Semi-physical neural modeling for linear signal restoration
Laurent Bourgois, Gilles Roussel, Mohammed Benjelloun

To cite this version:

HAL Id: hal-01421228
https://hal.archives-ouvertes.fr/hal-01421228
Submitted on 21 Dec 2016

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L’archive ouverte pluridisciplinaire HAL, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d’enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.
Semi-physical neural modeling for linear signal restoration

Laurent Bourgois*, Gilles Roussel**, Mohammed Benjelloun

Lille Nord de France, F-59000 Lille, France
ULCO, LISIC, F-62228, Calais
Université du Littoral - Côte d’Opale
50 rue Ferdinand Buisson, B.P. 699, F-62228, Calais Cedex, France

Abstract

This paper deals with the design methodology of a neural network for inverse modeling. We examine the performances of an inverse dynamic model resulting from the multi-model fusion of statistical learning and deterministic modelling. This inverse semi-physical or gray-box model is carried out using a recurrent neural network. The last one design is structured from a naive discrete reverse-time state-space form. In order to validate the approach, some tests are performed on two dynamic models. The first suggested model is a dynamic system characterized by an unspecified r-order Ordinary Differential Equation (ODE). The second one concerns in particular a mass balance equation for dispersion phenomenon governed by a Partial Differential Equation (PDE) discretized on a basic mesh. The performances are numerically analysed in terms of generalization, regularization and training effort.

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

1. Introduction

Many applications require data inversion. Inverse problems or signals restoration are solved by the inversion of a direct representation modeling the real system, with...
some different techniques: variational method, optimization of a criterion, inverse filter, analytical solution from direct model, etc.. All these methods use a mathematical description of the real system. According to how much a priori information is available, it is possible to contract rather a knowledge-based (white-box) model based on the physical, chemical, biological or sociological principles, or an empirical model (black-box) model based on an a priori well fitting analytical function and a data identification procedure. Of course, the quality of the restoration by data inversion depends on the observation noise, on the model accuracy, and on the inversion method. However, it is difficult to find an analytical solution when the system is quite complex, often non-linear and time varying. Modeling imprecise or complex system can be considered in combining knowledge on the physical system and the data measured during system operation. Such a model is named semi-physical or grey box concept. Although, this approach is often reserved for the direct modeling. The following idea is to construct a semi-physical inverse model merging physical knowledge of an inverse relaxed mechanistic model and data accumulated during a statistical learning phase. Thus, a robust Neural Network Inverse Model (NNIM) is ensured using a priori knowledge on the physical laws which govern the system. With the NNIM, we think to propose a technique having a faculty of learning and adaptability but also having good efficiency relative to inverse problem difficulties. In order to test the method, we have studied the deconvolution problem by examining linear models defined by an ODE, or a spatio-temporal model governed by a PDE.

Establishing a robust white-box model within the meaning of exhaustiveness compared to the variations of context is often tricky to express for several reasons. One needs a perfect expertise to enumerate all the physical laws and influential variables brought into play. Besides, an exhaustive spatial and temporal system description is also required. However, even if the previous stage is completed, some parameters may not be measured or precisely known. It is then advisable to estimate these parameters starting from observable data. Once the physical model has been fixed, it is endowed with a good robustness.

A black-box model is a behavior model very suited for complex system representation (Sjoberg et al., 1995), but does not take any prior information into account. Many
standard process forms link system’s outputs (or states) to inputs from experimental data: e.g. ARMA, ARMAX [Ljung, 1999], NARMAX, Box-Jenkins [Box et al., 1994], NOE (Nonlinear Output Error), etc. can be considered as black-box models. Another approach based on classic neural networks do not specify a mathematical form but rather a neural design which best fits with the system dynamics. One of the main advantages of neural networks is their great adaptability to static, dynamic, linear or not functions, thanks to the universal approximation property [Sontag, 1997]. Moreover, neural networks have been successfully used to nonlinear dynamic systems modeling. The form of usual nonlinear activation functions (e.g. sigmoid activation functions) results in parsimonious estimation (least error with minimum parameters) [Barron, 1993]. Nevertheless, black-box models are often less parsimonious than knowledge-based ones. Indeed, the mathematical functions used to describe white-box models are more accurate and minimize output errors without noise.

Between the two models previously exposed, grey box model is a tool emerged in the 1995s. The approach termed as grey-box modeling [Duarte et al., 2004], [Beghi et al., 2007], as hybrid modeling [Zorzetto et al., 2000] or semi physical modeling [Lindskog and Ljung, 1995] can be found in the literature. In [Lei, 2008], authors distinguish two categories of approaches called serial and parallel modeling. These two patterns of grey-box are different by the manner to combine black-box and white-box. Serial grey box [Nelles, 2001] makes a numerical separation between the known and unknown physical part of the system. Parallel grey box introduces a kind of competition between black-box and white-box. Mainly, the black-box corrects the predicted outputs of the white-box model. Between these two ways, another way is closer to the notion of models fusion. For example, [Oussar and Dreyfus, 2001], [Ploix and Dreyfus, 1997] have introduced a semi-physical direct model by modifying the design of a recurrent neural network. The idea is to design a recurrent neural network using engineers’ knowledge on the fundamental laws which govern the system. In this case, a priori information is based on the network structure. One or more degrees of freedom (e.g. additional neurons) may also be added to help the network successfully adapt to the ignored parts of the system [Oussar and Dreyfus, 2001]. Measurements on process are then used to learn the network. The recall phase then supplies predicted output values in real-time.
Other approaches have been proposed by (Krasnopolsky and Fox-Rabinovitz, 2006). They consist in carrying out the emulation of physically-based process models using neural network training starting from white-box model simulations. Semi-physical or gray-box modeling has often been used in the case of direct models. This type of model fulfills at the same time precision requirements, robustness and parsimony of the knowledge-based models, and also possesses the faculty of training and adaptability. Our idea is to inspire by such a concept for inverse problem.

2. Inverse neural modeling

2.1. Principle

The objective of many applications such as inverse problem in meteorology, tomography, software sensor, deconvolution 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. There are several numerical ways to deal with this problem such as state-space transformations (e.g. Laplace, Fourier, etc.), direct state-space model discretization followed by a matrix inversion, or the definition of a performance function to minimize (Groetsch, 1993), (Tarantola, 1987).

Our proposed additional objective is to realize the inverse model training. Some ideas for forward and inverse models training in physical measurement applications have been proposed by (Krasnopolosky and Schiller, 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).

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). Implicitly, this method looks like the error propagation through the adjoint network.

2.2. Regularization

Inverse problems are often ill-posed in the Hadamard sense (Groetsch, 1993). They can present an absence of solution, multiple solutions, or an unstable solution. To
transform ill-posed problems into well-conditioned ones, it is necessary to add a priori knowledge on the system before inversion. There are two approaches which differ according to the type of a priori knowledge introduced. The first procedure employs regularization methods based on deterministic information (Thikhonov and Arsenin, 1977). The second strategy considers techniques based on probabilistic information such as Bayesian methods (Marroquin et al., 1987), (Demoment, 1989) or maximum entropy methods (Mohammad-Djafari et al., 2002).

But, can we discuss the regularization problem in the case of the NNIM? Let us underline that a neural network always provides an output, regardless of the appropriateness of the input, due to its autoassociative memory property. That answers the two main difficulties of ill-posed inverse problems, 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 over-learning which results in an instability. It is also remarkable that early stopping procedure, i.e. stopping the
gradient descent before learning process reaches the optimal solution on the training set, supplies solutions with smaller generalization error. Besides, some Bayesian techniques have been developed to adjust the regularization coefficients of the performance function (MacKay, 1992). This confirms our opinion to use the neural network like an inverse model.

3. Design of an inverse semi-physical neural model

The construction of a gray-box neural direct model is generally performed in three steps:

**Step 1**: Discrete-time neural network design derived from the knowledge-based model;

**Step 2**: Training of the semi-physical neural model from knowledge-based simulations in order to obtain appropriate initial values;

**Step 3**: Training of the semi-physical neural model from experimental data.

The knowledge-based model is usually represented in the form of a set of coupled, differential, partial differential, algebraic and sometimes nonlinear equations. The starting model can be described by the standard state-space form:

\[
\begin{align*}
\frac{dx}{dt} &= f[x(t), u(t)] \\
y(t) &= g[x(t)] + b(t)
\end{align*}
\]  

(1)

Where \( x \) is the vector of state variables, \( y \) is the vector of outputs, \( u \) is the vector of control inputs and \( b \) corresponds to the noise. The vector functions \( f \) and \( g \) are known, but they may also be partially known or inaccurate. In black-box neural modeling, functions \( f \) and \( g \) are approximated during the training step from experimental data. In gray-box neural modeling, those functions are described by their analytical form and implemented as neural models with some fixed parameters. Other unknown parameters are computed during the training step from experimental data.

The discretized equations of the neural model can be written under the canonical form \( (2) \), where \( \phi^{NN} \) corresponds to the transition vector function, \( \Psi^{NN} \) represents the
output vector function and $b(n)$ is the output noise at time instant $n$. Since the output noise only appears in the observation equation, it does not have any influence on system dynamics.

\[
\begin{align*}
  x(n+1) &= \varphi^{NN}[x(n), u(n)] \\
  y(n) &= \psi^{NN}[x(n)] + b(n)
\end{align*}
\]

(2)

Figure 3 represents the graphical form of the neural state-space model.

Figure 3: Neural state-space model. The $q^{-1}$ operator stands for one $T$ sample time delay.

Similarly, we have carried out the inverse semi-physical neural model by adding an inversion step before the training. The reverse-time equations design has consisted in the expression of $u(n)$ according to the noisy observation $y_{obs}(n)$. Then, the state variables at time instant $n$ have been extracted to obtain a new system, according to the state variables at time instant $n + 1$.

Consequently, the inverse neural model can be described by the canonical form (3), where $\varphi^{NN}_I$ corresponds to the reverse-time transition vector function and $\psi^{NN}_I$ represents the restoring vector function of the input.

\[
\begin{align*}
  x(n) &= \varphi^{NN}_I[x(n+1), y_{obs}(n)] \\
  u(n) &= \psi^{NN}_I[x(n+1), y_{obs}(n)]
\end{align*}
\]

(3)

Figure 4 represents the graphical form of the inverse neural state-space model.
4. Inversion of a semi-physical ODE model

In the first part of this section, we obtain the canonical form of the inverse model which refers to (3) in the case of a dynamic system characterized by a $r$-order ODE. In the second part, an illustrative example we will present a study concerning a second order example. Some promising results about semi-physical ODE models have already been developed by (Bourgois et al., 2007b).

4.1. General case study: an $r$-order ODE without input derivative

Let us consider a continuous, mono input and mono output system governed by an ordinary differential equation:

$$
a_r \frac{d^r y}{dt^r} + a_{r-1} \frac{d^{r-1} y}{dt^{r-1}} + \cdots + a_1 \frac{dy}{dt} + a_0 y = c_1 u(t) \tag{4}
$$

The corresponding continuous state-space form is:

$$
\begin{align*}
\frac{dx(t)}{dt} &= Ax(t) + Bu(t) \\
y(t) &= Cx(t) + b(t)
\end{align*} \tag{5}
$$

State-space matrices $A$, $B$ and $C$ are worth:

$$
A = \text{Comp}(P), \quad B^T = \begin{bmatrix} 0 & \cdots & 0 & \frac{c_1}{a_r} \end{bmatrix}, \quad C = \begin{bmatrix} 1 & 0 & \cdots & 0 \end{bmatrix}
$$

Here, $\text{Comp}(P)$ is the companion matrix of the monic polynomial obtained starting from (3) and defined by $P(q) = \frac{a_0}{a_r} + \frac{a_1}{a_r} q + \cdots + \frac{a_{r-1}}{a_r} q^{r-1} + q^r$. By choosing
the explicit Euler method and supposing the sampling period \( T \) such as \( t = nT \), the
discretized state-space form of (5) leads to (6):

\[
\begin{align*}
\frac{x(n+1) - x(n)}{T} &= Ax(n) + Bu(n) \\
y(n) &= Cx(n) + b(n)
\end{align*}
\]

(6)

The new state-space matrices are expressed by

\[
F = TA + I_r, \quad G = TB, \quad H = C
\]

Here, \( I_r \) is the identity matrix with \( \dim(I_r) = \dim(F) = r \times r \), \( \dim(G) = r \times 1 \) and \( \dim(H) = 1 \times r \).

By referring to the demonstration [Appendix A] of the appendix, we have carried out the reverse-time state-space equations system (7) which fits to the canonical form (3). Results equations (A.10) and (A.12) of the appendix are remember as following :

\[
\begin{align*}
x(n) &= FIx(n+1) + G_I[y(n) - b(n)] \\
u(n) &= HIx(n+1) + I_I[y(n) - b(n)]
\end{align*}
\]

(7)

Where the reverse-time state-space matrices are worth :

\[
F_I = \begin{bmatrix}
0 & 0 & \cdots & 0 \\
\frac{1}{T} & 0 & \cdots & 0 \\
\vdots & \ddots & \ddots & \vdots \\
-\left(\frac{1}{T}\right)^{r-1} & \frac{1}{T} & \cdots & 0
\end{bmatrix}, \quad G_I = \begin{bmatrix}
1 \\
-1 \\
\vdots \\
\left(-\frac{1}{T}\right)^{r-1}
\end{bmatrix}
\]

\[
H_I = \begin{bmatrix}
0 & \cdots & 0 & \frac{a_r}{Tc_1} \\
\frac{a_0}{c_1} & \cdots & \frac{a_{r-2}}{c_1} & \frac{1}{Tc_1} (a_{r-1}T - a_r)
\end{bmatrix}F_I + \begin{bmatrix}
\frac{a_0}{c_1} & \cdots & \frac{a_{r-2}}{c_1} & \frac{1}{Tc_1} (a_{r-1}T - a_r)
\end{bmatrix}G_I
\]

\[I_I = \begin{bmatrix}
\frac{a_0}{c_1} & \cdots & \frac{a_{r-2}}{c_1} & \frac{1}{Tc_1} (a_{r-1}T - a_r)
\end{bmatrix}F_I + \begin{bmatrix}
\frac{a_0}{c_1} & \cdots & \frac{a_{r-2}}{c_1} & \frac{1}{Tc_1} (a_{r-1}T - a_r)
\end{bmatrix}G_I
\]

4.2. Study of a second order ODE model

We have studied the deconvolution problem for linear models governed by an ordinary differential equation in order to test the method. Let us suppose a system represented by the differential equation:
\[
\frac{d^2y}{dt^2} + 2\zeta\omega_n \frac{dy}{dt} + \omega_n^2 y = c_1 u(t)
\]  

(8)

This second order ordinary differential equation may be either the representation of a mechanical system (e.g. mass, spring, shock absorber, etc.) or the representation of an electrical one (e.g. RLC filter) excited by a time-dependent input \( u(t) \). The damping parameter \( \zeta \), the natural pulsation \( \omega_n \), and the static gain \( c_1 \) are not \textit{a priori} known in this physical model. By referring to the relation (5), the model can be represented by the following state-space system :

\[
\begin{align*}
\frac{dx(t)}{dt} &= \begin{bmatrix} 0 & 1 \\ -\omega_n^2 & -2\zeta\omega_n \end{bmatrix} x(t) + \begin{bmatrix} 0 \\ c_1 \end{bmatrix} u(t) \\
y(t) &= \begin{bmatrix} 1 & 0 \end{bmatrix} x(t) + b(t)
\end{align*}
\]  

(9)

In discrete form, the state-space matrices \( F \), \( G \) and \( H \) of (6) are expressed by :

\[
F = \begin{bmatrix} 1 & T \\ -\omega_n^2 & 1 - 2\zeta\omega_n \end{bmatrix}, \quad G = \begin{bmatrix} 0 \\ Tc_1 \end{bmatrix}, \quad H = \begin{bmatrix} 1 & 0 \end{bmatrix}
\]

By referring to (7), we have finally obtained the inverse state-space model :

\[
\begin{align*}
x(n) &= \begin{bmatrix} 0 & 0 \\ \frac{1}{T} & 0 \end{bmatrix} x(n+1) + \begin{bmatrix} 1 \\ -\frac{1}{T} \end{bmatrix} [y(n) - b(n)] \\
u(n) &= \begin{bmatrix} \alpha & \beta \end{bmatrix} x(n+1) + \gamma [y(n) - b(n)]
\end{align*}
\]  

(10)

Where the parameters \( \alpha \), \( \beta \) and \( \gamma \) are worth :

\[
\alpha = \frac{2\zeta\omega_n T - 1}{T^2 c_1}, \quad \beta = \frac{1}{Tc_1}, \quad \gamma = \frac{(\omega_n T)^2 + (1 - 2\zeta\omega_n T)}{c_1 T^2}
\]

Of course, this non-causal system can be implemented only if the state variables at time instant \( n+1 \) are known before the calculation of state variables at time instant
Inverse problems are more familiar with this concept. It is the case during the input sequences restoration at the initial time instant. In the reconstructed input, the observation noise $b(n)$ now appears as a correlated noise relative and is also amplified by the real $\gamma$. Let us underline that the reverse-time system remains stable for any $T$ since the eigenvalues of the state-space matrix are all null for this example. The inverse neural model of figure 5 is carried out starting from (10). Here, the activation functions $f$ are all linear. Besides, even if the sampling period $T$ is generally known, the physical parameters $c_1$, $\xi$ and $\omega_n$ may be imprecise, or completely unknown. The degrees of freedom may relate to these coefficients.

Figure 5: Second order neural inverse model representation. The output of network is the input of the system. The design fits with the inverse state space equations. The neural network is not completely connected.

5. Study of a Dispersion Model

In this section we will test the previous method on an atmospheric pollutant dispersion model governed by a partial differential equation in order to fulfill the pollution sources deconvolution and the receptors concentrations estimation.
5.1. Atmospheric Pollutant Dispersion Modeling

In this section, we will develop and increase the results we have obtained in (Bourgois et al., 2007a). Let us suppose a system represented by the following PDE (Turner, 1994):

\[ \frac{\partial x(p_1,t)}{\partial t} = D(p_1,t) \left( \frac{\partial^2 x(p_1,t)}{\partial p^2} \right) - \nabla \cdot (v(p_1,t) x(p_1,t)) + k x(p_1,t) + \Gamma(x(p_1,t)) + \sum_{i=1}^{n_s} u(s_i,t) \delta(p - s_i) \]

(11)

- \( x(p_1,t) \) is the concentration (in \( g.m^{-3} \)) at a receptor location \( p = (p_1, p_2, p_3) \) at time \( t \) in the referential \( \{O, \hat{i}, \hat{j}, \hat{k}\} \). It comes from the air dispersion of \( n_s \) pollutant sources of intensity \( u(s_i,t) \) at the position \( s_i = (s_{i,1}, s_{i,2}, s_{i,3}) \), inside a bounded open domain \( \Omega \) of dimension \( l \times L \times H \);

- \( D \) is the diffusion tensor (in \( m^2.s^{-1} \)) defined by its diagonal elements \( d_i(p_1,t) \);

- \( v(p_1,t) = (v_1(p_1,t), v_2(p_1,t), v_3(p_1,t))^T \) 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;

- \( \delta \) represents the Dirac function.

The observatory is configured by a network of \( n_c \) sensors at the positions \( c_i = (c_{i,1}, c_{i,2}, c_{i,3}) \). To simplify the presentation, we have chosen to present the method in the one-dimensional case. By projecting on \( O \hat{i} \), choosing the explicit Euler method and supposing the sampling period \( T \) such as \( t = nT \) and the spatial sampling step \( \Delta p_1 \) such as \( p_1 = k\Delta p_1 \), we have obtained the recurrent equation (12).

\[
x(k,n+1) = m_1(k,n)x(k+1,n) + m_2(k,n)x(k,n) + m_3(k,n)x(k-1,n) + T \Gamma(x(k,n)) + T \sum_{i=1}^{n_s} u(s_i,n) \delta(k - s_{i,1})
\]

(12)

Where the parameters \( m_1(k,n), m_2(k,n) \) and \( m_3(k,n) \) are worth:
\[
\begin{align*}
m_1(k, n) &= \frac{T d_1(k, n)}{(\Delta p_1)^2} - \left(1 - \frac{\text{sgn}(v_1(k, n))}{2}\right) \left(\frac{T v_1(k, n)}{\Delta p_1}\right) \\
m_2(k, n) &= 1 - KT - sgn(v_1(k, n)) \left(\frac{T v_1(k, n)}{\Delta p_1}\right) - \frac{2Td_1(k, n)}{(\Delta p_1)^2} \\
m_3(k, n) &= \frac{T d_1(k, n)}{(\Delta p_1)^2} + \left(1 + sgn(v_1(k, n))\right) \left(\frac{T v_1(k, n)}{\Delta p_1}\right)
\end{align*}
\]

Here, \(\text{sgn}\) defines the signum function. The equation (12) characterizes the de-convolution 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 \(M = \left[\frac{l}{\Delta p_1}\right]+1\) meshes on one dimension, \(x(n) = \begin{bmatrix} x(1, n) & \cdots & x(M, n) \end{bmatrix}^T\) and \(u(n) = \begin{bmatrix} u(s_1, n) & \cdots & u(s_n, n) \end{bmatrix}^T\), we have obtained the direct state-space equation (13).

\[
x(n+1) = Fx(n) + Gu(n) + TT(x(n)) \quad (13)
\]

The tridiagonal matrix \(F\) of size \(\text{dim}(F) = M \times M\) takes the form:

\[
F = \begin{bmatrix}
m_2(1, n) & m_1(1, n) & 0 & \cdots & 0 \\
m_3(2, n) & m_2(2, n) & m_1(2, n) & \vdots \\
0 & \ddots & \ddots & \ddots & \vdots \\
\vdots & \ddots & \ddots & \ddots & \ddots \\
0 & \cdots & 0 & m_3(M, n) & m_2(M, n)
\end{bmatrix}
\]

The matrix \(G\) of size \(\text{dim}(G) = M \times n_s\) is worth:

\[
G = T \begin{bmatrix}
\delta(1-s_{(1,1)}) & \delta(1-s_{(2,1)}) & \cdots & \delta(1-s_{(n_s,1)}) \\
\delta(2-s_{(1,1)}) & \delta(2-s_{(2,1)}) & \cdots & \delta(2-s_{(n_s,1)}) \\
\vdots & \vdots & \ddots & \ddots \\
\delta(M-s_{(1,1)}) & \delta(M-s_{(2,1)}) & \cdots & \delta(M-s_{(n_s,1)})
\end{bmatrix}
\]

14
Let \( y(n) = \begin{bmatrix} y(1,n) & \cdots & y(n_c,n) \end{bmatrix}^T \) and \( b(n) = \begin{bmatrix} b(1,n) & \cdots & b(n_c,n) \end{bmatrix}^T \).

In equation (6) characterizing the observations, the placing matrix \( H \) of the \( n_c \) sensors of size \( \text{dim}(H) = n_c \times M \) is expressed by:

\[
H = \begin{bmatrix}
\delta(1 - c_{(1,1)}) & \delta(2 - c_{(1,1)}) & \cdots & \delta(M - c_{(1,1)}) \\
\delta(1 - c_{(2,1)}) & \delta(2 - c_{(2,1)}) & \cdots & \\
\vdots & \ddots & \ddots & \\
\delta(1 - c_{(n_c,1)}) & \delta(M - c_{(n_c,1)}) & \\
\end{bmatrix}
\]

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

5.2. 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. Only linear case will be considered in this study. For this basic mesh, the matricies \( F, G \) and \( H \) are worth:

\[
F = \begin{bmatrix} m_2(1,n) & m_1(1,n) & 0 \\
m_3(2,n) & m_2(2,n) & m_1(2,n) \\
0 & m_3(3,n) & m_2(3,n) \end{bmatrix}, \quad G = \begin{bmatrix} 0 \\ T \end{bmatrix}, \quad H = \begin{bmatrix} 0 & 0 & 1 \end{bmatrix}
\]

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 (7) where the inverse state-space matrices are expressed by:

$$F_I = \begin{bmatrix}
1 & 0 & -\frac{m_1(1,n)}{m_2(1,n)m_3(3,n)} \\
0 & 0 & \frac{m_2(3,n)}{m_3(3,n)} \\
0 & 0 & 0
\end{bmatrix}, \quad H_I = \left[ \frac{m_3(2,n)}{Tm_2(1,n)} 1 \right]^T \zeta$$

$$G_I = \left[ \frac{m_1(1,n)m_2(3,n)}{m_2(1,n)m_3(3,n)} \right]^{T}, \quad I_I = \frac{\eta - \kappa - \nu}{Tm_2(1,n)m_3(3,n)}$$

Where the parameters in $\zeta$, $\eta$, $\kappa$ and $\nu$ are worth:

$$\zeta = \frac{m_1(1,n)m_3(2,n) - m_2(2,n)m_2(1,n)}{Tm_2(1,n)m_3(3,n)}, \quad \kappa = m_1(2,n)m_2(1,n)m_3(3,n)$$

$$\eta = m_2(1,n)m_2(2,n)m_2(3, n), \quad \nu = m_1(1,n)m_2(3,n)m_2(2,n)$$

The inverse neural model (figure 6) is carried out starting from (??). 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 define the remaining connections symbolized by arrows in figure 6. 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 in terms of parameters number for the same error ([Barron, 1993]).

5.3. 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 $\rho(F) < 1$;

• On the other hand, it has been advisable to know the behavior of the inverse state-space model in term of stability. The stability is ensured if and only if $\rho(F_I) < 1$.

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 7).

![Figure 7: Spectral radius evolution according to $T$: the reverse-time state-space model stability zones b) and c) (zoomed) are antagonist relative to a). Spectral radius of the two cases are inverse](image)

The inverse state-space model stability zone is totally antagonist with the direct state-space model one. For non-minimum phase system, it is then not possible to find a sampling period which ensures the direct and the reverse-time state-space models stability. Consequently, we have chosen a sampling period $T$ such as $\rho(F) < 1$ to ensure the simulation model stability and to remain faithful to the reality. Of course, this choice is unfavorable to the inverse state-space model stability but does not have any influence on the inverse state-space neural model which remains stable.

6. Results

The goal of this section is to check the assumptions of awaited quality concerning the gray-box NNIM 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), and in term of gain about the training effort. For that, the semi-physical NNIM has been compared to a traditional black-box inverse neural model.

6.1. Networks design

The black-box NNIM is a fully connected Elman network. In the case of the ODE model, the network is constituted by two linear neurons on its recurrent layer and one linear neuron on its output layer. For the PDE dispersion model, the recurrent layer possesses three linear neurons. After being randomly initialized, all the synaptic weights and biases are left free during the whole training. Figure 8 represents a classic design of a two layer Elman network. We have called $IW_{i,j}$, the weight matrices connected to inputs and $LW_{i,j}$ weight matrices coming from layer outputs. The indices $i$ and $j$ have been used to identify the source (second index) and the destination (first index) for the various weights. Here, $b$ corresponds to the bias.

![Classic two layer Elman network](image)

Figure 8: Classic two layer Elman network for the black-box NNIM. $IW_{i,j}, LW_{i,j}$ are weight matrices. Practically, time forward $q$ is substituted by a time delay $q^{-1}$ and input $y_{obs}$ is in reverse time.

The gray-box NNIM is designed starting from the previous black-box model and modified to obtain the inverse neural structure of figure 5 (ODE case) or 6 (PDE case). For that, we have connected the inputs layer to the output layer, added a delay between the two layers, and some values in the weight matrix $LW_{1,1}$ 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. Figure 9 represents the gray-box network.
6.2. Numerical simulations

In the case of the ODE model, we have chosen a damping parameter $\xi = 0.9$, a natural pulsation $\omega_0 = 5 \text{ rad.s}^{-1}$, a static gain $c_1 = 30$ and a sampling period $T = 0.05 \text{ s}$. Let us underline that this choice of parameters ensures for the matrix $F$ of the system (6) a spectral radius lower than 1. The direct state-space model stability is then guaranteed. For the PDE dispersion model, we have fixed a spatial sampling step $\Delta p_1 = 5 \text{ m}$, a wind speed field such as $v_1(1,n) = 5 \text{ m.s}^{-1}$, $v_1(2,n) = 5 \text{ m.s}^{-1}$ and $v_1(3,n) = 4 \text{ m.s}^{-1}$, a diffusion tensor such as $d_1(1,n) = 1 \text{ m}^2 \cdot \text{s}^{-1}$, $d_1(2,n) = 2 \text{ m}^2 \cdot \text{s}^{-1}$, $d_1(3,n) = 2 \text{ m}^2 \cdot \text{s}^{-1}$, and a chemical reaction coefficient $K = 0$. For the reasons previously exposed, we have set a sampling period $T = 0.2 \text{ s}$, ensuring the simulation model stability. The two NNIM models have been subjected to a learning with pseudo-experimental noisy data.

To construct the set of training, we have generated four short random input sequences of length $N = 50$ samples. These signals are step functions resulting from the product of an amplitude level $A_e$ by a Gaussian law of average $\mu_e$ and variance $\sigma_e^2$. The period $T_e$ is adjustable and characterizes the changes of states. By simulating the direct knowledge-based model starting from these input signals, we have obtained four noisy synthetic output signals. The average $\mu_b$, the variance $\sigma_b^2$, and the period $T_b$ characterize the noise dynamic. We have fixed $A_e = 1$, $\mu_e = 0$, $\sigma_e^2 = 1$, $\mu_b = 0$ and $T_b = 3T_e$. Of course, $T_e$ influences the dynamic of the input signals and thus, the dynamic of
the noisy synthetic output signals. We have then generated for each input sequence a random value for \( T \) such as a significant variation of the output signals is visible.

The learning stops if the number of iterations reaches 400 or if the mean squared error (MSE) is lower than 0.001 (ODE case) or 0.005 (PDE case). The error is calculated as the difference between the target output \( t \) (the desired input) and the network output \( \hat{t} \) (the estimated input):

\[
MSE = \frac{1}{N} \sum_{k=1}^{N} [t(k) - \hat{t}(k)]^2
\]  

(14)

In order to prevent over-learning on the training data, we have memorized all the weight matrices obtained after each epoch with a training signal. We have then kept the weights which give the best performance function. Moreover, early stopping improves regularization and tends to reduce noise influence, but in this case input restoration errors are more visible at the level of the changes of states (discontinuities).

During the test step (recall phase), we have studied the semi-physical contribution in terms of generalization and regularization from a new test signal. For that, we have generated another long random input sequence of length \( N = 400 \) samples. The noise variance of the corresponding noisy synthetic output signal is also worth \( \sigma_b^2 \).

To measure the noise influence, we have reproduced the previous protocol for several values of \( \sigma_b^2 \). The signal-to-noise ratio (SNR) of the corresponding synthetic output signals lies between 20 dB and plus infinity. Sometimes, 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 high MSE. Consequently, we have chosen to repeat each test hundred times and to calculate the average performances of the two NNIM. Since each test is realized with new random signals, we have used the normalized mean squared error (NMSE):

\[
NMSE = \frac{1}{N} \sum_{k=1}^{N} \frac{[t(k) - \hat{t}(k)]^2}{\sigma_t^2(k)}
\]  

(15)
with the unbiased variance:

$$\sigma_{t(k)}^2 = \frac{1}{N-1} \sum_{k=1}^{N} (t(k) - \overline{t(k)})^2$$

(16)

6.3. Modeling Errors and Regularizing Effect

The estimated input signals obtained without noise in the ODE case are shown in figure [10]. Let us underline that it deals with reverse-time signals. Figure [11] gathers these signals with a SNR of 20 dB.

Without noise in the training and test sequences, the inverse state-space model supplies an accurate input restoration. The semi-physical NNIM provides a nearly perfect input signal restoration, except for discontinuous zones. Indeed, the model does not exactly reproduce the changes of states. The estimated input signal obtained with the black-box inverse neural model is relatively approximative and biased. With a SNR of 20 dB, the inverse state-space model is largely penalized. Indeed, the noise is amplified
and the restoration is incorrect. For the gray-box model, restoration errors remain weak and suitable, but there is a slightly noise influence on the estimated input dynamic, due to the deterministic part introduced in the semi-physical modeling. For the black-box model, the noise influence is less visible than for the semi-physical model. However, the restoration is always relatively approximative and biased.

Figure 12 presents the estimated input signals obtained without noise in the PDE case. Estimated input signals with a SNR of 20 dB are shown in figure 13.

Let us bear in mind that in this case only the direct scheme stability is ensured. Thus without surprise, the inverse state-space model fastly diverges in both unnoisy and noisy cases. For the semi-physical and black-box models, the results are approximatively the same as those obtained in the ODE case.

Figure 14 gathers the average NMSE of the inverse models according to the SNR in the case of the second order ODE model.

Without noise in training and test sequences, the semi-physical NNIM provides best average performances (NMSE ≃ 0.13). The black-box neural model is slightly less
Figure 12: Estimated input signals obtained without noise in the PDE case: a) Simulated signals, b) & c) Estimated input signals.

Figure 13: Estimated input signals obtained with a SNR of 20 dB in the PDE case: a) Simulated signals, b) & c) Estimated input signals.
effective (NMSE $\simeq 0.27$). Of course, the inverse state-space model provides accurate results. When the noise grows, the inverse state-space model is largely penalized, whereas the two inverse neural models are moderately sensitive. The regularizing effect is real. In high noise situation, the gap between the two NNIM tends to slightly reduce. Indeed, the constraint imposed by the structure of the gray-box network and the more reduced connection number decrease the robust effect to the noise (loss of the neural network autoassociative memory property).

For the PDE dispersion model, the evolution of the NMSE according to the SNR is represented by figure 15.

The semi-physical NNIM again provides best average performances without noise (NMSE $\simeq 0.28$). Indeed, the black-box neural model is less effective (NMSE $\simeq 0.53$). Since the inverse state-space model fastly diverges, we do not compare its average performance. When the noise grows, the two NNPS are moderately sensitive, due to the regularizing effect. In addition, having chosen a sampling period $T$ such as $\rho(F_T) < 1$ does not interfere with the NNIM. In high noise situation, the two inverse neural models keep the same tendencies. Indeed, in this case the number of connections
in both networks is approximatively the same.

6.4. Learning Effort

We have compared the product of the NMSE by the number of epochs, i.e. the final error amplified by the number of iterations of the training phase. The results obtained with the ODE model are illustrated figure 16.

On the other side, figure 17 gathers the results obtained in the case of the PDE dispersion model.

We note that the gray-box NNIM is more effective in both slight and high noise situation than the black-box model. Physical knowledge favors the convergence of the weights so that the behavior approaches the data. The black-box inverse neural model is largely penalized because of its lesser capacity of regularization. Finally, we remark that the learning effort is about two times more important in the PDE dispersion model than in the ODE model.
Figure 16: Learning effort according to the SNR in the ODE case.

Figure 17: Learning effort according to the SNR in the PDE case.
7. Conclusion

We have proposed an approach to realize an inverse dynamic model resulting from the fusion of statistical training and deterministic modeling. 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 NNIM has been compared to a traditional black-box inverse neural model. The tests realized on a dynamic system characterized by an ODE, and 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 NNIM. 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.

Appendix A. Reverse-time state-space equations system

By considering the relation (6), we have obtained:

\[
\begin{align*}
\frac{x(n+1) - x(n)}{T} &= Ax(n) + Bu(n) \\
y(n) &= Cx(n) + b(n)
\end{align*}
\]  
(A.1)

Let us split the matrix $A$ of (A.1) in two parts, and let us write:

\[
\begin{align*}
\frac{x(n+1) - x(n)}{T} &= \begin{bmatrix} \overline{A}_{r-1} & A_1 \\ \overline{A}_1 & B_1 \end{bmatrix} x(n) + \begin{bmatrix} \overline{B}_{r-1} \\ B_1 \end{bmatrix} u(n) \\
y(n) - b(n) &= Cx(n)
\end{align*}
\]  
(A.2)

Where $A_k$ (respectively $B_k$) is constituted by the $k$ first lines of $A$ (respectively $B$), and $A_k$ (respectively $B_k$) is constituted by the $k$ last lines of $A$ (respectively $B$).

As $x(n) = \begin{bmatrix} x_1(n) & x_2(n) & \cdots & x_r(n) \end{bmatrix}^T$ in (A.2), we have obtained:
\[
\begin{pmatrix}
A_{r-1} \\
A_1
\end{pmatrix}
\begin{pmatrix}
x(n) \\
x_1(n)
\end{pmatrix}
= \frac{1}{T}
\begin{pmatrix}
x_{r-1}(n+1) \\
x_1(n+1)
\end{pmatrix}
- \frac{1}{T}
\begin{pmatrix}
x_{r-1}(n) \\
x_1(n)
\end{pmatrix}
- \begin{pmatrix}
B_{r-1} \\
B_1
\end{pmatrix}u(n)
\]  
(A.3)

\[x_1(n) = [y(n) - b(n)]\]  
(A.4)

By remarking that \(A_{r-1}x(n) = x_{r-1}(n)\) and separating (A.3), we have obtained:

\[x_{r-1}(n) = \frac{1}{T}x_{r-1}(n+1) - \frac{1}{T}x_{r-1}(n)\]  
(A.5)

\[A_1x(n) = \frac{1}{T}x_1(n+1) - \frac{1}{T}x_1(n) - \frac{c_1}{a_r}u(n)\]  
(A.6)

By concatenating (A.4) and (A.5), we have expressed:

\[
\begin{pmatrix}
x_1(n) \\
x_{r-1}(n)
\end{pmatrix}
= \frac{1}{T}
\begin{pmatrix}
T[y(n) - b(n)] \\
x_{r-1}(n+1)
\end{pmatrix}
- \frac{1}{T}
\begin{pmatrix}
0 \\
x_{r-1}(n)
\end{pmatrix}
\]  
(A.7)

By setting \(\Box_{r-1} = \begin{pmatrix} T[y(n) - b(n)] \\
x_{r-1}(n+1) \end{pmatrix}\) in (A.7), we have written more concisely:

\[x(n) = \frac{1}{T}\Box_{r-1} - \frac{1}{T}\begin{pmatrix}
0 \\
x_{r-1}(n)
\end{pmatrix}\]  
(A.8)

By using a recursive decomposition of (A.8), we have obtained:

\[x(n) = \frac{1}{T}\Box_{r-1} - \frac{1}{T}\begin{pmatrix}
0 \\
\frac{1}{T}\Box_{r-2} - \frac{1}{T}\begin{pmatrix}
0 \\
x_{r-2}(n)
\end{pmatrix}
\end{pmatrix}\]

And we have finally expressed:
By expanding the expression (A.9), we have obtained:

\[
x(n) = -\sum_{i=1}^{r-1} \left(-\frac{1}{T}\right)^i \begin{bmatrix} 0 \\ \vdots \\ -\frac{1}{T}^{r-1} \end{bmatrix} \begin{bmatrix} 0 \\ \vdots \\ -\frac{1}{T}^{r-1} \end{bmatrix} \begin{bmatrix} y(n) - b(n) \end{bmatrix}
\]

We have thus carried out the reverse-time state-space equation (A.10), where the state-space matrices \( F_l \) and \( G_l \) depend on the sampling period \( T \).

\[
x(n) = F_l x(n+1) + G_l [y(n) - b(n)] \tag{A.10}
\]

The lower triangular matrix \( F_l \) of size \( \text{dim}(F_l) = r \times r \) and the matrix \( G_l \) of size \( \text{dim}(G_l) = r \times 1 \) are worth:

\[
F_l = \begin{bmatrix} 0 & 0 & \cdots & 0 \\ \frac{1}{T} & 0 & \cdots & \vdots \\ \vdots & \ddots & \ddots & \vdots \\ -\left(\frac{1}{T}\right)^{r-1} & -\frac{1}{T} & 0 & 0 \end{bmatrix}, \quad G_l = \begin{bmatrix} 1 \\ -\frac{1}{T} \\ \vdots \\ -\left(\frac{1}{T}\right)^{r-1} \end{bmatrix}
\]

In addition, the relation (A.6) allows us to write:

\[
u(n) = \frac{a_r}{T c_1} x_1(n+1) - \frac{a_r}{c_1} A_1 x(n) - \frac{a_r}{T c_1} x_1(n)
\]
By simplifying, we have obtained (A.11):

\[
u(n) = \begin{bmatrix} 0 & \cdots & 0 \end{bmatrix} x(n+1) - \frac{a_r}{c_1} \begin{bmatrix} \frac{1}{T} \end{bmatrix} x(n) - \frac{a_r}{c_1} \begin{bmatrix} a_1 + \begin{bmatrix} 0 & \cdots & 0 & \frac{1}{T} \end{bmatrix} \end{bmatrix} x(n)\]

(A.11)

By incorporating relation (A.10) in (A.11), we have designed the reverse-time state-space equation (A.12), where the state-space matrices \( H_I \) and \( I_I \) also depend on the sampling period \( T \).

\[
u(n) = H_I x(n+1) + I_I \left[y(n) - b(n)\right]
\]

(A.12)

The matrix \( H_I \) of size \( \text{dim}(H_I) = 1 \times r \) is expressed by (A.13).

\[
H_I = \begin{bmatrix} 0 & \cdots & 0 \end{bmatrix} + \begin{bmatrix} a_0 \ c_1 & \cdots & a_{r-2} \ c_1 & \frac{1}{T c_1} (a_{r-1} T - a_r) \end{bmatrix} F_I
\]

(A.13)

The matrix \( I_I \) of size \( \text{dim}(I_I) = 1 \times 1 \), is given by (A.14).

\[
I_I = \begin{bmatrix} a_0 \ c_1 & \cdots & a_{r-2} \ c_1 & \frac{1}{T c_1} (a_{r-1} T - a_r) \end{bmatrix} G_I
\]

(A.14)

With equations (A.10) and (A.12), we have thus carried out the reverse-time state-space equations system which corresponds to the canonical form (3).

Appendix B. References


