INVERSION OF A SEMI-PHYSICAL DISPERSION MODEL

Laurent Bourgois, Gilles Roussel, Mohammed Benjelloun

Laboratoire d'Analyse des Systèmes du Littoral (EA 2600)
Université du Littoral - Côte d'Opale
50 rue Ferdinand Buisson, B.P. 699, 62228, Calais Cedex, France
firstname.name@lasl.univ-littoral.fr

Abstract: This study proposes to examine the performances of an inverse dynamic model resulting from the fusion of deterministic modeling and statistical learning. An inverse semi-physical or gray-box model is then carried out using a recurrent neural network (NN). The suggested model concerns in particular a pollutant dispersion phenomenon governed by a partial differential equation (PDE), on a basic mesh. This technique leads to the realization of a neural network inverse problem solver (NNIPS). The network is structured by the discrete reverse-time state form of the direct model. The performances are numerically analyzed in terms of generalization, regularization and training effort.

Keywords: Semi-physical modeling, inverse problem, neural network, model fusion.

1. INTRODUCTION

The objective of many applications such as identification and deconvolution in numerical communication, data assimilation in meteorology, or open-loop control system is to realize the inversion of a physical model. It generally consists in estimating nonmeasurable parameters or inputs starting from the measurable observations and *a priori* information about the system. However, model inversion is often tricky to carry out. One needs to establish a robust direct model within the meaning of exhaustiveness compared to the variations of context, and to choose an efficient inversion method which minimize the effects of data and parameters errors. Our approach aims at simultaneously overcoming these two aspects. The idea is to design an inverse semi-physical neural model by fusion of deterministic modeling and statistical learning. Thus, a NNIPS is structured by the inverse model resulting from the direct deterministic model. A robust inverse model is then ensured using *a priori* knowledge on the physical laws which govern the system. Semi-physical neural modeling has been used by (Oussar and Dreyfus, 2001) in the case of direct models. This type of model fulfills at the same time precision requirements, generics, parsimony of the knowledge-based models, and also possesses the faculty of training and adaptability. Close approaches have been proposed by (Cherkassky *et al.*, 2006). They consist in carrying out the emulation of physically-based process models using neural network training starting from simulated data. The recall phase then supplies predicted output values in real-time (Krasnopolsky and Fox-Rabinovitz, 2006). We have already tested this method in the case of a dynamic system characterized by an ordinary differential equation (Bourgois *et al.*, 2007).

2. INVERSE NEURAL MODELING

2.1 Principle

Some ideas for forward and inverse models training in physical measurement applications have been proposed by (Krasnopolsky and Schillerb, 2003). Learning phase consists in weights estimation by backpropagation. The coefficients are then adjusted to move the network outputs closer to the desired inputs (figure 1).

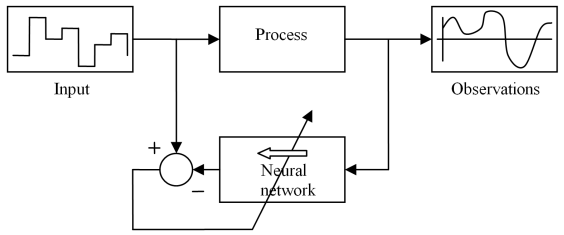

Fig. 1. Training phase of the inverse neural model.

In recall phase, the network estimates the inputs sequences, by supposing that the real model does not evolve any more after the last training (figure 2).

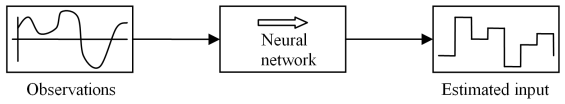

Fig. 2. Recall phase of the inverse neural model.

2.2 Regularization

Can we pose the regularization problem in the case of the NNIPS ? Let us underline that a neural network always provide an output, regardless of the appropriateness of the input, due to its autoassociative memory property. That answers the main difficulties of ill posed inverse problems (Groetsch, 1993), even if the suggested solution can prove to be false. In addition, regularization during training phase improves generalization with respect to the set of examples. It avoids the problem of overtraining which results in an instability. This confirms our opinion to use the neural network like an inverse model.

2.3 Representation

The inverse neural model can be described by the canonical form (1), where φ_I^{NN} corresponds to the reverse-time transition function and Ψ_I^{NN} represents the restoring function of the input.

$$\begin{cases} x(n) &= \varphi_I^{NN}[x(n+1), y_{obs}(n)] \\ u(n) &= \psi_I^{NN}[x(n+1), y_{obs}(n)] \end{cases} \tag{1}$$

The figure 3 represents the graphical form.

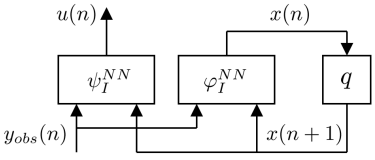

Fig. 3. Inverse neural state space model.

3. STUDY OF A DISPERSION MODEL

We have tested the previous method on an atmospheric pollutant dispersion model governed by a PDE in order to fulfill the pollution sources deconvolution and the receptors concentrations estimation.

3.1 Atmospheric Pollutant Dispersion Modeling

Let us suppose a system represented by the following partial differential equation (Turner, 1994) :

$$\begin{aligned} \frac{\partial x(\overrightarrow{p},t)}{\partial t} &= D(\overrightarrow{p},t)\left(\frac{\partial^2 x(\overrightarrow{p},t)}{\partial \overrightarrow{p}^2}\right) - \overrightarrow{V}(\overrightarrow{p},t)\left(\frac{\partial x(\overrightarrow{p},t)}{\partial \overrightarrow{p}}\right) \\ &- Kx(\overrightarrow{p},t) + \Gamma(x(\overrightarrow{p},t)) + \sum_{i=1}^{n_s} u(s_i,t)\delta(\overrightarrow{p}-\overrightarrow{s_i}) \end{aligned} \tag{2}$$

- $x(\overrightarrow{p},t)$ is the concentration (in $g.m^{-3}$) observed at a receptor location $\overrightarrow{p} = (p_1, p_2, p_3)$ at time t. It comes from the air dispersion of n_s pollutant sources of intensity $u(s_i,t)$ at the position $\overrightarrow{s_i} = (s_{(i,1)}, s_{(i,2)}, s_{(i,3)})$, inside a bounded open domain Ω of dimension $l \times L \times H$;
- D is the diffusion tensor (in $m^2.s^{-1}$) essentially defined by its diagonal elements $d_i(\overrightarrow{p},t)$;
- $\overrightarrow{V}(\overrightarrow{p},t) = (v_1(\overrightarrow{p},t), v_2(\overrightarrow{p},t), v_3(\overrightarrow{p},t))^T$ is the wind speed field (in $m.s^{-1}$), responsible for the $3D$ transport;
- K is the reaction coefficient of a first order chemical transformation;
- $\Gamma(x)$ appears when the chemical species presents nonlinear reactions;
- δ represents the Kronecker symbol.

The observatory is configured by a network of n_c sensors at the positions $\overrightarrow{c_i} = (c_{(i,1)}, c_{(i,2)}, c_{(i,3)})$. To simplify the presentation, we have chosen to present the method in the $1D$ case. For a $3D$ discrete representation, the reader can refer to (Roussel *et al.*, 2005). By choosing the explicit Euler method and supposing the sampling period T such as $t = nT$ and the spatial sampling step Δp_1 such as $p = k\Delta p_1$, we have discretized (2) and have obtained the recurrent equation (3).

$$\begin{aligned} x(k,n+1) &= m_1^{k,n} x(k+1,n) + m_2^{k,n} x(k,n) + m_3^{k,n} x(k-1,n) \\ &+ T\Gamma(x(k,n)) + T\sum_{i=1}^{n_s} u(s_i,n)\delta(k - s_{(i,1)}) \end{aligned} \tag{3}$$

With:

$$\begin{cases} m_1^{k,n} &= \frac{Td_1(k,n)}{(\Delta p_1)^2} - \left(\frac{1 - sgn(v_1(k,n))}{2}\right)\left(\frac{Tv_1(k,n)}{\Delta p_1}\right) \\ m_2^{k,n} &= 1 - KT - sgn(v_1(k,n))\left(\frac{Tv_1(k,n)}{\Delta p_1}\right) - \frac{2Td_1(k,n)}{(\Delta p_1)^2} \\ m_3^{k,n} &= \frac{Td_1(k,n)}{(\Delta p_1)^2} + \left(\frac{1 + sgn(v_1(k,n))}{2}\right)\frac{Tv_1(k,n)}{\Delta p_1} \end{cases} \tag{4}$$

Here, *sgn* defines the signum function. The equation (3) characterizes the deconvolution mask and presents

a linear part according to the coefficients $m_1^{k,n}$, $m_2^{k,n}$ and $m_3^{k,n}$. By supposing $M = E\left(\frac{l}{\Delta p_1}\right)+1$ meshes on one dimension, $x(n) = [x(1,n),\cdots,x(M,n)]^T$ and $u(n) = [u(s_1,n),\cdots,u(s_{n_s},n)]^T$, we have obtained the direct state space equation (5).

$$x(n+1) \;=\; Fx(n)+Gu(n)+T\Gamma(x(n)) \tag{5}$$

The matrix F, of size $dim(F) = M \times M$, tridiagonal, takes the form (6).

$$F \;=\; \begin{bmatrix} m_2^{1,n} & m_1^{1,n} & 0 & \cdots & 0 \\ m_3^{2,n} & m_2^{2,n} & m_1^{2,n} & & \vdots \\ 0 & & \ddots & & 0 \\ \vdots & & & & m_1^{M-1,n} \\ 0 & \cdots & 0 & m_3^{M,n} & m_2^{M,n} \end{bmatrix} \tag{6}$$

The matrix G, of size $dim(G) = M \times n_s$, is worth (7).

$$G \;=\; T\begin{bmatrix} \delta(1-s_{(1,1)}) & \delta(1-s_{(2,1)}) & \cdots & \delta(1-s_{(n_s,1)}) \\ \delta(2-s_{(1,1)}) & \delta(2-s_{(2,1)}) & & \\ \vdots & & \ddots & \\ \delta(M-s_{(1,1)}) & & & \delta(M-s_{(n_s,1)}) \end{bmatrix} \tag{7}$$

By setting $y(n) = [y(1,n),\cdots,y(n_c,n)]^T$ and $b(n) = [b(1,n),\cdots,b(n_c,n)]^T$, the equation characterizing the observations is given by:

$$y(n) \;=\; Hx(n)+b(n) \tag{8}$$

The placing matrix H of the n_c sensors, of size $dim(H) = n_c \times M$, is expressed by (9).

$$H \;=\; \begin{bmatrix} \delta(1-c_{(1,1)}) & \delta(2-c_{(1,1)}) & \cdots & \delta(M-c_{(1,1)}) \\ \delta(1-c_{(2,1)}) & \delta(2-c_{(2,1)}) & & \\ \vdots & & \ddots & \\ \delta(1-c_{(n_c,1)}) & & & \delta(M-c_{(n_c,1)}) \end{bmatrix} \tag{9}$$

The term $b(i,n) = b_{mod}(i,n)+b_{mes}(i,n)$ is a random vector, Gaussian centered $b(i,n) \sim \mathcal{N}(0,\sigma^2)$, of unknown variance σ^2, modeling the general uncertainty of the observations. It groups together model errors $b_{mod}(i,n)$ (phenomenon and wind fields uncertainty) and measurement uncertainty $b_{mes}(i,n)$ resulting from sensors or measurement environment.

3.2 Study Assumptions

We have considered a basic mesh to reproduce, constituted by three nodes or neurons. We have supposed there is only one source of flow $u(n)$ in this mesh, at the level of the central node. A sensor is positioned at the level of a lateral node. Wind speed is supposed to be constant in time, and the term of nonlinearity $\Gamma(y)$ is considered to be insignificant. This choice has been done in order to confirm the method in a linear case. The impact of nonlinearity will be mention further. For this basic mesh, we have obtained the direct state space equations system (10).

$$\begin{cases} x(n+1) \;=\; Fx(n)+Gu(n) \\ y(n) \;\;\;=\; Hx(n)+b(n) \end{cases} \tag{10}$$

With:

$$\begin{cases} F \;\;=\; \begin{bmatrix} m_2^{1,n} & m_1^{1,n} & 0 \\ m_3^{2,n} & m_2^{2,n} & m_1^{2,n} \\ 0 & m_3^{3,n} & m_2^{3,n} \end{bmatrix} \\ G^T \;=\; \begin{bmatrix} 0 & T & 0 \end{bmatrix},\;\; H \;=\; \begin{bmatrix} 0 & 0 & 1 \end{bmatrix} \end{cases} \tag{11}$$

The reverse-time equations design has consisted in the expression of the flow $u(n)$ according to the sensor observation. Then, the state variables at time n have been extracted to obtain a new system, according to the state variables at time $n+1$. We have thus carried out the reverse-time state space equations system (12) which corresponds to the canonical form (1).

$$\begin{cases} x(n) \;=\; F_I x(n+1)+G_I[y(n)-b(n)] \\ u(n) \;=\; H_I x(n+1)+I_I[y(n)-b(n)] \end{cases} \tag{12}$$

With:

$$\begin{cases} F_I \;=\; \begin{bmatrix} \frac{1}{m_2^{1,n}} & 0 & -\frac{m_1^{1,n}}{m_2^{1,n}m_3^{3,n}} \\ 0 & 0 & \frac{1}{m_3^{3,n}} \\ 0 & 0 & 0 \end{bmatrix},\;\; H_I \;=\; \begin{bmatrix} -\frac{m_3^{2,n}}{Tm_2^{1,n}} & \frac{1}{T} & \zeta \end{bmatrix} \\ G_I^T \;=\; \begin{bmatrix} \frac{m_1^{1,n}m_2^{3,n}}{m_2^{1,n}m_3^{3,n}} & -\frac{m_2^{3,n}}{m_3^{3,n}} & 1 \end{bmatrix},\quad I_I \;=\; \frac{\eta+\kappa-\nu}{Tm_2^{1,n}m_3^{3,n}} \end{cases} \tag{13}$$

Where the coefficients in H_I and I_I are expressed by:

$$\begin{cases} \zeta \;=\; \frac{m_1^{1,n}m_3^{2,n}-m_2^{2,n}m_2^{1,n}}{Tm_2^{1,n}m_3^{3,n}},\;\; \kappa \;=\; m_1^{2,n}m_2^{1,n}m_3^{3,n} \\ \eta \;=\; m_2^{1,n}m_2^{2,n}m_2^{3,n},\qquad\quad \nu \;=\; m_1^{1,n}m_2^{3,n}m_3^{2,n} \end{cases} \tag{14}$$

The inverse neural model (figure 4) is carried out starting from (12). But, even if previous results provide accurate coefficients, we do not need them to design the shape of the inverse neural model. One only needs to know the structure, *i.e.* the location of non-zero values. Indeed, the non-zero coefficients in (13) define the remaining connections symbolized by arrows in figure 4. The corresponding weights (degrees of

freedom) are then estimated during the training. Here, the activation functions f are linear. However, neural networks have been successfully used to nonlinear dynamic systems modeling. Indeed, the form of the usual nonlinear activation functions (*e.g.* sigmoid activation functions) results in more parsimonious approximations (Barron, 1993). We have supposed that the results obtained in the linear case may reasonably be kept in the nonlinear case with the same advantages.

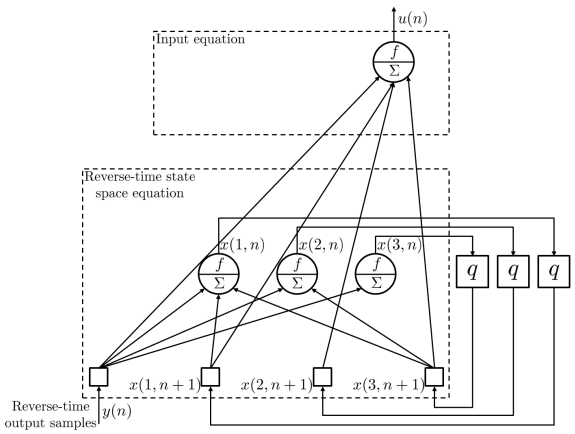

Fig. 4. Inverse neural model representation.

3.3 Study of Causality and Stability

The problem of causality have been raised at two levels :

- During the error calculation associated with each training exemple and during the recall phase, we have truncated all the sequences by deleting the $r-1$ first samples because of the unknown initial conditions (r being the system order);
- During numerical simulations, the simulated data have been rearranged before the training to obtain reverse-time sequences (the first element has become the last one, *etc*). The q operator has assumed the role of q^{-1} operator which stands for one T sample time delay to ensure causality is not violated.

This study have led us to treat stability conditions in two times :

- During the training phase, data are simulated starting from the direct state space model. It has been necessary to check the stability of the simulation model. The stability is ensured if and only if the spectral radius $\rho(F) \leqslant 1$;
- Since the inverse neural model is designed starting from the inverse state space model, it has been advisable to know the behavior of the last one in term of stability. The stability is ensured if and only if $\rho(F_I) \leqslant 1$.

However, the matrices F and F_I being essentially composed of fixed physical coefficients, the only adjustable parameter is the sampling period T. Thus, for invariant simulation parameters, we have studied the spectral radius evolution of the matrices F and F_I according to T (figure 5).

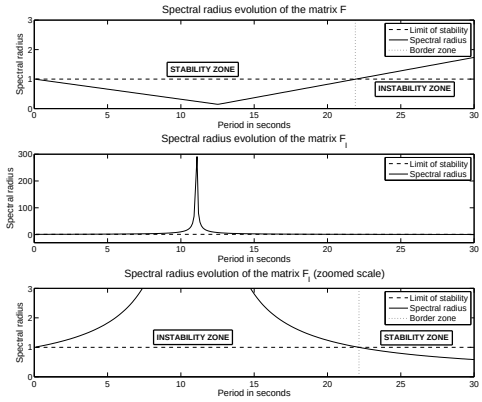

Fig. 5. Spectral radius evolution according to T.

The inverse state space model stability zone is totally unconnected with the direct state space model one. In order to keep the stability of numerical schemes, we have considered two distinct sampling periods T_{direct} and $T_{inverse}$. But, this resampling have involved :

- Restoration errors of the input signal;
- Natural regularization (indeed, the spectral radius $\rho(F_I)$ diminishes when T grows).

Of course, it is not conceivable to validate the study since results have been biased. Consequently, the resampling has been rejected and we have chosen a single sampling period T such as $\rho(F) \leqslant 1$ to ensure the simulation model stability. Nevertheless, this choice is unfavorable to the inverse state space model stability.

4. RESULTS

The goal of this section is to check the assumptions of awaited quality concerning the gray-box NNIPS :

- In term of robustness with respect to an unknown input from the training base;
- In term of robustness with respect to the noise on the output (*i.e.* the regularizing effect);
- In term of gain about the training effort.

The semi-physical NNIPS has been compared to a traditional black-box inverse neural model. The black-box NNIPS is an Elman network constituted by three linear neurons on its hidden layer and one linear neuron on its output layer. The network is fully connected. After being initialized by the Nguyen-Widrow method (Nguyen and Widrow, 1990), all the synaptic weights and biases are left free during the whole training. The gray-box NNIPS is designed starting from the previous black-box model and modified to obtain the inverse neural structure of the figure 4. For that, we have connected the inputs layer to the output layer, and

six weights have been forced to be null to delete corresponding connections. No neuron has been added. The remaining coefficients are left free during the whole training. The two NNIPS models have been subjected to a learning with pseudo-experimental noisy data.

For numerical simulations, we have chosen a spatial sampling step $\Delta p_1 = 100\ m$, a wind speed field such as $v_1(1,n) = 9\ m.s^{-1}$, $v_1(2,n) = 7\ m.s^{-1}$ and $v_1(3,n) = 8\ m.s^{-1}$, a diffusion tensor such as $d_1(1,n) = d_1(2,n) = d_1(3,n) = 2\ m^2.s^{-1}$, and a chemical reaction coefficient $K = 0$. For the reasons previously exposed, we have set a sampling period $T = 12.5\ s$, ensuring the simulation model stability.

To construct the set of training, we have generated a N samples input random sequence in order to simulate the direct knowledge-based model. This signal is a step function, resulting from the product of an amplitude level A_e by a Gaussian law of average μ_e and variance σ_e^2. The period T_e is adjustable and characterizes the change of state. Moreover, T_e influences the input signal dynamic, and thus the spectrum of the system excitation random signal. From this input signal, we have obtained a noisy synthetic output signal. The average μ_b, the variance σ_b^2, and the period T_b characterize the noise dynamic. For all the tests, we have fixed $A_e = 1$, $\mu_e = 0$, $\sigma_e^2 = 1$, $T_e = 60T$, $\mu_b = 0$ and $T_b = 3T$.

Sometimes, coefficients initial values may be deterministic for the quality of the results. Indeed, the backpropagation algorithm may converge to unsatisfactory local minima, and may not be able to find weights that minimize the error during the training phase. This may cause unstable network outputs and extremely high mean squared errors (MSE). Consequently, we have chosen to repeat each following test hundred times, and to exclude these kinds of results before calculating the average performances of the two NNIPS.

4.1 Modeling Errors and Regularizing Effect

We have measured the semi-physical contribution in terms of generalization and regularization. For that, we have simulated five distinct training sequences of length $N = 300$ samples. The noise variances σ_b^2 of the pseudo-experimental signals are worth 0, 0.96, 2.88, 8, and 32. The signal-to-noise ratio (SNR) of the corresponding synthetic output signals lies between 20 dB and plus infinity. After the training phase, we have generated five new noisy random signals with the same variances (test sets) and have compared the MSE obtained during the recall phases of the two NNIPS.

Let us underline that only thirty-seven experiments have been retained for the average performances calculation. Indeed, the traditional black-box inverse neural model have not supplied suitable restoration in 63% of the cases, *i.e.* the backpropagation algorithm have converged to unsatisfactory local minima.

The figure 6 shows the estimated input signals obtained with a SNR of 34 dB. The figure 7 gathers the average MSE of the two inverse neural models according to the SNR.

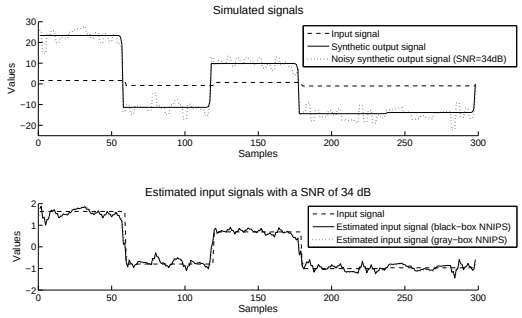

Fig. 6. a) Simulated signals, b) Estimated input signals with a SNR of 34 dB.

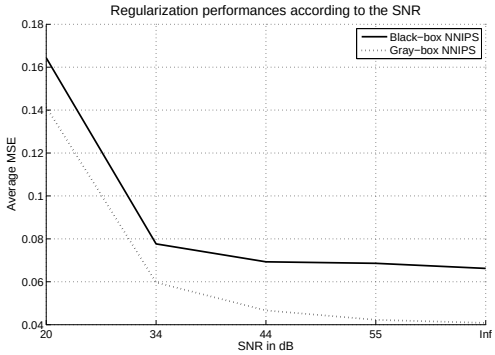

Fig. 7. Average MSE of the two NNIPS according to the SNR.

Without noise in training sequences and test sets, the semi-physical NNIPS provides the best performances (MSE $\simeq 0.040$). The black-box inverse neural model is slightly less effective (MSE $\simeq 0.065$). When the noise grows, the two inverse neural models are moderately sensitive and keep the same tendencies (figure 7). In addition, having chosen a sampling period T such as $\rho(F_I) > 1$ do not interfere with the NNIPS. The regularizing effect is real.

In a high noise situation, the gap between the two NNIPS tends to reduce. Indeed, the constraint imposed by the structure of the network and the more reduced connections number decrease the robust effect to the noise (loss of the neural network autoassociative memory property).

4.2 Learning Effort

We have compared the product of the MSE by the number of epochs, *i.e.* the final error amplified by the iteration count of the training phase. The learning stops if the MSE is lower than 0.005 or if the number of iterations reaches 250. Moreover, we have made a distinction between the errors obtained at the end

of the training phases (MSE_1) and the test sets errors (MSE_2).

On the figure 8, we note that the gray-box NNIPS is more effective with slight noise. Physical knowledge favors the convergence of the weights so that the behavior approaches the data. This seems to be true until a SNR of about 34 dB. Beyond that, performances are equal.

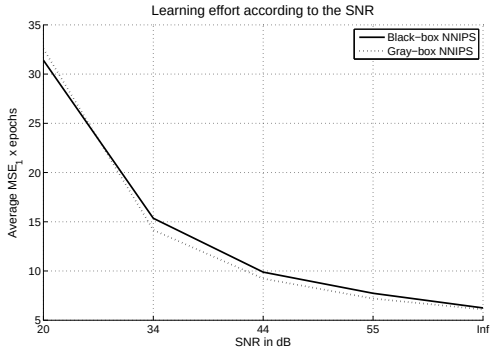

Fig. 8. Learning effort with the errors obtained at the end of the training phases.

On the figure 9, results are similar for the semi-physical NNIPS. The black-box inverse neural model is largely penalized with slight noise test sets, because of its lesser capacity of regularization.

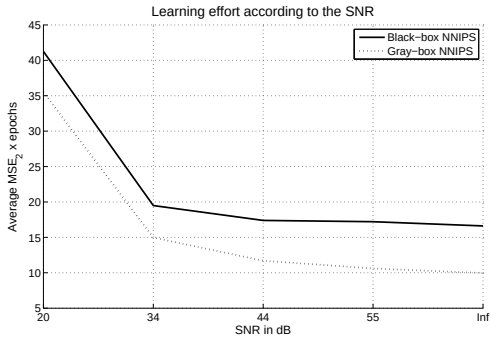

Fig. 9. Learning effort with the test sets errors.

5. CONCLUSION

We have examined the performances of an inverse model resulting from the fusion of deterministic modeling and statistical learning. We have chosen to carry out this inverse semi-physical model starting from a recurrent neural network to exploit typical properties of neural algorithms. Indeed, experimental results have shown that neural learning plays the part of statistical regressor and regularization operator. Moreover, input restoration errors are weak. In order to evaluate the semi-physical contribution, the gray-box NNIPS has been compared to a traditional black-box inverse neural model. The test realized on a basic mesh of an atmospheric pollutant dispersion model have reveal that the semi-physical inverse model is more parsimonious than the black-box NNIPS. Besides, gray-box modeling provides better performances in term of training effort than black-box modeling, due to the knowledge introduced by the deterministic model. Nevertheless, this work is only a first step before the realization of a more general inverse problem which aims at carrying out the sources restoration on a monitored area starting from pollutant concentration measurements at some other sites.

REFERENCES

Barron, A. (1993). Universal approximation bounds for superposition of a sigmoidal function. *IEEE Transactions on Information Theory* **39**, 930–945.

Bourgois, L., G. Roussel and M. Benjelloun (2007). Inversion of a semi-physical ode model. *IFAC, Proceedings of the International Conference on Informatics in Control, Automation and Robotics*.

Cherkassky, V., V. M. Krasnopolsky, D. Solomatine and J. Valdes (2006). Computational intelligence in earth sciences and environmental applications: Issues and challenges. *Neural Networks* **19, issue 2**, 113–121.

Groetsch, C. W. (1993). *Inverse Problems in the Mathematical Sciences*. Vieweg Sohn. Wiesbaden.

Krasnopolsky, V. M. and H. Schillerb (2003). Some neural network applications in environmental sciences. part i: Forward and inverse problems in geophysical remote measurements. *Neural Networks* **16**, 321–334.

Krasnopolsky, V. M. and S. F. Fox-Rabinovitz (2006). Complex hybrid models combining deterministic and machine learning components for numerical climate modeling and weather prediction. *Neural Networks* **19**, 122–134.

Nguyen, D. and B. Widrow (1990). Improving the learning speed of 2-layer neural networks by choosing initial values of the adaptive weigths. *Proceedings of the International Joint Conference on Neural Networks* **3**, 21–26.

Oussar, Y. and G. Dreyfus (2001). How to be a gray box : Dynamic semi-physical modeling. *Neurocomputing* **14**, 1161–1172.

Roussel, G., G. Delmaire and D. Hamad (2005). Détection et estimation de sources d'un pic de pollution par un réseau de capteurs. *RS-JESA* **39**, 455–474.

Turner, D. B. (1994). *Workbook of Atmospheric Dispersion Estimates : an Introduction to Dispersion Modeling*. CRC Press, 2nd edition. Boca Raton.