Journal of Machine Learning Research 3 (2003) 1383-1398 Submitted 5/02, Published 3/03

MLPs (Mono-Layer Polynomials and Multi-Layer Perceptrons) for Nonlinear Modeling

Isabelle Rivals ISABELLE.RIVALS@ESPCI.FR
Léon Personnaz LEON.PERSONNAZ@ESPCI.FR
Équipe de Statistique Appliquée
École Supérieure de Physique et de Chimie Industrielles
Paris, F75005, FRANCE

Editors: Isabelle Guyon and André Elisseeff

Abstract

This paper presents a model selection procedure which stresses the importance of the classic polynomial models as tools for evaluating the complexity of a given modeling problem, and for removing non-significant input variables. If the complexity of the problem makes a neural network necessary, the selection among neural candidates can be performed in two phases. In an additive phase, the most important one, candidate neural networks with an increasing number of hidden neurons are trained. The addition of hidden neurons is stopped when the effect of the round-off errors becomes significant, so that, for instance, confidence intervals cannot be accurately estimated. This phase leads to a set of approved candidate networks. In a subsequent subtractive phase, a selection among approved networks is performed using statistical Fisher tests. The series of tests starts from a possibly too large unbiased network (the full network), and ends with the smallest unbiased network whose input variables and hidden neurons all have a significant contribution to the regression estimate. This method was successfully tested against the real-world regression problems proposed at the NIPS2000 Unlabeled Data Supervised Learning Competition; two of them are included here as illustrative examples.

Keywords: additive procedure, approximate leave-one-out scores, confidence intervals, input variable selection, Jacobian matrix conditioning, model approval, model selection, neural networks, orthogonalization procedure, overfitting avoidance, polynomials, statistical tests.

1. Introduction

Many researchers in the neural network community devote a large part of their work to the development of model selection procedures (Moody, 1994, Kwok and Yeung, 1997a,b, Anders and Korn, 1999, Rivals and Personnaz, 2000b and 2003a, Vila, Wagner and Neveu, 2000). However, most of the corresponding publications assume the use of neural networks, and do not insist enough on the importance of a preliminary evaluation of the regression complexity, and of a removal of non-significant input variables, before considering neural networks. Our experience with real-world industrial problems has confirmed that polynomial models can be efficient tools for both tasks.

Because they are linear in their parameters, polynomials are computationally easier to handle than neural networks, and their statistical properties are well established. However, when the non-linearity of the regression is significant, so that a high degree polynomial of many monomials would be needed, the modularity and the parsimony of neural networks can be taken advantage of. Thus, it is only once the significant input variables have been selected, and once the regression has been identified as highly nonlinear, that specific "neural" modeling methods can be most successful. For this purpose, we also propose a novel model selection procedure for neural modeling, which is mainly based on least squares (LS) estimation, on the analysis of the numerical conditioning of the candidate models, and on statistical tests. This neural network selection procedure was successfully tested on artificial modeling problems by Rivals and Personnaz (2000b and 2003a).

In Section 2, the goal of the modeling method is described. In Section 3, we present a polynomial ranking and selection procedure, and a strategy for deciding whether the polynomial should be kept, or put in competition with other models (networks of radial basis functions, multi-layer perceptrons, etc.). In Section 4, the specific neural modeling procedure is presented. In Section 5, the whole method is applied to two representative modeling problems drawn from the NIPS2000 Unlabeled Data Supervised Learning Competition.

2. Goal of the Proposed Selection Procedure

We deal with the modeling of processes having an n-input vector $\boldsymbol{x}$ and a measured scalar output y that is considered as the actual value of a random variable depending on $\boldsymbol{x}$. We assume that there exists an unknown function of $\boldsymbol{x}$, the regression $f(\boldsymbol{x})$, such that for any fixed value $\boldsymbol{x}_a$ of $\boldsymbol{x}$:

$$y_a = E(y_a) + \varepsilon_a = f(\boldsymbol{x}_a) + \varepsilon_a \tag{1}$$

where $E(y_a)$ is the mathematical expectation of y_a, and ε_a is a random noise variable with zero expectation, the inexplicable part of y_a. We consider families of parameterized functions $\{g(\boldsymbol{x}, \boldsymbol{w}), \boldsymbol{x} \in \mathbb{R}^n, \boldsymbol{w} \in \mathbb{R}^q\}$. Such a family of functions contains the regression if there exists a value $\boldsymbol{w}^*$ of $\boldsymbol{w}$ such that $g(\boldsymbol{x}, \boldsymbol{w}^*) = f(\boldsymbol{x})$. In real-world black-box modeling problems, such a family of functions is not known *a priori*, so that candidate families of various complexities must be put in competition. In this work, we consider models whose output is linear with respect to the parameters, such as polynomials, networks of fixed Radial Basis Functions (RBFs), and multi-layer perceptrons that are nonlinear in their parameters. A data set of m input-output pairs $\{\boldsymbol{x}_k, y_k\}_{k=1 \text{ to } m}$ is assumed available for the parameter estimation.

The output value $g(\boldsymbol{x}_a, \boldsymbol{w})$ of such a model for a given input $\boldsymbol{x}_a$ of interest is a point estimate of the regression, and may be meaningless if the variance of the model output is large, due to a too small or not well distributed data set, and/or due to a high non-linearity of the model. In order to quantify this uncertainty on the point estimate of the regression, it is necessary to estimate a confidence interval that takes the model variance into account (Rivals and Personnaz, 1998 and 2000a). Finally, the model must be numerically well conditioned for the confidence intervals to be reliable. The goal of the modeling is therefore to select a model of minimal complexity that possesses only the significant input variables, approximates the regression as accurately as possible within the input domain delimited by the data set, and is well-conditioned enough to allow estimation of a confidence interval.

The selection procedure we propose begins with the removal of possibly non-significant input variables and with evaluating the regression complexity using polynomials (Section 3). When the regression is judged sufficiently complex, i.e., when it is a very nonlinear function of many significant input variables, we proceed with the construction and selection of neural networks (Section 4).

3. Polynomial Modeling

In the above formulation of the problem, it is assumed that the input vector $\boldsymbol{x}$ contains all the input variables necessary to "explain" the regression $f(\boldsymbol{x})$. When dealing with industrial processes, these input variables are the values of control variables and measurable disturbances, and may be easy to identify. When dealing with processes which are not built by man (economical, ecological, biological systems, etc.), the designer of the model may only have a list of all the input variables judged potentially significant, created without *a priori* restrictions by specialists of the process. In this case, the design of the model relies heavily on the removal of non-significant or redundant input variables. This removal can be economically performed with polynomials prior to the use of neural networks.

Polynomial modeling can be performed in two steps: first, nested polynomials are constructed, and second, a selection between the best candidates is performed.

3.1. Construction of Nested Polynomials

Given n potentially significant input variables, we build a polynomial of monomials up to degree d. For example, if the list of the input variables is $x_1, x_2, \ldots, x_n$, the polynomial of degree 2 involves a constant term and the monomials: x_1, x_2, …, x_n, $x_1 x_2$, $x_1 x_3$, …, $x_{n-1} x_n$, x_1^2, …, x_n^2. The polynomial of degree d possesses a number of monomials equal to:

$$N_{mono}(n, d) = \sum_{i=1}^{d} K_n^i = \sum_{i=1}^{d} C_{n+i-1}^i \tag{2}$$

where K_n^i is the number of i to i combinations of n objects with repetitions, and $C_n^i = i!/n!(n-i)!$ is the number of i to i combinations of n objects without repetitions. For example, the polynomial of degree $d = 2$ of $n = 100$ input variables possesses 5 150 monomials, but that of degree $d = 3$ possesses 176 850 monomials.

Following Golub (1983) and Chen et al. (1989), the monomials are iteratively ranked in order of decreasing contribution to their explanation of the output. We denote the m-output vector of the data set by $\boldsymbol{y}$, and the m-vectors corresponding to the monomials by the $\{\xi_i\}$. The monomial j that most decreases the residual sum of squares (RSS) which is also the monomial such that $|\cos(\boldsymbol{y}, \xi_j)|$ is the largest is ranked as first. The remaining $\{\xi_i\}$ and the output vector $\boldsymbol{y}$ are orthogonalized with respect to ξ_j using the modified Gram-Schmidt orthogonalization algorithm. The procedure is repeated in the subspace orthogonal to ξ_j, and so on. The RSS of polynomials with more than $m - 1$ parameters being equal to zero, the procedure is stopped when the $m - 1$ first monomials (and hence the corresponding nested polynomials) are ranked. The ranking may not be absolutely optimal in the sense that, for a given polynomial size, there is a small probability that a polynomial of different monomials than those chosen by Gram-Schmidt has a slightly smaller RSS (Stoppiglia, 1997). However we know of no practical example where Gram-

Schmidt applied to polynomials fails to capture a significant input or an important non-linearity. Note that the computational cost of this procedure is very low: it merely involves computation of inner products.

The Gram-Schmidt procedure is related to the construction of Multivariate Adaptive Regression Splines (MARS, see Hastie et al., 2001), whose polynomial form also allows selecting significant inputs. As with the Gram-Schmidt procedure, the splines are chosen *a posteriori* (the knots of the splines are located at the training examples), and at each step, MARS adds the spline which leads to the largest decrease of the RSS.

Conversely, Gram-Schmidt should not be compared with the polynomial construction method used in Structural Risk Minimization approaches, as described by Vapnik (1982). There, the nested polynomial models are chosen *a priori*, usually in the "natural" order of increasing degree x, x^2, x^3, etc. However, there is no "natural" order in the multi-dimensional case. Moreover, it means that the product $x_i\, x_j\, x_k$ will only appear in a model already containing all monomials of degree 1 and 2, and whose number of parameters might already be larger than m. In our approach, the data set is used to rank the monomials, so a significant monomial of high degree can appear among the first ranked.

3.2. Selection of a Sub-Polynomial

A model can be selected with a) Fisher tests, or b) on the basis of the leave-one-out (LOO) scores of the models, or possibly c) using replications of measurements.

a) When the size m *of the data set is sufficiently large with respect to the complexity of the regression, the statistical requirements for hypotheses tests to be valid are satisfied.* The tests must be started from a polynomial that gives a good estimate of the regression, and hence of the noise variance, but may be too complicated: this polynomial is termed *the full polynomial* (Chen and Billings, 1988). It can be chosen to be a polynomial with a suitably large number of the first ranked monomials, for example $m/4$. The following tests (Seber, 1977, Vapnik, 1982, Leontaritis and Billings, 1987) can then be used to discard possibly non-significant monomials of the full polynomial (see also Rivals and Personnaz, 1999, for a comparison with LOO selection).

Let us suppose that a polynomial with q_u parameters, called the "unrestricted" model, contains the regression. We are interested in deciding whether a sub-polynomial with $q_r < q_u$ parameters, called the "restricted" model, also contains the regression. For this purpose, we define the null hypothesis H_0 (null effect of the $q_u - q_r$ parameters), i.e., the hypothesis that the restricted model also contains the regression. We denote by $\boldsymbol{r}_{q_u}$ and $\boldsymbol{r}_{q_r}$ the residual vectors[1] of the unrestricted model and those of the restricted model. When H_0 is true, and under the assumption of homoscedastic (i.e., uncorrelated and with the same variance) Gaussian noise, the following ratio ρ is the value of a Fisher distributed random variable with $q_u - q_r$ and $m - q_u$ degrees of freedom:

$$\rho = \frac{\boldsymbol{r}_{q_r}'\boldsymbol{r}_{q_r} - \boldsymbol{r}_{q_u}'\boldsymbol{r}_{q_u}}{\boldsymbol{r}_{q_u}'\boldsymbol{r}_{q_u}} \frac{m - q_u}{q_u - q_r} \tag{3}$$

The decision to reject H_0 with an *a priori* fixed risk α% of rejecting it while it is true

[1] The k-th component of the residual m-vector $\boldsymbol{r}$ of a polynomial model with the least squares parameters $\boldsymbol{w}_{LS}$ is: $r_k = y_k - g(\boldsymbol{x}_k, \boldsymbol{w}_{LS}) = y_k - (\xi_k)'\, \boldsymbol{w}_{LS}$.

is taken when $\rho > F^{q_u - q_r}_{m - q_u}(1-\alpha)$, where $F^{q_u - q_r}_{m - q_u}$ is the inverse of the Fisher cumulative distribution. When $\rho \leq F^{q_u - q_r}_{m - q_u}(1-\alpha)$, we may conclude that the restricted model is also a good approximation of the regression.[2]

In practice, a sequence of tests is performed starting with the full polynomial as the unrestricted model: if the null hypothesis is not rejected for a restricted model with one monomial less ($q_u - q_r = 1$), this sub-polynomial is taken as new unrestricted model, and so on, until the null hypothesis is rejected.

b) When m *is not large enough, the conditions that are necessary for the tests to be valid may not be fulfilled.* In this case, cross-validation scoring of the candidate polynomials may be preferred for the selection of a polynomial. Recall that, in the case of a model that is linear in its parameters, the LOO errors are expressed analytically as functions of the residuals, and their evaluation does not require m parameter estimations (see Vapnik 1982). The k-th LOO error e_k can be computed according to:[3]

$$e_k = \frac{r_k}{1 - [P_X]_{kk}} \qquad k=1 \text{ to } m \tag{4}$$

where r_k denotes the corresponding residual, and P_X (often called the "hat" matrix), is the orthogonal projection matrix onto the range of X, the experiment matrix $X = [\xi_1 \ldots \xi_q]$. In the general case, the hat matrix is expressed with the generalized inverse X^I of X: $P_X = X X^I$. When X is full rank, $P_X = X (X' X)^{-1} X'$. It is then possible to compute the LOO score as:

$$MSE_{LOO} = \frac{1}{m} \sum_{k=1}^{m} (e_k)^2 \tag{5}$$

and to select the smallest polynomial corresponding to a minimum of the MSE_{LOO}. This model may be biased, but is a reasonable approximation of the regression given the too small data set.

Note that it is often difficult to state *a priori* whether m is large enough.. Because both the tests and the LOO scoring are computationally inexpensive, the designer can always perform both selection procedures. Generally, a too small m can be suspected when the tests and LOO lead to polynomials of very different sizes.

c) Finally, the selection is easier when replications of measurements are available (or when additional replications can be made) for different values of the inputs. In this case, the value of the noise variance can be estimated independently from any model. This leads to a test for lack of fit (Draper and Smith, 1989, Rivals and Personnaz, 2003a), which allows discarding biased polynomials and selection of an unbiased one.

---

[2] In the frequent case where $m - q_u$ is large (> 100), and when H_0 is true, we have approximately:

$$\rho\,(q_u - q_r) = \frac{\mathbf{r}_{q_r}{}' \mathbf{r}_{q_r} - \mathbf{r}_{q_u}{}' \mathbf{r}_{q_u}}{\mathbf{r}_{q_u}{}' \mathbf{r}_{q_u}} (m - q_u) \rightsquigarrow \chi^2(q_u - q_r)$$

[3] The k-th LOO error is the error for the k-th example of the model obtained with a least squares solution when this k-th example is left out from the data set. Note that LOO is termed "moving control" in the book of Vapnik, 1982.

3.3. Strategy for the Choice between Polynomials and Neural Models

If the selected polynomial model satisfies the performance specifications and does not have too many monomials, it must be adopted. The advantage of sticking to a polynomial model is that the unique LS solution is straightforward, as is the statistical analysis of its properties and its mathematical manipulation in general (e.g., the construction of a confidence interval). Conversely, for a model whose output is nonlinear in the parameters (such as a neural network), the same operations involve approximations whose quality depends on the curvature of the solution surface (Seber and Wild, 1989, Antoniadis, 1992).

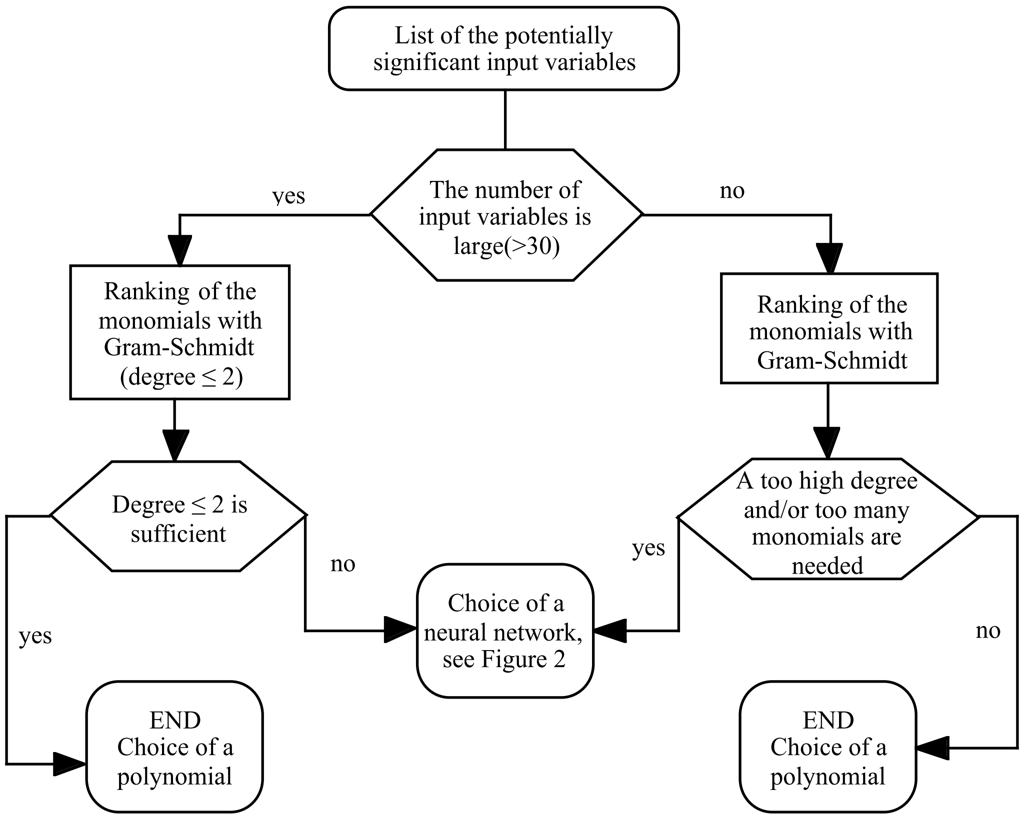

Figure 1. Strategy for the choice between MLPs (polynomials and neural networks):
Left) When the number of input variables is large, a degree larger than 2 can usually not be considered, see Equation (2). Fortunately, in practice, the cross-product terms are able to model interactions between input variables, and the square terms are often nonlinear enough. However, if the performance of such a polynomial is not satisfactory, one can resort to a more parsimonious model like a neural network, which can perform highly nonlinear functions while its number of parameters does not explode. The input variables of the neural network are those appearing in the first ranked monomials only.
Right) When the number of input variables is small, a higher degree can be considered. If the polynomial does not meet the specifications, or if it is satisfactory but has a high degree and/or possesses many monomials, one will resort to a neural network, again with the significant input variables only.

If the selected polynomial does not satisfy the performance specifications, or if it does but has a high degree and possesses many monomials, it reveals that the regression is a complex function, and the designer may choose to try non polynomial models, whose input variables are those appearing among the first-ranked monomials (significant interaction monomials may also be fed to the model):

– **Networks of fixed RBFs or wavelets**: the outputs of these networks are linear in the weighting parameters, so the selection method presented for polynomials can be used to discard the non-significant RBFs or wavelets.
– **Neural networks:** the outputs of these networks are nonlinear with respect to the parameters; for such models, we propose the selection procedure described in Section 4.3.

The procedure for input variable pre-selection using polynomials is extremely useful when the potentially significant input variables are numerous; this is illustrated in Section 5.1 on a real-world example. These considerations concerning the strategy for the choice of the families of functions are summarized on the organization chart of Figure 1.

4. Neural Modeling

Before we go into the details of the selection procedure for neural networks, we need to recall the properties of a nonlinear LS solution, which are obtained using a linearization of the model output with respect to the parameter vector (Seber and Wild, 1989, Rivals and Personnaz, 2000a).

4.1. Least Squares Estimation

A LS estimate $\boldsymbol{w}_{LS}$ of the parameters of a model $g(\boldsymbol{x}, \boldsymbol{w})$ minimizes the cost function:[4]

$$l(\boldsymbol{w}) = \frac{1}{2} \sum_{k=1}^{m} \left(y_k - g(\boldsymbol{x}_k, \boldsymbol{w})\right)^2 \tag{6}$$

Efficient iterative algorithms must be used for the minimization of $l(\boldsymbol{w})$, e.g., the Levenberg-Marquardt algorithm, as in this work, or quasi-Newton algorithms.

The Jacobian matrix Z of the network evaluated at $\boldsymbol{w}_{LS}$ plays an important role in the statistical properties of LS estimation. It is defined as the (m,q) matrix with elements:

$$[Z]_{ki} = \left.\frac{\partial g(\boldsymbol{x}_k, \boldsymbol{w})}{\partial w_i}\right|_{\boldsymbol{w}=\boldsymbol{w}_{LS}} \tag{7}$$

If the family of functions contains the regression and if the noise ε is homoscedastic with variance σ^2, we have the following properties:

a) The covariance matrix of the LS parameter estimator $\boldsymbol{w}_{LS}$ is asymptotically (i.e., as m tends to infinity) given by $\sigma^2 (Z' Z)^{-1}$.

b) The variance of the LS estimator $g(\boldsymbol{x}_a, \boldsymbol{w}_{LS})$ of the regression for an input value $\boldsymbol{x}_a$ is asymptotically given by $\sigma^2 (z_a)' (Z' Z)^{-1} z_a$, where $z_a = \partial g(\boldsymbol{x}_a, \boldsymbol{w})/\partial \boldsymbol{w} \,|\boldsymbol{w}=\boldsymbol{w}_{LS}$.

[4] For a multilayer neural network, due to symmetries in its architecture (function-preserving transformations are neuron exchanges, as well as sign flips for odd activation functions like the hyperbolic tangent), the absolute minimum of the cost function can be obtained for several values of the parameter vector; as long as an optimal parameter is unique in a small neighborhood, the following results are valid.

c) The m-vector $\boldsymbol{r}$ of the residuals with components $\{r_k = y_k - g(\boldsymbol{x}_k, \boldsymbol{w}_{LS})\}$ is uncorrelated with $\boldsymbol{w}_{LS}$ and has the asymptotic property:

$$\frac{\boldsymbol{r}'\boldsymbol{r}}{\sigma^2} \rightsquigarrow \chi^2(m-q)$$

d) An estimate of the $(1-\alpha)$% confidence interval for the regression for any input value $\boldsymbol{x}_a$ is given by:

$$g(\boldsymbol{x}_a, \boldsymbol{w}_{LS}) \pm t_{m-q}(\alpha)\, s \sqrt{(z_a)'\,(Z'\,Z)^{-1}\, z_a} \tag{8}$$

where s^2 is the following asymptotically unbiased estimate of the noise variance σ^2:

$$s^2 = \frac{\boldsymbol{r}'\boldsymbol{r}}{m-q} \tag{9}$$

and where t_{m-q} is the inverse of the Student cumulative distribution with $m-q$ degrees of freedom.

In the case of linear models, the Jacobian matrix Z reduces to the experiment matrix X, and properties (a-d) are exact whatever the value of m > q.

e) If the null hypothesis is true, the ratio ρ of Equation (3) is approximately Fisher distributed; provided the considered networks are nested, the same tests as discussed for the polynomials can be performed (Bates and Watts, 1988, Seber and Wild, 1989).

Finally, we showed (see Rivals and Personnaz, 2000a,b) that the k-th LOO error e_k can be *approximated* with:

$$e_k \approx \frac{r_k}{1 - [P_Z]_{kk}} \qquad k=1 \text{ to } m \tag{10}$$

where P_Z is the orthogonal projection matrix on the range of Z. It is hence possible to compute an economic approximate LOO mean square error, denoted by MSE_{ALOO}, which requires one parameter estimation only. The approximation of Equation (10) holds for any network, i.e., not only for a network which contains the regression.

Note that all the above results are asymptotic; if m is not large, the curvature of the solution surface is not negligible, and the above approximations will be rough. Moreover, the network must be well-conditioned enough for the above expressions to be reliably computed.

4.2. Jacobian Conditioning and Model Approval

The inverse of the squared Jacobian is involved in the expression of the covariance matrix, and hence in that of the confidence intervals. The most robust way to compute this inverse is to perform a singular value decomposition $Z = U\,\Sigma\,V'$ (Golub and Van Loan, 1983), where the elements $\{[\Sigma]_{ii}\}$ denoted by $\{\sigma_i\}$ are the singular values of Z. However, when Z is ill-conditioned, that is, when Z's condition number $cond(Z) = max\{\sigma_i\}/min\{\sigma_i\}$ is large, $(Z'\,Z)^{-1}$ cannot be computed accurately. Since the elements of the Jacobian represent the sensitivity of the network output with respect to its parameters, the ill-conditioning of Z is generally also a symptom that some parameters are superfluous, i.e., that the network is too complex and that the LS solution overfits[5] (Rivals and Personnaz, 1998).

[5]. Such a situation might also correspond to a relative minimum. To check the conditioning of Z thus also helps to discard a neural network trapped in a relative minimum, and leads to retrain this candidate with different initial parameter values.

In practice, a well known approach to limiting ill-conditioning is to center and normalize the input variable values. However, ill-conditioned neural candidates whose Jacobian condition number still exceeds 10^8 (for usual computers) should not be approved (Rivals and Personnaz, 2000a,b and 2003b). This model approval criterion therefore allows discarding overly complex networks.

This criterion also applies to models that are linear in their parameters, such as polynomials. However, unless the degree of the polynomial is very large and the size m of the data set very small, ill-conditioning will seldom appear, provided the input variable values are properly centered and normalized (this is why we did not mention the problem of the conditioning of X in Section 3).

4.3. Neural Model Construction and Selection Procedure

Networks with a single hidden layer of neurons with hyperbolic tangent activation function and a linear output neuron possess the universal approximation property. Thus, we choose not to consider more complex architectures, e.g., networks with multiple hidden layers and/or a different connectivity. Therefore, determining the smallest unbiased network reduces to the problem of identifying the significant input variables and the minimum necessary number of hidden neurons. We propose the following procedure, which consists of two phases:

a) An additive (or growing) estimation and approval phase of networks with the input variables appearing in the first ranked monomials, and an increasing number of hidden neurons.
b) A subtractive selection phase among the approved networks using statistical tests, which removes possibly non-significant hidden neurons and input variables.

The principle of the procedure is summarized on the organization chart of Figure 2.

4.3.1. Additive Phase

In an additive phase, the most important one, we consider candidate neural networks with the input variables pre-selected during the polynomial modeling. Networks with an increasing number of hidden neurons are trained. Note that several minimizations must be performed for each candidate network in order to increase the chance of reaching an absolute minimum, i.e., a LS solution. This additive phase is stopped when the Jacobian matrix Z of the candidates becomes too ill-conditioned ($cond(Z) > 10^8$). The weights, RSS and MSE_{ALOO} of the approved candidates are stored in memory for the subsequent phase.

4.3.2. Subtractive Phase

We have shown that the LOO score alone is often not sufficient to perform a good selection (see Rivals and Personnaz, 1999). We propose to perform statistical tests, and to use the MSE_{ALOO} only for the choice of a full network model. The full network model is chosen as the most complex approved network before the MSE_{ALOO} starts to increase. This network can then be considered as a good approximation of the regression if the ratio of its MSE_{ALOO} to its mean square training error $MSE_{train} = 2/m\ l(\boldsymbol{w}_{LS})$ is of the order of one. Note that, when replications of measurements are available, the test for lack of fit (Seber and Wild, 1989) allows the selection of a full network.

The Fisher tests of Section 4.1 are used to establish the usefulness of all the neurons of the full network. A sequence of tests is performed starting with the full network as the unrestricted model, and with the candidate with one neuron less as restricted model. If the null hypothesis is not rejected, the restricted model is taken as new unrestricted one, and so on, until a null hypothesis is rejected. The RSS of all sub-networks are available after the additive phase, so this series of tests is inexpensive. If there are still doubts about the significance of any of the input variables, additional Fisher tests can be performed on the corresponding sub-networks (Rivals and Personnaz, 2003a).

Note that this subtractive phase is only a refinement of the additive one: the removal of non-significant input variables in the polynomial phase and the approval criterion usually prevents the full network from having too many non-significant input variables and superfluous hidden neurons.

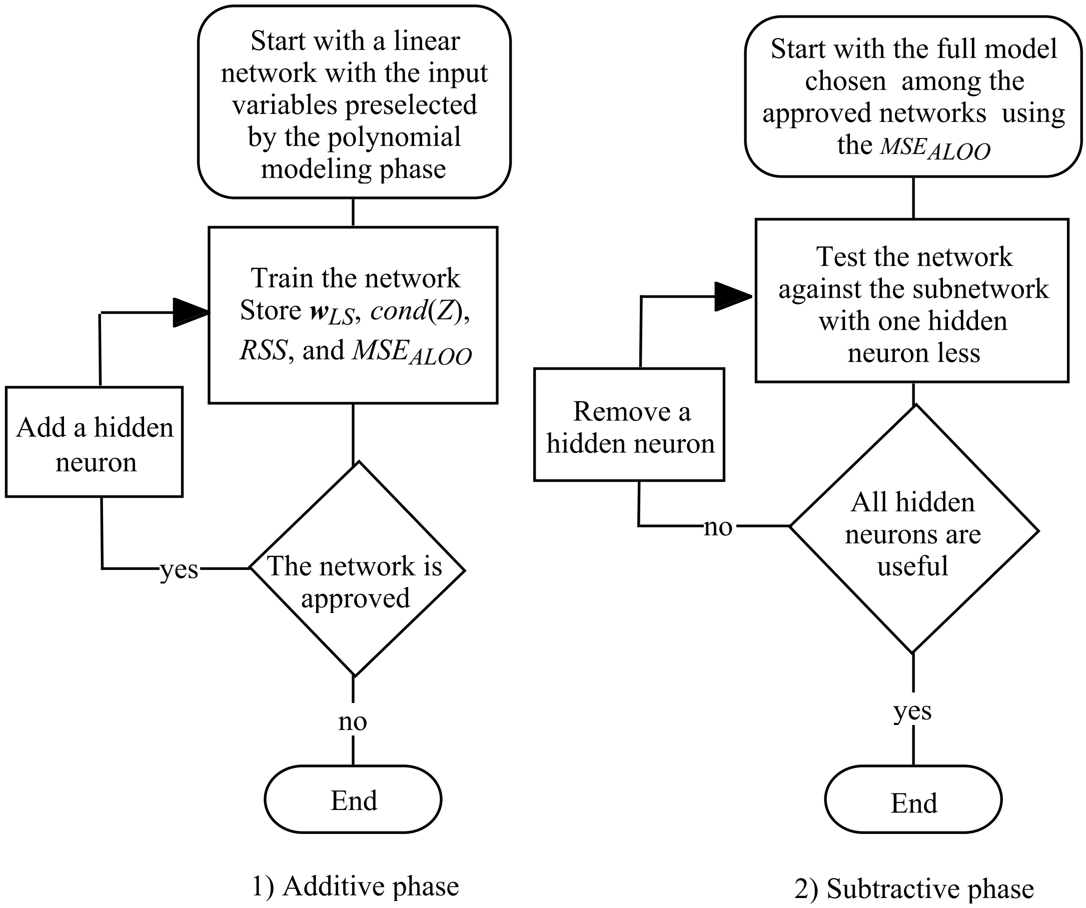

Figure 2. Proposed neural modeling procedure.

Finally, this subtractive phase offers several advantages with respect to pruning methods, such as OBD (for Optimal Brain Damage, see Le Cun et al., 1990) and OBS (for Optimal Brain Surgeon, Hassibi and Stork, 1993). It has been shown that these methods are related to the tests of linear hypotheses (Anders, 1997, Rivals and Personnaz, 2003a), but the notion of risk is absent, and thresholds must be set in an *ad hoc* fashion. In our method, the threshold for the rejection of the null hypothesis depends automatically on the risk chosen, the size of the data set, the number of the network parameters and the number of tested parameters.

5. Application to Real-World Modeling Problems of the NIPS2000 Competition

The performance of the specific neural modeling procedure of Section 4 has already been evaluated on several artificial problems (Rivals and Personnaz, 2000b). Here, we wish to stress the importance of the polynomial modeling presented in Section 3. For this purpose, we present here the results obtained on regression problems proposed at the NIPS2000 Unlabeled Data Supervised Learning Competition. We entered the competition for all four regression problems (Problems 3, 4, 8, 10), and won all of them. These are real world problems, and at the time of the competition, the nature of their input variables, was unknown to the candidates.[6] Here, we present Problems 8 and 3, which are representative of two typical situations: the complexity of Problem 8 arises from the large number of potentially significant inputs, whereas that of Problem 3 is due to the high non-linearity of the regression. Moreover, data are missing in the two other problems: this is an issue that we do not tackle in the present paper.

5.1. Molecular Solubility Estimation (Problem 8 of the Competition)

This problem involves predicting the solubility of pharmaceutical compounds based on the properties of the compound's molecular structure. The input variables are 787 real-valued descriptors of the molecules, based on their one-dimensional, two-dimensional, and electron density properties. The training set contains 147 examples; the test set contains 50 examples. The mean square errors on the training and test sets are denoted by MSE_{train} and MSE_{test} respectively. With such a small training set, a reasonable solution to the problem can be obtained only if the output can be explained by a small number of the input variables. A preliminary removal of the non-significant inputs using polynomials is therefore necessary. As a matter of fact, an affine model with all the variables (except 12 of them which are zero!), and whose parameters are the least squares solution estimated with the generalized inverse, has a MSE_{test} of 4.10. This is a poor performance since the MSE_{test} of a constant model (the mean of the process outputs) equals 7.68.

5.1.1. Polynomial Modeling

The whole training set is used for training. Because the number of input variables (775) is very large, polynomials of degree $d = 3$ are built omitting cross-product monomials, as they would require too much memory. We consider m as small, and perform the selection on the basis of MSE_{LOO}. The selected polynomial possesses 6 monomials among the initial 2 325: ${x_{17}}^2$, x_{36}, x_{159}, ${x_{172}}^2$, ${x_{222}}^3$, ${x_{565}}^3$. Table 1 summarizes the results obtained with this polynomial.[7] Recall that q is the number of parameters of the model, here the number of monomials plus one.

---

6. See: `http://q.cis.uoguelph.ca/~skremer/NIPS2000/` for further details concerning the highest scores and about the data. A goal of the competition was to test whether the use of unlabeled data could improve supervised learning, so in addition to the training and test sets, an additional unlabeled set was available. The results presented in this paper were obtained without using this unlabeled set.

7. During the competition, we obtained a score of 1.376846 (the highest score), that is a MSE_{test} of $5.27\ 10^{-1}$, using Fisher tests. The above value of $5.23\ 10^{-1}$ corresponds to the squared inverse of a score of 1.382578 we obtained after the competition, thanks to the use of the MSE_{LOO} for the selection, the data set size m being rather small.

d	q	MSE_{train}	MSE_{LOO}	MSE_{test}
3	7	$2.66\ 10^{-1}$	$2.89\ 10^{-1}$	$5.23\ 10^{-1}$

Table 1. Selected polynomial.

The ratio of the MSE_{LOO} to the MSE_{train} of the selected polynomial is close to 1; its MSE_{LOO} is indeed a rough estimate of the MSE_{test}.

5.1.2. Neural Modeling

The monomials selected above involve 6 input variables: x_{17}, x_{36}, x_{159}, x_{172}, x_{222}, x_{565}. These inputs were fed to networks with one layer of an increasing number of hidden neurons with a hyperbolic tangent activation function, and a linear output neuron. The networks were trained many times with a Levenberg-Marquardt algorithm in order to increase the chance of reaching an absolute minimum. The results are shown in Table 2.

N_{hid}	MSE_{train}	MSE_{ALOO}	$cond(Z)$
0	$3.77\ 10^{-1}$	$4.77\ 10^{-1}$	9.7
1	$3.60\ 10^{-1}$	$4.50\ 10^{-1}$	$9.5\ 10^{1}$
2	$2.41\ 10^{-1}$	$3.84\ 10^{-1}$	$1.1\ 10^{3}$
3	$2.03\ 10^{-1}$	–	$1.5\ 10^{18}$

Table 2. Training of neural networks with the 6 pre-selected input variables.

The condition numbers of the networks with more than 2 hidden neurons are larger than 10^8, so these networks were not approved. A Fisher test with risk 5% showed that 2 hidden neurons were necessary. The performance of the selected 2 hidden neuron network is summarized in Table 3. It is a little better than that of the best polynomial (the MSE_{test} equals $4.17\ 10^{-1}$ instead of $5.23\ 10^{-1}$). However, the test set of only 50 examples chosen by the organizers of the competition was too small to determine whether this improvement is really significant (large variance of the MSE_{test}).

q	MSE_{train}	MSE_{ALOO}	MSE_{test}
15	$2.41\ 10^{-1}$	$3.84\ 10^{-1}$	$4.17\ 10^{-1}$

Table 3. Selected neural network (6 pre-selected input variables, 2 hidden neurons).

5.1.3. Discussion

This example with many potentially significant inputs is a typical illustration of the necessity of using polynomials before resorting to neural networks: the complexity of the problem was due to the large number of potentially significant input variables rather than to the non-linearity of the regression.

Note that, on the whole and on this particular problem, B. Lucic obtained the second best results using "CROMRsel", a descriptor selection algorithm based on multi-regression models. According to the text available at the competition website, CROMRsel uses polynomials and an orthogonalization method. This algorithm is hence probably close to our procedure for models that are linear in their parameters. However, CROMRsel does not use statistical tests, and there is no mention of neural networks.

Finally, aqueous solubility is a key in understanding drug transport, and the polynomial or the neural network could be used to identify compounds likely to possess desirable pharmaceutical properties.

5.2. Mutual Fund Value Prediction (Problem 3 of the Competition)

This problem consisted of predicting the value of a mutual fund of a popular Canadian bank given the values of five other funds. The values were taken at daily intervals for a period of approximately three years, week-ends and holidays excluded. The training and the test sets both contain 200 examples.

5.2.1. Polynomial Modeling

Because the training set size m was rather large, the selection of the optimal polynomial could be performed with statistical tests. Because the number of inputs was small, we could build a high degree polynomial with all its monomials; we chose a degree 4 (126 monomials). Starting from a full polynomial of the first $m/4 = 50$ ranked monomials, the Fisher tests selected 44 monomials. Some of them were of degree 4, and all 5 inputs were involved in the selected monomials. Table 4 summarizes the results obtained with the selected polynomial.[8] For comparison, the MSE_{test} of a constant model equals 5.17, and Table 4 also displays the results of an affine model with all 5 input variables.

d	q	MSE_{train}	MSE_{LOO}	MSE_{test}
1	6	$7.22\ 10^{-2}$	$7.74\ 10^{-2}$	$6.33\ 10^{-2}$
4	45	$6.72\ 10^{-3}$	$1.26\ 10^{-2}$	$1.14\ 10^{-2}$

Table 4. Selected polynomial, and degree one polynomial for comparison.

Because the necessary degree and the number of monomials are relatively large, this polynomial was put into competition with neural networks.

5.2.2. Neural Modeling

Table 5 summarizes the results obtained when training networks with all 5 input variables and an increasing number of hidden neurons on the whole data set. The networks with more than 4 hidden neurons were not approved, because of the too large condition number of their matrix Z.

N_{hid}	MSE_{train}	MSE_{ALOO}	$cond(Z)$
0	$7.22\ 10^{-2}$	$7.74\ 10^{-2}$	$1.1\ 10^{1}$
1	$4.60\ 10^{-2}$	$5.16\ 10^{-2}$	$1.2\ 10^{2}$
2	$2.13\ 10^{-2}$	$3.06\ 10^{-2}$	$2.9\ 10^{2}$
3	$1.38\ 10^{-2}$	$1.82\ 10^{-2}$	$2.8\ 10^{3}$
4	$1.07\ 10^{-2}$	$1.75\ 10^{-2}$	$1.5\ 10^{3}$
5	$8.14\ 10^{-3}$	–	$3.0\ 10^{8}$

Table 5. Training of neural networks.

[8] The value of the MSE_{test} of the selected polynomial corresponds to a score of 9.386007, obtained when the competition was over. During the competition, we used only neural networks for this problem.

A Fisher test with risk 5% shows that all 4 hidden neurons are necessary. Further tests performed on the 5 input variables show that they are indeed significant. Table 6 presents the results[9] obtained with the selected network.

q	MSE_{train}	MSE_{ALOO}	MSE_{test}
29	$1.07\ 10^{-2}$	$1.75\ 10^{-2}$	$1.24\ 10^{-2}$

Table 6. Selected neural network (all 5 input variables, 4 hidden neurons).

From the point of view of the MSE_{test}, these results are a little less satisfactory than those obtained with the selected polynomial (the value of the MSE_{test} is $1.14\ 10^{-2}$ with the polynomial, against $1.24\ 10^{-2}$ with the neural network). However, the neural network is well conditioned, and its MSE_{train} is close to the MSE_{test}: it is hence very unlikely to overfit, whereas the MSE_{train} of the polynomial is much smaller than its MSE_{test}.

5.2.3. Discussion

In contrast to the previous example, the complexity of the present problem is not due to a large number of inputs, but due to the high non-linearity of the regression: 4 hidden neurons or a degree 4 are needed to represent the behavior of the process. This example also illustrates that polynomials are powerful models, provided their monomials are properly selected.

Finally, the excellent performance obtained with both models confirms that one mutual fund's price is related to the other (even though this relation is complex). This is quite a reasonable assumption because the commodities in which they invest are the same, and because they are affected by the same general economic trends (boom/bust).

6. Conclusion

The success of the proposed method for practical modeling problems is due to the following features:

– beginning with polynomial models provides an evaluation of the complexity of the regression at low computational cost, and leads to the removal of the less significant inputs;
– thanks to this preliminary selection, the training of neural networks is facilitated;
– networks with far too many hidden neurons are automatically discarded on the basis of the condition number of their Jacobian matrix, so that no test set is necessary to discard overfitting networks;
– superfluous neurons and non-significant inputs are removed at low computational cost using Fisher tests; the main advantage of these tests over pruning methods like OBD or OBS is that the decision to discard a neuron or an input variable takes into account the fixed risk, the size of the data set, and the number of parameters.

[9] The value of the MSE_{test} of the selected neural network corresponds to the squared inverse of the score of 8.982381 we obtained at the competition, which was the highest score. Note that we made multiple submissions at the NIPS competition resulting in scores between 7.56 and the latter value, depending on the random initialization of the selected 4 hidden neuron network parameters. The second best score obtained at NIPS equals 8.849004.

As a result of the previous features, the procedure is essentially constructive. These advantages can be exploited in the numerous applications involving a large number of potential descriptors: solubility prediction (topological, geometrical, electronic descriptors), avalanche forecasting (weather and snow factors), economical and financial problems, etc.

Further research includes improving the diagnosis on whether a data set size is too small, possibly by analyzing the correlations of the residuals. We will also focus on the extension of the method to dynamic modeling, where the fact that the inputs are correlated with the outputs and the large variety of predictors (input-output versus state-space, non-recursive versus recursive) make the selection task more difficult.

Acknowledgements

We are very grateful to Stefan C. Kremer and to the other organizers of the NIPS2000 workshop Unlabeled Data Supervised Learning Competition.

References

U. Anders. Neural network pruning and statistical hypotheses tests. In Progress in Connectionnist-Based Information Systems (Addendum), Proceedings of ICONIP'97. Springer, Berlin: 1-4, 1997.

U. Anders and O. Korn. Model selection in neural networks. Neural Networks 12: 309-323, 1999.

A. Antoniadis, J. Berruyer and R. Carmona. Régression non linéaire et applications. Economica, 1992.

D. M. Bates and D. G. Watts. Nonlinear regression analysis and its applications. John Wiley & Sons, 1988.

S. Chen and S. A. Billings. Prediction-error estimation algorithm for non-linear output-affine systems. Int. Journal of Control 47(1): 309-332, 1988.

S. Chen, A. Billings and W. Luo. Orthogonal least-squares methods and their application to non-linear system identification. Int. Journal of Control 50(5): 1873-1896, 1989.

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

G. H. Golub and C.F. Van Loan. Matrix computations. John Hopkins University Press, Baltimore, 1983.

B. Hassibi, D. Stork, G. Wolff and T. Watanabe. Optimal brain surgeon: Extensions and performance comparisons. In Advances in Neural Information Processing Systems 6. Morgan Kaufman, San Mateo, CA: 263-270, 1994.

T. Hastie, R. Tibshirani and J. Friedman. The elements of statistical learning. Springer, New York, 2001.

T.-Y. Kwok and D.-Y. Yeung. Constructive algorithms for structure learning in feedforward neural networks for regression problems. IEEE Transactions on Neural Networks 8(3): 630-645, 1997a.

T.-Y. Kwok and D.-Y. Yeung. Objective functions for training new hidden units in constructive neural networks. IEEE Transactions on Neural Networks 8(5): 1131-1148, 1997b.

Y. Le Cun, J. S. Denker and S. A. Solla. Optimal brain damage. In D. S. Touretzki (Ed.), Advances in Neural Information Processing Systems, vol. 2. Morgan Kaufmann, San Mateo, CA: 598-605, 1990.

I. J. Leontaritis and S. A. Billings. Model selection and validation methods for non-linear systems. Int. J. Control 45(1): 311-341, 1988.

J. Moody. Prediction risk and architecture selection for neural networks. In V. Cherkassy, J. H. Friedman and H. Wechsler (eds.), From statistics to neural networks: theory and pattern recognition applications, NATO ASI Series, Springer Verlag, 1994.

I. Rivals and L. Personnaz. 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, 1998.

I. Rivals and L. Personnaz. On cross-validation for model selection. Neural Computation 11(4): 863-870, 1999.

I. Rivals and L. Personnaz. Construction of confidence intervals for neural networks based on least squares estimation. Neural Networks 13(1): 463-484, 2000a.

I. Rivals and L. Personnaz. A statistical procedure for determining the optimal number of hidden neurons of a neural model. Proceedings of the Second ICSC Symposium on Neural Computation NC'2000, Berlin-Germany, 2000b.

I. Rivals and L. Personnaz. Neural network construction and selection in nonlinear modeling. IEEE Transactions on Neural Networks, to appear 2003a.

I. Rivals and L. Personnaz. Jacobian conditioning analysis for model validation. Neural Computation, to appear 2003b.

G. A. F. Seber. Linear regression analysis.Wiley, New York, 1977.

G. A. F. Seber and C. Wild. Nonlinear regression. Wiley, New York, 1989.

H. Stoppiglia. Méthodes statistiques de sélection de modèles neuronaux ; applications financières et bancaires. Thèse de Doctorat de l'Université Paris 6, 1997.

V. Vapnik. Estimation of dependences based on empirical data. Springer Verlag, New York, 1982.

J.-P. Vila, V.Wagner and P. Neveu. Bayesian nonlinear model selection and neural networks: a conjugate prior approach. IEEE Transactions on Neural Networks 11(2): 265-278, 1999.