# Comment on "Discrete-Time Recurrent Neural Network Architectures: a Unifying Review"

Léon PERSONNAZ, Gérard DREYFUS
ESPCI, Laboratoire d'Électronique
10, rue Vauquelin, 75005 Paris, France
Gerard.Dreyfus@espci.fr

## Abstract

Paper [1] aimed at providing a unified presentation of neural network architectures. We show in the present comment (i) that the canonical form of recurrent neural networks presented by Nerrand et al. [2] many years ago provides the desired unification, (ii) that what Tsoi and Back call Nerrand's canonical form is *not* the canonical form presented by Nerrand et al. in [2], and that (iii) contrarily to the claim of Tsoi and Back, all neural network architectures presented in their paper can be tranformed into Nerrand's canonical form. We show that the contents of Tsoi and Back's paper obscures the issues involved in the choice of a recurrent neural network instead of clarifying them: this choice is definitely much simpler than it might seem from Tsoi and Back's paper.

In [1], Tsoi and Back present a number of different discrete-time recurrent neural network architectures and intend to clarify the links between them. The authors must be commended for trying to perform such a task; they must also be commended for their lucid statement that they are unable to answer the simple question: which neural network to use for a particular given problem. Unfortunately, the authors are not content with not answering this important question; they considerably obscure the issues related to it, by making wrong statements, and by misunderstanding a powerful unifying tool which was published many years ago [2]. In the present comment, we disprove some unjustified and incorrect claims, with special emphasis on those expressed in an aggressive way in the Tsoi and Back paper, and we prove that the choice of a recurrent neural network architecture can be made on a principled basis.

Back in 1993, Nerrand et al. [2] proposed a general approach to the training of recurrent networks, either adaptively (on-line) or non-adaptively (off-line). One of the main points of that paper was the introduction of the canonical form, or minimal state-space form (hereinafter termed Nerrand's canonical form), defined in relations (4) and (4') of their paper as:

$\underline{z}(n+1) = \varphi\,[\underline{z}(n), \underline{u}(n)]$ (state equation) (1)

$\underline{y}(n+1) = \psi[\underline{z}(n+1), \underline{u}(n+1)]$ (output equation) (2)

where $\underline{z}(n)$ is the *minimal* set of variables necessary for computing completely the state of the model at time $n$+1 if the state of the model and its external input vector $\underline{u}(n)$ (control inputs, measured disturbances, ...) are known at time $n$; $\underline{y}(n)$ is the output vector. A graphic representation of the canonical form is shown in Figure 1, assuming that functions $\varphi$ and $\psi$ are computed by a single feedforward network.

In appendix 1 of their paper, Nerrand et al. showed how to compute the order (i.e. the number of state variables) of any recurrent network, and, in appendix 2 they derived the canonical form of various recurrent network architectures, which had been proposed by other authors.

In section 4.9.2 of [1], Tsoi and Back claim that Nerrand's canonical form, presented in [2], can be written as:

$\underline{y}(n+1) = c^T[\underline{z}(n+1)]$ (3)

$\underline{z}(n+1) = \varphi\,[y(n), y(n\text{-}1), ..., y(n\text{-}m)]$ (4)

where $\varphi$ is the vector function performed by the neurons of the hidden layer.

This claim is wrong, for several reasons:

- in Nerrand's canonical form, the output cannot be expressed, in general, as a nonlinear function of the past outputs,
- in Nerrand's canonical form, the output is not, in general, a linear function of the state variables,
- in Nerrand's canonical form, there are, in general, external inputs,
- in Nerrand's canonical form, function $\varphi$ can be implemented as an arbitrary feedforward network, not necessarily a layered one.

What Tsoi and Back claim to be a canonical form is actually a specific input-output form, without external inputs. It is a very special case of Nerrand's canonical form, where $\psi$ is the identity function, where there are no external inputs, and where $\varphi$ is implemented as a one-hidden layer feedforward network.

Actually, Figure 2 of the paper by Nerrand et al. shows, as a special case of Nerrand's canonical form, a graphical representation of the most general input-output form of a recurrent neural network; it is more general than the form (3) - (4) of Tsoi and Back (termed "static GFGR" in their paper) in the sense that function $\varphi$ is implemented as a fully connected feedforward network, and that it does have external inputs.

In an aggressively repetitious way (in section 4.9.2, then in the introduction of section 6, then in section 6.1, and in section 6.2, and also, in addition, in another paper by Tsoi [3] published in the same issue), Tsoi and Back claim that the statement made in [2] that any neural network can be cast into Nerrand's canonical form is not supported in [2], and wrong in general. They also claim repetitively that there is no systematic way of transforming a neural network into Nerrand's canonical form. This would be correct if

Nerrand's canonical form was the simple input-output form (3) - (4); as shown above, this is not the case. Therefore, all claims made by Tsoi and Back related to this issue are definitely incorrect.

The fact that any discrete-time model, of a class which contains all the neural network architectures presented by Tsoi and Back, can be transformed into Nerrand's canonical form, was proved in [4]. A systematic procedure for transforming any neural network (i.e. a neural network with any graph of connections, any delays in the connections, and any nonlinearity or absence of nonlinearity in the nodes of the graph - which includes what Tsoi and Back call "networks with dynamic connections") into a canonical form was proposed, and several practical examples were given, in [4] and [5].

As an additional comment, we disprove the claim that a dynamic GFGR cannot be transformed into Nerrand's canonical form. Consider a dynamic GFGR

$$\boldsymbol{z}(t) = F_n(A\,\boldsymbol{v}(t) + B\,\boldsymbol{u}(t) + C\,\boldsymbol{i}(t) + \tau_z)$$

$$\boldsymbol{v}(t) = H\,\zeta(t)$$

$$\zeta(t) = F\,\zeta(t\text{-}1) + G\,\boldsymbol{z}(t\text{-}1)$$

$$\text{with } \boldsymbol{u}(t) = \begin{vmatrix} y(t\text{-}1) \\ : \\ y(t\text{-}m) \end{vmatrix} \text{ and } \boldsymbol{i}(t) = \begin{vmatrix} e(t) \\ : \\ e(t\text{-}m'\text{+}1) \end{vmatrix}\text{, e(t) being the external input}$$

$$y(t) = \boldsymbol{c}^{\mathrm{T}}\,\boldsymbol{z}(t)$$

An example is shown on Figure 2.

The transformation of such a network into Nerrand's canonical form can be readily performed. We denote by $\boldsymbol{x}(t)$ the vector $\boldsymbol{x}(t) = \begin{bmatrix} \zeta(t) \\ \boldsymbol{z}(t) \\ y(t) \\ \boldsymbol{:} \\ y(t\text{-}m\text{+}1) \end{bmatrix}$; then the state equation of the GFGR can be written as $\boldsymbol{x}(t) = F_{NN}\big(\boldsymbol{x}(t\text{-}1), \boldsymbol{i}(t)\big)$

$$y(t) = [0 \ldots 0\ 1\ 0 \ldots 0]\ \boldsymbol{x}(t);$$

where $F_{NN}$ is the feedforward part of Nerrand's canonical form of the considered dynamic GFGR. This canonical form is shown on Figure 3

As a practical consequence, there is definitely no point in considering all the architectures described at length in the Tsoi and Back paper, since they are all amenable to the genuine Nerrand's canonical form as appropriately described by equations (1) and (2) of the present comment. All the terminology and endless architectural variations proposed in the Tsoi and Back paper (as well as in many other papers in the literature) only bring about extreme confusion. For practitioners of modeling with recurrent neural networks, the situation is actually reasonably simple; two possibilities arise:

- no prior knowledge on the process or time series to be modeled is available (black-box modeling); then one should try Nerrand's canonical form, since it is the most general possible form. Training should be performed by evaluating the gradient of the cost function with the algorithms described in [2], and weight updates should be computed with a second-order algorithm for best results; the determination of the order of the network and of the number of hidden neurons may be done by trial and error, or by making use of model selection methods [6]; a real-life example, showing clearly the superiority of state-space models (1) - (2) over input-output models (3) - (4), is given in [7];
- some prior knowledge is available in mathematical form; then these equations, which are generally in a state-space form, should be used in order to build the architecture of the neural model; if the latter is not in a canonical form, it should be transformed into a canonical form, either manually or by the procedure described in [4], and subsequently trained as described above. This methodology, termed "knowledge-based neural modeling", and an industrial application thereof, are presented in [8] and [9].

To summarize, we make the following points:
(i) what Tsoi and Back call "canonical form" or "static GFGR neural architecture" is not the canonical form introduced several years ago by Nerrand et al.; it is a very particular case of the input-output form also introduced by Nerrand et al.;
(ii) contrarily to the claims of Tsoi and Back, the genuine Nerrand's canonical form is indeed the most general form of recurrent neural architecture; despite Tsoi and Back's heavily repeated - but nevertheless unjustified - criticisms, any neural network architecture can be transformed into Nerrand's canonical form;
(iii) from a practical standpoint, there is definitely no point in trying an endless variety of architectures, such as presented in Tsoi and Back's paper, which are all amenable to Nerrand's canonical form. In the absence of any prior knowledge on the process to be modeled, Nerrand's canonical form should be tried first; if prior knowledge is available, in the form of mathematical equations, this knowledge should be used for determining a tentative architecture, which should be transformed into a canonical form and subsequently trained.

As a final point, it is the firm belief of the authors of the present comment that, since their paper was subject to heavy criticism by Tsoi and Back in [1], they should have been given a fair chance to reply to these authors in the same issue.

## LITERATURE REFERENCES

[1] A.C. Tsoi, A.D. Back, Discrete Time Recurrent Neural Network Architecture, a Unifying Review, Neurocomputing 15 (1997) 183 - 223.

[2] O. Nerrand, P. Roussel-Ragot, L. Personnaz, G. Dreyfus, Neural Networks and Nonlinear Adaptive Filtering: Unifying Concepts and New Learning Algorithms, Neural Comput. 5 (1993) 165-199.

[3] A.C. Tsoi, S. Tan, Recurrent Neural Networks: a Constructive Algorithm, and its Properties, Neurocomputing 15 (1997), 309-326

[4] G. Dreyfus, Y. Idan, The Canonical Form of Discrete-Time Nonlinear Models, Neural Comput. 10 (1998) 133-164.

[5] G. Dreyfus, Y. Idan, Discrete-time Neural Networks and their Canonical Form, Machines that Learn Conference, Snowbird, Utah (1997).

[6] D. Urbani, P. Roussel-Ragot, L. Personnaz, G. Dreyfus, The Selection of Neural Models of Non-linear Dynamical Systems by Statistical Tests, Neural Networks for Signal Processing IV, 229-237 (IEEE, 1993).

[7] I. Rivals, L. Personnaz, G. Dreyfus, J.L. Ploix, Modélisation, classification et commande par réseaux de neurones : principes fondamentaux, méthodologie de conception et illustrations industrielles, *in* Les réseaux de neurones pour la modélisation et la commande de procédés, J.P. Corriou, ed. (Lavoisier Tec & Doc, 1995).

[8] J.L. Ploix, G. Dreyfus, Knowledge-based Neural Modeling: Principles and Industrial Applications, *in* Industrial Applications of Neural Networks, F. Fogelman-Soulié, P. Gallinari, eds. (World Scientific, 1997).

[9] J.L. Ploix, G. Dreyfus, Early Fault Detection in a Distillation Column: an Industrial Application of Knowledge-based Neural Modeling, *in* Best Neural Network Practice in Europe, SNN'97 ( Amsterdam, 1997).

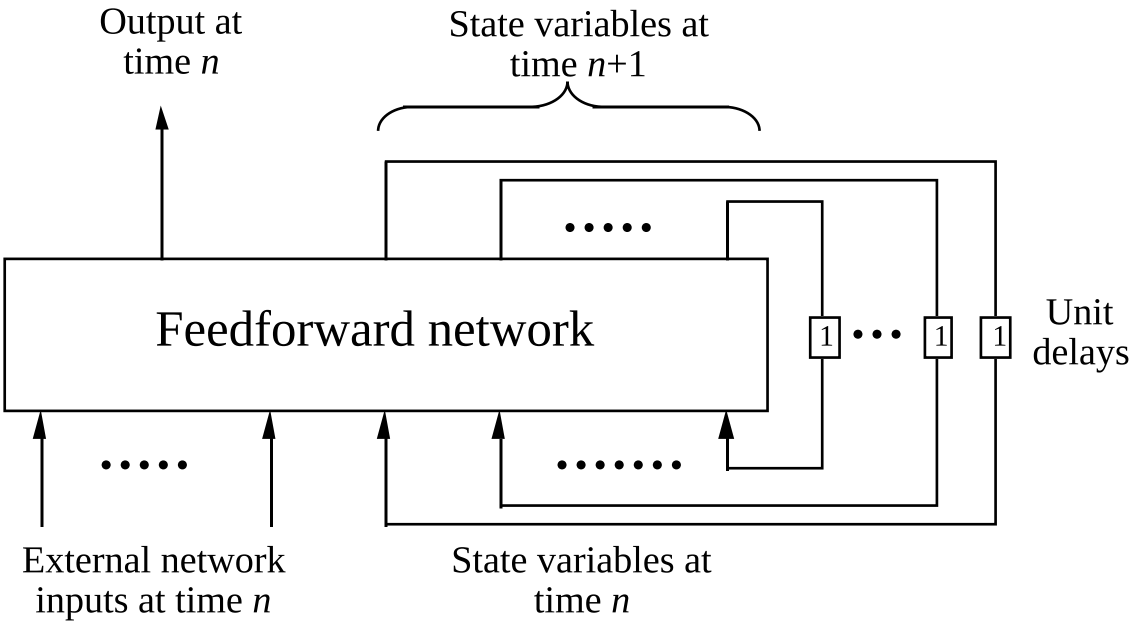
Output at
time *n*
State variables at
time *n*+1
Feedforward network
1
1
1
Unit
delays
External network
inputs at time *n*
State variables at
time *n*

Figure 1

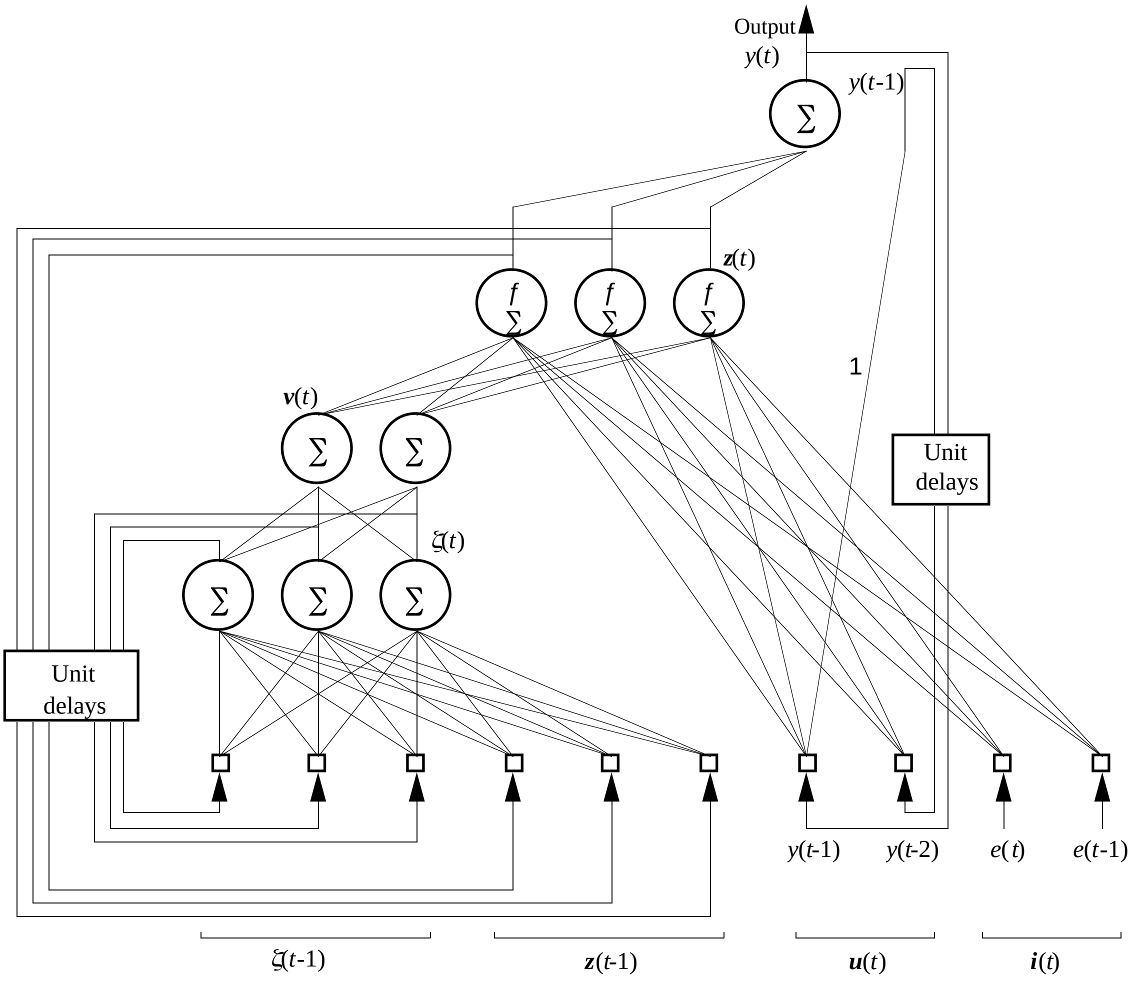
Output
y(t)
y(t-1)
∑
z(t)
f ∑
f ∑
f ∑
1
v(t)
∑
∑
Unit delays
ξ(t)
∑
∑
∑
Unit delays
y(t-1)
y(t-2)
e(t)
e(t-1)
ξ(t-1)
z(t-1)
u(t)
i(t)

Figure 2

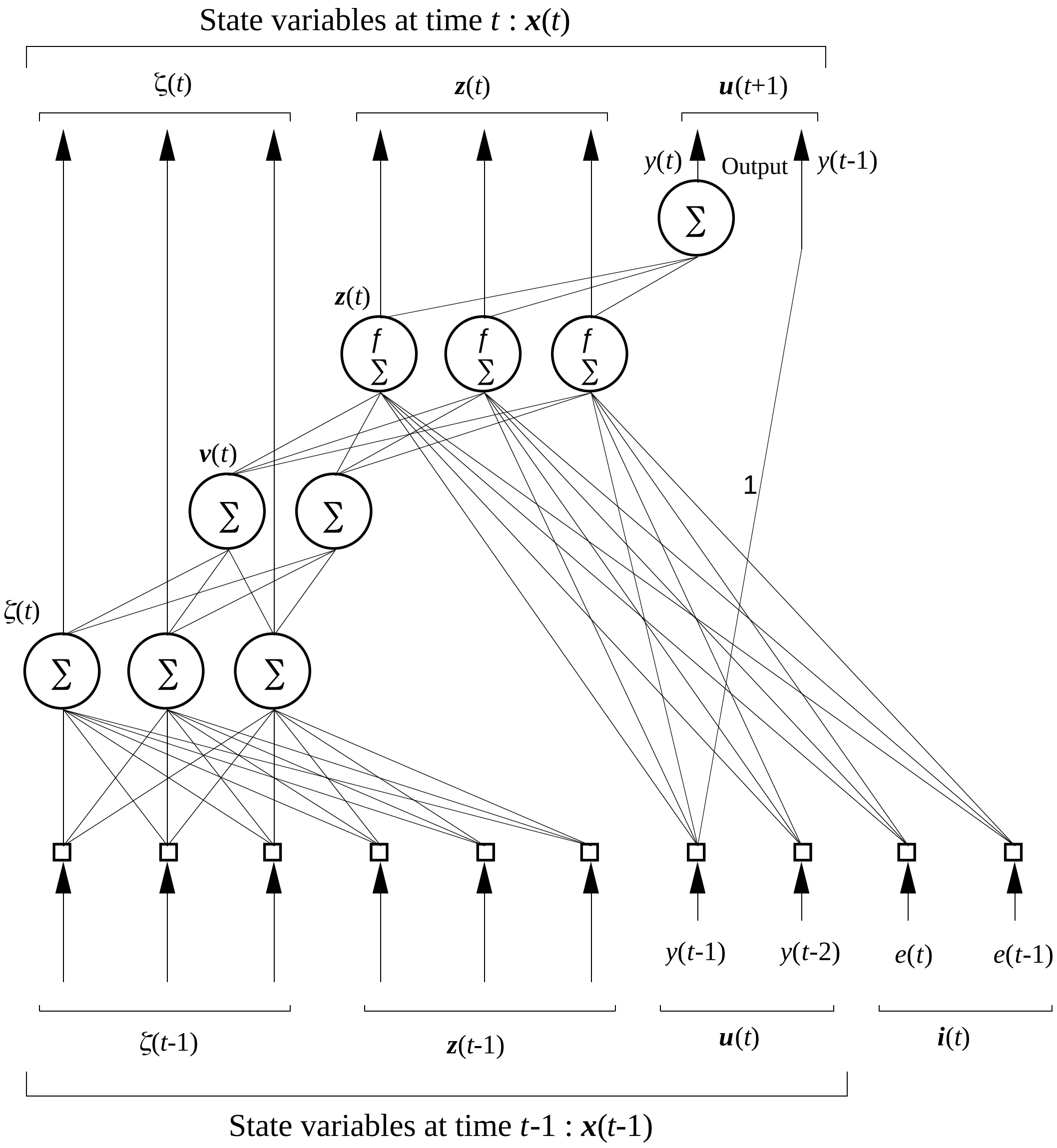
State variables at time *t* : $\boldsymbol{x}(t)$
ζ(*t*)
$\boldsymbol{z}(t)$
$\boldsymbol{u}(t+1)$
*y*(*t*)
Output
*y*(*t*-1)
$\boldsymbol{z}(t)$
*f*
Σ
$\boldsymbol{v}(t)$
1
ζ(*t*)
Σ
*y*(*t*-1)
*y*(*t*-2)
*e*(*t*)
*e*(*t*-1)
ζ(*t*-1)
$\boldsymbol{z}(t-1)$
$\boldsymbol{u}(t)$
$\boldsymbol{i}(t)$
State variables at time *t*-1 : $\boldsymbol{x}(t-1)$

Figure 3

# FIGURE CAPTIONS

FIGURE 1
Nerrand's canonical form of a recurrent neural network.

FIGURE 2
Example of GFGR network

FIGURE 3
A canonical form of the GFGR network of Figure 2