The "Weight Smoothing" Regularization of MLP for Jacobian Stabilization

Filipe Aires, Michel Schmitt, Alain Chedin, and Noëlle Scott

***Abstract*— In an approximation problem with a neural network, a low-output root mean square (rms) error is not always a universal criterion. In this paper, we investigate problems where the Jacobians—first derivative of an output value with respect to an input value—of the approximation model are needed and propose to add a quality criterion on these Jacobians during the learning step. More specifically, we focus here on the approximation of functionals $\mathcal{A}$. from a space of continuous functions (discretized in pratice) to a scalar space. In this case, the approximation is confronted with the compensation phenomenon: a lower contribution of one input can be compensated by a larger one of its neighboring inputs. In this case, profiles (with respect to the input index) of neural Jacobians are very irregular instead of smooth. Then, the approximation of $\mathcal{A}$ becomes an ill-posed problem because many solutions can be chosen by the learning process. We propose to introduce the smoothness of Jacobian profiles as an *a priori* information *via* a regularization technique and develop a new and efficient learning algorithm, called "weight smoothing." We assess the robustness of the weight smoothing algorithm by testing it on a real and complex problem stemming from meteorology: the neural approximation of the forward model of radiative transfer equation in the atmosphere. The stabilized Jacobians of this model are then used in an inversion process to illustrate the improvement of the Jacobians after weight smoothing.**

***Index Terms*—Inverse problems, ill-posed problems, MLP, neural jacobians, regularization.**

I. Introduction

We study in this paper the approximation of a functional $\mathcal{A}$ by a multilayered perceptron g_W. The functional $\mathcal{A}$ describes dependencies between a space $\mathcal{M}$ of smooth functions f: $\mathbb{R} \rightarrow \mathbb{R}$ continuous, $\mathcal{C}^1$, $\mathcal{C}^2$, etc. and the space $\mathbb{R}^m$

$$\mathcal{A}\text{: } f \rightarrow y\text{; where } f \in \mathcal{M} \text{ and } y \in \mathbb{R}^m. \tag{1}$$

In our application f is a temperature profile in the atmosphere, f: pressure $\rightarrow$ temperature. In practice, the function f is discretized ($f \leftrightarrow x = (x_i;\ i = 1, \cdots, n)$) but we cannot consider the components $(x_i;\ i = 1, \cdots, n)$ in the input of the neural network independently. The x_i are ordered by the index i (for example the altitude) and hence possess some regularities of f. So, the regularity properties of function f must be transposed in its discretization x. In our case, the *a priori* knowledge is that the neural approximator describes a functional that has a smooth contribution of ordered inputs to each output. So, the *a priori* knowledge we have on the problem consists in the smoothness—in terms of the minimization of some derivatives—of the m neural Jacobian profiles: $\{J_{*k};\ k = 1, \cdots, m\}$ where $J_{*k} \overset{\text{def}}{=} ((\partial y_k/\partial x_i);\ i = 1, \cdots, n)$ is the kth Jacobian profile. A profile is defined as a discretized function with respect to some ordered index and a neural Jacobian $J_{ik} \overset{\text{def}}{=} (\partial y_k/\partial x_i)$ is the first derivative of the output y_k with respect to the input x_i of the neural network g_W. We focus on neural models but the following discussion could be applied in the general context of statistics. In our case, the neural Jacobians have an important physical meaning expressing the link between the frequency of the measurement channels to the atmospheric layers sounded (see our application on Section III).

A problem is said ill-posed if its solution may not exist, be nonunique or nonstable. The approximation of $\mathcal{A}$ is an ill-posed problem due to the nonunicity and the nonstability of the solution. Regularization [21] is one way to make our problem well-posed by stabilizing the neural Jacobians. The idea of regularization is to add a penalty term C_2 to the usual quality criterion C_1 in the learning process, with C_1 usually chosen as the mean square error in neural ouputs. This penalty term uses a regularizer (or stabilizer) $\Omega(\cdot)$ which forces the solution of the optimization problem to satisfy some constraints expressing the *a priori* knowledge about the approximation problem.

According to [22], a regularizer is a lower semicontinuous functional $\Omega(\cdot)$ that possesses the following three properties.

- The solution z of the inverse problem belongs to the domain of definition of the functional $\Omega(\cdot)$.
- On its domain of definition, the functional $\Omega(\cdot)$ admits real-valued non negative values.
- The sets $\mathcal{M}_c = \{z\text{: } \Omega(z) \leq c\}$, $c \geq 0$, are all compact.

Regularization decreases the representation's capability of the network but increases the bias (bias/variance dilemma [10]). So, the principle of regularization is to choose a well-defined regularizer to decrease the variance and to affect the bias as little as possible [4].

Examples of regularizers are the double backpropagation (DBP) [7] and the input perturbation (IP) [3] wich both force the neural function g_W to have small perturbations in the outputs y for small perturbations in the inputs x. During the learning step, they add the constraint of minimizing the magnitude of neural Jacobians J_{ik}. With the same goal of smoothing the neural function g_W, a constraint (i.e., a smoothing operator in the network g_W) is used in [17] and [11] to define directly the structure of the network g_W, called generalized regularization networks (GRN's). In [2] the authors

Manuscript received July 28, 1998; revised May 20, 1999 and July 19, 1999.

F. Aires, A. Chedin, and N. Scott are with the Laboratoire de Météorologie Dynamique du C.N.R.S., École Polytechnique, Palaiseau, France.

M. Schmitt is with the Centre de Géostatistique de l'École Nationale Supérieure des Mines de Paris, Fontainebleau, France.

Publisher Item Identifier S 1045-9227(99)09000-1.

have also tried to minimize the second derivative's magnitude. And recently, [14] have proposed to learn simultaneously a functional $\mathcal{A}$ and its Jacobians.

In this study, we develop a specific and efficient algorithm, the "weight smoothing" (WS) regularization, to introduce the constraint of Jacobian profile smoothness during the learning step. This is a new approach for regularization, very different from previously quoted methods using neural Jacobians like DBP, IP, or GRN that only minimize the neural Jacobian amplitude to smooth the neural-network behavior. The WS regularization is only appropriate when inputs of network are the discretization of a smooth function (so a natural ordering and regularity exists in inputs). This type of functional approximation is widespread. Examples include the retrieval of temperature or gas concentration profiles in the atmosphere from space observed outgoing radiances and, generally, all problems with smooth input data, like the salinity in the ocean (smoothness due to the diffusion), the propagation speed in a geological layer (smoothness due to the homogeneity of the layer), etc.

This paper is organized as follows: we first describe the regularizer and the resulting general learning algorithm. Then, an additional specification is introduced into the algorithm in order to speed up the learning step. Finally, a complex and real example stemming from meteorology is described. It concerns the approximation of the radiative transfer in the atmosphere (direct and inverse problem).

II. Regularizing by Smoothing Jacobian Profiles

A. Specifying the Smoothness of Jacobian Profiles

The MLP neural network $g_W(\cdot)$ carries out a function $x \to y$, where $x \in \mathbb{R}^n$, $y \in \mathbb{R}^m$, and W is the set of parameters of the neural network. For example, in a MLP network with one hidden layer S_1, the kth output component of the network is defined by

$$y_k(x) = \sum_{j \in S_1} w_{jk} \cdot \sigma(a_j) = \sum_{j \in S_1} w_{jk} \cdot \sigma\left(\sum_{i \in S_0} w_{ij} \cdot x_i \right) \quad (2)$$

where σ is a sigmoid function, a_j is the activity of unit j and S_i is the ith layer of the network (with $i = 0$ for the input layer). We have deliberately omitted the usual bias terms in expression (2) to simplify the notation without loss of generality.

To choose the parameters W, we apply a learning algorithm using a data set of P examples $((\hat{x}^p, \hat{y}^p); p = 1, \cdots, P)$. We assume that these axamples are independent and identically distributed and generated by the joint distribution of (x, y) [4]. A quality criterion C_1 is defined on this training set. We used the error backpropagation (BP) algorithm [19] for the optimization step. We introduce *a priori* information to regularize the learning step by adding a constraint to smooth the neural Jacobian profiles $J_{*k} = ((\partial y_k / \partial x_i); i = 1, \cdots, n)$. The neural Jacobians in the previous example (an MLP network with one hidden layer) are

$$J_{ik}(W, x) \stackrel{\text{def}}{=} \frac{\partial y_k}{\partial x_i} = \sum_{j' \in S_1} w_{j'k} \cdot \sigma'\left(\sum_{i' \in S_0} w_{i'j'} \cdot x_{i'} \right) \cdot w_{ij'}. \quad (3)$$

For a more complex MLP network, with many hidden layers, there exists a BP algorithm computing efficiently the neural Jacobians [4].

We smooth the kth Jacobian profile J_{*k}, using a linear Tikhonov's regularizer [21] (also called the Phillips–Twomey method [16]) of the form: $\Omega(J_{*k}) = (B \cdot J_{*k})^2$. The regularization matrix B could have different meanings. We give two examples of Tikhonov's regularizer with matrix B_1 and B_2

$$B_1 = \begin{pmatrix} -1 & 2 & -1 & 0 & \cdots & & 0 \\ 0 & -1 & 2 & -1 & 0 & & \vdots \\ \vdots & \ddots & \ddots & \ddots & \ddots & \ddots & \\ & & 0 & -1 & 2 & -1 & 0 \\ 0 & \cdots & & 0 & -1 & 2 & -1 \end{pmatrix}$$

$$B_2 = \begin{pmatrix} -1 & 3 & -3 & 1 & 0 & \cdots & & 0 \\ 0 & -1 & 3 & -3 & 1 & 0 & & \vdots \\ \vdots & \ddots & \ddots & \ddots & \ddots & \ddots & \ddots & 0 \\ & & 0 & -1 & 3 & -3 & 1 & 0 \\ 0 & \cdots & & 0 & -1 & 3 & -3 & 1 \end{pmatrix}.$$

For matrix B_1, we want to minimize the discrete second-derivative of the Jacobian profile (with respect to the input index), which means that the Jacobian profile should be as linear as possible (this is called a linear profile constraint). The regularizer $\Omega(z) = (B_1 \cdot z)^2 = z^t \cdot B_1^t \cdot B_1 \cdot z = z^t \cdot H_1 \cdot z$ is real-valued and nonnegative because the matrix $H_1 = B_1^t \cdot B_1$ is degenerate and possesses two zero eigen values that correspond to the two undetermined parameters of a linear profile z.

Similarly for B_2, we want to minimize the discrete third-derivative of the Jacobian profile to have a profile as quadratic as possible (this is called a quadratic profile constraint). The regularizer $\Omega(z) = z^t \cdot H_2 \cdot z$ is real-valued and nonnegative because the matrix $H_2 = B_2^t \cdot B_2$ is degenerate and possess three zero eigen values that correspond to the three undetermined parameters of a quadratic profile z.

All the *a priori* information introduced is on matrix B. The new criterion becomes: $C(W) = C_1(W) + \gamma \cdot C_2(W)$, where γ is the regularization parameter balancing the importance of the two criteria and $C_2(W) = (1/2P) \sum_{p=1}^{P} \sum_{k=1}^{m} \Omega(J_{*k}(W, \hat{x}^p))$.

For B equal to the identity matrix, the regularizer minimizes the neural Jacobian's magnitude, which leads to an equivalent technique to that quoted in introduction (DBP, IP, and GRN techniques).

B. Global Smoothing of Jacobian Profiles

To estimate parameters of the network by gradient-based optimization (e.g., stochastic gradient descent or conjugate gradient optimization), we require the derivative of $C(W)$.

The term $\partial C_1(W)/\partial w$ is the same as in the usual BP algorithm and the computation of $\partial C_2(W)/\partial w$ follows:

$$\frac{\partial C_2(W)}{\partial w} = \frac{1}{2}\sum_{k=1}^{m}\frac{\partial}{\partial w}\Omega(J_{*k}) = \sum_{k=1}^{m}(B\cdot J_{*k})^t\cdot\left(B\cdot\left(\frac{\partial}{\partial w}J_{*k}\right)\right) \quad (4)$$

where A^t is the transpose of matrix A. We give some practical details about the derivative in (4) with the example of the previous matrix B_2 in the Appendix.

The stochastic gradient version of the regularized learning algorithm becomes:

- propagate example x in the network by relation (2);
- compute the Jacobians J_{ik} with formula (3);
- compute the derivative $\partial C_1(W)/\partial w$ by usual BP algorithm;
- compute the derivative $\partial C_2(W)/\partial w$ by formula (4) (see (14) in the Appendix for a more detailed expression);
- modify the synaptic weights W by stochastic gradient descent.

This algorithm is computer intensive. For functions in spaces of relatively low dimensions, it can be used as presented. In higher dimensions, it is possible to begin the learning step by the usual BP algorithm and then use the regularization technique to smooth the Jacobian profiles.

C. The "Weight Smoothing" Regularization

In this section we search for a less complex, more rapid and efficient algorithm. Let us examine in more details the minimization of the cost C_2. We have

$$(B\cdot J_{*\hat{k}})^2 = \left(\sum_{j'\in S_1}\sigma'(a_{j'})\cdot\omega_{j'\hat{k}}\cdot(B\cdot W_{*j'})\right)^2 = (W_2^{\hat{k}}\cdot\Phi(W_1))^2 \quad (5)$$

where $W_2^{\hat{k}}$ is the vector $((\omega_{j'\hat{k}});\ j'\in S_1)$ and $\Phi(W_1)$ is the vector $((\sigma'(a_{j'})\cdot(B\cdot W_{*j'}));\ j'\in S_1)$. The minimization of the smoothing criterion is then equivalent to the orthogonalization of the two vectors in (5). By the Cauchy–Schwartz inequality

$$\begin{aligned}0 &\le (B\cdot J_{*\hat{k}})^2\\ &\le \left(\sum_{j'\in S_1}\omega_{j'\hat{k}}^2\right)\cdot\left(\sum_{j'\in S_1}\sigma'(a_{j'})^2\cdot(B\cdot W_{*j'})^2\right)\\ &\le M^2\cdot\left(\sum_{j'\in S_1}\omega_{j'\hat{k}}^2\right)\cdot\left(\sum_{j'\in S_1}(B\cdot W_{*j'})^2\right)\end{aligned} \quad (6)$$

where $M=\max_\nu\sigma'(\nu)<+\infty$.

The minimization of the terms $(\omega_{j'\hat{k}})^2$ is the classical "weight decay" regularization. And the minimization of $(B\cdot W_{*j'})^2$ (independent of the ouput $\hat{k}$) smoothes the weight profiles entering units $j'\in S_1$, and then smoothes the contribution of inputs to the hidden layer (Fig. 1). This approach is more constrained than the algorithm developed in Section II-B because if we orthogonalize the two vectors, the global

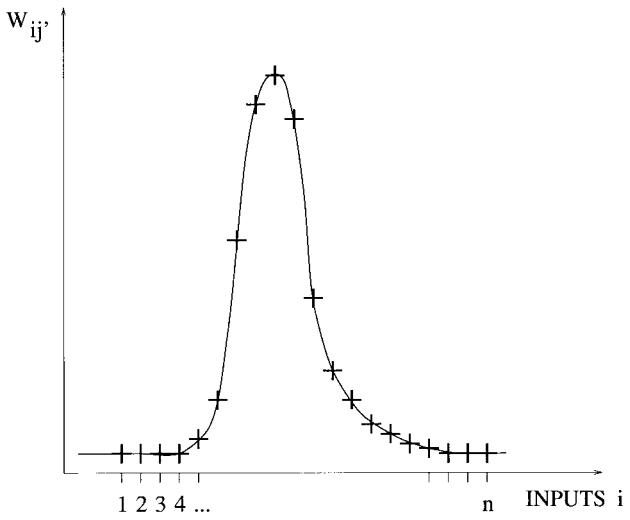

Fig. 1. Example of smooth of weight vector $W_{*j'}$.

Jacobians (input–output) will be smooth, but the internal profiles in the hidden-layer may still be very irregular and exhibit the compensation phenomenon: a lower contribution of one input can be compensated by a larger one of its neighboring inputs (this is a particular kind of overfitting concerning the smoothness of the underlying Jacibian profiles). If we minimize the norm of the two vectors, the internal profiles have to be smooth. The minimization of the norm of the second vector specifies that the hidden-layer of the network is a kind of filtering preprocessing step solving the compensation phenomenon directly. So, we substitute the new penalty term C_3 for the previous C_2

$$C_3(W)=\frac{1}{2P}\sum_{p=1}^{P}\left(\gamma'\sum_{\hat{k}\in S_2}\sum_{j'\in S_1}(w_{j'\hat{k}})^2+\sum_{j'\in S_1}(B\cdot W_{*j'})^2\right) \quad (7)$$

with γ' a parameter balancing the importance of the weight decay and the preprocessing filter regularizers. Then during learning, we iteratively modify the weights w_{ij} as in the usual BP but with an additional term, for every example $(\hat{x},\hat{y})$ in the training set

$$\begin{cases}\text{for } i\in S_0\text{: } w_{ij}^{t+1}\\ \quad = w_{ij}^t-\rho\cdot\left(\dfrac{\partial(g(W^t,\hat{x})-\hat{y})^2}{\partial w_{ij}^t}+\gamma_1\cdot\dfrac{\partial(B\cdot W_{*j}^t)^2}{\partial w_{ij}^t}\right),\\ \text{for } i\in S_1\text{: } w_{ij}^{t+1}\\ \quad = w_{ij}^t-\rho\cdot\left(\dfrac{\partial(g(W^t,\hat{x})-\hat{y})^2}{\partial w_{ij}^t}+\gamma_2\cdot w_{ij}^t\right).\end{cases} \quad (8)$$

For example, with the matrix B_2 we have

$$\frac{\partial(B\cdot W_{*j}^t)^2}{\partial w_{i,j}^t}=(-w_{i-3,j}^t+6w_{i-2,j}^t-15w_{i-1,j}^t+20w_{i,j}^t-15w_{i+1,j}^t+6w_{i+2,j}^t-w_{i+3,j}^t). \quad (9)$$

This algorithm, called WS regularization, is very cheap because it requires only few additions and multiplications more

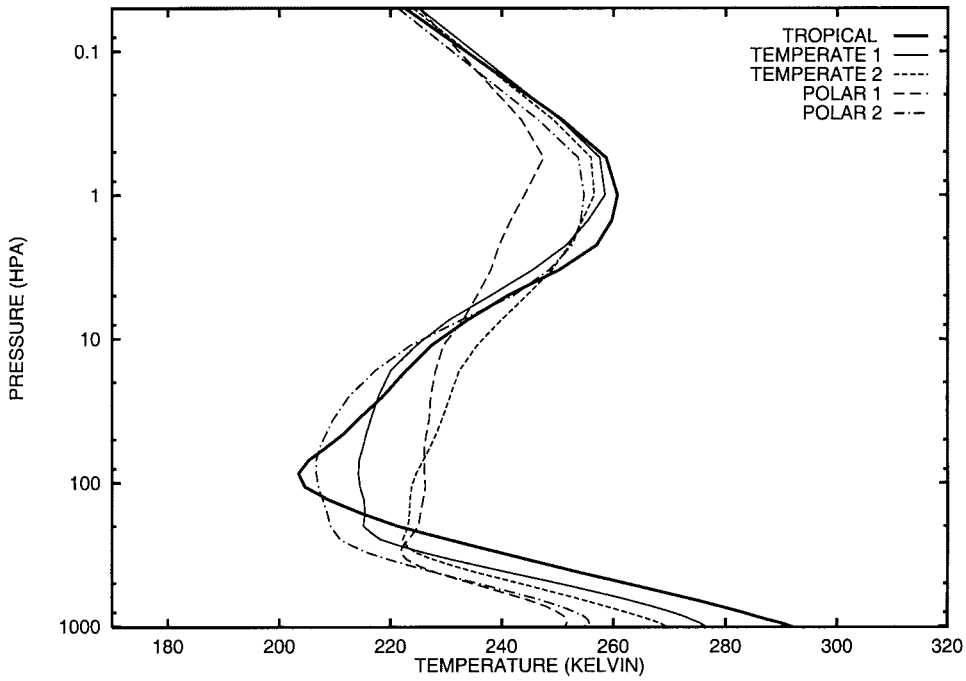

Fig. 2. Typical temperature profiles in the atmosphere.

than the usual BP for each weight modification. And as we will see in the next section, the results of learning are exactly what we expect.

III. Application to the Radiative Transfer in the Atmosphere

Our application is based on the radiative transfer equation (RTE) in the earth's atmosphere, which can be summarized by

$$I = \mathcal{A}(G) \tag{10}$$

where $I \in \mathbb{R}^n$ is the measured brigthness temperatures and $G \in \mathbb{R}^m$ the geophysical parameters describing the atmospheric situation (surface and atmospheric temperature, concentration of various gases like water vapor or ozone at different altitudes on the atmosphere). For the complete equation involving the physical parameters, see [20]. Among these geophysical parameters, we only consider the vertical temperature profile. It is a function of the altitude or of the atmospheric log-pressure (these two quantities are quasilinearly dependent). We show five typical temperature profiles in Fig. 4, belonging to five different air masses (tropical, midlatitude type 1 and 2 and polar type 1 and 2). Note that the temperature increases smoothly with the log-pressure. The direct problem concerns the determination of the measured brightness I from the geophysical variables G. In the inverse problem, we try to retrieve the geophysical variables G [like the vertical atmospheric temperature profile $T(P)$ in (10)] from the measured brightness temperatures $I(\nu)$ in various spectral intervals ν.

A. Regularization of the Direct Model

The TOVS instrument (TIROS-N Operational Vertical Sounder), flown aboard the satellites of the TIROS-N/NOAA (National Oceanic and Atmospheric Administration of the United States) series since 1979, measures the brightness temperature emitted by the atmosphere in the infrared and the microwaves ranges. Of its 27 channels (that correspond to 27 frequencies) measuring the brightness temperature, 11 are "sensitive" to temperature and then are used to restitute this profile.

In this section, we want first to determine the magnitude $I(\nu)$ of these 11 channels, using only the information from the temperature profile. One limitation of the direct model of (10) is that here G represents the temperature profile and I the brightness temperatures measured in the subset of the 11 temperature sensitive channels. The atmosphere is discretized in 60 atmospheric levels, so the variable $G \in \mathbb{R}^{60}$.

This application is clearly a functional because the input space (the space of temperature profiles) is a space of real, discretized and continuous functions f: pressure $\rightarrow$ temperature. The compensation phenomenon is present in this model because the error on the transmission factor of a given atmospheric layer can be compensated by one of its neighboring layers [18]. So we apply our WS regularization technique with a quadratic constraint for the Tikhonov's regularizer (matrix B_2 of Section II-A).

The network used here is a MLP network with one hidden layer. The architecture has 60 units in the input layer (the 60 atmospheric temperatures between zero and 59 km), 50 units in the hidden layer (this number was chosen empirically by trial in the training set) and 11 units in the output layer (the 11 brightness temperatures of TOVS channels sensitive to temperature).

For the training and the testing data set we have merged the two climatology data bases TIGR2 [8] and TIGR1 [5]. TIGR stands for "Thermodynamical Initial Guess Retrieval" and is

a vast and as diversified as possible set of real atmospheres. For each of these atmospheric situations, the corresponding TOVS brightness temperatures are then computed by the 4A (automatized atmospheric absorption atlas) algorithm [20]. These two data sets come from a sampling process of about 100 000 *in situ* measurements (geophysical parameters). They are two reasons to perform this sampling process: first, some of the 100 000 radiosondes measurements have poor quality (missing data, instrumental noise, etc.), so quality criteria have to be used to select only adequate measurements. Second, the simulation of the brightness temperature spectrum emitted by the atmosphere for each radiosonde measurements is very computer expensive. We extract from the TIGR data base 1761 atmospheres for the learning step and 681 atmospheres for the generalization step.

We apply the BP algorithm with and without WS regularization. We empirically choose $\gamma = 1.0$ to tend to make the two parts of the criterion data/regularization have comparable weights. During the WS learning, we have observed that the weight decay term (acting in connections between layer S_1 and layer S_2) has no significant effect on the results. So, we take $\gamma' = 0.0$ in (8). The most important part of the WS regularization is the smoothing of input contributions to the hidden layer ("filtering preprocessing" step).

We have tested similarly two classical regularization techniques: the "weight decay" and the "weight elimination." The penalization parameter was choosen empirically by trial to ensure a maximum performance level on both the training and the testing set (a proper method would use a cross-validation approach). The root mean square (rms) errors obtained with these two regularization techniques during the generalization step are close to 0.45 Kelvin. This test error is too high for our purpose, so these two techniques are not well adapted for this kind of problems. On the other hand, rms test errors for the neural estimation with or without ws regularization are good: an rms test error in the brightness temperatures lower than 0.2° Kelvin, in data between 150 and 300° Kelvin. This is a good result because the errors are comparable to the noise of the TOVS instrument. As WS restricts the representation capabilities of the neural network, the training performances should be worse. The fact that we obtain same learning errors with and without WS illustrates perfectly the compensation phenomenon: many physical models can give the same level of error in output because the learning problem is under-constrained. The goal of WS regularization is to avoid solutions created by the compensation phenomenon, and choose the most realistic and physically acceptable solution to the estimation problem among all solutions.

In Fig. 3, we have represented the neural Jacobian profiles for two networks (with and without regularization) and for four different channels of TOVS taken among the 11. Fig. 3 shows that without WS, the profiles of neural Jacobians are very irregular. This is due to the compensation phenomenon: an error in the contribution of one atmospheric layer is compensated by errors in the neighboring layers. This is why a bad solution (in term of neural Jacobians) can give a good rms test error. On the other hand, the neural Jacobian profiles of the regularized network are smooth.

The WS regularization is then satisfactory because it produces a neural estimation $g_W(\cdot)$ with smooth neural Jacobian profiles. These neural Jacobian profiles are also consistent with the physical knowledge we have about the real Jacobian profiles of $\mathcal{A}$. They are positive and the pressure where the Jacobian of a channel is maximum indicates the atmospheric layer making the largest contribution to the brightness temperature.

The availability of good quality neural Jacobians is very important. One application of these neural Jacobians is their use in the numerical weather prediction model of the Meteorological Operational Centers. These models uses the technique of variational assimilation [18]: if we have an initial estimation of the state of the atmosphere given by a first guess $\overline{f}$, and a satellite measurement of brightness temperature F perturbed by an instrumental noise ε, the estimation of the state of the atmosphere becomes

$$\hat{f} = \overline{f} + S_{\overline{f}} \cdot {K_f}^t \cdot (K_f \cdot S_{\overline{f}} \cdot {K_f}^t + S_\varepsilon)^{-1}(F - K_f \cdot \overline{f}) \tag{11}$$

where $S_{\overline{f}}$ is the covariance matrix of first guess error and K_f is the Jacobian matrix $(\partial F_k/\partial f_i)_{ki}$. The major advantage of the neural Jacobians is the rapidity for their computation, a hard limitation in operational models.

B. Neural Inversion of the Radiative Transfer Equation

To show that we obtain a more robust estimation of the RTE with the WS learning, we study the impact of this regularization in a neural inversion technique. The inverse problem of radiative transfer is very important: a vertical sounder determines the thermodynamical variables in the atmosphere [5]. So we have to invert the previous direct model $\mathcal{A}$ of radiative transfer. The approximation of $\mathcal{A}^{-1}$ is also an ill-posed problem because its solution can be: nonexistent (due to noise in measurements), nonunique (with the problem of compensation phenomenon) or nonstable (because the numerical computations are ill-conditioned) [18].

One strategy for the resolution of the inverse problem is to use a neural network h_W to compute directly the temperature profile (in the output y of h_W) with the brightness temperature measurements (in the input x of h_W). See [9] for this direct inversion with the TOVS instrument and [1] for the application of this technique to the IASI interferometer. The neural model h_W could also be regularized, for example to impose a constraint of smoothness $(B \cdot y)^2$ on the temperature profiles of the output of h_W, where B is one smoothing matrix of Section II-A. An approach to solve this general problem in the linear case could also be found on [15].

Other neural techniques of inversion, such as "distal learning" [12], "indirect inversion" [6] or "iterative inversion" [13], first estimate the direct model $\mathcal{A}$ by a MLP and then, invert it using its neural Jacobians. This strategy solves some difficulties, particularly the multivalued aspect of the inverse function [12].

Here, we have used the iterative inversion because it is an excellent test for the quality of the direct model and its Jacobians: after estimating the direct model $\mathcal{A}$ by a network

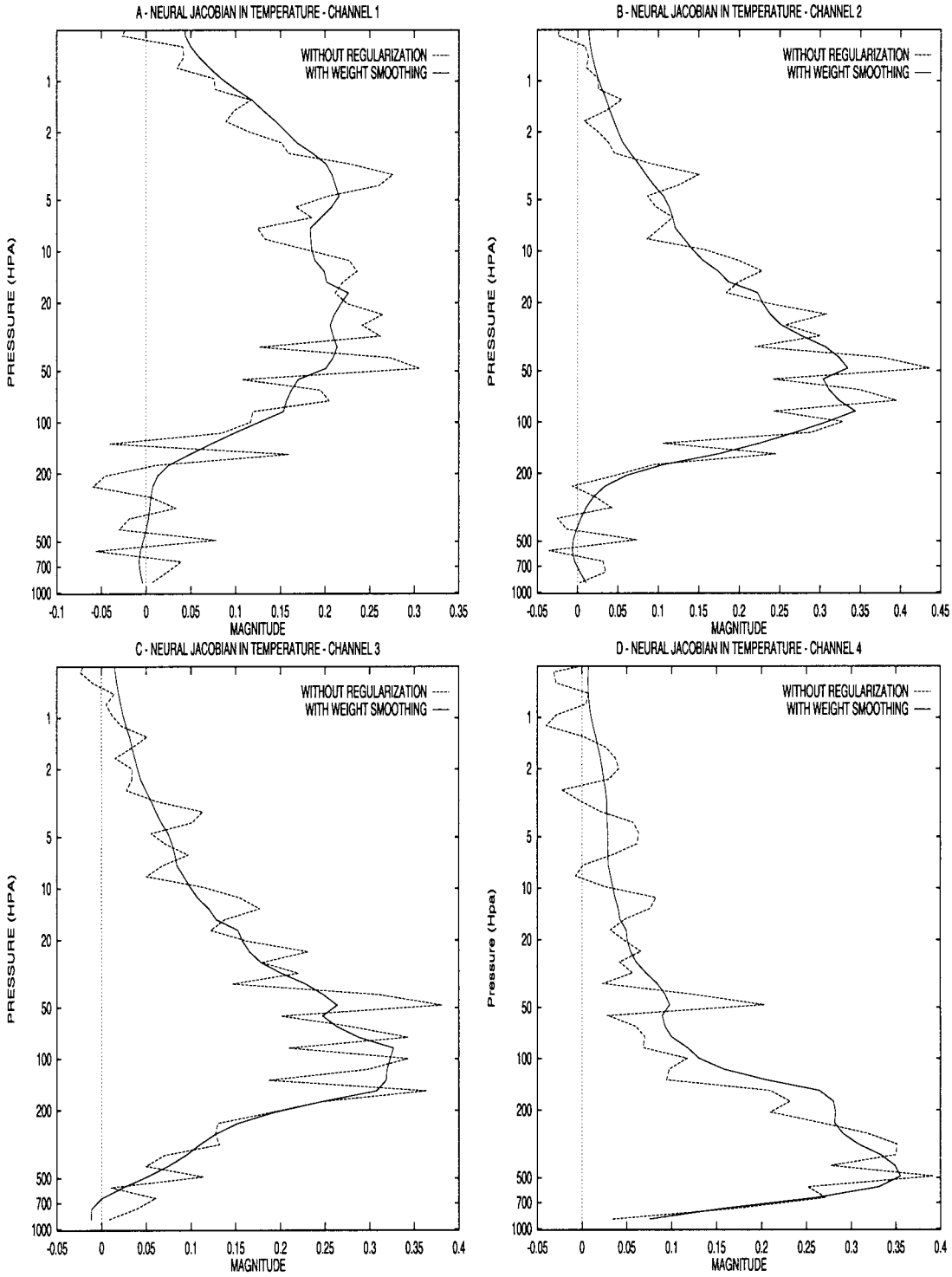

Fig. 3. Neural Jacobian profiles of the direct model for TOVS, for one atmospheric situation of the TIGR data base, with and without "weight smoothing" ($\gamma = 1.0$, $\gamma' = 0.0$ and learning rate = 0.1).

$g_W(\cdot)$ (Section III-A), for a given radiance measurement $y \in \mathbb{R}^m$, we search for the temperature profile $x \in \mathbb{R}^n$ such that $g_W(x) = y$. This is achieved by iterative modifications in the current solution x^t using the neural Jacobians, according to:

- take a first guess x^0 (we always choose the mean temperature profile $\langle x \rangle$ in our TIGR data bases);
- modify the current solution x^t using: $x^{t+1} = x^t - \lambda \cdot (\partial/\partial x^t)(g_W(x^t) - y)^2$, where λ is the learning step-rate of the inversion algorithm.

This algorithm can be used to help solve an inverse problem. But it can also be used to analyze what a neural network $g_W(\cdot)$ has learned. Depending on the initial guess x^0, this inversion

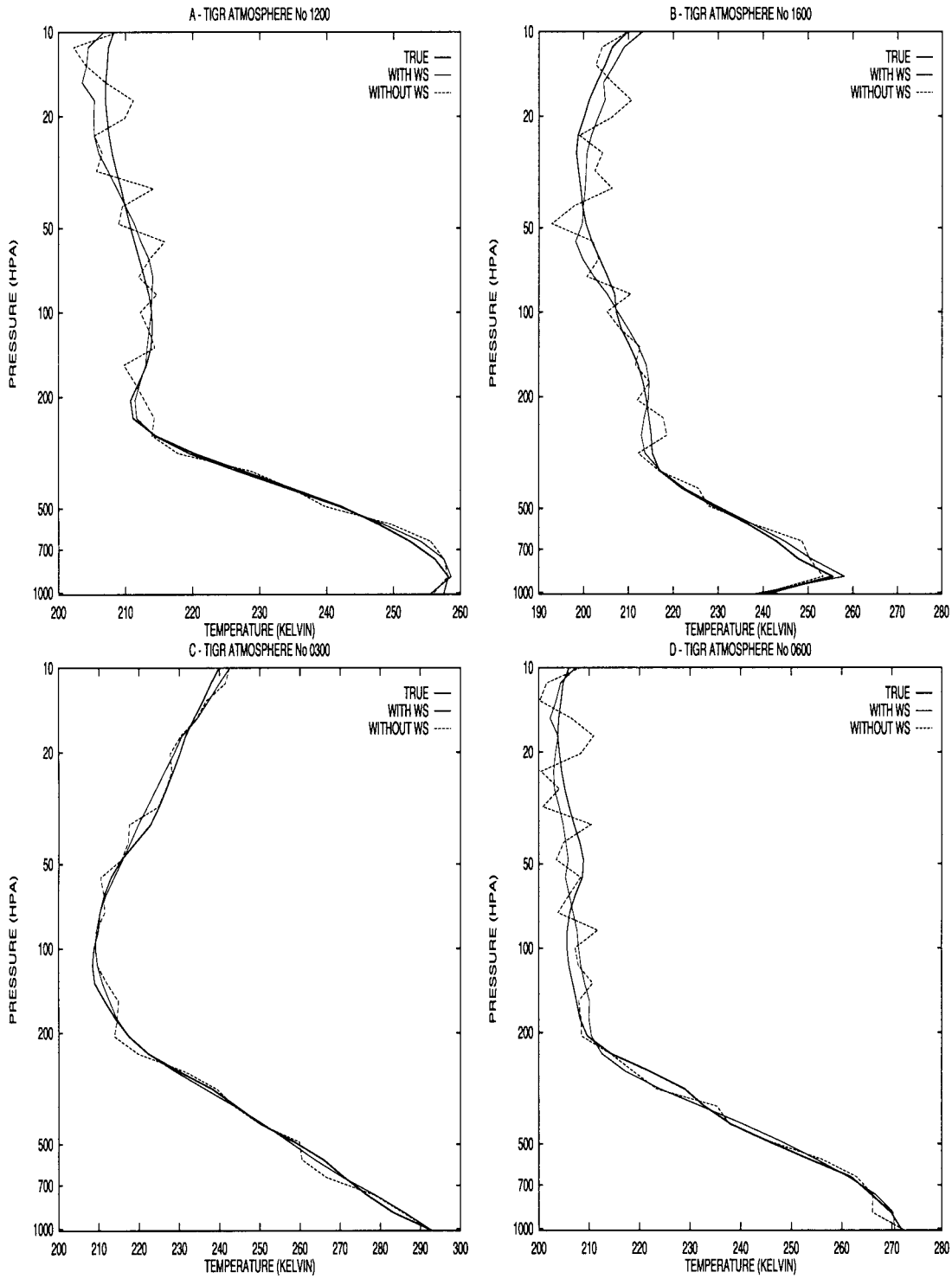

Fig. 4. Retrieved temperature profiles from the TIGR data base, with the iterative inversion algorithm, using the neural direct model with and without "weight smoothing" ($\lambda = 0.1$ and $x_0 = \langle x \rangle$).

process can give different solutions and, thus, can indicate whether the direct model is robust or not.

We have tested this inversion technique with the two networks $g_W(\cdot)$ (with and without WS) of Section II-C which simulate the direct radiative transfer. Then, for a given TOVS measurement y (the 11 brightness temperatures of the TOVS channels sensitive to atmospheric temperature), we search for the corresponding temperature profile x such that $g_W(x) = y$. In Fig. 4, we represent four examples of this inversion process. Each graph compares results with and without WS regularization. The root mean square error in the restitution of these four temperature profiles is given in Fig. 5. Fig. 4 shows that, without WS regularization, the retrieved temperature profiles are very irregular: oscillations around the real profile are observed. The increase of the oscillations in the top of the atmosphere is due to the discretization (atmospheric layers of

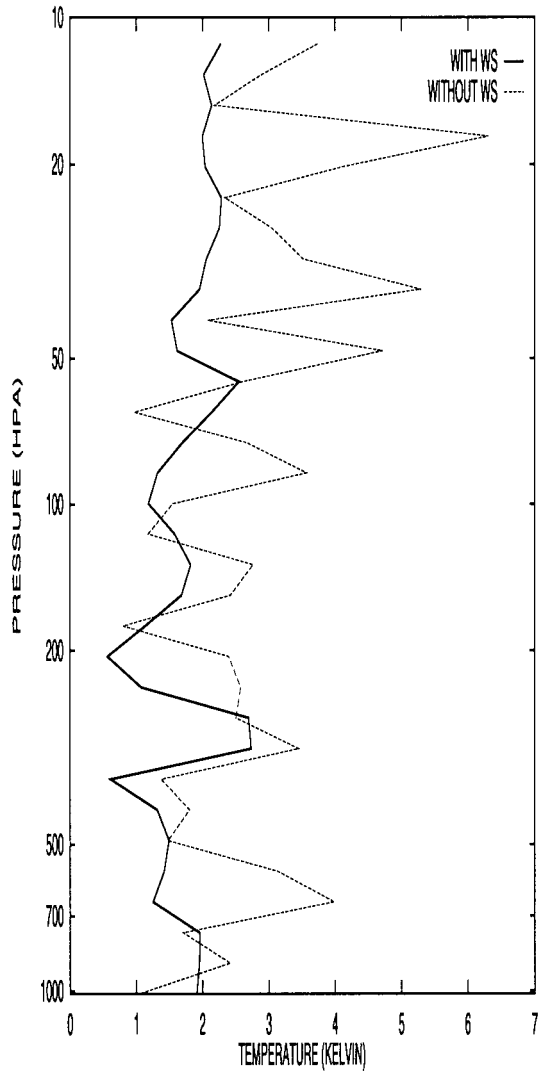

Fig. 5. Root mean square error for the four restituted temperature profiles from the TIGR data base, with the iterative inversion algorithm, using the neural direct model with and without WS regularization ($\lambda = 0.1$ and $x_0 = \langle x \rangle$).

1 km) used for the temperature profile restitution. The TOVS radiometer do not have channels sounding this pressure range at such a high vertical resolution, so the number of degrees of freedom in the restituted profile is higher than the number of pieces of information given by the TOVS instrument. However; this problem of oscillations from the surface to the top of the atmosphere, is solved to a large extent with the WS procedure.

The results of the inversion algorithm are then substantially improved if we use our regularized model of the RTE. The profile retrieved with WS regularization is a good estimation of the real smooth profile both in term of errors (Fig. 4) and in term of the smoothing characteristics of the profile.

Other advantages of this regularized inversion technique are that the inversion process is rapid, the Jacobians of the neural model are available (this is a very important fact for the "variational assimilation" technique in numerical weather prediction models) and that the inversion technique is well-adapted for the analysis of the different solutions of the inversion problem.

We have then shown that our regularized model of the direct radiative transfer equation in Section III-A is more robust than those without WS. We have shown the advantages that the WS regularization can offer to neuronal inversion algorithms (like "distal learning," "iterative inversion," or "indirect inversion") in cases where the function to invert is a functional sensitive to the compensation phenomenon.

IV. Conclusion

We have presented an original approach to the regularization of MLP: the WS algorithm. For functional approximation with inputs resulting from the discretization of a continuous function, the WS regularization smoothes the neural Jacobian profiles with respect to the input index. Solving the compensation phenomenon in such approximations, this algorithm makes it possible to estimate a physically acceptable and more robust solution. This specific regularization technique appears to be more efficient, in our case, than other classical regularization algorithms like "weight decay" or "weight elimination."

We have illustrated the efficiency of this algorithm through a real and quite complex problem stemming from space meteorology: the radiative transfer model in the earth's atmosphere and more particularly, the role of the temperature. This application has demonstrated the rapidity of the learning algorithm and the quality of solutions. The more robust neural Jacobians obtained with WS regularization may be used, for example, in a Numerical Weather Prediction models. They could also be used in neural techniques of inversion like "iterative inversion," "distal learning" or "indirect inversion" that use the neural Jacobians. We have tested our regularized direct neuronal model in an iterative inversion algorithm where the quality of neural Jacobians is essential to estimate atmospheric temperature profiles from space measurements of the earth outgoing radiance.

Appendix

If $\hat{\imath} \in S_0$ is an input component of the network, $\hat{\jmath}$ in the hidden layer S_1 and $\hat{k}$ an output component in S_2, the derivatives in (4) become

$$\begin{cases} \dfrac{\partial J_{ik}}{\partial w_{\hat{\imath}\hat{\jmath}}} = w_{\hat{\jmath}k} \cdot (\sigma''(a_{\hat{\jmath}}) \cdot x_{\hat{\imath}} \cdot w_{i\hat{\jmath}} + \sigma'(a_{\hat{\jmath}}) \cdot \delta_{i\hat{\imath}}) \\ \dfrac{\partial J_{ik}}{\partial w_{\hat{\jmath}\hat{k}}} = w_{i\hat{\jmath}} \cdot \sigma'(a_{\hat{\jmath}}) \cdot \delta_{k\hat{k}} \end{cases} \tag{12}$$

where δ is the Kronecker delta symbol. Then

$$\begin{cases} \dfrac{\partial C_2(W)}{\partial w_{\hat{\imath}\hat{\jmath}}} = (B \cdot (\sigma''(a_{\hat{\jmath}}) \cdot x_{\hat{\imath}} \cdot W_{*\hat{\jmath}} + \sigma'(a_{\hat{\jmath}}) \cdot \delta_{*\hat{\imath}}))^t \\ \qquad\qquad \cdot \left(\displaystyle\sum_{k \in S_2} w_{\hat{\jmath}k} \cdot (B \cdot J_{*k}) \right) \\ \dfrac{\partial C_2(W)}{\partial w_{\hat{\jmath}\hat{k}}} = \sigma'(a_{\hat{\jmath}}) \cdot (B \cdot W_{*\hat{\jmath}})^t \cdot (B \cdot J_{*\hat{k}}). \end{cases} \tag{13}$$

For example, with the previous matrix B_2, we obtain

$$\begin{cases} \dfrac{\partial C_2(W)}{\partial w_{\hat{\imath}\hat{\jmath}}} = \displaystyle\sum_{\imath' \in \overline{S}_0} (\sigma''(a_{\hat{\jmath}}) \cdot (-w_{\imath',\hat{\jmath}} + 3w_{\imath'+1,\hat{\jmath}} \\ \qquad \cdot x_{\hat{\imath}} - 3w_{\imath'+2,\hat{\jmath}} + w_{\imath'+3,\hat{\jmath}}) + \sigma'(a_{\hat{\jmath}}) \\ \qquad \cdot (-\delta_{\imath',\hat{\imath}} + 3\delta_{\imath'+1,\hat{\imath}} - 3\delta_{\imath'+2,\hat{\imath}} + \delta_{\imath'+3,\hat{\imath}})) \\ \qquad \cdot \left(\displaystyle\sum_{k \in S_2} w_{\hat{\jmath}k} \cdot (-J_{\imath',k} + 3J_{\imath'+1,k} - 3J_{\imath'+2,k} \right. \\ \qquad \left. + J_{\imath'+3,k}) \right) \\ \dfrac{\partial C_2(W)}{\partial w_{\hat{\jmath}\hat{k}}} = \displaystyle\sum_{\imath' \in \overline{S}_0} \sigma'(a_{\hat{\jmath}}) \cdot (-w_{\imath',\hat{\jmath}} + 3w_{\imath'+1,\hat{\jmath}} - 3w_{\imath'+2,\hat{\jmath}} \\ \qquad + w_{\imath'+3,\hat{\jmath}}) \cdot (-J_{\imath',\hat{k}} + 3J_{\imath'+1\hat{k}} - 3J_{\imath'+2,\hat{k}} \\ \qquad + J_{\imath'+3,\hat{k}}) \end{cases} \tag{14}$$

where $\overline{S}_0 = \{1, \cdots, n-3\}$ takes into account the edge condition of J_{*k}.

REFERENCES

[1] F. Aires, R. Armante, A. Chédin, and N. Scott, "Surface and atmospheric temperature retrieval with the high resolution interferometer iasi," in *AMS Conf.*, Paris, France, 1998, pp. 181–186.

[2] C. Bishop, "Curvature-driven smoothing: A learning algorithm for feedforward networks," *IEEE Trans. Neural Networks*, vol. 4, pp. 882–884, 1993.

[3] ——, "Training with noise is equivalent to tikhonov regularization," *Neural Comput.*, vol. 7, no. 1, pp. 108–116, 1995.

[4] ——, *Neural Networks for Pattern Recognition*. Oxford, U.K.: Clarendon, 1996.

[5] A. Chédin, N. Scott, C. Wahiche, and P. Moulinier, "The improved initialization inversion method: A high-resolution physical method for temperature retrievals from tiros-n series," *J. Clim. Appl. Meteor.*, vol. 24, pp. 128–143, 1985.

[6] S. Colombano, M. Compton, and M. Bualat, "Goal directed model inversion: Adaptation to unexpected model changes," in *Neuro-Nimes*, 269–278, 1991.

[7] H. Drucker and Y. Le Cun, "Improving generalization performance using double backpropagation," *IEEE Trans. Neural Networks*, vol. 3, pp. 991–997, 1992.

[8] J. Escobar, *Base de données pour la restitution de paramètres atmosphériques à l'échelle globale; étude sur l'inversion par réseaux de neurones des données des sondeurs verticaux atmosphériques satellitaires présents et à venir*, Ph.D. dissertation, Université Denis Diderot (Paris VII), 1991.

[9] J. Escobar, A. Chédin, F. Cheruy, and N. Scott, "Réseaux de neurones multicouches pour la restitution de variables thermodynamiques atmosphériques à l'aide de sondeurs verticaux satellitaires," *CRAS*, vol. 317, no. II, pp. 911–918, 1993.

[10] S. Geman, E. Bienenstock, and R. Doursat, "Neural networks and the bias-variance dilema," *Neural Comput.*, vol. 1, no. 4, pp. 1–58, 1992.

[11] F. Girossi, M. Jones, and T. Poggio, "Regularization theory and neural networks architectures," *Neural Comput.*, vol. 7, pp. 219–269, 1995.

[12] M. Jordan and D. Rumelhart, "Forward models: Supervised learning with a distal teacher," *Cognitive Sci.*, vol. 16, pp. 307–354, 1992.

[13] J. Kindermann and A. Linden, "Inversion of neural networks by gradient descent," *Parallel Computing*, 270–286, 1990.

[14] J.-W. Lee and J.-H. Oh, "Hybrid learning of mapping and its Jacobian in multilayer neural networks," *Neural Comput.*, vol. 9, pp. 937–958, 1997.

[15] M. Nashed and G. Wahba, "Generalized inverses in reproducing kernel spaces: an approach to regularization of linear operator equations," *SIAM J. Math. Anal.*, vol. 5, no. 6, pp. 974–987, 1974.

[16] D. Phillips, *J. Assoc. Comput. Machinery*, vol. 9, pp. 84–97, 1962.

[17] T. Poggio and F. Girosi, "Networks for approximation and learning," in *Proc. IEEE*, vol. 78, pp. 1481–1497, 1990.

[18] C. Rodgers, "Retrieval of atmospheric temperature and composition from remote measurements of thermal radiation," *Rev. Geophys. Space Phys.*, vol. 14, no. 4, pp. 609–624, Nov. 1976.

[19] D. Rumelhart, G. Hinton, and R. Williams, *Learning Representations by Backpropagating Error*, vol. I. Cambridge, MA: MIT Press, 1986, ch. 8, pp. 318–362.

[20] N. Scott and A. Chédin, "A fast line-by-line method for atmospheric absorption computations: The automatized atmospheric absorption atlas," *J. Appl. Meteor.*, vol. 20, pp. 802–812, 1981.

[21] A. Tikhonov and V. Arsenin, *Solutions of Ill-Posed Problems*. Washington, D.C.: V. H. Vinsten, 1977.

[22] V. Vapnik, *Statistical Learning Theory*. New York: Wiley, 1998.