To appear in Neural Networks.

CONSTRUCTION OF CONFIDENCE INTERVALS FOR NEURAL NETWORKS BASED ON LEAST SQUARES ESTIMATION

Isabelle Rivals and Léon Personnaz

Laboratoire d'Électronique, École Supérieure de Physique et de Chimie Industrielles,
10 rue Vauquelin, 75231 Paris Cedex 05, France.

Isabelle Rivals
Laboratoire d'Électronique, École Supérieure de Physique et de Chimie Industrielles,
10 rue Vauquelin, 75231 Paris Cedex 05, France.
Phone: 33 1 40 79 45 45 Fax: 33 1 40 79 44 25
E-mail: Isabelle.Rivals@espci.fr

CONFIDENCE INTERVALS FOR NN

CONSTRUCTION OF CONFIDENCE INTERVALS FOR NEURAL NETWORKS
BASED ON LEAST SQUARES ESTIMATION

Abstract

We present the theoretical results about the construction of confidence intervals for a nonlinear regression based on least squares estimation and using the linear Taylor expansion of the nonlinear model output. We stress the assumptions on which these results are based, in order to derive an appropriate methodology for neural black-box modeling; the latter is then analyzed and illustrated on simulated and real processes. We show that the linear Taylor expansion of a nonlinear model output also gives a tool to detect the possible ill-conditioning of neural network candidates, and to estimate their performance. Finally, we show that the least squares and linear Taylor expansion based approach compares favourably with other analytic approaches, and that it is an efficient and economic alternative to the non analytic and computationally intensive bootstrap methods.

Keywords

Nonlinear regression, neural networks, least squares estimation, linear Taylor expansion, confidence intervals, ill-conditioning detection, model selection, approximate leave-one-out score.

Notations

We distinguish between random variables and their values (or realizations) by using upper- and lowercase letters; all vectors are column vectors, and are denoted by boldface letters; non random matrices are denoted by light lowercase letters.

.

$\mathbf{x}$	non random n-input vector
$Y_p = Y_p \mid \mathbf{x}$	random scalar output depending on $\mathbf{x}$
$E(Y_p \mid \mathbf{x})$	mathematical expectation, or regression function, of Y_p given $\mathbf{x}$
W	random variable with zero expectation denoting additive noise
s^2	variance of W
$\{\mathbf{x}^k, y_p^k\}_{k=1 \text{ to } N}$	data set of N input-output pairs, where the $\{\mathbf{x}^k\}$ are non random n-vectors, and the $\{y_p^k\}$ are the corresponding realizations of the random outputs $\{Y_p^k = Y_p \mid \mathbf{x}^k\}$
$\{(\mathbf{x}^k)^T \boldsymbol{q},\ \boldsymbol{q} \in \mathbb{R}^n\}$	family of linear functions of $\mathbf{x}$ parameterized by $\boldsymbol{q}$
$\boldsymbol{q}_p$	unknown true q-parameter vector (q=n in the linear case)
$x = [\mathbf{x}^1\ \mathbf{x}^2 \dots \mathbf{x}^N]^T$	non random (N,n) input matrix
$\mathbf{Y}_p = [Y_p^1\ Y_p^2 \dots Y_p^N]^T$	random N-vector of the outputs of the data set
$\mathbf{W} = [W^1\ W^2 \dots W^N]^T$	random N-vector with $E(\mathbf{W}) = \mathbf{0}$
$J(\boldsymbol{q})$	value of the least squares cost function
$\boldsymbol{Q}_{LS}$	least squares estimator of $\boldsymbol{q}_p$
$\boldsymbol{q}_{LS}$	least squares estimate of $\boldsymbol{q}_p$
$\mathbf{R} = \mathbf{Y}_p - x\,\boldsymbol{Q}_{LS}$	least squares residual random N-vector in the linear case
$\mathbf{r}$	value of $\mathbf{R}$
$\mathcal{M}(x)$	range of x (linear manifold)
p_x	orthogonal projection matrix on $\mathcal{M}(x)$
S^2	estimator of s^2
s^2	value of S^2
$\{f(\mathbf{x}, \boldsymbol{q}),\ \boldsymbol{q} \in \mathbb{R}^q\}$	family of nonlinear functions of $\mathbf{x}$ parameterized by $\boldsymbol{q}$
$\mathbf{f}(x, \boldsymbol{q})$	N-vector $[f(\mathbf{x}^1, \boldsymbol{q}) \dots f(\mathbf{x}^k, \boldsymbol{q}) \dots f(\mathbf{x}^N, \boldsymbol{q})]^T$
$\mathbf{R} = \mathbf{Y}_p - \mathbf{f}(x, \boldsymbol{Q}_{LS})$	least squares residual random N-vector
$\xi = [\boldsymbol{x}^1\ \boldsymbol{x}^2 \dots \boldsymbol{x}^N]^T$	unknown non random (N, q) matrix with $\boldsymbol{x}^k = \left.\frac{\partial f(\mathbf{x}^k, \boldsymbol{q})}{\partial \boldsymbol{q}}\right\|_{\boldsymbol{q} = \boldsymbol{q}_p}$
$\mathcal{M}(\xi)$	range of ξ
p_ξ	orthogonal projection matrix on $\mathcal{M}(\xi)$

$z = [\mathbf{z}^1\ \mathbf{z}^2 \dots \mathbf{z}^N]^T$	matrix approximating ξ with $\mathbf{z}^k = \left.\frac{\partial f(\mathbf{x}^k, \boldsymbol{q})}{\partial \boldsymbol{q}}\right\|_{\boldsymbol{q} = \boldsymbol{q}_{LS}}$
$\mathcal{M}(z)$	range of z
p_z	orthogonal projection matrix on $\mathcal{M}(z)$
I_N	(N,N) identity matrix
$\boldsymbol{q}_{LS}^{(k)}$	leave-one-out (the k-th example) least squares estimate
$\{e^k\}_{k=1 \text{ to } N}$	leave-one-out errors
n_h	number of hidden neurons of a neural network
H	random Hessian matrix of the cost function
h	value of the Hessian matrix of the cost function
$s^2_{ref}(\mathbf{x})$	reference variance estimate
$\widehat{var}(f(\mathbf{x}, \boldsymbol{Q}_{LS}))_{LTE}$	LTE estimate of the variance of a nonlinear model output
$\widehat{var}(f(\mathbf{x}, \boldsymbol{Q}_{LS}))_{Hessian}$	Hessian estimate of the variance of a nonlinear model output
$\widehat{var}(f(\mathbf{x}, \boldsymbol{Q}_{LS}))_{sandwich}$	sandwich estimate of the variance of a nonlinear model output

Abbreviations

CI	confidence interval
LOO	leave-one-out
LS	least squares
LTE	linear Taylor expansion
SISO	single input - single output
MISO	multi input - single output
MSTE	mean square training error
MSPE	mean square performance error

I. INTRODUCTION

For any modeling problem, it is very important to be able to estimate the reliability of a given model. This problem has been investigated to a great extent in the framework of linear regression theory, leading to well-established results and commonly used methods to build confidence intervals (CIs) for the regression, that is the process output expectation [Seber 1977]; more recently, these results have been extended to nonlinear models [Bates & Watts 1988] [Seber & Wild 1989]. In neural network modeling studies however, these results are seldom exploited, and generally only an average estimate of the neural model reliability is given through the mean square model error on a test set; but in an application, one often wishes to know a CI at any input value of interest. Nevertheless, thanks to the increase of computer power, the use of bootstrap methods has increased in the past years [Efron & Tibshirani 1993]. These non analytic methods have been proposed to build CIs for neural networks [Paass 1993] [Tibshirani 1993] [Heskes 1997], but with the shortcoming of requiring a large number of trainings.

This paper presents an economic alternative to the construction of CIs using neural networks. This approach being built on the linear least squares (LS) theory applied to the linear Taylor expansion (LTE) of the output of nonlinear models, we first recall how to establish CIs for linear models in section II, and then approximate CIs for nonlinear models in section III. In section IV, we exploit these known theoretical results for practical modeling problems involving neural models. We show that the LTE of a nonlinear model output not only provides a CI at any input value of interest, but also gives a tool to detect the possible ill-conditioning of the model, and, as in [Monari 1999] [Monari & Dreyfus], to estimate its performance through an approximate leave-one-out (LOO) score. A real-world illustration is given through an industrial application, the modeling of the elasticity of a complex material from some of its structural descriptors. Section V compares the LS LTE approach to other analytic approaches, and discusses its advantages with respect to bootstrap approaches.

We consider single-output models, since each output of a multi-output model can be handled separately. We deal with static modeling problems for the case of a non random (noise free) n-input vector $\mathbf{x} = [x_1\ x_2\ \dots\ x_n]^T$*, and a noisy measured output* y_p *which is considered as the actual value of a random variable* $Y_p = Y_p \mid \mathbf{x}$ *depending on* $\mathbf{x}$*. We assume that there exists an unknown regression function* $E(Y_p \mid \mathbf{x})$ *such that for any fixed value of* $\mathbf{x}$ *:*

$$Y_p \mid \mathbf{x} = E(Y_p \mid \mathbf{x}) + W \mid \mathbf{x} \qquad (1)$$

where $W \mid \mathbf{x}$ *is thus a random variable with zero expectation. A family of parameterized functions* $\{f(\mathbf{x}, \boldsymbol{q}), \mathbf{x} \in \mathbb{R}^n, \boldsymbol{q} \in \mathbb{R}^q\}$ *is called an assumed model. This assumed model is said to be true if there exists a value* $\boldsymbol{q}_p$ *of* $\boldsymbol{q}$ *such that,* $\forall\ \mathbf{x}$ *in the input domain of interest,* $f(\mathbf{x}, \boldsymbol{q}_p) = E(Y_p \mid \mathbf{x})$*. In the following, a data set of N input-output pairs* $\{\mathbf{x}^k, y_p^k\}_{k=1 \text{ to } N}$ *is available, where the* $\mathbf{x}^k = [x_1^k\ x_2^k\ \dots\ x_n^k]^T$ *are non random n-vectors, and the* $\{y_p^k\}$ *are the corresponding realizations of the random variables* $\{Y_p^k = Y_p \mid \mathbf{x}^k\}$[1]*. The goal of the modeling procedure is not only to estimate the regression* $E(Y_p \mid \mathbf{x})$ *in the input domain of interest with the output of a model, but also to compute the value of a CI for the regression, that is the value of a random interval with a chosen probability to contain the regresssion. For the presentation of the results of linear and nonlinear regression estimation, we deal with the true model (a model which is linear in the parameters in section II, a nonlinear one in section III), i.e. we consider that a family of functions containing the regression is known. In*

[1] *We recall that we distinguish between random variables and their values (or realizations) by using upper- and lowercase letters, e.g.* Y_p^k *and* y_p^k*; all vectors are column vectors, and are denoted by boldface letters, e.g. the n-vectors* $\mathbf{x}$ *and* $\{\mathbf{x}^k\}$*; non random matrices are denoted by light lowercase letters (except the unambiguous identity matrix).*

section IV, we consider the general realistic case of neural black-box modeling where a preliminary selection among candidate neural models is first performed because a true model is not known a priori.

II. CONFIDENCE INTERVALS FOR LINEAR MODELS

We consider a true linear assumed model, that is the associated family of linear functions $\left\{\mathbf{x}^T \boldsymbol{q}, \mathbf{x} \in \mathbb{R}^n, \boldsymbol{q} \in \mathbb{R}^n\right\}$ *contains the regression; (1) can thus be uniquely rewritten as:*

$$Y_p \mid \mathbf{x} = \mathbf{x}^T \boldsymbol{q}_p + W \mid \mathbf{x} \qquad (2)$$

where $\boldsymbol{q}_p$ *is an unknown n-parameter vector. Model (2) associated to the data set leads to:*

$$\mathbf{Y}_p = x\, \boldsymbol{q}_p + \mathbf{W} \qquad (3)$$

where $x = \left[\mathbf{x}^1\ \mathbf{x}^2 \ldots \mathbf{x}^N\right]^T$ *is the non random (N,n) input matrix,* $\mathbf{Y}_p = \left[Y_p^1\ Y_p^2 \ldots Y_p^N\right]^T$ *and* $\mathbf{W} = \left[W^1\ W^2 \ldots W^N\right]^T$ *are random N-vectors, with* $E\left(\mathbf{W}\right) = \mathbf{0}$*. Geometrically, this means that* $E\left(\mathbf{Y}_p \mid x\right) = x\, \boldsymbol{q}_p$ *belongs to the solution surface, the linear manifold* $\mathcal{M}(x)$ *of the observation space* $\mathbb{R}^N$ *spanned by the columns of the input matrix*[2] *(the range of x). We assume that* $\mathcal{M}(x)$ *is of dimension n, that is rank* $(x) = n$*. In other words, the model is identifiable, i.e. the data set is appropriately chosen, possibly using experimental design.*

II.1. The linear least squares solution

The LS estimate $\boldsymbol{q}_{LS}$ *of* $\boldsymbol{q}_p$ *minimizes the empirical quadratic cost function:*

$$J\left(\boldsymbol{q}\right) = \frac{1}{2} \sum_{k=1}^{N} \left(y_p^k - \left(\mathbf{x}^k\right)^T \boldsymbol{q}\right)^2 = \frac{1}{2} \left(\mathbf{y}_p - x\, \boldsymbol{q}\right)^T \left(\mathbf{y}_p - x\, \boldsymbol{q}\right) \qquad (4)$$

[2] $\mathcal{M}(x)$ *is sometimes called the "expectation surface" [Seber & Wild 89]; as a matter of fact, the solution surface coincides with the expectation surface only when the assumed model is true.*

The estimate $\boldsymbol{q}_{LS}$ *is a realization of the LS estimator* $\boldsymbol{Q}_{LS}$ *whose expression is:*

$$\boldsymbol{Q}_{LS} = \left(x^T x\right)^{-1} x^T\, \mathbf{Y}_p = \boldsymbol{q}_p + \left(x^T x\right)^{-1} x^T\, \mathbf{W} \qquad (5)$$

Since the assumed model is true, this estimator is unbiased. The orthogonal projection matrix on $\mathcal{M}(x)$ *is* $p_x = x\left(x^T x\right)^{-1} x^T$*. It follows from (5) that the unbiased LS estimator of* $E\left(\mathbf{Y}_p \mid x\right)$ *is:*

$$x\, \boldsymbol{Q}_{LS} = x\, \boldsymbol{q}_p + p_x\, \mathbf{W} \qquad (6)$$

that is the sum of $E\left(\mathbf{Y}_p \mid x\right)$ *and of the projection of* $\mathbf{W}$ *on* $\mathcal{M}(x)$*, as shown in Figure 1. Let* $\mathbf{R}$ *denote the residual random N-vector* $\mathbf{R} = \mathbf{Y}_p - x\, \boldsymbol{Q}_{LS}$ *, that is the vector of the errors on the data set, then:*

$$\mathbf{R} = \left(I_N - p_x\right) \mathbf{W} \qquad (7)$$

Under the assumption that the $\left\{W^k\right\}$ *are identically distributed and uncorrelated (homoscedastic), i.e. the noise covariance matrix is* $K\left(\mathbf{W}\right) = \sigma^2 I_N$*, it follows from (5) that the variance of the LS estimator of the regression for any input* $\mathbf{x}$ *of interest is*[3]*:*

$$var\left(\mathbf{x}^T \boldsymbol{Q}_{LS}\right) = \sigma^2\, \mathbf{x}^T \left(x^T x\right)^{-1} \mathbf{x} \qquad (8)$$

Using (7), we obtain the unbiased estimator $S^2 = \dfrac{\mathbf{R}^T \mathbf{R}}{N - n}$ *of* s^2*; the corresponding (unbiased) estimate of the variance of* $\mathbf{x}^T \boldsymbol{Q}_{LS}$ *is thus:*

$$\widehat{var\left(\mathbf{x}^T \boldsymbol{Q}_{LS}\right)} = s^2\, \mathbf{x}^T \left(x^T x\right)^{-1} \mathbf{x} \qquad (9)$$

where s is the value of S.

II.2. Confidence intervals for a linear regression

If the $\left\{W^k\right\}$ *are homoscedastic gaussian variables, that is* $\mathbf{W} \rightarrow \mathbf{N}_N\left(\mathbf{0}, \sigma^2 I_N\right)$*:*

ThL1. $$\boldsymbol{Q}_{LS} - \boldsymbol{q}_p \rightarrow \mathbf{N}_n\left(\mathbf{0}, \sigma^2 \left(x^T x\right)^{-1}\right) \qquad (10)$$

ThL2. $$\frac{\mathbf{R}^T \mathbf{R}}{\sigma^2} \rightarrow \chi^2_{N-n} \qquad (11)$$

ThL3. $\boldsymbol{Q}_{LS}$ *is statistically independent from* $\mathbf{R}^T \mathbf{R}$.

[3] *We recall that* $\mathbf{x}$ *(boldface) is the input vector (n, 1) of interest, and that x is the experimental (N, n) input matrix.*

The proof of the above theorems follows from Figure 1 and from the Fisher-Cochrane theorem [Goodwin & Payne 1977], see for instance [Seber 1977].

The goal is to build a CI for the regression value $E(Y_p \mid \mathbf{x}) = \mathbf{x}^T \boldsymbol{q}_p$, *for any input vector* $\mathbf{x}$ *of interest. The variance of the measurements* s^2 *being unknown, let us build a normalized centered gaussian variable where both* $E(Y_p \mid \mathbf{x})$ *and* σ *appear:*

$$\frac{\mathbf{x}^T \boldsymbol{Q}_{LS} - E(Y_p \mid \mathbf{x})}{\sigma \sqrt{\mathbf{x}^T (x^T x)^{-1} \mathbf{x}}} \rightarrow \mathbf{N}\,(0,\, 1) \qquad (12)$$

Thus, using the Pearson variable (11), which is independent from (12) according to ThL3, we obtain the following Student variable:

$$\frac{\mathbf{x}^T \boldsymbol{Q}_{LS} - E(Y_p \mid \mathbf{x})}{S \sqrt{\mathbf{x}^T (x^T x)^{-1} \mathbf{x}}} \rightarrow Student\,(N-n) \qquad (13)$$

A 100(1–α*)% CI for* $E(Y_p \mid \mathbf{x})$ *is thus:*

$$\mathbf{x}^T \boldsymbol{q}_{LS} \pm t_{N-n}(\alpha)\, s \sqrt{\mathbf{x}^T (x^T x)^{-1} \mathbf{x}} \qquad (14)$$

where t_{N-n} *is the inverse of the Student(N-n) cumulative distribution.*

Note that (14) allows to compute a CI corresponding to any input vector, and that it is much more informative than average values such as that the mean square error on the data set, or the mean of the variance estimate over the data set[4]*; as a matter of fact, the latter invariably equals* $s^2 \frac{n}{N}$.

II.3. Example of a simulated linear SISO process (process #1)

We consider a simulated single input - single output (SISO) linear process:

$$y_p^k = q_{p_1} + q_{p_2} x^k + w^k \quad k=1 \text{ to } N \qquad (15)$$

We take $\theta_{p_1} = 1$, $\theta_{p_2} = 1$, $\sigma^2 = 0.5$, $N = 30$. *The inputs* $\{x^k\}$ *of the data set are uniformly distributed in [-3; 3], as shown in Figure 2a. The family of functions* $\{q_1 + q_2\, x\,,\ \boldsymbol{q} \in \mathbb{R}^2\}$ *is considered, that is the assumed model is true, and we choose*

[4] *The mean of the variance estimate over the training data set is:*
$\frac{1}{N}\sum_{k=1}^{N} s^2 (\mathbf{x}^k)^T (x^T x)^{-1} \mathbf{x}^k = \frac{s^2}{N}\sum_{k=1}^{N} [p_x]_{kk} = \frac{s^2}{N}$ *trace* (p_x). *Since* p_x *is the orthogonal projection matrix on a n-dimensional subspace, trace* $(p_x) = n$ *.*

a confidence level of 99% ($t_{28}(1\%) = 2.76$*). The LS estimation leads to* $s^2 = 0.29$*, i.e. underestimates the noise variance. Figure 2b displays the estimate (9) of the variance of* $\mathbf{x}^T \boldsymbol{Q}_{LS}$ *, and the true variance (8). The estimator* S^2 *of the noise variance being unbiased, the difference between the estimated variance (9) and the (usually unknown) true variance (8) is only due to the particular values of the measurement noise. Figure 2a shows the regression* $E(Y_p \mid \mathbf{x})$*, the data set, the model output and the 99% CI for the regression computed with (14).*

III. APPROXIMATE CONFIDENCE INTERVALS FOR NONLINEAR MODELS

We consider a family of nonlinear functions $\{f(\mathbf{x},\, \boldsymbol{q}),\ \mathbf{x} \in \mathbb{R}^n,\ \boldsymbol{q} \in \mathbb{R}^q\}$ *which contains the regression, that is the assumed model is true; (1) can thus be rewritten as:*

$$Y_p \mid \mathbf{x} = f(\mathbf{x},\, \boldsymbol{q}_p) + W \mid \mathbf{x} \qquad (16)$$

where $\boldsymbol{q}_p$ *is an unknown q-parameter vector. We denote by* $\mathbf{f}(x,\, \boldsymbol{q}_p)$ *the unknown vector* $[f(\mathbf{x}^1,\, \boldsymbol{q}_p) \ldots f(\mathbf{x}^k,\, \boldsymbol{q}_p) \ldots f(\mathbf{x}^N,\, \boldsymbol{q}_p)]^T$ *defined on the data set, thus:*

$$\mathbf{Y}_p = \mathbf{f}(x,\, \boldsymbol{q}_p) + \mathbf{W} \qquad (17)$$

As in section II, x denotes the (N,n) input matrix[5]*, and* $\mathbf{Y}_p$ *and* $\mathbf{W}$ *are random N-vectors with* $E(\mathbf{W}) = \mathbf{0}$*. Geometrically, this means that* $E(\mathbf{Y}_p \mid x)$ *belongs to the solution surface, the manifold* $\mathcal{M}(\mathbf{f}(x,\, \boldsymbol{q})) = \{\mathbf{f}(x,\, \boldsymbol{q}),\ \ \boldsymbol{q} \in \mathbb{R}^q\}$ *of* $\mathbb{R}^N$.

III.1. The linear Taylor expansion of the nonlinear least squares solution

A LS estimate $\boldsymbol{q}_{LS}$ *of* $\boldsymbol{q}_p$ *minimizes the empirical cost function*[6]*:*

[5] *In the case of a nonlinear model, x is merely a two-dimensional array.*

[6] *In the case of a multilayer neural network, the minimum value of the cost function can be obtained for several values of the parameter vector; but, since the only function-preserving transformations are*

$$J(\boldsymbol{q}) = \frac{1}{2}\sum_{k=1}^{N}\left(y_p^k - f(\mathbf{x}^k,\ \boldsymbol{q})\right)^2 = \frac{1}{2}\left(\mathbf{y}_p - \mathbf{f}(x,\ \boldsymbol{q})\right)^T\left(\mathbf{y}_p - \mathbf{f}(x,\ \boldsymbol{q})\right) \qquad (18)$$

The estimate $\boldsymbol{q}_{LS}$ is a realization of the LS estimator $\boldsymbol{Q}_{LS}$. Efficient algorithms are at our disposal for the minimization of the cost function (18): they can lead to an absolute minimum, but they do not give an analytic expression of the estimator that could be used to build CIs. In order to take advantage of the results concerning linear models, it is worthwhile considering a linear approximation of $\boldsymbol{Q}_{LS}$ which is obtained by writing the LTE of $f(\mathbf{x},\ \boldsymbol{q})$ around $f(\mathbf{x},\ \boldsymbol{q}_p)$:

$$f(\mathbf{x},\ \boldsymbol{q}) \approx f(\mathbf{x},\ \boldsymbol{q}_p) + \sum_{r=1}^{q}\left(\frac{\partial f(\mathbf{x},\ \boldsymbol{q})}{\partial \theta_r}\bigg|_{\boldsymbol{q}=\boldsymbol{q}_p}\left(\theta_r - \theta_{p_r}\right)\right) = f(\mathbf{x},\ \boldsymbol{q}_p) + \boldsymbol{x}^T(\boldsymbol{q} - \boldsymbol{q}_p) \qquad (19)$$

where $\boldsymbol{x} = \dfrac{\partial f(\mathbf{x},\ \boldsymbol{q})}{\partial \boldsymbol{q}}\bigg|_{\boldsymbol{q}=\boldsymbol{q}_p}$. Thus, with the matrix notation:

$$\mathbf{f}(x,\ \boldsymbol{q}) \approx \mathbf{f}(x,\ \boldsymbol{q}_p) + \xi\ (\boldsymbol{q} - \boldsymbol{q}_p) \qquad (20)$$

where $\xi = \left[\boldsymbol{x}^1\ \boldsymbol{x}^2\ \ldots\ \boldsymbol{x}^N\right]^T$ and $\boldsymbol{x}^k = \dfrac{\partial f(\mathbf{x}^k,\ \boldsymbol{q})}{\partial \boldsymbol{q}}\bigg|_{\boldsymbol{q}=\boldsymbol{q}_p}$. The (N,q) matrix ξ is the non random and unknown (since $\boldsymbol{q}_p$ is unknown) Jacobian matrix of $\mathbf{f}$. Using (20), one obtains, similarly to the linear case, the following approximation of $\boldsymbol{Q}_{LS}$ (see Appendix 1.1 for a detailed derivation of (21) and (23)):

$$\boldsymbol{Q}_{LS} \approx \boldsymbol{q}_p + \left(\xi^T\ \xi\right)^{-1}\ \xi^T\ \mathbf{W} \qquad (21)$$

The range $\mathfrak{M}(\xi)$ of ξ is tangent to the manifold $\mathfrak{M}(\mathbf{f}(x,\ \boldsymbol{q}))$ at $\boldsymbol{q} = \boldsymbol{q}_p$; this manifold is assumed to be of dimension q, hence $\mathrm{rank}(\xi) = q$. The matrix $p_\xi = \xi\left(\xi^T\ \xi\right)^{-1}\ \xi^T$ is the orthogonal projection matrix on $\mathfrak{M}(\xi)$. From (20) and (21), the LS estimator of $E(\mathbf{Y}_p \mid x)$ can be approximately expressed by:

$$\mathbf{f}(x,\ \boldsymbol{Q}_{LS}) \approx \mathbf{f}(x,\ \boldsymbol{q}_p) + p_\xi\ \mathbf{W} \qquad (22)$$

i.e. it is approximately the sum of $E(\mathbf{Y}_p \mid x)$ and of the projection of $\mathbf{W}$ on $\mathfrak{M}(\xi)$, as illustrated in Figure 3. If $K(\mathbf{W}) = \sigma^2\ I_N$ (homoscedasticity), the variance of the model output, that is the LS estimator of the regression, for an input $\mathbf{x}$ is approximately:

$$var\left(f(\mathbf{x},\ \boldsymbol{Q}_{LS})\right) \approx \sigma^2\ \boldsymbol{x}^T\left(x^T\ x\right)^{-1}\ \boldsymbol{x} \qquad (23)$$

neuron exchanges, as well as sign flips for odd activation functions like the hyperbolic tangent [Sussmann 92], we will legitimately consider the neighborhood of one of these values only.

In the following, approximation (23) will be termed "the LTE approximation" of the model output variance. Let $\mathbf{R}$ denote the LS residual vector $\mathbf{R} = \mathbf{Y}_p - \mathbf{f}(x,\ \boldsymbol{Q}_{LS})$, thus:

$$\mathbf{R} \approx \left(I_N - p_\xi\right)\mathbf{W} \qquad (24)$$

Under the assumption of appropriate regularity conditions on f, and for large N, the curvature of the solution surface[7] $\mathfrak{M}(\mathbf{f}(x,\ \boldsymbol{q}))$ is small; thus, using (24), one obtains the asymptotically (i.e. when $N \to \infty$) unbiased estimator $S^2 = \dfrac{\mathbf{R}^T\ \mathbf{R}}{N-q}$ of σ^2. In (23), the matrix ξ takes the place of the matrix x in the linear case. But, as opposed to x, ξ is unknown since it is a function of the unknown parameter $\boldsymbol{q}_p$. The (N,q) matrix x may be approximated by $z = \left[\mathbf{z}^1\ \mathbf{z}^2\ \ldots\ \mathbf{z}^N\right]^T$ where $\mathbf{z}^k = \dfrac{\partial f(\mathbf{x}^k,\ \boldsymbol{q})}{\partial \boldsymbol{q}}\bigg|_{\boldsymbol{q}=\boldsymbol{q}_{LS}}$, that is:

$$z_r^k = \frac{\partial f(\mathbf{x}^k,\ \boldsymbol{q})}{\partial \theta_r}\bigg|_{\boldsymbol{q}\ =\ \boldsymbol{q}_L} \qquad (25)$$

In the following, we assume that $\mathrm{rank}(z) = q$. Like the matrix ξ, the vector $\boldsymbol{x}$ is not available, and its value may be approximated by:

$$\mathbf{z} = \frac{\partial f(\mathbf{x},\ \boldsymbol{q})}{\partial \boldsymbol{q}}\bigg|_{\boldsymbol{q}=\boldsymbol{q}_{LS}} \qquad (26)$$

From (23), (25) and (26), the LTE estimate of the variance of the LS estimator of the regression for an input $\mathbf{x}$ is thus:

$$\widehat{var\left(f(\mathbf{x},\ \boldsymbol{Q}_{LS})\right)}_{LTE} = s^2\ \mathbf{z}^T\left(z^T\ z\right)^{-1}\ \mathbf{z} \qquad (27)$$

III.2. Approximate confidence intervals for a nonlinear regression

If $\mathbf{W} \to \mathbf{N}_N(\mathbf{0},\ \sigma^2\ I_N)$, and for large N, it follows from the above relations and from linear LS theory [Seber & Wild 1989] that:

ThNL1. $$\boldsymbol{Q}_{LS} \ \sim\!\!\to \mathbf{N}_q\left(\boldsymbol{q}_p,\ \sigma^2\left(\xi^T\ \xi\right)^{-1}\right) \qquad (28)$$

ThNL2. $$\frac{\mathbf{R}^T\ \mathbf{R}}{\sigma^2} \ \sim\!\!\to \chi^2_{N-q} \qquad (29)$$

ThNL3. *$\boldsymbol{Q}_{LS}$ is approximately statistically independent from $\mathbf{R}^T\ \mathbf{R}$.*

[7] The curvature is usually decomposed in two components: i) the intrinsic curvature, which measures the degree of bending and twisting of the solution surface $\mathfrak{M}(\mathbf{f}(x,\ \boldsymbol{q}))$, and ii) the parameter-effects curvature, which describes the degree of curvature induced by the choice of the parameters $\boldsymbol{q}$.

Using (23) and (28), let us again build a quasi normalized and centered gaussian variable where both $E(Y_p \mid \mathbf{x})$ *and* σ *appear:*

$$\frac{f(\mathbf{x}, \boldsymbol{Q}_{LS}) - E(Y_p \mid \mathbf{x})}{\sigma \sqrt{\boldsymbol{x}^T (\xi^T \xi)^{-1} \boldsymbol{x}}} \sim\!\!\to \mathbf{N}(0, 1) \qquad (30)$$

Thus, the variable (29) being approximately independent from (30) according to ThNL3, we have:

$$\frac{f(\mathbf{x}, \boldsymbol{Q}_{LS}) - E(Y_p \mid \mathbf{x})}{S \sqrt{\boldsymbol{x}^T (\xi^T \xi)^{-1} \boldsymbol{x}}} \sim\!\!\to Student\,(N - q) \qquad (31)$$

A 100(1–α*)% approximate CI for* $E(Y_p \mid \mathbf{x})$ *is thus:*

$$f(\mathbf{x}, \boldsymbol{q}_{LS}) \pm t_{N-q}(\alpha)\, s \sqrt{\mathbf{z}^T (z^T z)^{-1} \mathbf{z}} \qquad (32)$$

Note that, when N is large, the Student distribution is close to the normal distribution, and thus $t_{N-q}(\alpha) \approx n(\alpha)$*, where n is the inverse of the normal cumulative distribution.*

Like in the linear case, (32) allows to compute a CI at any input **x** *of interest, which gives much more information than the value of the mean variance estimate over the data set: as a matter of fact, the latter always approximately equals* $s^2 \frac{q}{N}$.

From a practical point of view, the construction of a CI for a neural model output at any input **x** *of interest involves once and for all the computation of the matrix z, that is the* $N \times q$ *partial derivatives of the model output with respect to the parameters evaluated at* $\boldsymbol{q}_{LS}$ *for the data inputs* $\{\mathbf{x}^k\}_{k=1 \text{ to } N}$*, and, for each* **x** *value, that of* **z***, i.e. the derivatives at* **x***. In the case of a neural model, these derivatives are easily obtained with the backpropagation algorithm.*

All the previous results and the above considerations are valid provided an absolute minimum of the cost function (18) is reached. In real-life, several estimations of the parameters must thus be made starting from different initial values, the estimate corresponding to the lowest minimum being kept in order to have a high probability to obtain an absolute minimum. In the examples of this work, the algorithm used for the minimization of the cost function is the Levenberg algorithm, as described for example in [Bates & Watts 1988], and several trainings are performed. The probability of getting trapped in a relative minimum increasing with the number of parameters of the network and decreasing with the size of the data set, the number of trainings is chosen accordingly.

III.3. Quality of the approximate confidence intervals

III.3.1. Theoretical analysis

The quality of the approximate CI depends essentially on the curvature of the solution surface $\mathcal{M}(\mathbf{f}(x, \boldsymbol{q}))$*, which depends on the regularity of f and on the value of N. In practice, f is often regular enough for a first-order approximation to be satisfactory, provided that N is large enough. Thus, if N is large: (i) as in the linear case, the estimator of the noise variance* S^2 *is unbiased, and the difference between* s^2 *and* σ^2 *is only due to the particular values of the noise; (ii) the variance of* $f(\mathbf{x}, \boldsymbol{Q}_{LS})$ *is small, and* $\boldsymbol{q}_{LS}$ *is likely to be close to* $\boldsymbol{q}_p$*: z and* **z** *are thus likely to be good approximations of respectively x and* ***x*** *A reliable estimate of a CI may thus be obtained from the LTE variance estimate (27). On the other hand, if N is too small: (i) as opposed to the linear case, even if the assumed model is true, the estimator of the noise variance* S^2 *is biased; (ii) the variance of* $f(\mathbf{x}, \boldsymbol{Q}_{LS})$ *is large, and* $\boldsymbol{q}_{LS}$ *is likely to differ from* $\boldsymbol{q}_p$*: z and* **z** *risk to be poor approximations of x and* ***x*** *Thus, if N is diagnosed as too small, one cannot “have confidence” in the confidence intervals (32), and additional data should be gathered.*

III.3.2. Quantitative analysis

As detailed for example in [Bates & Watts 1988] [Seber & Wild 1989] [Antoniadis et al. 1992], different measures of the curvature can be computed, and can be used in each particular case to evaluate the accuracy of the LTE. In section IV.3 dealing with neural network modeling, we give indications on how to judge if N is large enough for the approximate CI to be accurate.

In order to evaluate the accuracy of the LTE variance estimate (27) when dealing with simulated processes, we introduce an estimate of the unknown true variance of $f(\mathbf{x}, \boldsymbol{\theta}_{LS})$ that is not biased by curvature effects, the reference variance estimate $s^2_{ref}(\mathbf{x})$. This estimate is computed on a large number M of other sets of N outputs corresponding to the inputs of the training set, and whose values are obtained with different realizations (simulated values) of the noise $\mathbf{W}$*. The i-th LS estimate* $f(\mathbf{x}, \boldsymbol{\theta}^{(i)}_{LS})$ *of* $E(Y_p \mid \mathbf{x})$ *is computed with data set i (i=1 to M), and the reference estimate of the variance at input* $\mathbf{x}$ *is computed as:*

$$s^2_{ref}(\mathbf{x}) = \frac{1}{M}\sum_{i=1}^{M}\left(f\left(\mathbf{x}, \boldsymbol{\theta}^{(i)}_{LS}\right) - \left\langle f(\mathbf{x})\right\rangle\right)^2 \text{, where } \left\langle f(\mathbf{x})\right\rangle = \frac{1}{M}\sum_{i=1}^{M} f\left(\mathbf{x}, \boldsymbol{\theta}^{(i)}_{LS}\right) \qquad (33)$$

In the nonlinear case, we thus use three notions related to the (true) variance of $f(\mathbf{x}, \boldsymbol{\theta}_{LS})$*: 1) the LTE variance approximation (23), which is a good approximation of the true variance if the curvature is small, as we show in section III.4.3, and which can be computed only when the process is simulated; 2) the LTE variance estimate (27), which is the estimate that can be computed in real-life; 3) the reference variance estimate (33), which tends to the true variance when M tends to infinity because it is not biased by curvature effects, and which can be computed only when the process is simulated.*

III.4. Illustrative examples

Since we are concerned with neural models, and since, in this section, the assumed model is true, the following examples bring into play "neural" processes, that is to say processes whose regression function is the output of a neural network; the more realistic case of arbitrary processes for whom a family of nonlinear functions (a neural network with a given architecture) containing the regression is unknown is tackled in the next section.

III.4.1. Example of a simulated "neural" SISO process (process #2)

We consider a SISO process simulated by a neural network with one hidden layer of two hidden neurons with hyperbolic tangent activation function and a linear output neuron:

$$y^k_p = \theta_{p_1} + \theta_{p_2}\tanh\left(\theta_{p_3} + \theta_{p_4}x^k\right) + \theta_{p_5}\tanh\left(\theta_{p_6} + \theta_{p_7}x^k\right) + w^k \quad k=1 \text{ to } N \qquad (34)$$

We take $\boldsymbol{\theta}_p = [1;\ 2;\ 1;\ 2;\ -1;\ -1;\ 3]^T$, $\sigma^2 = 10^{-2}$, *N = 50. The inputs* $\{x^k\}$ *of the data set are uniformly distributed in [-3; 3], as shown in Figure 4a. The family of functions* $\left\{\theta_1 + \theta_2\tanh\left(\theta_3 + \theta_4 x\right) + \theta_5\tanh\left(\theta_6 + \theta_7 x\right),\ \boldsymbol{\theta} \in \mathbb{R}^7\right\}$ *is considered, that is the assumed model is true, and we choose a confidence level of 99%* ($t_{43}(1\%) = 2.58$)*. The minimization of the cost function with the Levenberg algorithm leads to* $s^2 = 1.02\ 10^{-2}$*. Figure 4b displays the LTE estimate of the variance of* $f(\mathbf{x}, \boldsymbol{\theta}_{LS})$ *(27), and the reference estimate (33) computed over M = 10 000 sets. Figure 4a shows the regression, the data set used for the LS estimation, and the corresponding model output and 99% CI (32). The model being true and the size N = 50 of the data set being relatively large with respect to the number of parameters and to the noise variance, we observe that: (i)* $s^2 \approx \sigma^2$*; (ii) the model output is close to the regression, leading to good approximations of* x *and of* $\boldsymbol{x}$ *by z and* $\mathbf{z}$*. Thus, (i) and (ii) lead to an accurate LTE estimate of the variance, and hence of the CI.*

III.4.2. Example of a simulated "neural" MISO process (process #3)

We consider a MISO process simulated by a neural network with two inputs, one hidden layer of two "tanh" hidden neurons and a linear output neuron:

$$\begin{aligned} y^k_p = {} & \theta_{p_1} + \theta_{p_2}\tanh\left(\theta_{p_3} + \theta_{p_4}x^k_1 + \theta_{p_5}x^k_2\right) \\ & + \theta_{p_6}\tanh\left(\theta_{p_7} + \theta_{p_8}x^k_1 + \theta_{p_9}x^k_2\right) + w^k \quad k=1 \text{ to } N \end{aligned} \qquad (35)$$

We take $\boldsymbol{\theta}_p = [1;\ 1;\ 0;\ 1;\ -1;\ -2;\ 0;\ 1;\ 1]^T$, $\sigma^2 = 10^{-1}$, *N = 100. The inputs* $\{x^k_1\}$ *and* $\{x^k_2\}$ *of the data set are uniformly distributed in [-3; 3]. As for process #2, the assumed model is true, i.e. the neural network associated to (35) is used; the minimization of the cost function with the Levenberg algorithm leads to* $s^2 = 9.73\ 10^{-2}$*. Figure 5a shows the inputs of the data set and the regression; Figure 5b displays the LTE estimate of the* *variance of $f\left(\mathbf{x}, \boldsymbol{Q}_{LS}\right)$ (27); Figure 5c displays the difference between the reference variance estimate (33) computed over $M = 10\,000$ sets, and the LTE variance estimate (27). Since s^2 is slightly smaller than s^2, the variance is globally slightly underestimated. But except in the domain around the corner (3, 3) where the density of the inputs is lower, and where the slope of the output surface is steep, the LTE variance estimate is satisfactory. A reliable estimate of the CI may thus be obtained.*

III.4.3 Accuracy of the LTE variance approximation (processes #2 and #3)

Let us now show on the example of processes #2 and #3 that the curvature of the solution surface is small enough for the LTE approximation of the variance (23) to be satisfactory. For both processes, we have computed approximation (23), using the values of x and of $\boldsymbol{x}$ (at $\boldsymbol{q}_p$) and the values of s^2 used for the noise simulation. As shown in Figures 6a and 6b for process #2, the LTE approximation of the variance (23) is very close to the reference variance estimate. As a matter of fact, the difference between them (Figure 6b) is only due to the curvature, which is small since $N = 50$ is large with respect to the complexity of the regression. Expression (23) also leads to satisfactory results in the case of process #3 as shown in Figures 7a and 7b ($N = 100$, two inputs). This tends to show that a first-order approximation of the variance is often sufficient, and that is not worth to bother with a higher-order approximation. In [Seber & Wild 89], a quadratic approximation of the LS estimator using the curvature of the solution surface is introduced. This approximation uses arrays of projected second derivatives, the intrinsic and parameter-effects curvatures arrays; but their presentation is beyond the scope of this paper.

IV. CONFIDENCE INTERVALS FOR NEURAL NETWORKS

In the previous sections, the model used for the construction of CIs is true. For real-world black-box modeling problems however, a family of functions which contains the regression is not known a priori. The first task is thus to select the less complex family of functions which contains a function approximating the regression to a certain degree of accuracy in the input domain defined by the data set. In practice, several families of increasing complexity (for example neural networks with one layer of an increasing number n_h of hidden units, and a linear output neuron) are considered, and the data set is used both to estimate their parameters, and to perform the selection between the candidates. In order to retain the less complex family containing a good approximation of the regression, it is of interest to perform the selection only between neural candidates which are not unnecessarily large, and which are (that is their matrix z is) sufficiently well-conditioned to allow the computation of the approximate CI (32). We propose to discard too large models by a systematic detection of ill-conditioning, and to perform the selection among the approved, i.e. well-conditionned models using an approximate value of their LOO score whose computation does not require further trainings. Both the ill-conditioning detection and the estimation of the LOO score of a neural candidate are based on the LTE of its output.

IV.1. Ill-conditioning detection for model approval

A too large neural model, trained up to convergence with a simple LS cost-function, will generally overfit. Overfitting is often avoided by using early stopping of the training algorithm or by adding regularization terms in the cost function, e.g. “weight decay” [Bishop 1995]. Unfortunately, since only the estimator whose value corresponds to an absolute minimum of the quadratic cost function (18) without weight decay terms is unbiased, both early stopping and weight decay introduce a *bias in the estimation of the regression: the corresponding estimates thus lead to questionable CIs for the regression.*

To detect and discard too large networks, we propose, after the training of each candidate up to a (hopefully) absolute minimum of the cost function (18), to check the conditioning of their matrix z (see [Rivals & Personnaz 1998]). The fact that z be ill-conditioned is the symptom that some parameters are useless, since the elements of z represent the sensibility of the model output with respect to the parameters. A typical situation is the saturation of a "tanh" hidden neuron, a situation which generates in the matrix z a column of +1 or –1 that corresponds to the parameter between the output of the saturated hidden neuron and the linear output neuron, and columns of zeros that correspond to the parameters between the network inputs and the saturated hidden neuron[8]. In practice, we propose to perform a singular value factorization of z, and to compute its condition number, that is the ratio of its largest to its smallest singular value, see for example [Golub & Van Loan 1983]. The matrix z can be considered as very ill-conditioned when its condition number reaches the inverse of the computer precision, which is of the order of 10^{-16}.

Furthermore, in order to be able to compute the approximate CI (32) which involve $(z^T z)^{-1}$, the cross-product Jacobian matrix $z^T z$ must also be well-conditioned. Since the condition number of $z^T z$ is the square of the condition number of z, the networks whose matrix z has a condition number much larger than 10^8 cannot be approved.

There are other studies of the ill-conditioning of neural networks, but they deal with their training rather than with their approval, like in [Zhou and Si 1998] where an algorithm avoiding the Jacobian rank deficiency is presented, or [Saarinen et al. 1993] where the Hessian rank deficiency is studied during training. In our view, rank deficiency is not very relevant during the training since, with a Levenberg algorithm, the matrix to be "inverted" is made well-conditioned by the addition of a scalar matrix $\lambda\, I_q$, $\lambda > 0$, to the cross-product Jacobian.

[8] *Such a situation might also correspond to a relative minimum; to check the conditioning of z helps thus also to discard neural networks trapped in relative minima, and leads to retrain the neural candidate starting from different initial weights.*

IV.2. Approximate leave-one-out scores for model selection

The selection among the networks which have been approved can be performed with statistical tests for example [Urbani et al. 1994] [Rivals & Personnaz 1998]. Another approach, cross validation, consists in partitioning the data set in training and test sets, and in selecting the smallest network leading to the smallest mean square error on the test sets[9]. One of the drawbacks of cross validation is to require a successful training of the candidate models on many test sets, that is N successful trainings in the case of LOO cross validation. Let us denote by e^k the error obtained on the left out example k with the model trained on the N-1 remaining examples (k-th test set). In this section, we derive an approximate expression of e^k, expression which allows an economic estimation of the LOO score without performing these N time-consuming trainings of each candidate network, as proposed in [Monari 1999] [Monari & Dreyfus].

In the case of a linear model, it is well-known [Efron & Tibshirani 1993] that the k-th LOO error e^k can be directly derived from the corresponding residual r^k:

$$\begin{cases} e^k = \dfrac{r^k}{1 - [p_x]_{kk}} & \text{if } [p_x]_{kk} < 1 \\ e^k = r^k = 0 & \text{if } [p_x]_{kk} = 1 \end{cases} \quad k=1 \text{ to } N \tag{36}$$

where, we recall, p_x denotes the orthogonal projection matrix on the range of x. Expression (36) holds irrespective of whether or not the assumed model is true.

[9] *Note that statistical tests may advantageously be used complementarily to cross validation in order to take a decision [Rivals & Personnaz 1999]; these tests can also be established by applying LS theory to the LTE of nonlinear models [Bates & Watts 1988], but this exceeds the scope of this paper.*

In the case of a nonlinear model, we show (see Appendix 2) that a useful approximation of the k-th LOO error can be obtained using the LTE of the model output at $\boldsymbol{q}_{LS}$:

$$e^k \approx \frac{r^k}{1-[p_z]_{kk}} \quad k=1 \text{ to } N \tag{37}$$

where p_z denotes the orthogonal projection matrix on the range of z. The approximation (37) is thus similar to (36)[10]. In the case where (numerically) $[p_z]_{kk} = 1$, we choose to take $e^k = r^k$ (the residual is not necessarily zero). Like in the linear case, expression (37) holds independently on the assumed model being true or not. Hence the LOO score:

$$\text{LOO score} = \frac{1}{N}\sum_{k=1}^{N}\left(e^k\right)^2 \tag{38}$$

This LOO score can be used as an estimate of the mean square performance error, and we thus denote it as MSPE, as opposed to $\frac{2}{N}J(\boldsymbol{q}_{LS})$, the mean square training error (MSTE). The interested reader will find in [Monari 1999] [Monari & Dreyfus] a systematic model selection procedure based on both the approximate LOO score and the distribution of the values of the $\{[p_z]_{kk}\}$. Nevertheless, another performance measure could be chosen as well (a 10-fold cross validation score, a mean square error on an independent set, etc.).

IV.3. Accuracy of the approximate confidence intervals

The quality of the selected model $f(\mathbf{x}, \boldsymbol{q}_{LS})$, and thus of the associated approximate CI, depends essentially on the size N of the available data set with respect to the complexity of the unknown regression function and to the noise variance s^2:

1. *N is large: it is likely that the selected family $\{f(\mathbf{x}, \boldsymbol{q}), \boldsymbol{q} \in \mathbb{R}^q\}$ contains the regression $E(Y_p \mid \mathbf{x})$, i.e. that the LS estimator is asymptotically unbiased, that the model $f(\mathbf{x}, \boldsymbol{q}_{LS})$ is a good approximation of $E(Y_p \mid \mathbf{x})$ in the domain of interest, and that the curvature is small. In this case, reliable CIs can be computed with (32).*
2. *N is small: it is likely that the selected family $\{f(\mathbf{x}, \boldsymbol{q}), \boldsymbol{q} \in \mathbb{R}^q\}$ is too small[11] to contain $E(Y_p \mid \mathbf{x})$, i.e. that the LS estimator is biased, and that the model $f(\mathbf{x}, \boldsymbol{q}_{LS})$ thus underfits. The approximate CIs are thus questionable, and additional data should be gathered.*

A good indicator of whether the data set size N is large enough or not is the ratio MSPE/MSTE of the selected candidate: if its value is close to 1, then N is probably large enough, whereas a large value is the symptom of a too small data set size, as shown in Figure 8 (and as illustrated numerically in the following examples).

IV.4. Example of a simulated nonlinear SISO process (process #4)

This first example is based on a simulated process. Like in the previous sections, a reference estimate of the variance of the output of a neural network is made, using M = 1000 other sets; in order to ensure that an absolute minimum is reached on each of the M sets, five to thirty trainings (depending on the network size) with the Levenberg algorithm for different initializations of the weights are performed, and the weights giving the smallest value of the cost function (18) are kept. We consider the SISO process simulated with:

$$y_p^k = \text{sinc}\left(2\left(x^k+5\right)\right) + w^k \quad k=1 \text{ to } N \tag{39}$$

where sinc denotes the cardinal sine function; we take $s^2 = 10^{-2}$.

First, a data set of N = 200 input-output pairs is computed, with input values uniformly distributed in [-5; 5]. Since a family of nonlinear functions (a neural network

[10] An expression similar to (37) is proposed in [Hansen & Larsen 1996], but unfortunately, it is not valid even in the linear case.

[11] It will generally not be too large since the approval procedure proposed in section IV.1 prevents from selecting a neural network with useless parameters.

with a given architecture) containing the regression is not known a priori, neural networks with a linear output neuron and a layer of n_h "tanh" hidden neurons are trained. The numerical results are summarized in Table 1. We list the number of parameters q, the MSTE (i.e. the smallest MSTE obtained with the network for its different random weight initializations), the condition number of z, and, if the latter is not too large, the MSPE (corresponding approximate LOO score computed with (37) and (38)) and the ratio MSPE/MSTE. The candidates with more than 6 hidden neurons cannot be approved, because cond(z) >> 10^8: for n_h = 7, cond(z) = 10^{11}. The optimal number of neurons $n_h^{opt}(200) = 4$ is selected on the basis of the MSPE. The fact that the corresponding ratio MSPE/MSTE is close to 1 is the symptom that N is large enough, so that the selected family of networks contains a good approximation of the regression, and that the curvature is small (case 1 of section IV.3). The results obtained for the selected neural network are shown in Figure 9. The model output is close to the regression, the LTE variance estimate (27) is close to the reference variance estimate (33), and the CI is thus accurate.

Second, a data set of N = 30 input-output pairs is computed, the numerical results being summarized in Table 2. The data set being much smaller, the candidates cannot be approved as soon as n_h > 4: for n_h = 5, cond(z) = 10^{15}. The optimal number of neurons $n_h^{opt}(30) = 2$ is selected on the basis of the MSPE. The ratio MSPE/MSTE of the selected network equals 2.1, symptom that N is relatively small, and that the selected family of networks is likely not to contain the regression (case 2 of section IV.3). The results obtained for the selected neural network are shown in Figure 10. The family of functions implemented by a network with two hidden units is obviously too small to contain a good approximation of the regression, and though the estimate of the output variance is good (it is close to the reference variance estimate), since the output of the neural network differs from the regression, the CI is less accurate than in the case where N = 200. Note that in the input domain [0, 5] where the model underfits, though there is a higher concentration of training examples around 1 and 3, the variance remains constant and low. This is due to the fact that, in this domain, the model output is insensitive to most parameters of the network (this is usually the case when, like here, the output of a network does not vary[12]): the elements of **z***'s in this domain are thus constant and small, hence a small and constant variance at the corresponding* **x***'s.*

IV.5. Industrial modeling problem

We apply here the presented methodology (LS parameter estimation, model approval, model selection, CI construction) to an industrial example first tackled in [Rivals & Personnaz 1998], that is the modeling of a mechanical property of a complex material from three structural descriptors. We have been provided with a data set of N = 69 examples; the inputs and outputs are normalized for the LS estimations. Thanks to repetitions in the data, and assuming homoscedasticity, the "pure error mean square" [Draper & Smith 1998] gives a good estimate of the noise variance: $\widehat{\sigma^2} = 3.38\ 10^{-2}$. Using this reliable estimate, statistical tests establish the significance of two inputs. An affine model with these two inputs gives the estimate $s^2 = 2.38\ 10^{-1}$ of the variance, hence the necessity of nonlinear modeling.

Neural networks with a linear output neuron and a layer of n_h "tanh" hidden neurons are trained. The numerical results are summarized in table 3. It shows that the candidates with more than 3 hidden neurons cannot be approved: for n_h = 4, cond(z) = 10^{12}. The optimal number of neurons $n_h^{opt}(69) = 2$ is selected on the basis of the MSPE. The fact that the corresponding ratio MSPE/MSTE equals 1.3 indicates that N is large enough, so that the selected family of networks contains probably a

[12] The output of a neural network with one layer of tanh hidden units remains constant in a given domain of its inputs when the "tanh" activation functions of all hidden units are saturated in this domain: the output of the network is thus insensitive to all the parameters of the hidden units.

good approximation of the regression, and that the curvature is small (case 1 of section IV.3). The function implemented by the selected network is shown in Figure 11.

The $N = 69$ outputs of the training set are presented in increasing order in Figure 12a, and the corresponding residuals and approximate LOO errors in Figure 12b: both appear quite uncorrelated and homoscedastic. A CI with a level of significance of 95% is then computed with (32); the half width of the 95% CI on the $N = 69$ examples of the data set is shown in Figure 12c. In order to check the confidence which can be attached to the model, the variance of its output must be examined in the whole input domain of interest. Figure 13 shows the isocontours of the LTE estimate of the standard deviation of the model output $s\sqrt{\mathbf{z}^T (z^T z)^{-1}\, \mathbf{z}}$ in the input domain defined by the training set. The computation of the LTE variance estimate thus allows not only to construct a CI at any input of interest, but also to diagnose that, at the top right corner of the input domain, the model standard deviation is larger than that of the noise itself (the highest isocontour value equals that of the estimate of the noise standard deviation $s = 1.39\ 10^{-1}$). Little confidence can thus be attached to the model output in this input domain, where more data should be gathered. On the opposite, there is a large region on the left of the diagram where there are very few training examples, but where the LTE estimate of the standard deviation is surprisingly rather small: like for the modeling of process #4 in section IV.4, this is due to the fact that the model output is little sensitive to most parameters of the network in this region (the model output varies very little, see Figure 11).

V. COMPARISONS

In this section, we discuss the advantages of the LS LTE approach to the construction of construction intervals for neural networks with respect to other analytic approaches and to the bootstrap methods, and compare them on simulated examples.

V.1. Comparison to other analytic approaches

Maximum likelihood approach

In the case of gaussian homoscedastic data, likelihood theory leads to the same approximate variance (23), but two different estimators of it are commonly encountered (see Appendix 1.2):

$$\widehat{var\left(f\left(\mathbf{x},\, \boldsymbol{Q}_{LS}\right)\right)}_{LTE} = \widehat{s^2}\; \mathbf{z}^T \left(z^T z\right)^{-1} \mathbf{z}$$

i.e. the same estimate as (27), and also:

$$\widehat{var\left(f\left(\mathbf{x},\, \boldsymbol{Q}_{LS}\right)\right)}_{Hessian} = \widehat{s^2}\; \mathbf{z}^T \left[h\left(\boldsymbol{q}_{LS}\right)\right]^{-1} \mathbf{z} \qquad (40)$$

which necessitates the computation of the Hessian. Efficient methods for computing the Hessian are presented in [Buntine & Weigend 1994].

Bayesian approach

The Bayesian approach is an alternative approach to sampling theory (or frequentist approach) for modeling problems, and also leads to the design of CIs. These two approaches are conceptually very different: the Bayesian approach treats the unknown parameters as random variables, whereas they are considered as certain in the frequentist approach. Nevertheless, as presented for example in [MacKay 1992a] [MacKay 1992b] [Bishop 1995], the Bayesian approach leads to a posterior distribution of the parameters with a covariance matrix whose expression is very similar to that of the covariance matrix of the least-squares estimator of the *parameters, and thus to CIs which are similar to those presented in this paper. We thus make a brief comparison between the CIs these two approaches lead to.*

The most important difference is that the estimator which is considered here is the one whose estimate minimizes the cost function (18), whereas in the Bayesian approach, a cost-function with an additional weight-decay regularization term is minimized; the presence of this weight-decay term stems from the assumption of a gaussian prior for the parameters.

Nevertheless, the least squares cost function (18) can be seen as the limit where the regularization term is zero, which corresponds to an uninformative prior for the parameters. In this case (that is (18) is minimized as in this paper), there is another small difference in the Bayesian approach as presented in [MacKay 1992a] [MacKay 1992b] [Bishop 1995]. Under hypotheses which we cannot detail here, the Bayesian approach leads to a posterior parameter distribution with approximate covariance matrix $s^2\left[h\left(\boldsymbol{q}_{LS}\right)\right]^{-1}$, $h\left(\boldsymbol{q}_{LS}\right)$ *being the Hessian of the cost function evaluated at the most probable value of the parameter, that is here* $\boldsymbol{q}_{LS}$. *A LTE of the estimator output leads then to the following estimate of its variance at input* **x**:

$$\widehat{var\left(f\left(\mathbf{x},\, \boldsymbol{Q}_{LS}\right)\right)}_{Hessian} = \widehat{s^2}\, \mathbf{z}^T\left[h\left(\boldsymbol{q}_{LS}\right)\right]^{-1}\mathbf{z}$$

i.e. it also leads to estimate (40).

Sandwich estimator

The sandwich estimate of the variance of a nonlinear model output can be derived in various frameworks (a possible derivation in the frequentist approach is given in Appendix 1.3):

$$\widehat{var\left(f\left(\mathbf{x},\, \boldsymbol{q}_{LS}\right)\right)}_{sandwich} = \widehat{s^2}\, \mathbf{z}^T\left[h\left(\boldsymbol{q}_{LS}\right)\right]^{-1} z^T z\left[h\left(\boldsymbol{q}_{LS}\right)\right]^{-1}\mathbf{z} \qquad (41)$$

The sandwich estimator is known to be robust to model incorrectness, i.e. the assumptions about the noise are incorrect or the considered family of functions is too small, see for example [Efron & Tibshirani 1993] [Ripley 1995].

Numerical comparison (processes #5 and #6)

Here, we perform a numerical comparison of the three variance estimates considered above on a very simple example. We consider a SISO process simulated by a single "tanh" neuron:

$$y_p^k = \tanh\left(q_{p_1} + q_{p_2}\, x^k\right) + w^k \quad k=1 \text{ to } N \qquad (42)$$

with $\sigma^2 = 0.01$, $N = 30$. *For this comparison, the noise variance* s^2 *is estimated with* s^2 *in the three (LTE, Hessian, sandwich) output variance estimates.*

We first simulate a process with: $\theta_{p_1} = 0$, $\theta_{p_2} = 1$ *(process #5). The corresponding results are shown in Figure 14. The variance reference estimate is computed on M = 10 000 data sets. The LTE approximation (23) of the variance is almost perfect. The LTE (27), Hessian (40), and sandwich (41) estimates are comparable: the parameter estimates being accurate (*$\theta_{LS_1} = 3.63\ 10^{-2}$, $\theta_{LS_2} = 0.996$*), the fact that they are overestimated is almost only due to the noise variance estimate* $s^2 = 1.32\ 10^{-2}$. *Nevertheless, the shape of the LTE estimate is closer to the reference estimate than that of the two others.*

We then simulate a process with: $\theta_{p_1} = 0$, $\theta_{p_2} = 5$ *(process #6). The corresponding results are shown in Figure 15. The function being steeper, the curvature is larger, and the LTE approximation (23) of the variance is a little less accurate. The three estimates are still very similar but, here, their overestimation is due not only to the noise variance estimate* $s^2 = 1.25\ 10^{-2}$, *but also to the bias of the parameter estimates (*$\theta_{LS_1} = 3.79\ 10^{-2}$, $\theta_{LS_2} = 6.58$*).*

The computational cost of the LTE estimate being lower (is does not necessitate the computation of the Hessian matrix), there is no reason to prefer one of the two other estimates. As a matter of fact, since the Hessian depends on the data set, it is the realization of a random matrix. Thus, in the maximum likelihood as well as in the Bayesian approach, it is often recommended to take the expectation of the Hessian, and to evaluate it at the available $\boldsymbol{q}_{LS}$, *i.e. to replace it by the cross-product Jacobian* $z^T z$ *[Seber & Wild 1989]: estimates (40) and (41) then reduce to estimate (27). As* *mentioned above, the sandwich variance estimator is known to be robust to model incorrectness, a property which is not tested with this simple setting, but this is beyond the scope of this paper.*

V.2. Comparison to bootstrap approaches

The bootstrap works by creating many pseudo replicates of the data set, the bootstrap sets, and reestimating the LS solution (retraining the neural network) on each bootstrap set; the variance of the neural model output, and the associated CI, are then computed over the trained networks, typically a hundred [Efron & Tibshirani 1993]. In the "bootstrap pairs approach" for example, a bootstrap set is created by sampling with replacement from the data set [Efron & Tibshirani 1993]. The first advantage of the LS LTE approach is to require only one successful training of the network on the data set to compute the LTE estimate of the variance of its output, whereas the bootstrap methods require a hundred successful trainings of the network on the different bootstrap sets.

Studies on bootstrap where only one training with a random initialization of the weights was performed for each bootstrap set show a pathological overestimation of the variance. This can be seen in [Tibshirani 1996], examples 2 and 3; but the overestimation of the bootstrap is not detected in this work because the reference estimate is also overestimated for the same reasons (one single training per set). As pointed out in [Refenes et al. 1997], a way to reduce this overestimation is to start each training on a bootstrap set with the weights giving the smallest value of the cost function (18) (that is on the original data set); but even so, the bootstrap method becomes untractable for large networks, and/or for multi input processes.

The claim that bootstrap methods are especially efficient for problems with small data sets (see [Heskes 1997] for example) may be subject to criticism. As an illustration, the variance was estimated for process #2 using the bootstrap pairs approach on 300 bootstrap sets, the network weights being initialized twice for each training, once with the true ones, and once with those obtained by training the network on the whole data set. As shown in Figure 16, though the size of the data set is not very small ($N = 50$), the bootstrap variance estimate is far away from the reference estimate. Increasing the number of bootstrap sets up to 1000 did not improve the variance estimate.

In fact, the bootstrap is especially suited to the estimation of the variance of estimators defined by a formula, like for example an estimator of a correlation coefficient [Efron & Tibshirani 1993]: for each bootstrap set, an estimate is computed using the formula, and the estimate of the variance is easily obtained. But the bootstrap is definitely not the best method if each estimation associated to a bootstrap set involves an iterative algorithm like the training of a neural network, which is the case for the construction of a CI with a neural model. However, if the data set is large enough, and if the training time is considered unimportant, the bootstrap pairs approach is a solution in the case of heteroscedasticity (that is $K(\mathbf{W})$ is not scalar anymore), whereas the LS LTE approach, as well as the "bootstrap residuals" approach [Efron & Tibshirani 1993], are no longer valid.

VI. CONCLUSION

We have given a thorough analysis of the LS LTE approach to the construction of CIs for a nonlinear regression using neural network models, and put emphasis on its enlightening geometric interpretation. We have stressed the underlying assumptions, in particular the fact that the approval and selection procedures must have led to a parsimonious, well-conditioned model containing a good approximation of the regression. Our whole methodology (LS parameter estimation, model approval, model selection, CI construction) has been illustrated on representative examples, bringing into play simulated processes and an industrial one.

We have also shown that, as opposed to the computationally intensive bootstrap methods, the LS LTE approach to the estimation of CIs is both accurate and economical in terms of computer power, and that it leads to CIs which are comparable to those obtained by other analytic approaches under similar assumptions, at a lower computational cost.

A rigorous assessment of the accuracy of the results obtained with the LS LTE approach, as well as with any statistical approach dealing with nonlinear models and assuming the local planarity of the solution surface, remains an open problem: it could be enlightened by a specific study of the curvature of the solution surface of neural networks.

ACKNOWLEDGEMENTS

We thank an anonymous reviewer whose constructive comments contributed to improve the quality of this paper. We also thank Howard Gutowitz whose advice we greatly appreciated.

APPENDIX 1. ESTIMATES OF A NONLINEAR MODEL OUTPUT VARIANCE

In order to make this paper self-contained, we provide derivations of the different variance estimates.

A1.1. LTE variance estimate in sampling theory

The well-known approximation [Seber & Wild 1989] we use in this paper is based on a single expansion, the LTE of the nonlinear model output for an input **x** *at the true parameter value* $\boldsymbol{q}_p$*:*

$$f(\mathbf{x}, \boldsymbol{q}) \approx f(\mathbf{x}, \boldsymbol{q}_p) + \boldsymbol{x}^T(\boldsymbol{q} - \boldsymbol{q}_p) \qquad (A1\text{-}1)$$

This expansion leads, for the data set, to:

$$\mathbf{f}(x, \boldsymbol{q}) \approx \mathbf{f}(x, \boldsymbol{q}_p) + \xi\,(\boldsymbol{q} - \boldsymbol{q}_p) \qquad (A1\text{-}2)$$

We now use (A1-2) in the expression of the cost-function:

$$J(\boldsymbol{q}) = \frac{1}{2}(\mathbf{y}_p - \mathbf{f}(x, \boldsymbol{q}))^T(\mathbf{y}_p - \mathbf{f}(x, \boldsymbol{q})) \qquad (A1\text{-}3)$$

This leads to:

$$J(\boldsymbol{q}) \approx \frac{1}{2}\left(\mathbf{y}_p - \mathbf{f}(x, \boldsymbol{q}_p) - \xi\,(\boldsymbol{q} - \boldsymbol{q}_p)\right)^T\left(\mathbf{y}_p - \mathbf{f}(x, \boldsymbol{q}_p) - \xi\,(\boldsymbol{q} - \boldsymbol{q}_p)\right)$$

$$\approx \frac{1}{2}\left(\mathbf{y}_p - \mathbf{f}(x, \boldsymbol{q}_p) + \xi\,\boldsymbol{q}_p\right)^T\left(\mathbf{y}_p - \mathbf{f}(x, \boldsymbol{q}_p) + \xi\,\boldsymbol{q}_p\right) - \boldsymbol{q}^T\,\xi^T\left(\mathbf{y}_p - \mathbf{f}(x, \boldsymbol{q}_p) + \xi\,\boldsymbol{q}_p\right) + \frac{1}{2}\,\boldsymbol{q}^T\,\xi^T\xi\,\boldsymbol{q}$$

An approximate expression of the gradient of the cost-function follows:

$$\frac{\partial J}{\partial \boldsymbol{q}} \approx -\xi^T\left(\mathbf{y}_p - \mathbf{f}(x, \boldsymbol{q}_p) + \xi\,\boldsymbol{q}_p\right) + \xi^T\,\xi\,\boldsymbol{q} \qquad (A1\text{-}4)$$

Hence an approximate expression of the least-squares estimate of the parameters:

$$\boldsymbol{q}_{LS} \approx \boldsymbol{q}_p + \left(\xi^T\,\xi\right)^{-1}\xi^T\left(\mathbf{y}_p - \mathbf{f}(x, \boldsymbol{q}_p)\right) \qquad (A1\text{-}5)$$

And hence the corresponding approximation of the least-squares estimator (i.e. the random vector $\boldsymbol{Q}_{LS}$*) of the parameters (expression (21) in the main text):*

$$\boldsymbol{Q}_{LS} \approx \boldsymbol{q}_p + \left(\xi^T\,\xi\right)^{-1}\xi^T\left(\mathbf{Y}_p - \mathbf{f}(x, \boldsymbol{q}_p)\right) \approx \boldsymbol{q}_p + \left(\xi^T\,\xi\right)^{-1}\xi^T\,\mathbf{W} \qquad (A1\text{-}6)$$

Using the linear Taylor expansion (A1-1), we obtain an approximation of the variance of the LS estimator of the regression for any input **x** *(expression (23) in the main text):*

$$var\left(f(\mathbf{x}, \boldsymbol{Q}_{LS})\right) \approx \sigma^2\,\boldsymbol{x}^T(x^T x)^{-1}\,\boldsymbol{x} \qquad (A1\text{-}7)$$

Sampling theory LTE variance estimate

The derivatives involved in $\boldsymbol{x}$ *and* x *being performed at the unknown* $\boldsymbol{q}° = \boldsymbol{q}_p$*, they may be estimated by the derivatives at* $\boldsymbol{q}° = \boldsymbol{q}_{LS}$*, that is by replacing* $\boldsymbol{x}$ *by* $\mathbf{z}$ *and* x *by* z*.*

Hence the LTE variance estimate presented in the paper:

$$\widehat{var\left(f(\mathbf{x}, \boldsymbol{Q}_{LS})\right)}_{LTE} = s^2\,\mathbf{z}^T(z^T z)^{-1}\,\mathbf{z} = \frac{\mathbf{r}^T\mathbf{r}}{N-q}\,\mathbf{z}^T(z^T z)^{-1}\,\mathbf{z} \qquad (A1\text{-}8)$$

A1.2. LTE variance estimates in maximum likelihood theory

For comparison, we sum up the results obtained with maximum likelihood theory (see for example [Efron & Tibshirani 1993] [Tibshirani 1996]). We make the same *assumptions as for sampling theory, i.e. that the nonlinear assumed model is true and that $K(\mathbf{W}) = s^2 I_N$ (homoscedasticity), and we consider a gaussian distributed noise. In this case, the log likelihood function is:*

$$L(\boldsymbol{q}) = -\frac{1}{2\,s^2}\left(\mathbf{y}_p - \mathbf{f}(x,\,\boldsymbol{q})\right)^T\left(\mathbf{y}_p - \mathbf{f}(x,\,\boldsymbol{q})\right) + cte \qquad (A1\text{-}9)$$

The parameters that maximize (A1-9) are those that minimize (A1-3), i.e. $\boldsymbol{q}_{ML} = \boldsymbol{q}_{LS}$.

It can be shown [Seber & Wild 1989] that the covariance matrix of $\boldsymbol{Q}_{ML}$ is given asymptotically by the inverse of the Fisher information matrix evaluated at $\boldsymbol{q}_p$. The Fisher information matrix being the mathematical expectation of the random matrix $M(\boldsymbol{q})$ of the second derivatives of the log likelihood function, we have:

$$\left[M(\boldsymbol{q})\right]_{ij} = -\frac{\partial^2 L}{\partial\theta_i\,\partial\theta_j} = \frac{1}{\sigma^2}\sum_{k=1}^{N}\left(\frac{\partial f(\mathbf{x}^k,\,\boldsymbol{q})}{\partial\theta_i}\frac{\partial f(\mathbf{x}^k,\,\boldsymbol{q})}{\partial\theta_j} + \left(Y_p^k - f(\mathbf{x}^k,\,\boldsymbol{q})\right)\frac{\partial^2 f(\mathbf{x}^k,\,\boldsymbol{q})}{\partial\theta_i\,\partial\theta_j}\right) \qquad (A1\text{-}10)$$

The assumed model being true, i.e. $E\left(Y_p^k - f(\mathbf{x}^k,\,\boldsymbol{q}_p)\right) = E\left(W^k\right) = 0$, the Fisher information matrix evaluated at $\boldsymbol{q}_p$ is given by:

$$\begin{aligned}\left[E(M(\boldsymbol{q}_p))\right]_{ij} &= -\left.\frac{\partial^2 L}{\partial\theta_i\,\partial\theta_j}\right|_{\boldsymbol{q}=\boldsymbol{q}_p} = \frac{1}{s^2}\sum_{k=1}^{N}\left[\left.\frac{\partial f(\mathbf{x}^k,\,\boldsymbol{q})}{\partial\theta_i}\right|_{\boldsymbol{q}=\boldsymbol{q}_p}\left.\frac{\partial f(\mathbf{x}^k,\,\boldsymbol{q})}{\partial\theta_j}\right|_{\boldsymbol{q}=\boldsymbol{q}_p}\right] \\ E(M(\boldsymbol{q}_p)) &= \frac{1}{s^2}\,x^T x\end{aligned} \qquad (A1\text{-}11)$$

Thus, the covariance matrix of $\boldsymbol{Q}_{ML} = \boldsymbol{Q}_{LS}$ is approximatively given by:

$$K(\boldsymbol{Q}_{ML}) \approx \left[E(M(\boldsymbol{q}_p))\right]^{-1} = s^2\left(x^T x\right)^{-1} \qquad (A\text{-}1\text{-}12)$$

Using the linear Taylor expansion (A1-1), the maximum likelihood approximation of the variance of the output in the gaussian case is obtained:

$$var\left(f(\mathbf{x},\,\boldsymbol{Q}_{LS})\right) \approx s^2\,\boldsymbol{x}^T\left(x^T x\right)^{-1}\mathbf{x} \qquad (A1\text{-}13)$$

Hence, the likelihood approximate variance (A1-13) is identical to the sampling theory approximate variance (A1-7).

Remark. The Hessian matrix h is the value of the random matrix H with elements:

$$\left[H(\boldsymbol{q})\right]_{ij} = \frac{\partial^2 J}{\partial\theta_i\,\partial\theta_j} = \sum_{k=1}^{N}\left(\frac{\partial f(\mathbf{x}^k,\,\boldsymbol{q})}{\partial\theta_i}\frac{\partial f(\mathbf{x}^k,\,\boldsymbol{q})}{\partial\theta_j} + \left(Y_p^k - f(\mathbf{x}^k,\,\boldsymbol{q})\right)\frac{\partial^2 f(\mathbf{x}^k,\,\boldsymbol{q})}{\partial\theta_i\,\partial\theta_j}\right) \qquad (A1\text{-}14)$$

Thus:

$$E\left(M(\boldsymbol{q}_p)\right) = \frac{1}{s^2}\,E\left(H(\boldsymbol{q}_p)\right) = \frac{1}{s^2}\,x^T x \qquad (A1\text{-}15)$$

Maximum likelihood theory LTE variance estimates

We can thus estimate the variance with:

$$\widehat{var\left(f(\mathbf{x},\,\boldsymbol{Q}_{LS})\right)}_{LTE} = \widehat{s^2}\,\mathbf{z}^T\left(z^T z\right)^{-1}\mathbf{z} \qquad (A1\text{-}16)$$

In likelihood theory, the variance of the noise is estimated with $\frac{R^T R}{N} = \frac{N-q}{N}\,s^2 \approx s^2$, but we will skip over this minor difference: (A1-16) is thus identical to (A1-8).

It is also proposed to estimate the Fisher information matrix $E\left(M(\boldsymbol{q}_p)\right)$ with the “observed information matrix” $m(\boldsymbol{q}_{LS})$; this leads to estimate the variance with:

$$\widehat{var\left(f(\mathbf{x},\,\boldsymbol{Q}_{LS})\right)}_{Hessian} = \widehat{s^2}\,\mathbf{z}^T\left[h(\boldsymbol{q}_{LS})\right]^{-1}\mathbf{z} \qquad (A1\text{-}17)$$

As opposed to estimate (A1-16), estimate (A1-17) necessitates the computation of the Hessian.

A1.3. Sandwich variance estimate

Let us propose a derivation of this estimate in the sampling approach. A second expansion is needed, the LTE of the gradient at the true parameter value $\boldsymbol{q}_p$:

$$\begin{aligned}\left.\frac{\partial J}{\partial\boldsymbol{q}}\right|_{\boldsymbol{q}=\boldsymbol{q}_{LS}} &\approx \left.\frac{\partial J}{\partial\boldsymbol{q}}\right|_{\boldsymbol{q}=\boldsymbol{q}_p} + \left.\frac{\partial^2 J}{\partial\boldsymbol{q}\;\partial\boldsymbol{q}^T}\right|_{\boldsymbol{q}=\boldsymbol{q}_p}\left(\boldsymbol{q}_{LS} - \boldsymbol{q}_p\right) \\ &= \left.\frac{\partial J}{\partial\boldsymbol{q}}\right|_{\boldsymbol{q}=\boldsymbol{q}_p} + h(\boldsymbol{q}_p)\left(\boldsymbol{q}_{LS} - \boldsymbol{q}_p\right)\end{aligned} \qquad (A1\text{-}18)$$

where $h(\boldsymbol{q}_p)$ is the value of the random Hessian matrix (see A1.2) evaluated at $\boldsymbol{q}_p$. Hence an approximate expression of the LS estimate of the parameters:

$$\boldsymbol{q}_{LS} \approx \boldsymbol{q}_p - \left[h(\boldsymbol{q}_p)\right]^{-1}\left.\frac{\partial J}{\partial\boldsymbol{q}}\right|_{\boldsymbol{q}=\boldsymbol{q}_p} \qquad (A1\text{-}19)$$

In (A1-19), we can replace the gradient by its expression:

$$\left.\frac{\partial J}{\partial\boldsymbol{q}}\right|_{\boldsymbol{q}=\boldsymbol{q}_p} = -\,\xi^T\left(\mathbf{y}_p - \mathbf{f}(x,\,\boldsymbol{q}_p)\right) = -\,\xi^T\,\mathbf{w} \qquad (A1\text{-}20)$$

Hence the corresponding approximation of the least-squares estimator (random vector $\boldsymbol{Q}_{LS}$) of the parameters:

$$\boldsymbol{Q}_{LS} \approx \boldsymbol{q}_p + \left[H(\boldsymbol{q}_p)\right]^{-1}\xi^T\,\mathbf{W} \qquad (A1\text{-}21)$$

Using the LTE of the model output (A1-1), we obtain:

$$f\left(\mathbf{x},\,\boldsymbol{Q}_{LS}\right) \approx f(\mathbf{x},\,\boldsymbol{q}_p) + \boldsymbol{x}^T\left[H(\boldsymbol{q}_p)\right]^{-1}x^T\;\mathbf{W} \qquad (A1\text{-}22)$$

Neglecting the random character of H (H being replaced by h), the output variance can be approximated by:

$$\mathrm{var}\left(f\left(\mathbf{x},\ \boldsymbol{Q}_{LS}\right)\right) \approx \sigma^2\ \mathbf{x}^T \left[h\left(\boldsymbol{q}_p\right)\right]^{-1} x^T x \left[h\left(\boldsymbol{q}_p\right)\right]^{-1} \boldsymbol{x} \tag{A1-23}$$

Sandwich variance estimate

This leads to propose the sandwich estimate:

$$\widehat{\mathrm{var}\left(f\left(\mathbf{x},\ \boldsymbol{Q}_{LS}\right)\right)}_{sandwich} = s^2\ \mathbf{z}^T \left[h\left(\boldsymbol{q}_{LS}\right)\right]^{-1} z^T z \left[h\left(\boldsymbol{q}_{LS}\right)\right]^{-1} \mathbf{z} \tag{A1-24}$$

This estimate also necessitates the computation of the Hessian of the cost-function.

APPENDIX 2. DERIVATION OF AN APPROXIMATE LOO ERROR

The following derivation is inspired from [Antoniadis et al. 1992]; it is valid irrespective of whether or not the assumed model is true. We denote by $\boldsymbol{q}_{LS}^{(k)}$ the LS estimate on the k-th LOO set $\left\{\mathbf{x}^i,\ y_p^i\right\}_{i=1 \text{ to } N,\ i\neq k}$. We have the k-th residual r^k and the k-th LOO error e^k:

$$\begin{cases} r^k = y_p^k - f\left(\mathbf{x}^k,\ \boldsymbol{q}_{LS}\right) \\ e^k = y_p^k - f\left(\mathbf{x}^k,\ \boldsymbol{q}_{LS}^{(k)}\right) \end{cases} \tag{A2-1}$$

Let us denote by $\mathbf{y}_p^{(k)}$ the (N-1)-vector obtained by deletion of the k-th component of the measured output vector $\mathbf{y}_p$, by $z^{(k)}$ the (N-1,q) matrix obtained by deletion of the k-th row of z, by $x^{(k)}$ the (N-1,q) matrix obtained by deletion of the k-th row of x. The LOO estimate $\boldsymbol{q}_{LS}^{(k)}$ minimizes the cost-function:

$$J^{(k)}(\boldsymbol{q}) = \frac{1}{2}\left(\mathbf{y}_p^{(k)} - \mathbf{f}\left(x^{(k)},\ \boldsymbol{q}\right)\right)^T \left(\mathbf{y}_p^{(k)} - \mathbf{f}\left(x^{(k)},\ \boldsymbol{q}\right)\right) \tag{A2-2}$$

We first approximate $\mathbf{f}\left(x^{(k)},\ \boldsymbol{q}\right)$ by its LTE at $\boldsymbol{q}_{LS}$:

$$\mathbf{f}\left(x^{(k)},\ \boldsymbol{q}\right) \approx \mathbf{f}\left(x^{(k)},\ \boldsymbol{q}_{LS}\right) + z^{(k)}\ \left(\boldsymbol{q} - \boldsymbol{q}_{LS}\right) \tag{A2-3}$$

Hence the approximation of $\boldsymbol{q}_{LS}^{(k)}$:

$$\boldsymbol{q}_{LS}^{(k)} \approx \boldsymbol{q}_{LS} + \left(z^{(k)T} z^{(k)}\right)^{-1} z^{(k)T} \left(\mathbf{y}_p^{(k)} - \mathbf{f}\left(x^{(k)},\ \boldsymbol{q}_{LS}\right)\right) \tag{A2-4}$$

In the previous expression, we have:

$$\begin{aligned} z^{(k)T}\left(\mathbf{y}_p^{(k)} - \mathbf{f}\left(x^{(k)},\ \boldsymbol{q}_{LS}\right)\right) &= z^T\left(\mathbf{y}_p - \mathbf{f}\left(x,\ \boldsymbol{q}_{LS}\right)\right) - \mathbf{z}^k\, r^k \\ &= z^T\,\mathbf{r} - \mathbf{z}^k\, r^k \\ &= -\,\mathbf{z}^k\, r^k \end{aligned} \tag{A2-5}$$

since the columns of z are orthogonal to the residual vector **r**. *Using the matrix inversion lemma, we can express $\left(z^{(k)T} z^{(k)}\right)^{-1}$ in (A2.4) in terms of $\left(z^T z\right)^{-1}$:*

$$\begin{aligned} \left(z^{(k)T} z^{(k)}\right)^{-1} &= \left(z^T z\right)^{-1} + \frac{\left(z^T z\right)^{-1} \mathbf{z}^{(k)}\, \mathbf{z}^{(k)T} \left(z^T z\right)^{-1}}{1 - \mathbf{z}^{(k)T} \left(z^T z\right)^{-1} \mathbf{z}^{(k)}} \\ &= \left(z^T z\right)^{-1} + \frac{\left(z^T z\right)^{-1} \mathbf{z}^{(k)}\, \mathbf{z}^{(k)T} \left(z^T z\right)^{-1}}{1 - \left[p_z\right]_{kk}} \end{aligned} \tag{A2-6}$$

where p_z denotes the orthogonal projection matrix on the range of z.

Replacing (A2-5) and (A2-6) into (A2-4), we finally obtain:

$$\boldsymbol{q}_{LS}^{(k)} \approx \boldsymbol{q}_{LS} - \left(z^T z\right)^{-1} \mathbf{z}^k\, \frac{r^k}{1 - \left[p_z\right]_{kk}} \tag{A2-7}$$

Expanding e^k at $\boldsymbol{q}_{LS}$ and replacing (A2-7) into this expansion, we obtain an approximate expression of the LOO error which is similar to the expression of the linear LOO error (36):

$$e^k \approx \frac{r^k}{1 - \left[p_z\right]_{kk}} \quad k=1 \text{ to } N \tag{A2-8}$$

assuming that $\left[p_z\right]_{kk} < 1$. In the case where $\left[p_z\right]_{kk} = 1$, we choose to take, similarly to the linear case, $e^k = r^k$ (the residual is not necessarily zero).

In practice, the diagonal terms of p_z are computed using the singular value factorization of $z = u\,\Sigma\,v$, where u is an orthogonal (N,N) matrix, Σ is a diagonal (N,q) matrix, and v is an orthogonal (q,q) matrix, see [Golub & Van Loan 1983]. Then:

$$\left[p_z\right]_{kk} = \sum_{i=1}^{q} \left[u\right]_{ki}^2 \quad k=1 \text{ to } N \tag{A2-9}$$

The diagonal elements of p_z that differ from 1 by a threshold consistent with the computer precision are considered as equal to 1 (theoretically, the values of the $\left[p_z\right]_{kk}$ are comprised between 1/N and 1).

REFERENCES

Antoniadis A., Berruyer J. & Carmona R. (1992). Régression non linéaire et applications. Paris: Economica.

Bates D. M. & Watts D. G. (1988). Nonlinear regression analysis and its applications. New York: Wiley.

Bishop M. (1995). Neural Networks for Pattern Recognition. Oxford: Clarendon Press.

Buntine W. & Weigend A. (1994). Computing second derivatives in feedforward neural networks: a review. IEEE Transactions on Neural Networks **5** *(3), 480-488.*

Draper N. R. & Smith H. (1998). Applied regression analysis. New York: John Wiley & Sons, inc.

Efron B. & Tibshirani R. J. (1993). An introduction to the bootstrap. New York: Chapman & Hall.

Golub G. H. & Van Loan C. F. (1983). Matrix computations. Baltimore: The John Hopkins University Press.

Goodwin G. C. & Payne R. L. (1977). Dynamic system identification; experiment design and data analysis. New York: Academic Press.

Hansen L. K. & Larsen J. (1993). Linear unlearning for cross validation, Advances in computational mathematics **5***, 296-280.*

Heskes T. (1997). Practical confidence and prediction intervals, to appear in M. Mozer, M. Jordan & T. Petsche (EDS.), Advances in Neural Information Processing Systems **9***. Cambridge, MA: MIT Press.*

MacKay D. J. C. (1992a). Bayesian interpolation. Neural computation **4***, 415-447.*

MacKay D. J. C. (1992b). A practical Bayesian framework for backprop networks. Neural computation **4***, 448-472.*

Monari G. (1999). Sélection de modèles non linéaires par leave-one-out ; étude théorique et application des réseaux de neurones au procédé de soudage par points, Thèse de Doctorat de l'Université Paris 6.

Monari G., Dreyfus G. Local linear least squares: performing leave-one-out without leaving anything out, submitted for publication.

Paass G. (1993). Assessing and improving neural network predictions by the bootstrap algorithm. In S. J. Hanson, J. D. Cowan, & C. L. Giles (Eds.), Advances in Neural Information Processing Systems **5** *(pp. 186-203). Cambridge, MA: MIT Press.*

Refenes A.-P. N., Zapranis A. D., Utans J. (1997). Neural model identification, variable selection and model adequacy. In "Decision technologies for financial engineering", A. S. Weigend, Y. Abu-Mostafa, A.-P. N. Refenes eds. World scientific, 243-261.

Ripley B. D. (1995). Pattern recognition and neural networks. Cambridge: Cambridge University Press.

Rivals I. & Personnaz L. (1998). Construction of confidence intervals in neural modeling using a linear Taylor expansion. Proceedings of the International Workshop on Advanced Black-Box Techniques for Nonlinear Modeling, 8-10 July 1998, Leuwen, 17-22.

Rivals I. & Personnaz L. (1999). On cross-validation for model selection, Neural Computation **11***, 863-870.*

Saarinen S., Bramley R. & Cybenko G. (1993) Ill-conditioning in neural network training problems, SIAM J. Sci. Stat. Comp. **14***, 693-714.*

Seber G. A. F. (1977). Linear regression analysis. New York: Wiley.

Seber G. A. F. & Wild C. (1989). Nonlinear regression. New York: Wiley.

Sussman H. J. (1992). Uniqueness of the weights for minimal feedforward nets with a given input-output map. Neural Networks **5***, 589-593.*

Tibshirani R. J. (1996). A comparison of some error estimates for neural models. Neural Computation **8***, 152-163.*

Urbani D., Roussel-Ragot P., Personnaz L. & Dreyfus G. (1994). The selection of neural models of non-linear dynamical systems by statistical tests, Neural Networks for Signal Processing, Proceedings of the 1994 IEEE Workshop.

Zhou G. & Si J. (1998). A systematic and effective supervised learning mechanism based on Jacobian rank deficiency, Neural Computation **10***, 1031-1045.*

TABLES

Table 1.

Results obtained on the modeling of the simulated SISO process #4 using neural networks, in the case N = 200.

n_h	q	$MSTE$	$cond(z)$	$MSPE$	$\frac{MSPE}{MSTE}$
1	4	$1.4\ 10^{-2}$	10	$1.4\ 10^{-2}$	1.0
2	7	$1.2\ 10^{-2}$	10^{3}	$1.3\ 10^{-2}$	1.1
3	10	$9.7\ 10^{-3}$	10^{6}	$1.1\ 10^{-2}$	1.1
4	**13**	$\mathbf{8.5\ 10^{-3}}$	$\mathbf{10^{2}}$	$\mathbf{9.8\ 10^{-3}}$	**1.1**
5	16	$8.4\ 10^{-3}$	10^{6}	$9.9\ 10^{-3}$	1.2
6	19	$8.2\ 10^{-3}$	10^{7}	$1.0\ 10^{-2}$	1.2
7	22	$7.9\ 10^{-3}$	10^{11}	–	–

Table 2.

Results obtained on the modeling of the simulated SISO process #4 using neural networks, in the case N = 300.

n_h	q	$MSTE$	$cond(z)$	$MSPE$	$\frac{MSPE}{MSTE}$
1	4	$2.4\ 10^{-2}$	10^{1}	$2.7\ 10^{-2}$	1.1
2	**7**	$\mathbf{1.1\ 10^{-2}}$	$\mathbf{10^{6}}$	$\mathbf{2.3\ 10^{-2}}$	**2.1**
3	10	$8.1\ 10^{-3}$	10^{3}	$2.4\ 10^{-2}$	3.0
4	13	$7.1\ 10^{-3}$	10^{4}	$4.3\ 10^{1}$	$6.1\ 10^{3}$
5	16	$5.0\ 10^{-3}$	10^{15}	–	–

Table 3.

Results obtained on the modeling of the industrial process using neural networks.

n_h	q	*MSTE*	*cond(z)*	*MSPE*	$\frac{MSPE}{MSTE}$
1	*5*	$5.2\ 10^{-2}$	10^{4}	$6.6\ 10^{-2}$	*1.3*
2	**9**	$\mathbf{1.6\ 10^{-2}}$	$\mathbf{10^{5}}$	$\mathbf{2.1\ 10^{-2}}$	**1.3**
3	*13*	$1.5\ 10^{-2}$	10^{4}	$1.7\ 10^{-1}$	$1.1\ 10^{1}$
4	*17*	$1.2\ 10^{-2}$	10^{12}	–	–

FIGURES

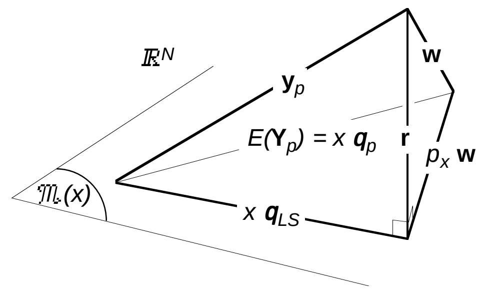

Figure 1.

Geometric representation of the linear LS solution (true assumed model).

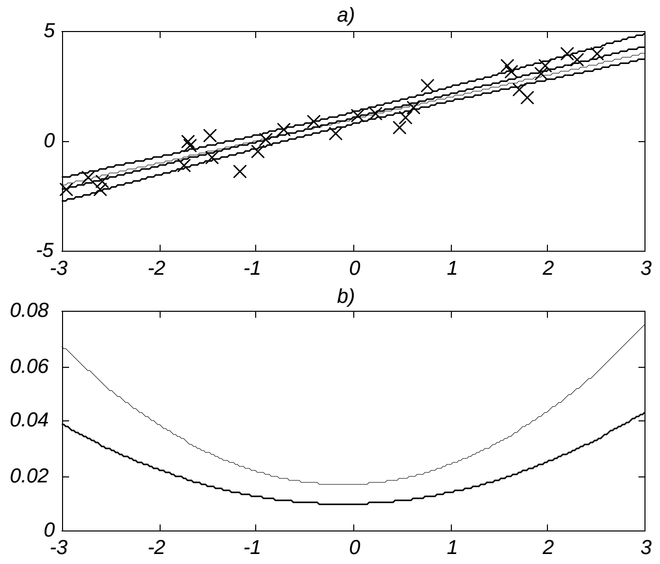

Figure 2.

CI for process #1, a simulated linear SISO process (true assumed model with $n = 2$ parameters): a) regression (thin line), data set (crosses), model output and 99% CI (thick lines); b) true variance (thin line) and LS estimate of the variance (thick line) of $\mathbf{x}^T \boldsymbol{Q}_{LS}$.

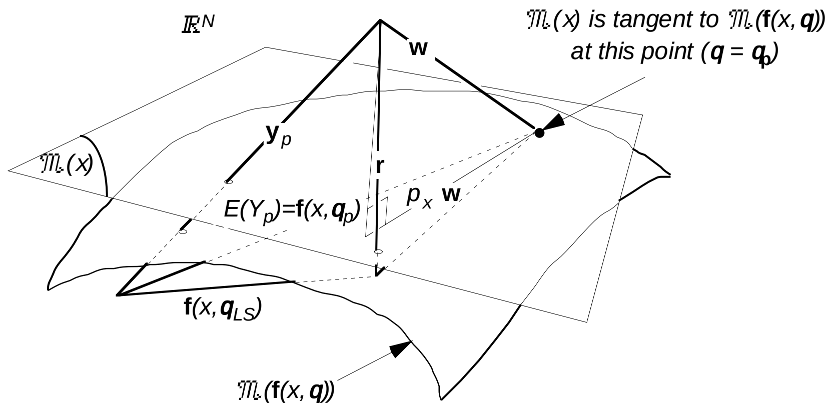
$\mathbb{R}^N$
$\mathcal{M}(x)$ is tangent to $\mathcal{M}(\mathbf{f}(x,\boldsymbol{q}))$ at this point ($\boldsymbol{q} = \boldsymbol{q}_p$)
w
$\mathbf{y}_p$
$\mathcal{M}(x)$
r
p_x w
$E(Y_p)=\mathbf{f}(x,\boldsymbol{q}_p)$
$\mathbf{f}(x,\boldsymbol{q}_{LS})$
$\mathcal{M}(\mathbf{f}(x,\boldsymbol{q}))$

Figure 3.

Geometric representation of the nonlinear LS solution and of its LTE approximation (true assumed model).

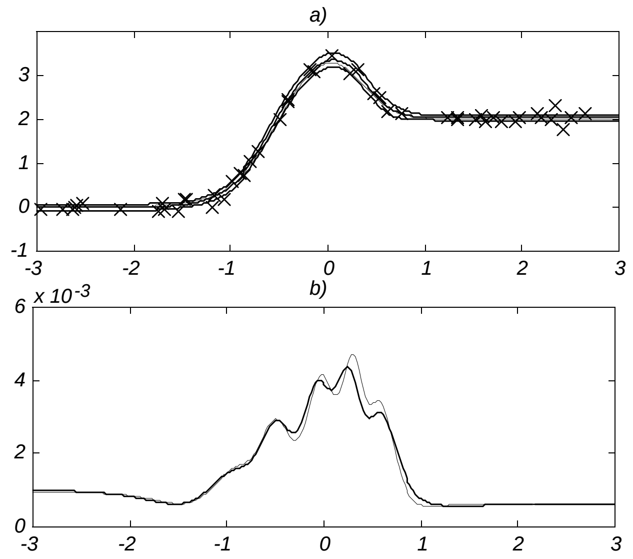

Figure 4.

CI for process #2, a simulated “neural” SISO process (the assumed model, a two hidden neurons network with q = 7 parameters, is true): a) regression (thin line), the N = 50 examples of the data set (crosses), model output and 99% approximate CI (thick lines); b) reference (thin line) and LTE (thick line) estimates of the variance of $f(\mathbf{x}, \boldsymbol{Q}_{LS})$.

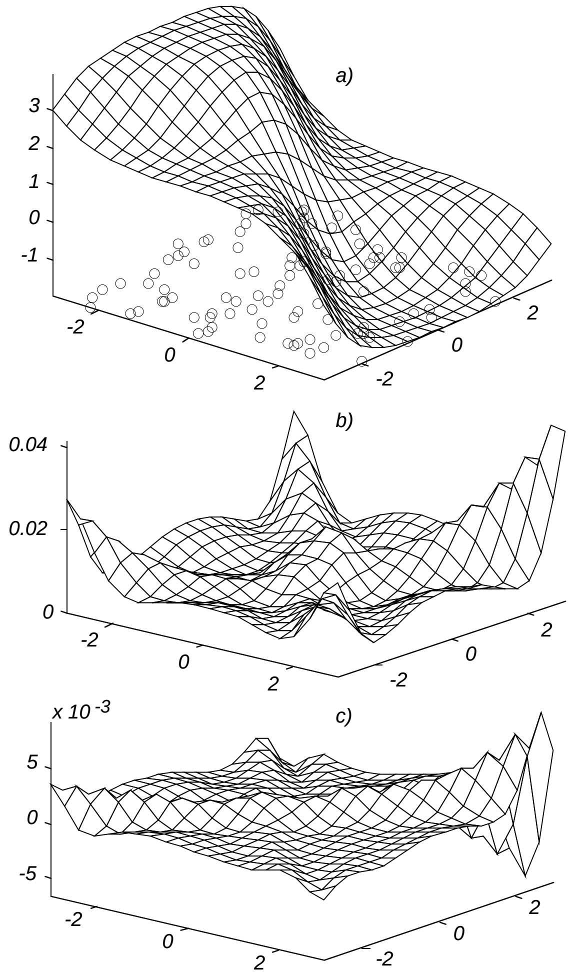

Figure 5.

CI for process #3, a simulated “neural” MISO process (the assumed model, a two hidden neurons network with q = 9 parameters, is true): a) the N = 100 inputs of the data set (circles) and regression; b) LTE estimate of the variance of $f(\mathbf{x}, \boldsymbol{Q}_{LS})$*; c) difference between the reference and the LTE estimates of the variance of* $f(\mathbf{x}, \boldsymbol{Q}_{LS})$.

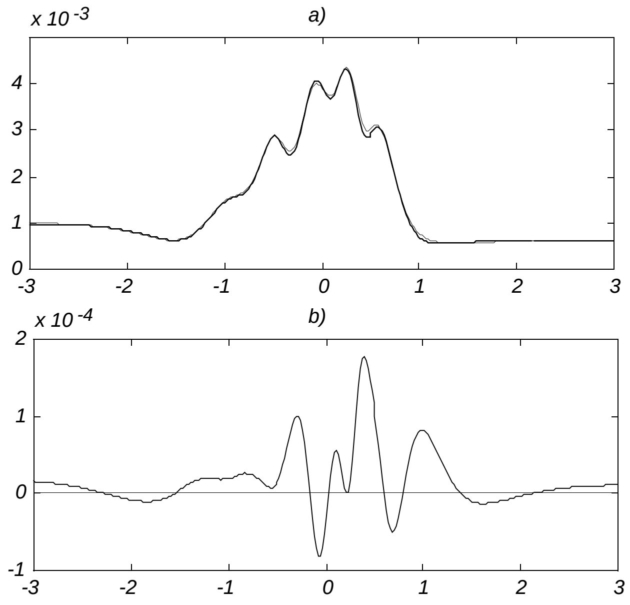

Figure 6.

Accuracy of the LTE approximation of the variance for process #2: a) reference estimate of the variance of $f(\mathbf{x}, \boldsymbol{Q}_{LS})$ *(thin line), LTE approximation obtained with the true values* $\boldsymbol{q}_p$ *and* σ^2 *(thick line); b) difference between the reference estimate of the variance of* $f(\mathbf{x}, \boldsymbol{Q}_{LS})$ *and the LTE approximation obtained with* $\boldsymbol{q}_p$ *and* σ^2*.*

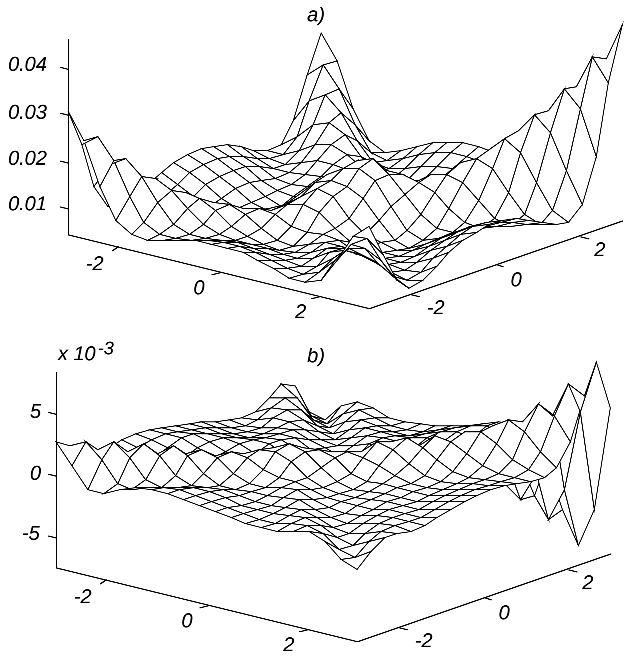

Figure 7.

Accuracy of the LTE approximation of the variance for process #3: a) reference estimate of the variance of $f(\mathbf{x}, \boldsymbol{Q}_{LS})$*; b) difference between the reference estimate of the variance of* $f(\mathbf{x}, \boldsymbol{Q}_{LS})$ *and the LTE approximation obtained with* $\boldsymbol{q}_p$ *and* σ^2*.*

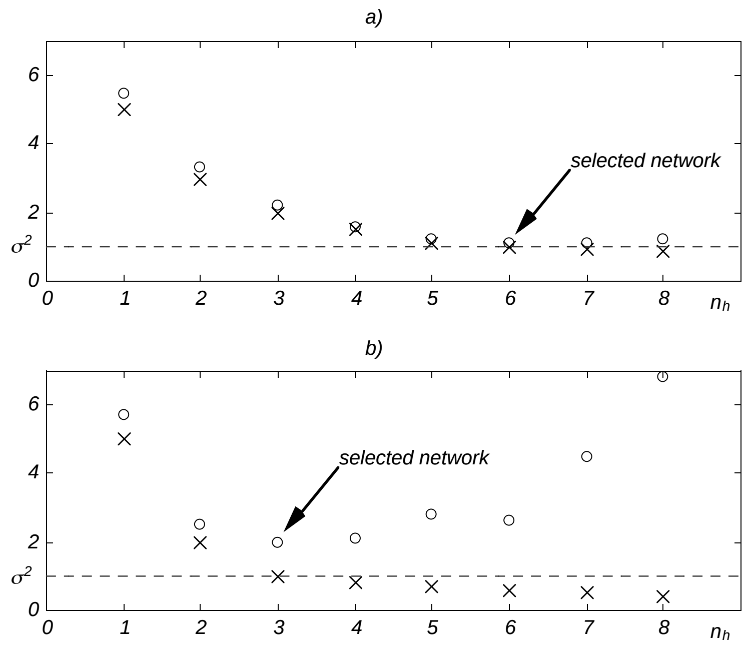

Figure 8.

Schematic evolution of the MSTE (crosses) and MSPE (circles) as a function of the number of hidden neurons of the neural network candidates, the network with the smallest MSPE being selected: a) large data set: the ratio MSPE/MSTE of the selected network (six hidden neurons) is roughly equal to 1, hint that the data set size N is large; b) small data set: the ratio MSPE/MSTE of the selected network (three hidden neurons) is roughly equal to 2, hint that the data set size N is small.

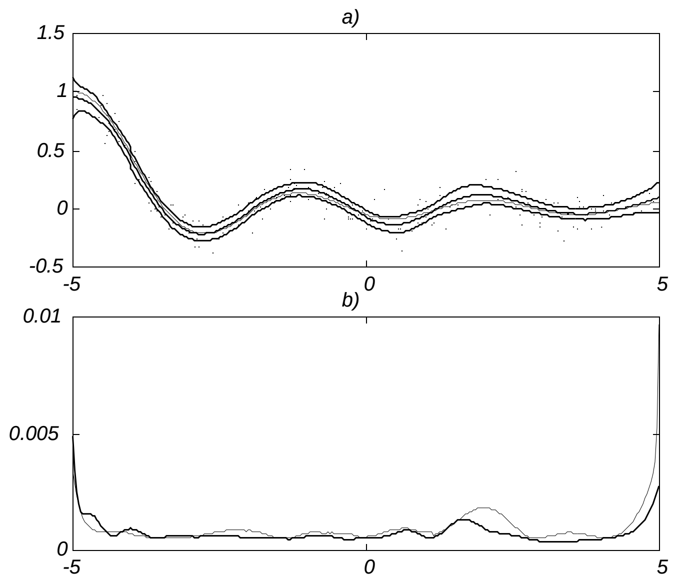

Figure 9.

CI for process #4, a simulated nonlinear SISO process, in the case of a data set of size N = 200 (the selected model is a four hidden neurons network with q = 13 parameters): a) regression (thin line), data set (small points), model output and 99% approximate CI (thick lines); b) reference (thin line) and LTE (thick line) estimates of the variance of $f(\mathbf{x}, \boldsymbol{\theta}_{LS})$.

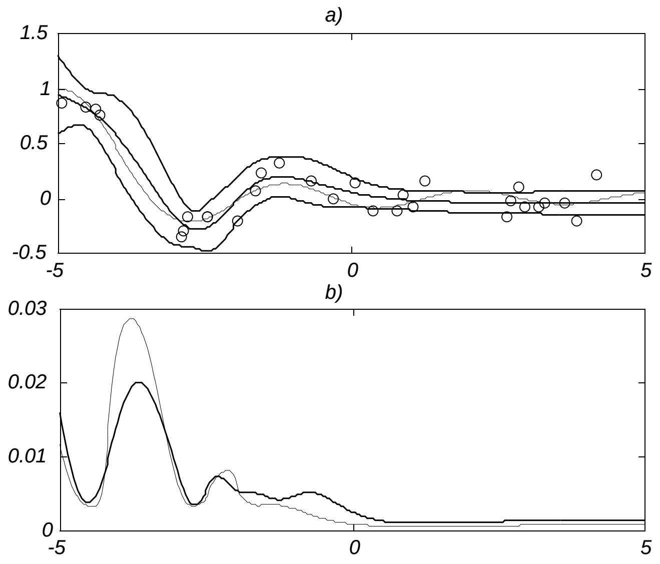

Figure 10.

CI for process #4, a simulated nonlinear SISO process, in the case of a data set of size N = 30 (the selected model is a two hidden neurons network with q = 7 parameters): a) regression (thin line), data set (circles), model output and 99% approximate CI (thick lines); b) reference (thin line) and LTE (thick line) estimates of the variance of $f(\mathbf{x}, \boldsymbol{\theta}_{LS})$.

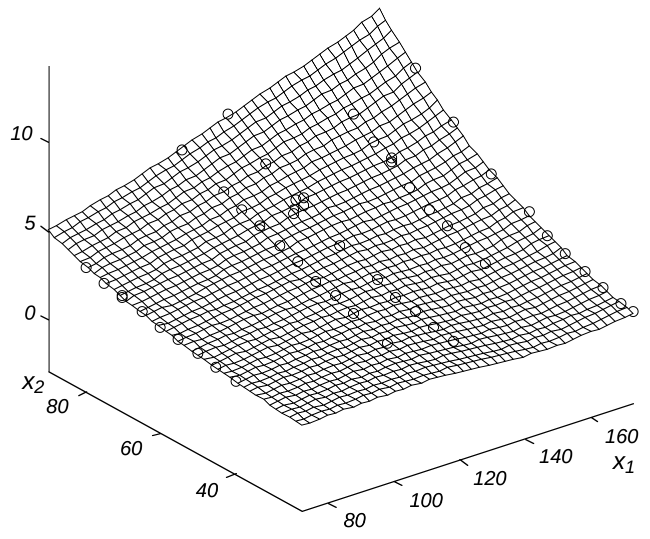

Figure 11.

Industrial modeling problem (the selected model is a two hidden neurons network with q = 9 parameters): model output, and the N = 69 examples of the data set (circles).

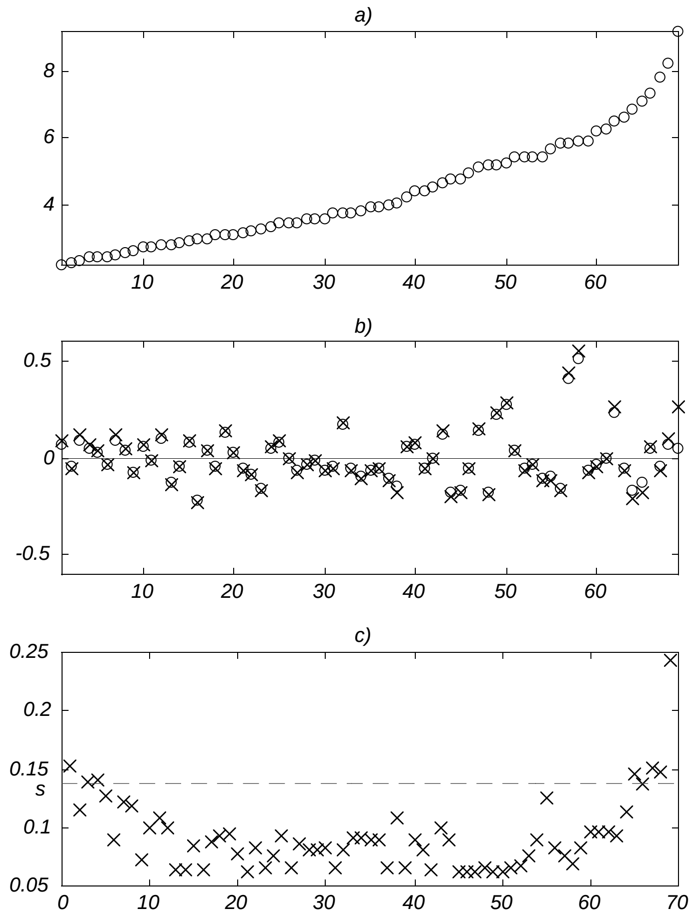

Figure 12.

Industrial modeling problem: a) the N = 69 outputs of the data set presented in increasing order of their values; b) the corresponding residuals (circles) and approximate LOO errors (crosses); c) half width of the 95% approximate CI at the N = 69 examples of the data set, and LS estimate s of the noise standard-deviation s (dotted line).

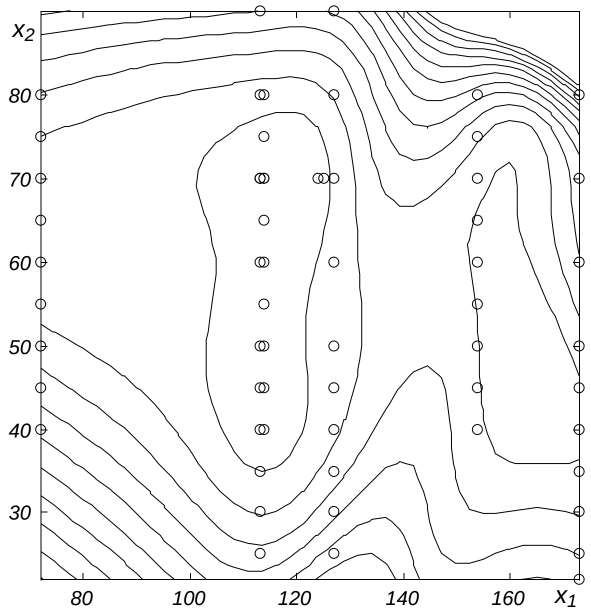

Figure 13.

Industrial modeling problem: isocontours of the LTE estimate of the standard deviation of $f(\mathbf{x}, \boldsymbol{Q}_{LS})$, and the N = 69 inputs of the data set (circles).

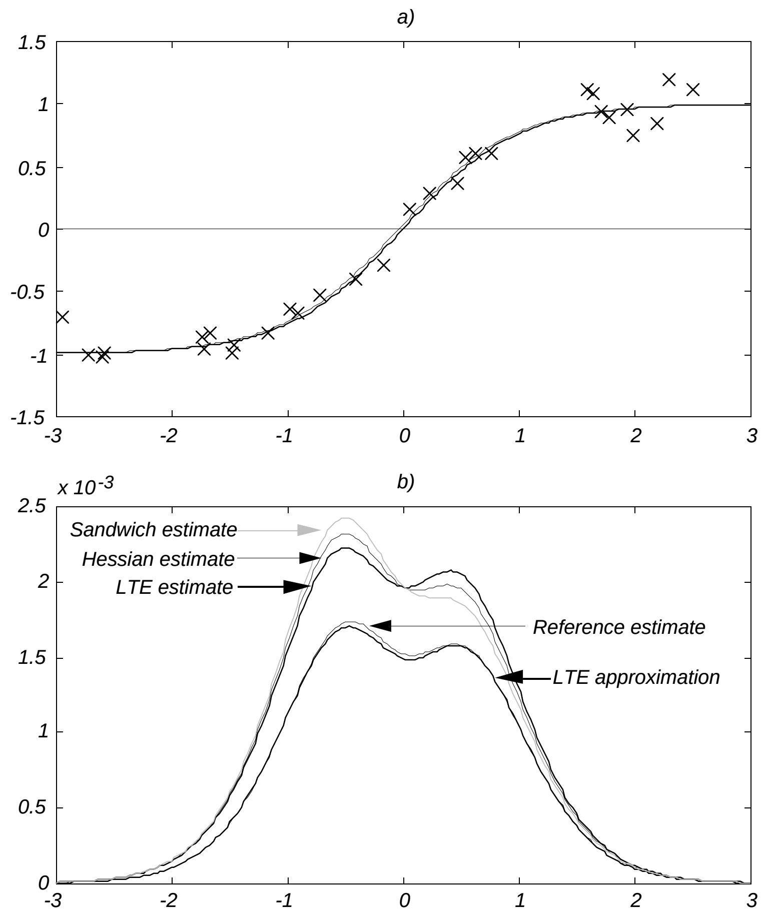

Figure 14.

Comparison of different estimates of the variance of a nonlinear model output for process #5, a simulated "neural" SISO process (the assumed model, a single nonlinear neuron with q = 2 parameters, is true): a) regression with a "gentle" slope (thin line), the N = 30 examples of the data set (crosses), model output (thick line); b) LTE approximation and estimates of the variance of $f\left(\mathbf{x}, \boldsymbol{\theta}_{LS}\right)$.

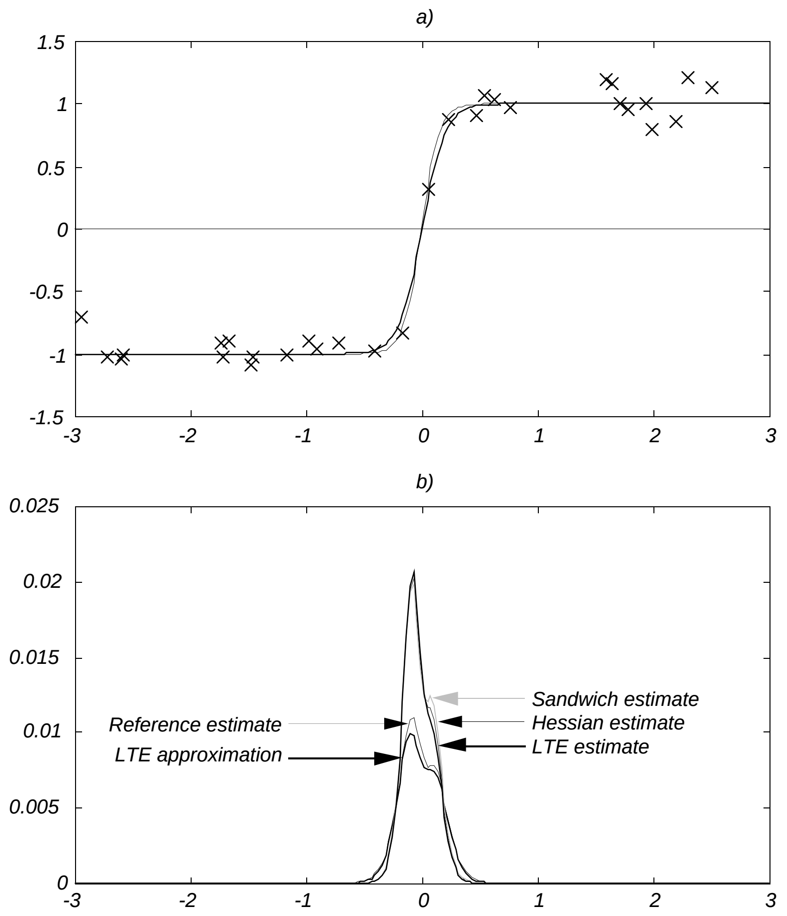

Figure 15.

Comparison of different estimates of the variance of a nonlinear model output for process #6, a simulated "neural" SISO process (the assumed model, a single nonlinear neuron with q = 2 parameters, is true): a) regression with a "steep" slope (thin line), the N = 30 examples of the data set (crosses), model output (thick line); b) LTE approximation and estimates of the variance of $f\left(\mathbf{x}, \boldsymbol{\theta}_{LS}\right)$.

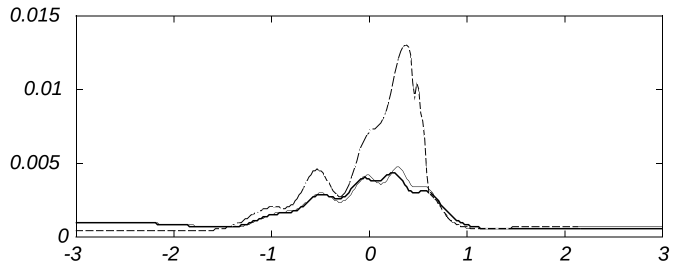

Figure 16.

Comparison of the LS LTE and bootstrap pairs approach estimates of the variance for process #2: reference (thin line), LTE (thick line), and bootstrap pairs (dotted line) estimates of the variance of $f\left(\mathbf{x}, \boldsymbol{Q}_{LS}\right)$.