r^M(t)$ to the nonlinear activation function $f^M(\cdot)$ so that the output of the activation function gives the state of the r-th output neuron, $x_r^M(t)$:

$$x_r^M(t) = f^M[s_r^M(t)] \quad (4)$$

The extension of the above calculations to the case of multiple hidden layers ($M > 2$) is straightforward. The time evolution of the generic neuron j belonging to the generic layer $k = 1,2,\ldots,M$ is described by the following equations:

$$x_j^k(t) = \begin{cases} 1 & j = 0 \\ f^k\left[s_j^k(t)\right] & otherwise \end{cases} \quad (5)$$

$$s_j^k(t) = \sum_{l=0}^{N^{k-1}} y_{jl}^k(t) \quad (6)$$

$$y_{jl}^k(t) = \sum_{p=0}^{L_{jl}^k-1} w_{jl(p)}^k \cdot x_l^{k-1}(t-p) + \sum_{p=1}^{I_{jl}^k} v_{jl(p)}^k \cdot y_l^{k-1}(t-p) \quad (7)$$

2.2 Recursive Back-Propagation (RBP) Algorithm for Batch-training an IIR-LRNN

The IIR-LRNN bases its dynamic modelling capabilities on the particular input-output functional form built into its architecture and on the values of its internal parameters, i.e., the network weights $w_{jl(p)}^k$ and $v_{jl(p)}^k$. These latter figures are calibrated through a training procedure aimed at the best-fitting set of exemplary trajectories representative of the dynamic behaviour of the system to be modelled. In particular, the method adopted here is the RBP training algorithm [7], a gradient-based minimization algorithm that makes use of a particular chain rule expansion for the computation of the necessary derivatives. When used in batch mode, it is equivalent to Real-Time Recurrent Learning (RTRL) [8] and Back-Propagation-Through-Time (BPTT) [9].

For each dynamic trajectory of duration T (discretized in time steps $t = 1,2,\ldots,T$) in the training set, the following steps are taken:

1. Assign at random the values of the network weights $w_{jl(p)}^k$ and $v_{jl(p)}^k$, e.g., by uniform sampling in a given range of values (here, [-0.5, 0.5] for $w_{jl(p)}^k$, [-0.1, 0.1] for $v_{jl(p)}^k$).
2. Perform the IIR-LRNN forward calculations (2)-(5)

for the entire sequence of values of the dynamic trajectory and save the states $x_j^k(t)$, $j=1,2,\ldots,N^k$, $k=0,1,\ldots,M$ of the network nodes at all times $[1, T]$.

3. Start the backward propagation by computing the mean squared error E^2 between the network-computed outputs $x_r^M(t)$ and the corresponding real values on the dynamic trajectory $d_r(t)$, $r=1,2,\ldots,N^M$:

$$E^2 = \sum_{t=1}^{T}\sum_{r=1}^{N^M}[e_r(t)]^2 \qquad e_r(t) = d_r(t) - x_r^M(t) \tag{8}$$

4. Starting from null initial conditions, iteratively compute the derivatives

$$\frac{\partial y_{qj}^{k+1}(t+p)}{\partial x_j^k(t)} = \sum_{\tau=1}^{\min(L_{qj}^{k+1},p)} v_{qj(\tau)}^{k+1}\frac{\partial y_{qj}^{k+1}(t+p-\tau)}{\partial x_j^k(t)} + \begin{cases} w_{qj(p)}^{k+1} & \text{if } 0 \le p \le L_q^{k+1}-1 \\ 0 & \text{otherwise} \end{cases} \tag{9}$$

5. By back-propagating the error from the last layer, compute the quantities $\delta_j^{k+1}(t)$ and $e_j^k(t)$ for every instant of time in $[1, T]$

$$\delta_j^{k+1}(t) = -\frac{1}{2}\frac{\partial E^2}{\partial s_j^{k+1}(t)} = e_j^{k+1}(t)f'(t) \tag{10}$$

where, $f'(\cdot)$ is the derivative of the activation function $f(\cdot)$ and

$$e_j^k(t) = \begin{cases} e_j(t) \quad (eq.8) & \text{for } k = M \\ \sum_{p=0}^{T-t}\sum_{q=1}^{N^{k+1}}\delta_q^{k+1}(t+p)\frac{\partial y_{qj}^{k+1}(t+p)}{\partial x_j^k(t)} & \text{for } k = M-1, M-2, \ldots, 1 \end{cases} \tag{11}$$

6. Compute the weight variations $\Delta w_{jl(p)}^k$ and $\Delta v_{jl(p)}^k$

$$\Delta w_{jl(p)}^k = -\frac{\mu}{2}\sum_{t=1}^{T}\frac{\partial E^2}{\partial s_j^k(t)}\frac{\partial s_j^k(t)}{\partial w_{jl(p)}^k} = \sum_{t=1}^{T}\mu\delta_j^k(t)\left[x_l^{k-1}(t-p) + \sum_{\tau=1}^{l_M^k} v_{jl(\tau)}^k\frac{\partial s_j^k(t-\tau)}{\partial w_{jl(p)}^k}\right] \tag{12}$$

$$\Delta v_{jl(p)}^k = -\frac{\mu}{2}\sum_{t=1}^{T}\frac{\partial E^2}{\partial s_j^k(t)}\frac{\partial s_j^k(t)}{\partial v_{jl(p)}^k} = \sum_{t=1}^{T}\mu\delta_j^k(t)\left[y_{jl}^k(t-p) + \sum_{\tau=1}^{l_M^k} v_{jl(\tau)}^k\frac{\partial s_j^k(t-\tau)}{\partial v_{jl(p)}^k}\right] \tag{13}$$

where, μ is the so-called learning coefficient which modulates the step along the gradient.

7. Update the weights

The procedure is terminated when the mean squared error E^2 falls below a pre-assigned acceptable threshold.

3. IR-LRNN MODEL FOR THE SIMULATION OF LBE-XADS REACTOR TRANSIENTS

3.1 Model of the LBE-XADS

The Lead-Bismuth Eutectic eXperimental Accelerator

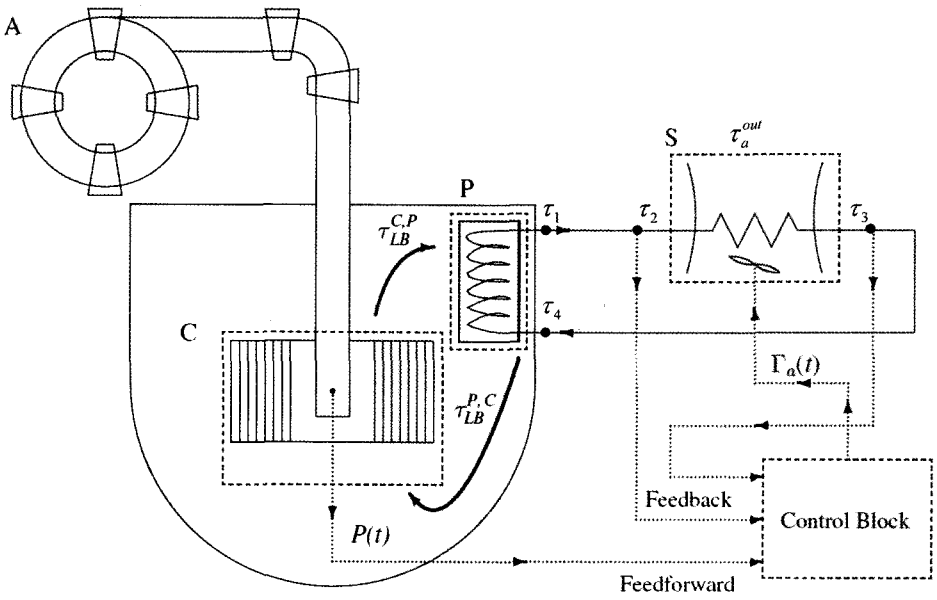


Fig. 2. LBE-XADS Simplified Schematics. A = Accelerator; C = Core; P = Primary Heat Exchanger; S = Secondary Heat Exchanger; $\tau_{LB}^{C,P}$ = Pb-Bi Core Outlet Temperature; $\tau_{LB}^{P,C}$ = Pb-Bi Core Inlet Temperature; $P(t)$ = Reactor Power; $\Gamma_a(t)$ = Cooler Air Flow; τ_a^{out} = Cooler Air Output Temperature [15]

Driven System (LBE-XADS) is a sub-critical, fast reactor in which the fission process is sustained by an external neutron source through the spallation reaction by a proton beam accelerated by a synchrotron on a lead-bismuth eutectic target [5,6,10-12]. In the current design [13], the core contains Uranium and Plutonium dioxide fuel rods; future developments are aimed at also housing long-lived transuranic elements. The reactor can be considered "cold" as the average fuel temperature is about 1073 K [14].

The primary cooling system is the pool-type with Lead-Bismuth Eutectic (LBE) liquid metal coolant. The configuration is similar to that of most sodium-cooled reactors (e.g., the French Phenix and SuperPhenix). A simplified scheme of the plant is sketched in Figure 2.

The primary coolant leaving the top of the core at 673 K is pushed by natural circulation enhanced by argon gas injection into the heat exchangers of the secondary cooling circuit and then re-enters the core from the bottom through the down-comer at 573 K.

The secondary cooling system is a flow of organic diathermic oil at 553-593 K, at full power conditions, through two independent loops, each one consisting of two intermediate heat exchangers. Cooling of the diathermic oil in each loop is provided by three air coolers connected in a series.

A dedicated, dynamic simulation model has been implemented in SIMULINK for providing a simplified, lumped and zero-dimensional description of the coupled neutronic and thermo-hydraulic evolution of the system [15]. The control is realized by a PID controller designed to keep a steady state value of 573 K of the average temperature of the diathermic oil (controlled variable) by manipulating the mass flow rate of air (control variable) on the basis of both a feedforward action (whose intensity is governed by the value of the reactor power) and a feedback action (whose intensity is governed by the difference between the average diathermic oil temperature and its reference value of 573 K) (Fig. 2). In passing, notice that the model allows the simulation of the system controlled dynamics described above as well as of the free dynamics when the control module is deactivated: in this latter case, neither feedforward nor feedback actions are taken to regulate the air cooler flow, which is kept constant regardless of the reactor power level and of the difference between the average diathermic oil temperature and its reference value of 573 K.

The model has been used to simulate transients which in this work are taken as representative of the real behaviour of the system; hence, these transients have been used both as reference trajectories for the IIR-LRNN training and as terms of comparison for the verification and validation of the IIR-LRNN's performance.

3.2 IIR-LRNN Training

In Table 1, the relevant system input variables and the output quantities of interest are reported, together with their corresponding steady state values.

The IIR-LRNN used to model the input-output relationship

Table 2. Structure of the IIR-LRNN for the Simulation of the LBE-XADS

IIR-LRNN structure		
Input nodes (bias included)		7
Hidden nodes (bias included)		6
Output nodes		3
Type of activation functions (hidden/output nodes)		Sigmoidal
L_{jl}^{k}	Hidden ($k=1$, $l=1, 2, \ldots, 6$, $j=1, 2, \ldots, 5$)	3
	Output ($k=M=2$, $l=1, 2, \ldots, 5$, $j=1, 2, 3$)	2
Γ_{jl}^{k}	Hidden ($k=1$, $l=1, 2, \ldots, 6$, $j=1, 2, \ldots, 5$)	2
	Output ($k=M=2$, $l=1, 2, \ldots, 5$, $j=1, 2, 3$)	3

Table 1. IIR-LRNN Inputs and Outputs

	Variable Name	Variable Description	Steady State Value
Input	$Q(t)$	Input 1: accelerator driven neutron source	$7.3638 \cdot 10^{4}$ n / s
	$\tau_1(t)$	Input 2: diathermic oil temperature after the oil heat exchanger	593.3 K
	$\tau_2(t)$	Input 3: diathermic oil temperature before the oil-air heat exchanger	553 K
	$\tau_3(t)$	Input 4: diathermic oil temperature after the oil-air heat exchanger	593.3 K
	$\tau_4(t)$	Input 5: diathermic oil temperature before the oil heat exchanger	553 K
	$\tau_a^{out}(t)$	Input 6: cooler air output temperature	398.15 K
Output	$\tau_{fuel}^{av}(t)$	Output 1: fuel average temperature	1073.2 K
	$\tau_{LB}^{av}(t)$	Output 2: Pb-Bi eutectic average temperature	625.65 K
	$P(t)$	Output 3: reactor power	$8 \cdot 10^{4}$ W

is composed of three layers: the input, with seven nodes (bias included); the hidden, with six nodes (bias included); and the output, with three nodes. A sigmoidal activation function has been adopted for the hidden and output nodes.

The number of taps and delays have been chosen experimentally to obtain a satisfactory modelling performance of the trained IIR-LRNN, measured in terms of a small Root Mean Square Error (RMSE) of the network estimates with respect to the model simulated output values. The IIR-LRNN structure is summarized in Table 2.

Two IIR-LRNNs with this architecture have been trained: one for the free and one for the controlled dynamics. In both cases, the training set is made up of $N_t = 100$ transients, each one lasting $T = 1000$ seconds. The time-discretization is made with steps of $\Delta t = 1\ s$, thus giving rise to patterns made of 1000 values. All transients have been created by varying at a random time T_s the value of the external neutron source from Q_0 to $Q_0+\Delta Q$, the step ΔQ being uniformly sampled in $[-0.1\cdot Q_0, 0.1\cdot Q_0]$:

$$Q(t) = \begin{cases} Q_0 & t \le T_s \\ Q_0 + \Delta Q & T_s < t \le T = 1000\,s \end{cases} \quad (14)$$

Table 3. Training Parameters of the IIR-LRNN for Simulating the LBE-XADS

Principal training parameters	
Number of transients in the training set	90
Number of patterns in each transient	1000
μ (Learning coefficient)	0.001
α (Momentum coefficient)	0
Learning epochs (n_{epoch})	40
Consecutive repetitions of each transient (n_{rep})	10
Data normalization range	[0.3;0.7]

Of the $N_t = 100$ transients generated, 90 have been used to train the IIR-LRNN; the remaining 10 have been kept for validation.

The training procedure has been carried out iteratively for $n_{epoch} = 40$ epochs. During each epoch, every transient is repeatedly presented to the IIR-LRNN for $n_{rep} = 10$ consecutive times. The training process is monitored by computing the RMSE at each learning epoch. The principal training parameters are summarized in Table 3.

Additional transients have been created for testing the trained IIR-LRNN. To generate these transients, the neutron source is varied from its steady state value Q_0, according to a random ramp function starting from a random time T_s and lasting T_r seconds

$$Q(t) = \begin{cases} Q_0 & t \le T_s \\ Q_0 + \Delta Q \dfrac{t - T_s}{T_r} & T_s < t \le T_s + T_r \\ Q_0 + \Delta Q & T_s + T_r < t \le T = 1000\,s \end{cases} \quad (15)$$

in which ΔQ is uniformly sampled in $[-0.1\cdot Q_0, 0.1\cdot Q_0]$.

Training and test transients have been created in such a way for both the free and controlled dynamic simulations. All data have been normalized within the range [0.3, 0.7].

Figure 3 and Figure 4 report the RMSE as a function of the iterative learning epoch, for each of the three outputs of both the free and controlled dynamics networks, respectively.

3.3 IIR-LRNN Results

Figure 5 and Figure 6 show the 'real' (model-simulated) and IIR-LRNN-estimated evolution of the three output quantities for step transients of free and controlled dynamics different from those used in training, respectively. The IIR-LRNN estimation of the outputs is in satisfactory agreement with the real transients, confirming the ability of the IIR-LRNN to deal with the short-term dynamics governed by

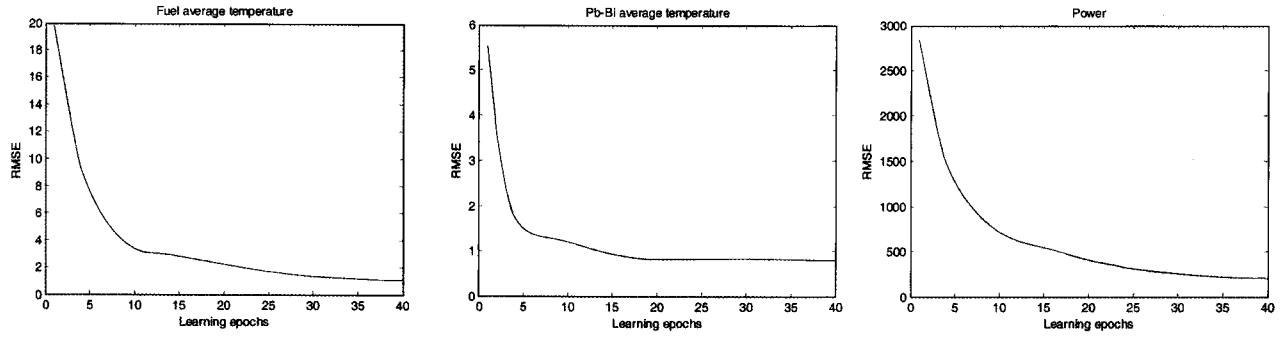


Fig. 3. RMSE Trend During Training of the Free Dynamics IIR-LRNN. The RMSE Final Values are (a) 1.08 K (b) 0.78 K (c) 201 [n/s]

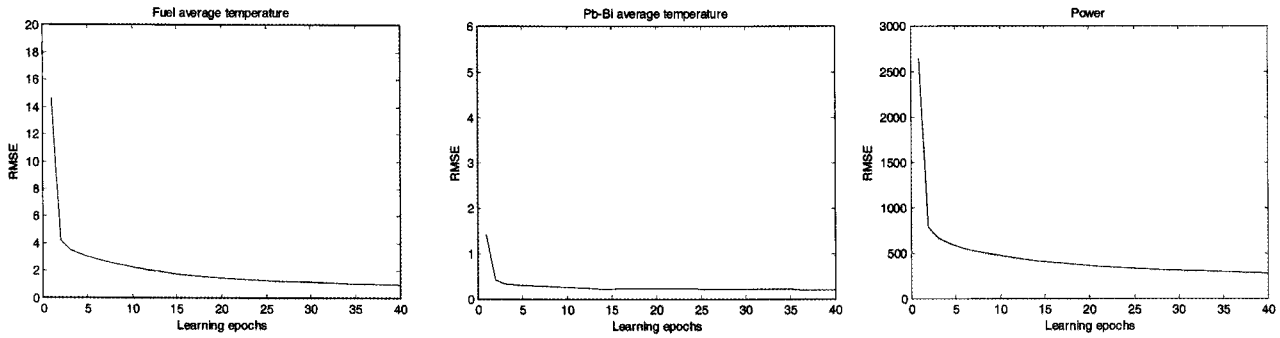


Fig. 4. RMSE Trend During Training of the Controlled Dynamics IIR-LRNN. The RMSE Final Values are (a) 0.98 K (b) 0.19 K (c) 278 [n/s]

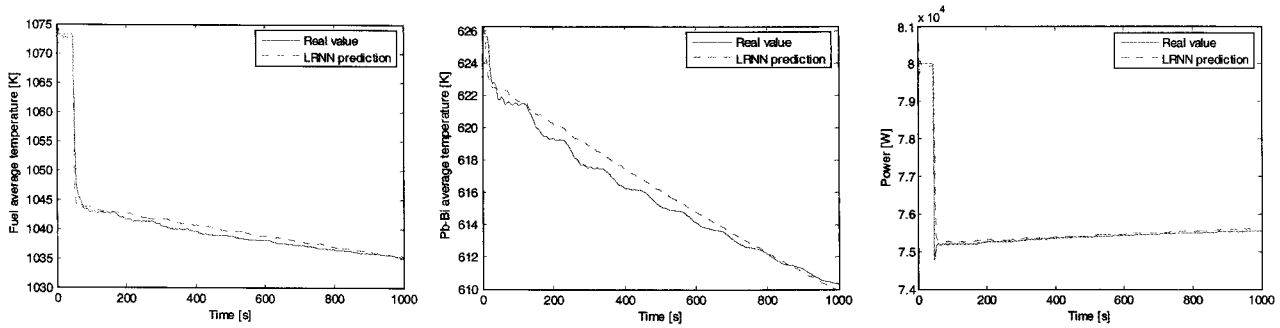


Fig. 5. Free Dynamics. Validation Patterns

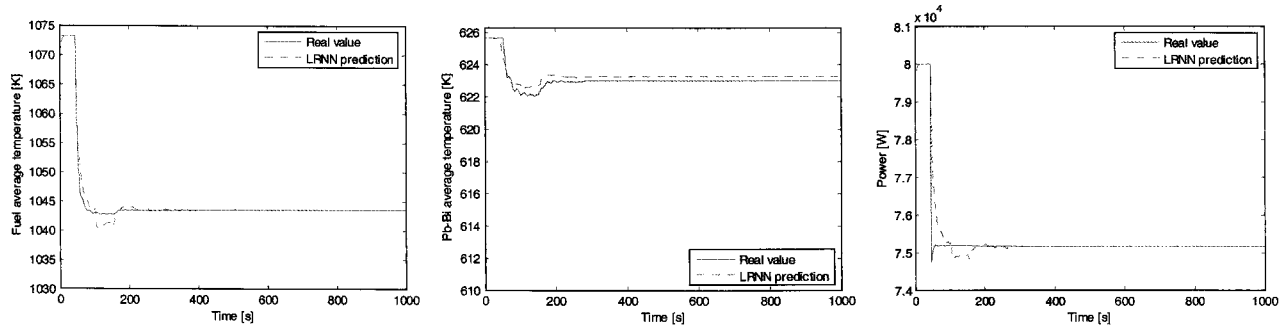


Fig. 6. Controlled Dynamics. Validation Patterns

the instantaneous variations of the forcing function.
The capabilities of the IIR-LRNN have been successfully verified also on test transients generated by forcing function variations functionally different from those used in the training phase, i.e., the ramp variations (15) in the neutron source (Figure 7 and Figure 8).

Looking more closely at the results of the controlled dynamics, one can conclude that the trained networks are capable of reproducing the evolution of the three output quantities, albeit they introduce some small numerical oscillations in the power and fuel average temperature, creating a dependence between the three output variables.

3.4 Tests on the Capabilities of IIR-LRNN

To further analyze the capabilities of the IIR-LRNN and highlight its advantages over the conventional dynamic modelling approaches, the extrapolation and interpolation capabilities of the network, its fault tolerance and its transient recovery capabilities are investigated. This is done through a number of ad hoc tests, as suggested in [16].

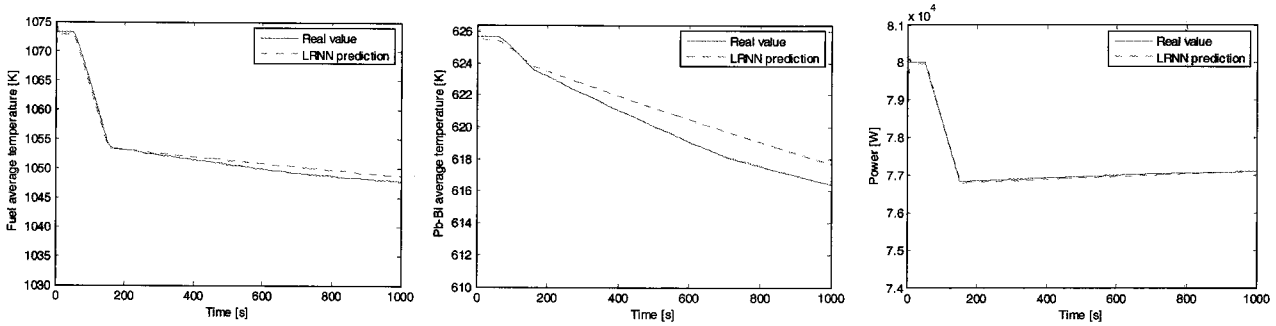


Fig. 7. Free Dynamics. Test Patterns

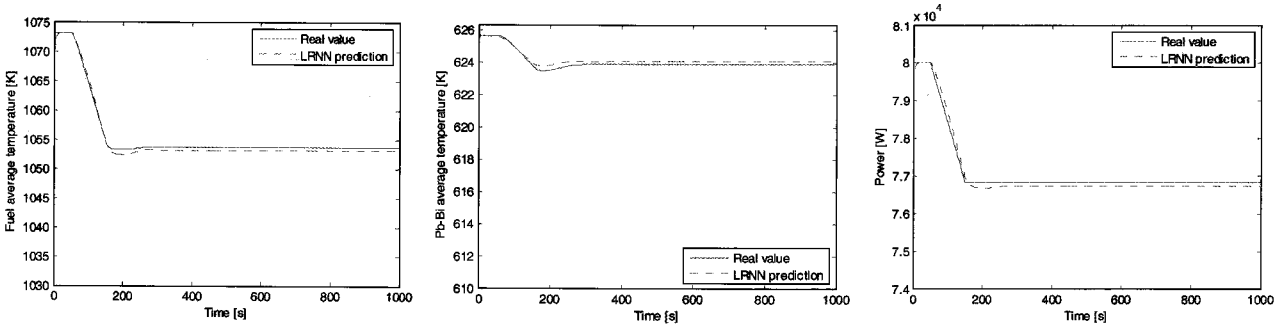


Fig. 8. Controlled Dynamics. Test Patterns

3.4.1 Extrapolation Tests

One important issue in system modelling is long-term prediction. For testing the corresponding capabilities of the trained IIR-LRNN, the following transient lasting $T=$ 2000 s is considered (Figure 9):

$$Q(t)=\begin{cases} Q_0 & t<T_{s1} \\ Q_0+\Delta Q\dfrac{t-T_{s1}}{T_{r1}} & T_{s1}\le t<T_{s1}+T_{r1} \\ Q_0+\Delta Q & T_{s1}+T_{r1}\le t<T_{s2} \\ Q_0-\Delta Q\dfrac{t-T_{s2}}{T_{r2}} & T_{s2}\le t<T_{s2}+T_{r2} \\ Q_0 & T_{s1}+T_{r1}\le t<T_{end} \end{cases} \quad (16)$$

where, $Q_0=7.3638\cdot10^4$ n/s, $\Delta Q=0.05\cdot Q_0$, $T_{s1}=50$ s, $T_{r1}=$ 400 s, $T_{s2}=1000$ s, $T_{r2}=1100$ s, and $T_{end}=2000$ s.

As seen in the Figure, the transient is characterized by a variation of the neutron source $Q(t)$ made up by two ramps: the first one is a slow-increasing ramp lasting 450 s whereas the second one is a fast-decreasing ramp lasting 50 s.

Figure 10 and Figure 11 compare the "real" output values with the IIR-LRNN predictions for both the free and controlled dynamics, respectively. Even in the long term extrapolation part, beyond the usual 1000 s of training transient duration, the IIR-LRNN provides a very good prediction.

3.4.2 Interpolation Tests

The interpolation capabilities are tested by training two new IIR-LRNNs, for the free and controlled dynamics, with a data set down–sampled with a 5:1 ratio; that is, only one datum in every 5 samples is taken for training. This aspect is important in cases where limitations on the size of the training data exist or need to be imposed.

The results in Figure 12 and Figure 13 show the "real" output values and those estimated by the down-sampled IIR-LRNN for the fully-sampled transient (15); in spite of the down-sampled training, the network shows good interpolation capabilities for all output quantities of the controlled dynamics, with maximum relative errors of 0.0019, 0.0037, 0.0105, for the fuel average temperature, the Pb-Bi average temperature and the power in the free dynamics prediction, respectively.

3.4.3 Fault Tolerance Tests

During on-line operation, it is possible that the signal values provided by the sensors might be inaccurate or even wrong, due to faults or miscalibrations of the components on the sensing lines. In this respect, it is important to test

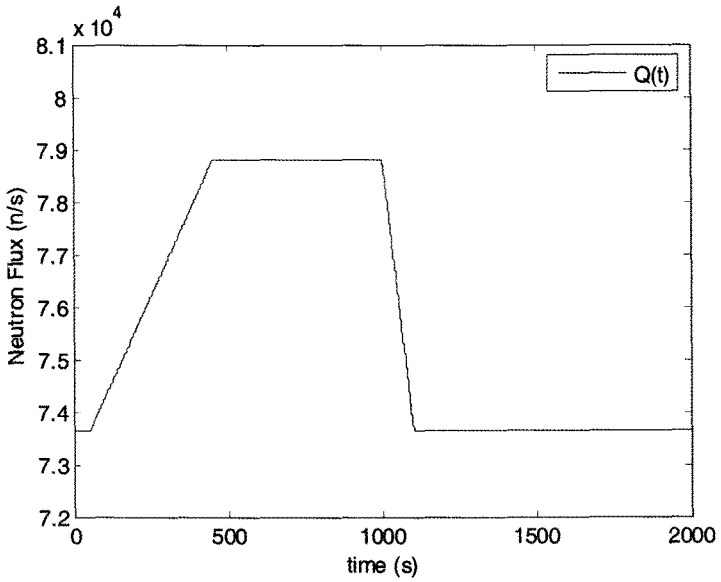


Fig. 9. Neutron Source $Q(t)$ Used in the Long Term Extrapolation Test

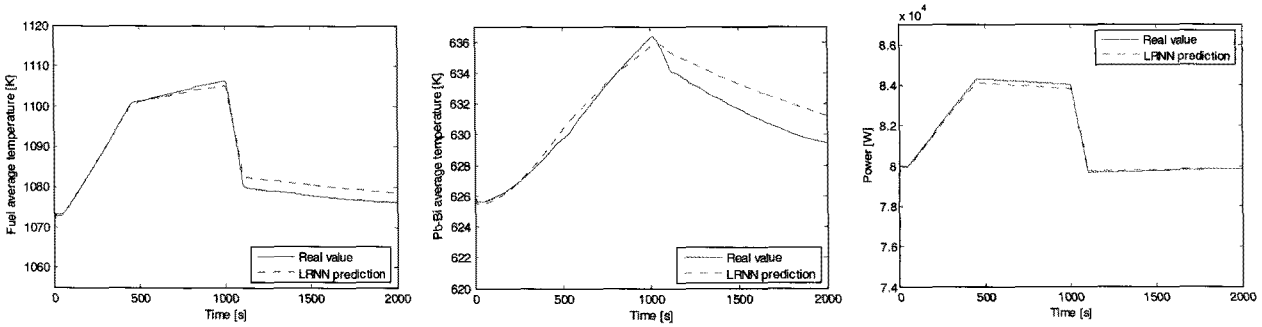


Fig. 10. Free Dynamics. Extrapolation

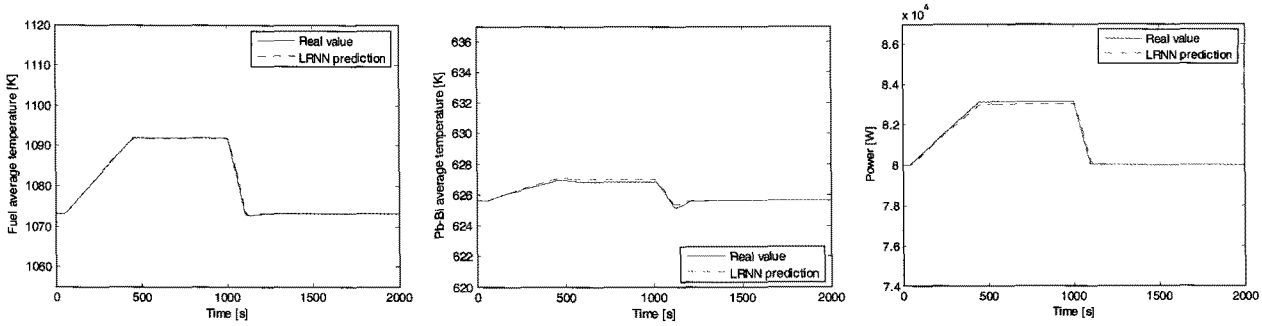


Fig. 11. Controlled Dynamics. Extrapolation

the response of the neural network model developed for fault tolerance, i.e., when fed with incomplete and/or inaccurate data because of equipment malfunction. In the present work, this has been done two ways:

i) one of the input variables, the diathermic oil temperature after the oil heat exchanger ($\tau_1(t)$), has been fixed to a

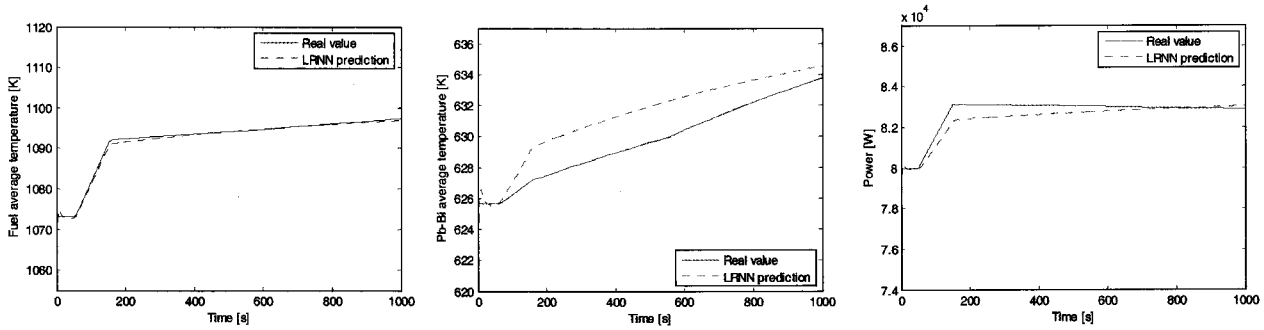


Fig. 12. Free Dynamics. Interpolation

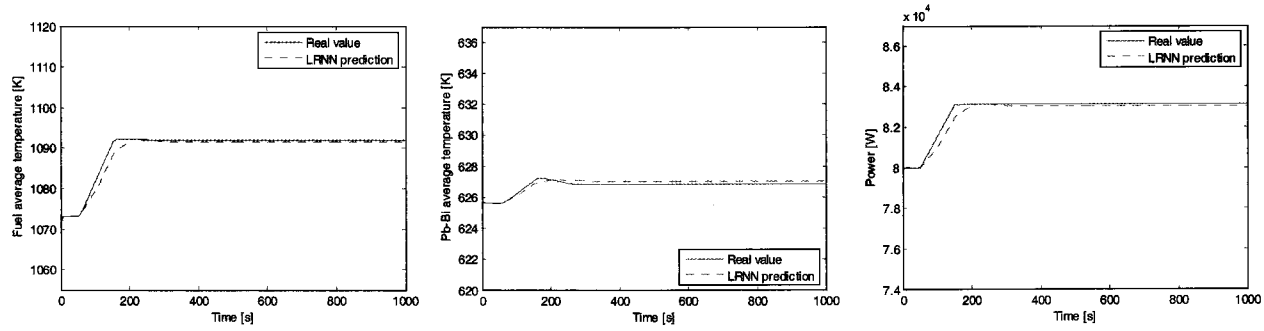


Fig. 13. Controlled Dynamics. Interpolation

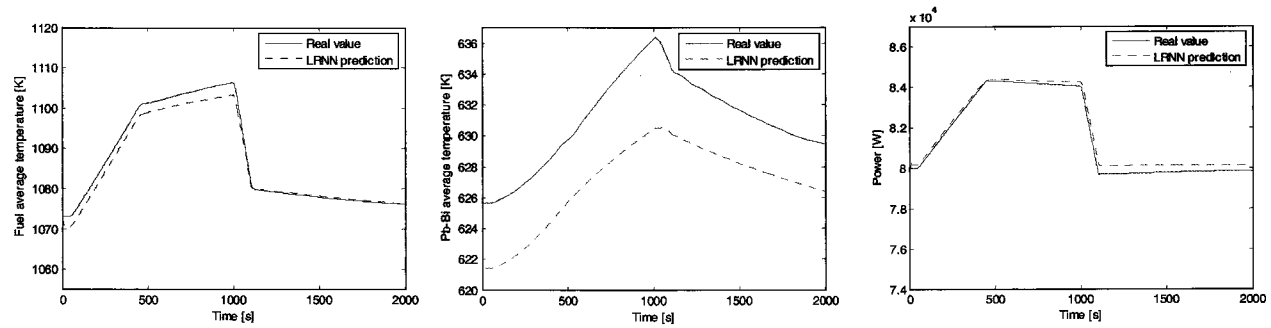


Fig. 14. Free Dynamics. Fault Tolerance: Constant Oil Temperature τ_1 = 589.94 K

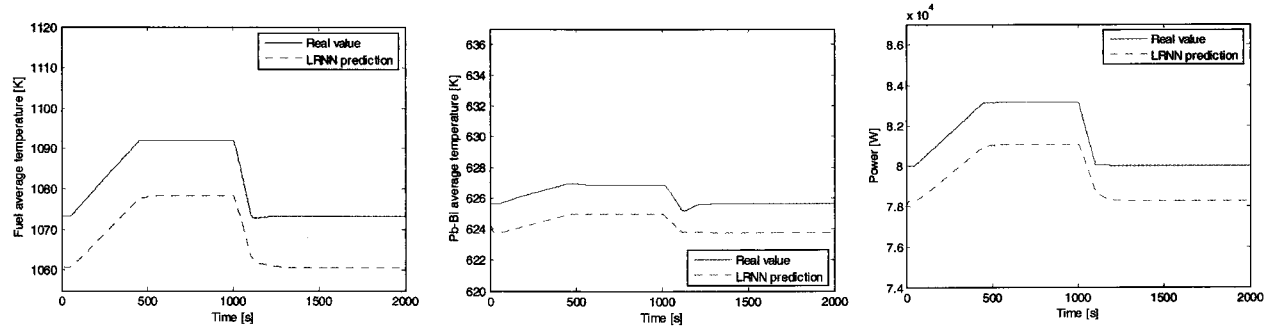


Fig. 15. Controlled Dynamics. Fault Tolerance: Constant Oil Temperature τ_1= 589.94 K

constant value equal to the lower normalization bound in the training transients (589.94 K), for that input;

ii) the input of the diathermic oil temperature before the oil heat exchanger ($\tau_4(t)$) has been set to 90% of its actual value.

The motivation beyond the choice of damaging these inputs is that the challenge to the network modelling capabilities is highest, since they fully describe the temporal evolution of the diathermic oil in the secondary coolant loop, with full information about the time-delays in that circuit.

As shown in Figure 14-17, the IIR-LRNN performs relatively poorly in both fault tolerance tests compared with previous performances. This is due to the fact that there are only temperature inputs, all bearing equally important information to the network, so that the loss of one input significantly alters the network outputs compared to the case with all valid inputs. Yet, the maximum relative errors for the output quantities are limited to 0.0065, 0.0070, 0.0106 and 0.0132, 0.0032, 0.0265 for malfunction i) in the free and controlled dynamics cases, respectively, and 0.0069, 0.0148, 0.0251 and 0.0174, 0.0082, 0.0549 for

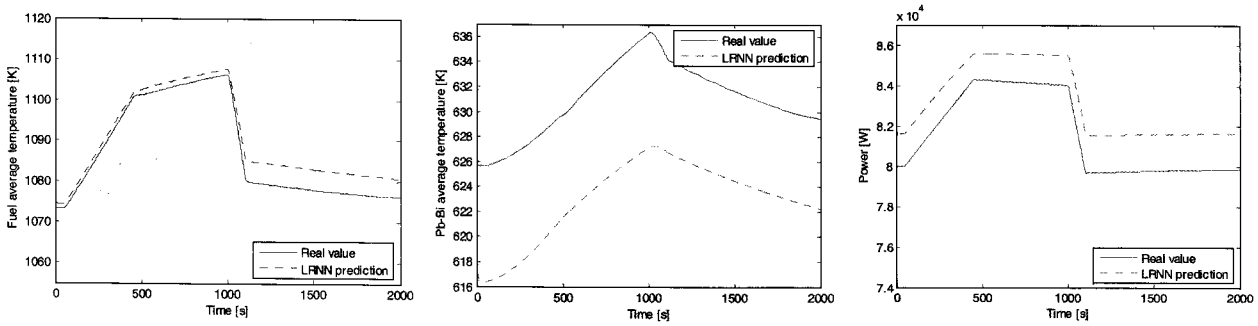


Fig. 16. Free Dynamics. Fault Tolerance: 90% Value of the Diathermic Oil Temperature before the Oil Heat Exchanger τ_4

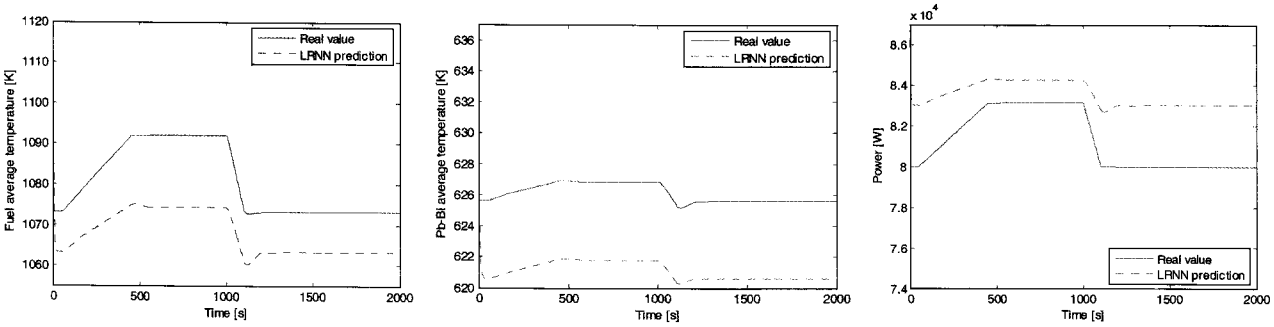


Fig. 17. Controlled Dynamics. Fault Tolerance: 90% Value of the Diathermic Oil Temperature before the Oil Heat Exchanger τ_4

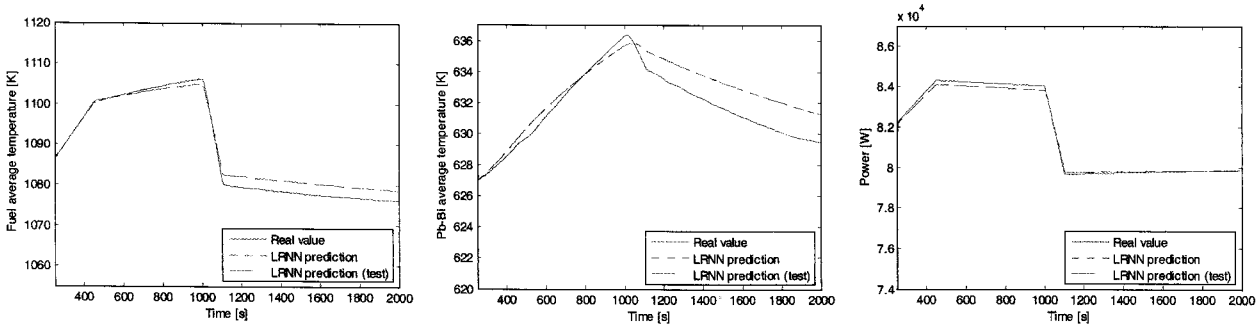


Fig. 18. Free Dynamics. Arbitrary Starting Point

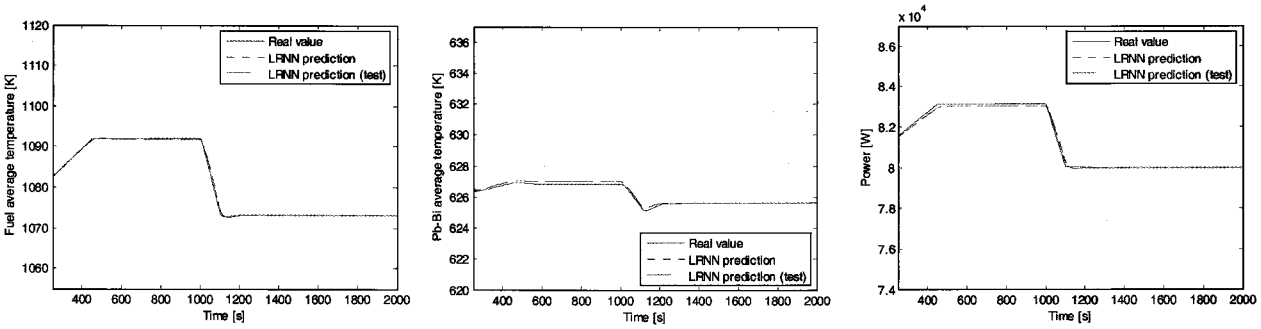


Fig. 19. Controlled Dynamics. Arbitrary Starting Point

malfunction ii) in the free and controlled dynamics cases, respectively.

3.4.4 Arbitrary Initial Conditions Tests

Model-based predictions, in general, require proper setting of some internal variables which might create problems in on-line situations where predictions may be needed starting from arbitrary initial conditions. These situations may arise in real-time control and recovery applications, where sudden variations of the plant state must be followed and the corresponding transients recovered.

In this respect the IIR-LRNN prediction capabilities were tested on the transient of 2000 s duration defined in (16), with starting conditions at $t = 250$ s. As shown in Figure 18 and Figure 19 for the free and controlled dynamics, respectively, at the start, the network output presents a "spike" because of the lack of dynamic information on the values of the inputs at the previous time instants; after a few time steps the network stabilizes its outputs, achieving prediction performances comparable to those of the network that processes the whole transient from the initial conditions at $t = 0$ s.

4. CONCLUSIONS

The ability to model dynamic systems has become a fundamental aspect for the safety of modern industrial plants. System design and testing, fault diagnosis, and control design are just a few of the tasks in modern engineering which rely on the ability of identifying and modelling dynamic systems. These tasks are particularly relevant in critical applications, such as nuclear ones.

Many of the analyses needed for the above tasks (e.g., sensitivity analysis for the identification of the most important parameters of a process, probabilistic uncertainty analysis, on-line and real-time control design) entail several repeated calculations, thus forbidding the use of large, detailed dynamic codes due to the impractical computing times involved. In these cases, one has to resort to either simplified, reduced analytical models, such as those based on lumped effective parameters or empirical models. In both cases, the model parameters have to be estimated so as to best fit the available plant data.

With respect to the latter option of empirical modeling, artificial neural networks are being used with increasing frequency as an alternative to traditional models in a variety of engineering applications including monitoring, prediction, diagnostics, control and safety. Yet, in the critical safety and control applications required by the nuclear industry, their acceptance is still very limited, particularly by the regulatory bodies. A major concern regarding the use of ANN-based monitoring and diagnostic systems in nuclear power plants is that their accuracy as dynamic process estimators must be demonstrated under various realistic conditions of operation.

In this respect, in the previous paper the dynamic modelling performance of the so-called infinite impulse response-locally recurrent neural network (IIR-LRNN) was studied with reference to the nonlinear dynamics of an innovative nuclear reactor design, the so-called Lead Bismuth Eutectic eXperimental Accelerator Driven System: the main focus of that paper was the comparison between batch and on-line algorithms to train the IIR-LRNN architecture.

This paper has extended the previous work by considering an *additional* number of performance tests of the IIR-LRNN under various *different* conditions of operation; for simplicity of illustration only the batch mode of recursive training has been employed.

Two IIR-LRNNs, one for the free and one for the controlled dynamics, have been successfully designed and trained, with a recursive back-propagation (RBP) algorithm, to reproduce the evolution of a number of relevant system parameters, knowing the external neutron source, the diathermic oil temperature and the cooling air evolution.

The modelling and generalization capabilities of the trained IIR-LRNN have been first validated and tested on unknown transients of the same temporal length as that of the training ones: the performance of the IIR-LRNN in these cases has turned out to be very satisfactory.

Additional tests have been successfully performed

under various conditions to verify the generalization and modelling capabilities of the network for i) extrapolation to transients of duration beyond that of training, ii) interpolation tasks, iii) for fault tolerance and iv) transients initiated with arbitrary initial conditions.

As a final remark, the obtained results preliminarily show the potential feasibility of the use of IIR-LRNN as a tool for simulating real complex system dynamics under various challenging conditions; a full verification on more complex transients and on a wider range of power manoeuvring (i.e., larger than the 10% variation considered in this study) will be needed to confirm the capability of the IIR-LRNN in nuclear applications.

ACKNOWLEDGEMENT

The authors express their gratitude to Doctor Antonio Cammi of the Department of Nuclear Engineering of the Polytechnic of Milan for providing the simulator of the LBE-XADS used for data generation and for his fruitful contribution in the definition of the network architecture.

ACRONYMS AND SYMBOLS

ANN	Artificial Neural Network
LRNN	Locally Recurrent Neural Network
IIR	Infinite Impulse Response
MLP	Multi-Layer Perceptron
RBP	Recursive Back-Propagation
RTRL	Real-Time Recurrent Learning
BPTT	Back-Propagation-Through-Time
RMSE	Root Mean Square Error
LBE-XADS	Lead Bismuth Eutectic-eXperimental Accelerator Driven System
k	layer index. In particular, $k=0$ and $k=M$ denote the input and the output layers, respectively
N^k	number of neurons in the k-th layer. In particular, N^0 and N^M denote the number of input and output neurons, respectively
j	neuron index
t	continuous time index
$x_j^k(t)$	output of the j-th neuron of the k-th layer, at time t. In particular, $j=0$ refers to the bias inputs: $x_0^k(t)=1$. Note that $x_j^0(t)$, $j=1, 2, \ldots, N^0$, are the input signals
L_{jl}^k-1	order of the MA part of the synapse of the j-th neuron of the k-th layer relative to the l-th output of the (k-1)-th layer. $L_{jl}^k \geq 1$ and $L_{j0}^k=1$
I_{jl}^k	order of the AR part of the synapse of the j-th neuron of the k-th layer relative to the l-th output of the (k-1)-th layer. $I_{jl}^k \geq 1$ and $I_{j0}^k=1$
$w_{jl(p)}^k$	($p=0, 1, \ldots, L_{jl}^k-1$) coefficients of the MA part of the corresponding synapse. If $L_{jl}^k=1$, the synapse has no MA part and the weight notation becomes $w_{jl}^k \cdot w_{j0}^k$ is the bias
$v_{jl(p)}^k$	($p=1, 2, \ldots, I_{jl}^k$) coefficients of the AR part of the synapse. If $I_{jl}^k=1$ the synaptic filter is purely MA
$f^k(\cdot)$	nonlinear activation function relative to the k-th layer
$f_k'(\cdot)$	derivative of $f^k(\cdot)$
$y_{jl}^k(t)$	synaptic filter output at time t relative to the synapse connecting the j-th neuron of the k-th layer to the l-th input
$s_j^k(t)$	"Net" input to the activation function of the j-th neuron of the k-th layer at time t
$d_r(t)$	($r=1, 2, \ldots, N^M$) desired target of output node r at time t
μ	learning coefficient
α	momentum coefficient
n_{epoch}	number of learning epochs during training
n_{rep}	number of consecutive repetitions of each transient during training
N_t	number of transients in the training/validation/test set
T	temporal length of a transient
Δt	time step for the numerical simulation of a transient
n_p	number of patterns in a training/validation/test transient, $n_p=T/\Delta t$
T_s	steady-state time interval (in step and ramp forcing functions)
T_r	ramp duration (in ramp forcing functions)

REFERENCES

[1] Seidl, D. R., and Lorenz, D., "A Structure by which a Recurrent Neural Network can Approximate a Nonlinear Dynamic System," *Proc. Intl. Joint Conf. on Neural Networks*, Seattle, USA, Jul. 8-14 (1991).

[2] Funashi, K.-I., and Nakamura, Y., "Approximation of Dynamical systems by Continuous Time Recurrent Neural Networks," *Neural Networks*, **6**, 801, (1993).

[3] Siegelmann, H., and Sontag, E., "On the Computational Power of Neural Nets," *Journal of Computer and System Sciences*, **50**, 132, (1995).

[4] Zio, E., Broggi, M., and Pedroni, N., "Nuclear Reactors Dynamics On-line Estimation by Locally Recurrent Neural Networks," *Progress in Nuclear Energy*, **51**, 573 (2009).

[5] Bowman, C. D., Arthur, E. D., Lisowski, P. W., Lawrence, G. P., Jensen, R. J., Anderson, J. L., Blind, B., Cappiello, M., Davidson, J. W., England, T. R., Engel, L. N., Haight, R. C., Hughes, H. G., Ireland, J. R., Krakowski, R. A., LaBauve, R. J., Letellier, B. C., Perry, R. T., Russell, G. J., Staudhammer, K. P., Versamis, G., and Wilson, W. B., "Nuclear energy generation and waste transmutation using an accelerator-driven intense thermal neutron source," *Nucl. Instr. Meth. Phys. Res. A*, **320**, 336 (1992).

[6] Rubbia, C., Rubio, J. A., Buono, S., Carminati, F., Fitier, N., Galvez, J., Gels, C., Kadi, Y., Klapisch, R., Mandrillon, P., Revol, J. P., and Roche, C., "Conceptual Design of a Fast Neutron Operated High Power Energy Amplifier," CERN-AT-95-44(ET), Conseil Europeen pour la Recherche Nucleaire (CERN) (1995).

[7] Campolucci, P., Uncini, A., Piazza, F., and Rao, B. D., "On-Line Learning Algorithms of Locally Recurrent Neural Networks," *IEEE Trans. Neural Networks*, **10**, 253 (1999).

[8] Williams, R., and Zipser, D., "A Learning Algorithm for Continually Running Fully Recurrent Neural Networks," *Neural Computation*, **1**, 270 (1989).

[9] Rumelhart, D. E., Hinton, G. E., and Williams, R. J., "Learning Internal Representations by Error Propagation," *Parallel*

Distributed Processing: Explorations in the Microstructure of Cognition, Rumelhart, D. E. and McClelland, J. L., Eds., MIT Press, Cambridge (1986).

[10] Van Tuyle, G. J., Todosow, M., Geiger, M. J., Aronson, A. L., and Takahashi, H., "Accelerator-driven subcritical target concept for transmutation of nuclear wastes," *Nucl. Technol.*, **101**, 1 (1993).

[11] Venneri, F., Bowman, C. D., and Jameson R., "Accelerator-driven Transmutation of Waste (ATW) - A New Method for Reducing the Long-term Radioactivity of Commercial Nuclear Waste," LA-UR-93-752, Los Alamos National Laboratory (1993).

[12] Carminati, F., Klapisch, R., Revol, J. P., Roche, Ch., Rubio, J. A., and Rubbia, C., "An Energy Amplifier for Cleaner and Inexhaustible Nuclear Energy Production Driven by a Particle Beam Accelerator," CERN-AT-93-47(ET), Conseil Europeen pour la Recherche Nucleaire (CERN) (1993).

[13] Ansaldo, "XADS PbBi Cooled Experimental Accelerator Driven System Reference Configuration," Summary Report ANSALDO-ADS-1-SIFX-0500, Ansaldo Nuclear Division (2001).

[14] Luzzi, L., Vettraino, F., and Calabrese, R., "ADS-demo fuel rod performance analysis," *Global Environment and Nuclear Energy System/Advanced Nuclear Power Plants International Conference (GENES4/ANP2003)*, Kyoto, Japan, Sept. 15-19 (2003).

[15] Cammi, A., Luzzi, L., Porta, A. A., and Ricotti, M. E., "Modelling and control strategy of the Italian LBE-XADS," *Progress in Nuclear Energy*, **48**, 578 (2006).

[16] Adali, T., Bakal, B., Sönmez, M. K., Fakory, R., and Tsaoi, C. O., "Modeling nuclear reactor core dynamics with recurrent neural networks," *Neurocomputing*, **15**, 363 (1997).