# UNDERSAMPLING FOR THE TRAINING OF FEEDBACK NEURAL NETWORKS ON LARGE SEQUENCES; APPLICATION TO THE MODELING OF AN INDUCTION MACHINE

*L. Constant *, B. Dagues *, I. Rivals **, L. Personnaz ***

(*) Laboratoire d'Électrotechnique et d'Électronique Industrielle
UMR CNRS n°5828
2, rue Camichel, BP 7122, 31071 TOULOUSE Cedex 7, FRANCE
E-mail: Constant@leei.enseeiht.fr

(**) Laboratoire d'Électronique de l'École Supérieure de Physique et de Chimie Industrielles
UPR CNRS n°9081 (NETS)
10, rue Vauquelin, 75231 PARIS Cedex 05, FRANCE
E-mail: Isabelle.Rivals@espci.fr

## ABSTRACT

This paper proposes an economic method for the nonlinear modeling of dynamic processes using feedback neural networks, by undersampling the training sequences. The undersampling (i) allows a better exploration of the operating range of the process for a given size of the training sequences, and (ii) it speeds up the training of the feedback networks. This method is successfully applied to the training of a neural model of the electromagnetic part of an induction machine, whose sampling period must be small enough to take fast variations of the input voltage into account, i.e. smaller than 1 $\mu s$.

## 1. INTRODUCTION

We are interested in the real-time emulation of systems consisting of a static converter, an induction machine, and of the associated sensors [1]. Implementations of classic induction machine models are computationally intensive, which prevents real-time processing. The ability to parsimoniously approximate nonlinear mappings, as well as the possibility of parallel computing, make neural networks efficient in terms of accuracy and computation time, and make their use interesting in this context. This paper thus presents the neural modeling [2] [3] of the electromagnetic part of an induction machine [4] [5] [6]. A difficulty of this type of dynamic modeling is the design of training sequences exploring the entire range of operation of the induction machine. Moreover, even if the dynamics of the induction machine itself does not necessitate a very high sampling frequency, the latter should be high enough to take fast variations of the input voltage into account, constraint which leads to huge training sequences. Under these conditions, the training is not easily achievable from the point of view both of its duration and of its success, due to local minima. To overcome these problems, we propose (i) to undersample the training sequences, and (ii) to perform an adequate modification of the parameters obtained after training so that the feedback neural model finally operates at the desired, higher, sampling frequency. Thanks to this method, it is possible for a given number of samples to explore a larger portion of the range of operation of the process, i.e. to make the training sequences more informative, and to perform a more efficient training.

## 2. NEURAL MODELING OF AN INDUCTION MACHINE

This section introduces the process (simulated induction machine) to be modeled, the neural model architecture, and the training procedure used to estimate the network parameters.

### 2.1. Reference model

The simulated process is based on the classic two-phase model of an induction machine with a three-phase stator, $p$ pairs of poles and a squirrel-cage rotor. The two-phase model in the reference frame $(\alpha, \beta)$ fixed on the stator consists of:

– a system of differential equations for the fluxes:

$$\begin{cases} \dfrac{d\phi_{s\alpha}}{dt} = -R_s \dfrac{1}{\sigma L_s} \phi_{s\alpha} + R_s \dfrac{1-\sigma}{\sigma M} \phi_{r\alpha} + V_{s\alpha} \\ \dfrac{d\phi_{s\beta}}{dt} = -R_s \dfrac{1}{\sigma L_s} \phi_{s\beta} + R_s \dfrac{1-\sigma}{\sigma M} \phi_{r\beta} + V_{s\beta} \\ \dfrac{d\phi_{r\alpha}}{dt} = +R_r \dfrac{1-\sigma}{\sigma M} \phi_{s\alpha} - R_r \dfrac{1}{\sigma L_r} \phi_{r\alpha} - p\, \Omega\, \phi_{r\beta} \\ \dfrac{d\phi_{r\beta}}{dt} = +R_r \dfrac{1-\sigma}{\sigma M} \phi_{s\beta} - R_r \dfrac{1}{\sigma L_r} \phi_{r\beta} + p\, \Omega\, \phi_{r\alpha} \end{cases} \quad (1)$$

– a relation between the fluxes and the electromagnetic torque:

$$T_{em} = p \frac{1-\sigma}{\sigma M} \left( \phi_{s\beta}\, \phi_{r\alpha} - \phi_{s\alpha}\, \phi_{r\beta} \right) \quad (2)$$

– relations between the fluxes and the stator currents:

$$\begin{cases} I_{s\alpha} = \dfrac{1}{\sigma L_s} \phi_{s\alpha} - \dfrac{1-\sigma}{\sigma M} \phi_{r\alpha} \\ I_{s\beta} = \dfrac{1}{\sigma L_s} \phi_{s\beta} - \dfrac{1-\sigma}{\sigma M} \phi_{r\beta} \end{cases} \quad (3)$$

A discrete-time model is obtained by discretizing the above continuous-time model using the Runge-Kutta method of order 4. This discrete-time model, termed “reference model”, is used to generate training and test sequences for the neural model.

### 2.2. Neural model architecture

The neural network architecture is based on the equations of the two-phase model, and consists thus of two distinct parts. The first one, a feedback (dynamic) input-output neural network shown in **Fig. 1**, corresponds to the system of differential equations for the fluxes (1). Its state outputs are the 4 fluxes, which are computed according to the stator voltages, the mechanical speed, and the fluxes at the previous time step. According to the reference model, the 2 stator fluxes are estimated by two linear functions, and the 2 rotor fluxes are estimated by 2 nonlinear subnetworks. These subnetworks have only the input variables involved in the corresponding differential equations, a layer of 4 hidden neurons with hyperbolic tangent activation function, and a linear output neuron. The second part of the neural model is a feedforward (static) neural network, which implements relations (2) and (3): it computes the stator currents and the electromagnetic torque according to the fluxes [4]. The training of the feedforward network being independent from the sampling period, we focus here on the training of the feedback network.

### 2.3. Training on undersampled sequences

A very small sampling period is not necessary to obtain an accurate discretization of the continuous-time model of the electromagnetic part of the induction machine, but at the end, the induction machine model must be associated to a static converter model. Phenomena such as the dead-times of the converter must thus be taken into account, so that the working sampling period must be at most of 1 $\mu s$. The design of an appropriate training sequence mainly necessitates the determination of the domain defined by the ranges of mechanical speed, flux amplitude, and load torque that must be explored in the training sequence so that the network is able to reproduce unlearned modes of operation. In [7], such a training sequence has been designed. The latter being extremely large (it consists of hundreds of thousands of samples), it is undersampled here at 10 $\mu s$ for the training of the neural model.

The training is performed in an undirected (parallel) fashion using an iterative non recursive quasi-Newtonian algorithm; the gradient of the cost-function is computed by backpropagation [2] [3]. Section 3 presents how the network parameters values obtained after training can be modified so that the network operates at the desired sampling period, i.e. 1 $\mu s$.

## 3. SAMPLING PERIOD MODIFICATION

We suppose that a SISO first-order nonlinear discrete-time predictive model of an unknown continuous-time process has been obtained using measurements at a sampling period $T$:

$$y(k+1) = a\, y(k) + b\, u(k) + c\, f\big(y(k), u(k)\big) \quad (4)$$

where $f$ is a nonlinear function. The problem is to derive from (4) a discrete-time model working at a smaller sampling period $T'$:

$$y(k+1) = a'\, y(k) + b'\, u(k) + c'\, f\big(y(k), u(k)\big) \quad (5)$$

For this purpose, we propose to consider that the discrete-time model (4) is a discretization of the following fictitious continuous-time model (6) with one of the usual discretization rules:

$$\frac{dy}{dt} = F(y, u) = a_c\, y + b_c\, u + c_c\, f(y, u) \quad (6)$$

Subsection 3.1 determines which discretization rules allow the computation of $a'$, $b'$, $c'$ from $a$, $b$, $c$.

### 3.1. Possible discretization rules

We search for the discretization rules on which it is possible to base a transformation of the discrete-time model (4) into the discrete-time model (5).

***a) Euler’s backward rule***

The approximate discretization of (6) using Euler’s backward rule is obtained by assuming that $dy/dt$ is constant between $kT$ and $(k+1)T$, and that it is equal to $F(kT)$. Hence the model for a sampling period $T$:

$$\begin{cases} y_e(k+1) = \big(1+a_cT\big)y_e(k) + b_cTu(k) + c_c\,T f\big(y_e(k), u(k)\big) \\ \qquad\qquad\;\; = a\, y_e(k) + b\, u(k) + c\, f\big(y_e(k), u(k)\big) \end{cases} \quad (7)$$

Performing Euler at $T'$, and eliminating $a_c$, $b_c$, $c_c$ with (7), one obtains:

$$\begin{cases} a' = 1 + \dfrac{T'}{T}(a-1) \\ b' = \dfrac{T'}{T}\, b \\ c' = \dfrac{T'}{T}\, c \end{cases} \quad (8)$$

***b) Euler’s forward rule***

The discretization using Euler’s forward rule is obtained by assuming that $dy/dt$ is constant between $kT$ and $(k+1)T$, and equal to $F((k+1)T)$. Because of the non-linearity $f$ in $F$, an explicit difference equation is generally impossible to derive (for instance if $f$ is implemented by nonlinear neurons). This rule is thus not suited to our problem.

***c) Tustin’s (trapezoidal) rule***

For the same reason as for Euler’s forward rule, this rule is also not suited to our problem.

***d) Other rules***

For a linear model, there exist many other rules to derive a discrete-time transfer function [8], in particular those leading to a behavior of the discrete-time system that is identical to that of the continuous-time system at the sampling instants for a given type of input (impulse, step, ramp, sinusoid, etc.). Unfortunately,

there is no general rule for exact matching even for specific inputs in the nonlinear case.
To conclude, the only rule that can generically be used for our problem is Euler's backward rule.

### 3.2. Application to feedback neural networks

For simplicity, we consider a first-order feedback network with $Ne$ external inputs (possibly including a constant unit input), a layer of $Nh$ nonlinear neurons, and a linear state output with direct connections to the inputs (see **Fig. 2**). We denote the external inputs at time $k$ by $\{x_i(k)\}$ for $i$=1 to $Ne$, and the outputs of the hidden neurons by $\{x_{Ne+i}(k)\}$ for $i$=1 to $Nh$. The elements of the parameter vector $\theta$ corresponding to all the connections to the output are numbered as in **Fig. 2.** The network behavior is described by:

$$y\,(k+1) = \theta_{Ne+Nh+1}\, y(k) + \sum_{i=1}^{Ne+Nh} \theta_i\, x_i(k) \qquad (9)$$

We look for the parameters $\theta'$ of the output neuron of the network working at a smaller sampling period $T'$. This network behavior is by definition described by:

$$y\,(k+1) = \theta'_{Ne+Nh+1}\, y(k) + \sum_{i=1}^{Ne+Nh} \theta'_i\, x_i(k) \qquad (10)$$

As shown in the previous section, Euler's backward rule can be used to derive the parameters $\theta'$ :

$$\theta'_{Ne+Nh+1} = 1 + \frac{T'}{T}\left(\theta_{Ne+Nh+1} - 1\right) \qquad (11)$$

$$\theta'_i = \frac{T'}{T}\,\theta_i \quad \text{for } i=1 \text{ to } Ne+Nh \qquad (12)$$

The elements of $\theta$ and $\theta'$ corresponding to the connections from the inputs to the hidden neurons are identical.
This transformation is easily generalized to the case of a state-space neural network [3] with several states: for each state output neuron, the parameter of the connection from the corresponding state input is modified according to (11), and all the others according to (12).

## 4. TEST OF THE NEURAL MODEL

A neural model working at 1 $\mu s$ is obtained by performing the parameter transformations (11) (12) for the 4 state outputs $\phi_{s\alpha}$, $\phi_{s\beta}$, $\phi_{r\alpha}$, $\phi_{r\beta}$, of the neural model of the induction machine electromagnetic part trained at 10 $\mu s$ as described in section 2.
A test sequence, shown in **Fig. 3**, is used to assess the accuracy of the neural model predictions. This sequence consists of a machine start-up, i.e. involves operating conditions which are not explored by the training sequences. The error of the neural networks computing $\phi_{s\alpha}$ and $\phi_{r\alpha}$ for example is less than 0.5 percent of the corresponding reference fluxes $\phi_{s\alpha\ ref}$ and $\phi_{r\alpha\ ref}$, as shown on **Fig. 4**. The networks operating at the reduced sampling period ($T'$ = 1 $\mu s$) are even more accurate than those operating at $T$ = 10 $\mu s$ .
An increase of accuracy would be guaranteed if the neural model had been perfectly trained on sequences obtained by Euler's backward discretization. If this is not the case (i.e. if the sequences are obtained with another discretization method, or more generally if they are measured on a real process), an improved accuracy can still be expected since the fast variations of the inputs are better taken into account at a smaller sampling period. This explains the better performance observed for the neural networks operating at the small sampling period $T'$ = 1 $\mu s$ .

## 5. CONCLUSIONS

This paper demonstrates that it is possible to design a neural model of the induction machine electromagnetic part dedicated to a real-time operation with a sampling period of 1 $\mu s$. The difficulty of carrying out the training of a feedback neural model with such a small sampling period has led us to develop a general method of undersampling, which both facilitates the training and makes it more efficient by allowing a thorough exploration of the operating range of the process. This method will be further tested on the training of a neural model of the induction machine taking magnetic saturation phenomena into account.

## 6. REFERENCES


[1] S. Ben Saoud, F. Chouzal, B. Dagues, J. C. Hapiot, "Emulator of static converters/electrical machines; application to the test of new control algorithms", *EPE'95*, Sept. 1995, Sevilla, Spain.

[2] O. Nerrand, P. Roussel-Ragot, L. Personnaz, G. Dreyfus, "Neural networks and nonlinear adaptive filtering: unifying concepts and new algorithms", Neural Computation **5**, 1993, pp. 165-199.

[3] I. Rivals, L. Personnaz, "Black-box modeling with state-space neural networks", in *Neural Adaptive Control Technology*, R. Zbikowski and K. J. Hunt eds, 1996, World Scientific, pp. 237-264.

[4] L. Constant, P. Lagarrigues, B. Dagues, I. Rivals, L. Personnaz, "Modeling of electromechanical systems using feedback neural networks", in *Computational Intelligence and Applications*, P.S. Szczepaniak eds, 1999, Physica-Verlag, pp. 137-143.

[5] L. Constant, P.-E. Lagarrigues, B. Dagues, I. Rivals, L. Personnaz, "Neural modeling of an induction machine", *System Modeling and Control*, Apr. 1998, Zakopane, Poland.

[6] J. F. Martins, A. J. Pires, J. A. Dente, "Automatic input/output modeling of a squirrel-cage induction motor drive system using neural network", *EPE'97,* Sept. 1997, Trondheim, Norway.

[7] L. Constant, R. Ruelland, B. Dagues, I. Rivals, L. Personnaz, "Identification and validation of a neural network estimating the fluxes of an induction machine", *Electrimacs'99*, sept. 1999, Lisbon, Portugal.

[8] R. H. Middleton, G. C. Goodwin, *Digital control and estimation; a unified approach.* Prentice-Hall International, London, 1990.

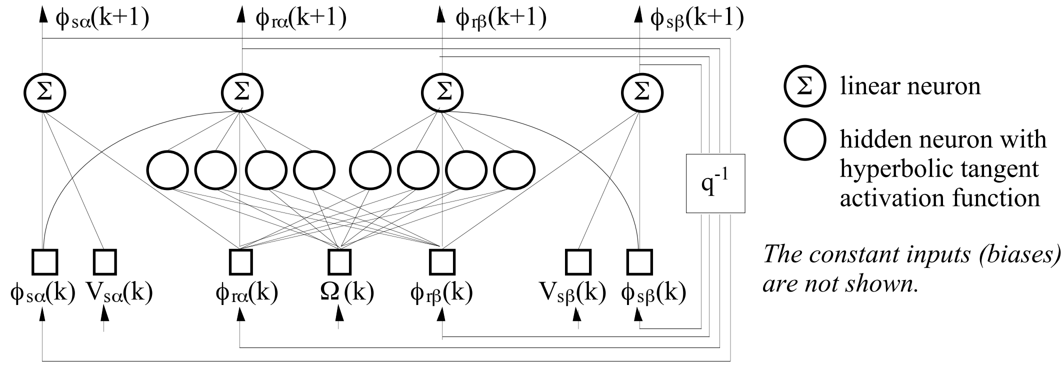


**Fig. 1**: Feedback neural model of the induction machine electromagnetic part.

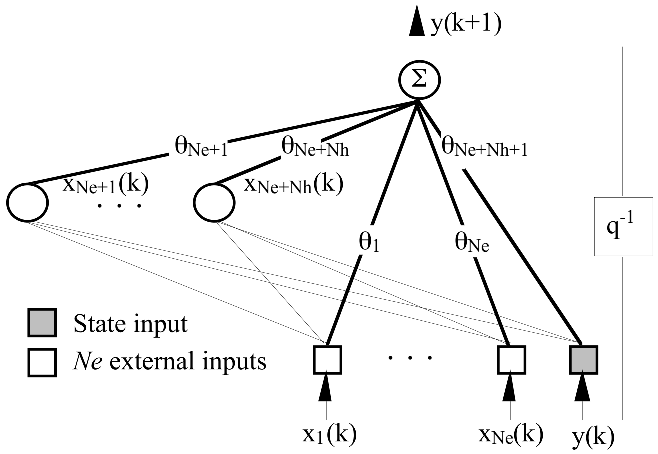


**Fig. 2:** Feedback neural network of order 1.

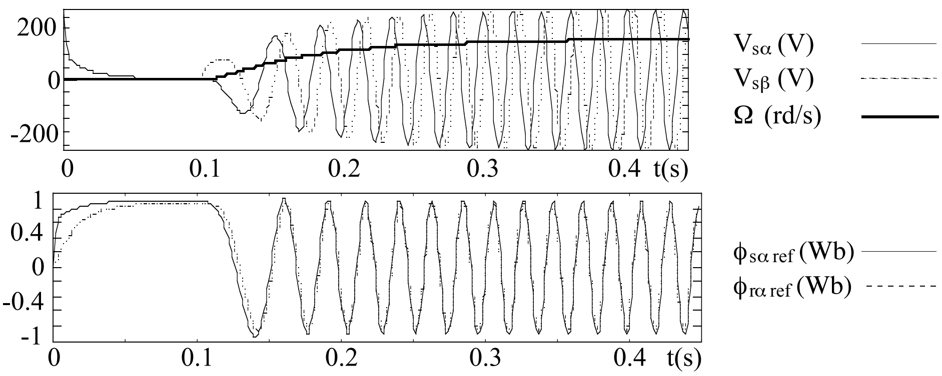


**Fig. 3**: Inputs ( $V_{s\alpha}$, $V_{s\beta}$, $\Omega$) and outputs ( $\phi_{s\alpha\, ref}$, $\phi_{r\alpha\, ref}$) of the test sequence.

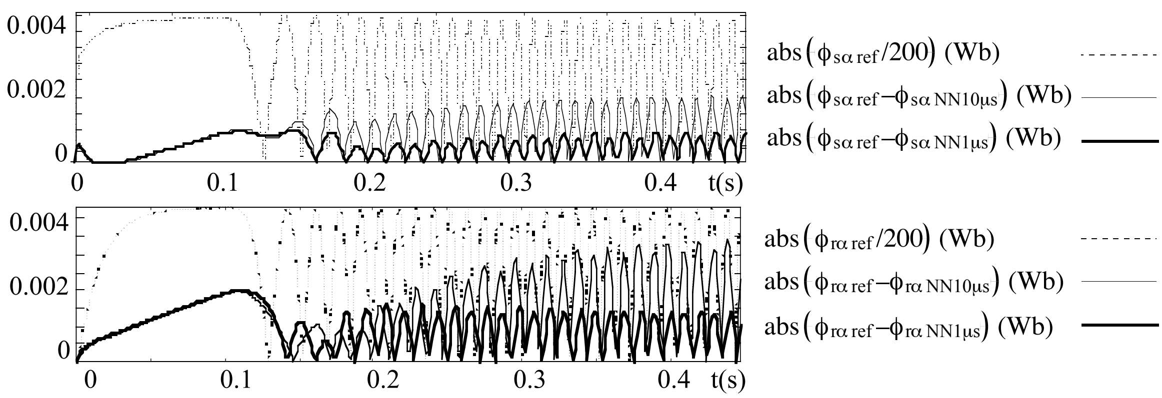


**Fig. 4:** Errors of the neural networks working at $T = 10\ \mu s$ and $T' = 1\ \mu s$.