ON-LINE VALIDATION OF MEASUREMENTS ON JET ENGINES USING AUTOMATIC LEARNING METHODS

Pierre Dewallef Olivier Léonard

University of Liège - LTAS (Aerospace Laboratory) Turbomachinery Group
Chemin des chevreuils 1 (B52/3) - 4000 Liège - BELGIUM

p.dewallef@ulg.ac.be o.leonard@ulg.ac.be

Abstract

Nowadays, turbine engine tests are processed using an open loop, i.e. the measurements are verified and treated *a posteriori*, sometimes weeks or months after the end of the test. The purpose of the present project is to develop a new methodology which enables real time detection of faulty measurements and the suppression of the source of these faults during the test.

The validation of the measurements is achieved by a "robust" parameter identification [1]. Such a method is called robust in the sense that it can cope with 20 to 30% of faulty measurements. The robustness is insured by a distribution of the measurement noise, as introduced by Huber [7, 8], that takes into account the possibility of faults.

The purpose of a parameter identification is to find the set of parameters which has most likely generated the measurements observed on the process. This leads to an optimisation problem that has to be solved for the parameters. The measurements are linked to the parameters through a non-linear model, leading to a large system of equations for modern jet engines. If no physical model of the process can be made available or if this model is too complex to allow real time validation, automatic learning methods may provide a solution:

- either a mathematical representation is generated, directly based on the measurements (on-line learning),
- or a database is first generated, based on the existing (but expensive) physical model, the database being subsequently used to build a statistical model (off-line learning).

Neural networks seem to be very suitable for modeling the behavior of turbojets, avoiding the resolution of a time-consuming non-linear system. In this paper neural networks are tested to generate a mathematical representation of a single flow, single spool and variable geometry nozzle turbojet, from a data base of "measurements" generated by a physical model of the engine. Only the off-line learning approach is considered.

Statement of the problem

The problem of fault identification can be stated as follows: given a process (i.e. a turbojet, a chemical process, ...) on which several measurements are taken, one would like to verify whether some of these measurements are faulty and whether the process works correctly.

A process can be described by a set of independent parameters θ (for our turbojet, the inlet engine temperature and pressure, the fuel flow, the area of the exhaust nozzle, ...). The exact value of these parameters θ is not available, but may be estimated using a set of measurements Y and any model of the process. This is called a parameter identification: a set $\tilde{\theta}$ is sought that maximizes the probability of observing the measurements Y.

A robust validation procedure has been developed by the authors and is described in [1]. This procedure has been tested with a physical model of the engine. The results demonstrated that multiple sensor faults can be detected. It is now proposed to investigate how a model can be built using automatic learning methods such as artificial neural networks (ANN), and how this mathematical representation can be matched to the robust validation procedure.

Model development

A physical model of a jet-engine is based on mass, momentum and energy balances, that have to be satisfied. However such a model, as good as it is, must be fitted to the available measured data to be sufficiently accurate. Once this calibration has been performed the model can be used to predict the performance of the engine.

The development of a physical model is usually a complex and time consuming task because it implies to gather a large amount of information from different sections of the engine (compressor, turbine, combustion chamber, ...). Moreover, such a model will result in a high index non-linear system of equations which may be difficult to solve.

These reasons lead to the idea of testing the ability of a neural network to replace the physical model. Recent data acquisition systems make it possible to store a huge amount of measurements from an engine. Automatic learning methods allow to make use of these data to train a mathematical representation of the engine, i.e. to infer the mapping between the inputs θ and the output data Y.

Neural Network Structure

An artificial neural network consists of a pool of simple processing units which communicate by sending signals to each other over a large number of weighted connections [14].

Following this definition, an artificial neural network (ANN) can be considered as a non-linear regression technique. The mapping of the network is determined by its structure (number of units and layers) and a set of parameters (weights). A very short introduction to neural networks is proposed hereafter. A more complete description can be found in [12], [13], [14] and [2].

The structure of a neural network can be summarized as follows: neurons (fig. 1) are the basic processing units of the network. Each unit performs a relatively simple job: receive input a_i from neighbours or external sources and from this deliver an output z_i signal which is propagated to other units. Neurons are also characterized by an activation function g

$$z_i = g(a_i) \tag{1}$$

In most cases, it is assumed that each neuron provides an additive contribution to the input of its neighbours with which it is connected. In other words, the total input for unit i is simply the weighted sum of the separate outputs from each of the connected units plus an offset term (fig. 1).

$$a_i = \sum_j w_{ij} z_j + \theta_i \tag{2}$$

The network topology considered here is the so-called feed-forward, where the data flows from input to output neurons and only this way. The data processing can extend over multiple layers of neurons (fig. 3).

Let:

- θ_k for $k = 1...p$ be a set of parameters,
- $\tilde{\theta_k}$ for $k = 1...p$ be the set of input parameters of the model,
- Y_j for $j = 1...c$ be a set of data measured on the process,
- $\tilde{Y_j}$ for $j = 1...c$ be a set of measurements predicted by a model of the process for a given $\tilde{\theta}$.

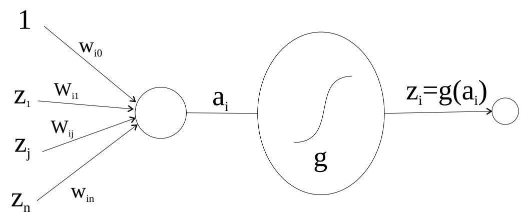

Figure 1: Single neuron

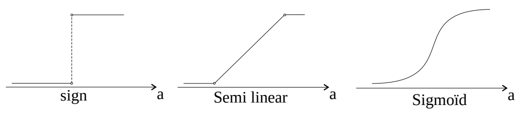

Figure 2: Different activation functions

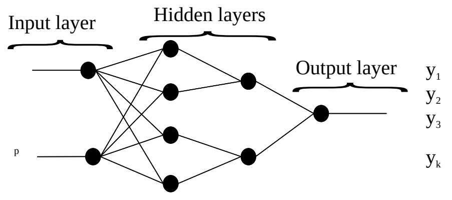

Figure 3: Artificial Neural Network scheme

Neural Network training

Training a network is determining the mapping between the inputs and the outputs. Provided the structure of the network, the training phase reduces to the determination of the weights characterizing the connections between neurons, on the basis of a collection of available data.

The database being $B(\theta^l, Y^l)$ and containing n items (an item is the set of both input and output data), we are looking for the underlying generator of the data. In other words, training a neural network is finding the weights which maximize the probability $P(\theta, Y)$ of observing Y together with θ:

$$\begin{aligned} L &= P(\theta, Y) = P(Y|\theta) \times P(\theta) && (3) \\ &= \prod_l P(Y^n|\theta^l) \times P(\theta^l) && (4) \end{aligned}$$

Defining the error function E:

$$\begin{aligned} E &= -\ln L && (5) \\ &= -\sum_l \ln P(Y^l|\theta^l) - \underbrace{\sum_n P(\theta^l)}_{constant} && (6) \end{aligned}$$

Minimizing P corresponds to the minimization of:

$$E = -\sum_l \ln P(Y^l|\theta^l) \qquad (7)$$

A measurement Y_j can be expressed as a function of the predicted measurement $\tilde{Y}_j(\theta; w)$:

$$Y_j = \tilde{Y}_j(\theta; w) + \epsilon_j \qquad (8)$$

where ϵ_j is the measurement noise. Therefore, if gaussian noise is assumed, the error function is (for details see [6]):

$$E = \sum_{l=1}^{n} \sum_{j=1}^{c} (\tilde{Y}_j(\theta^l; w) - Y_j^l)^2 \qquad (9)$$

and the network weights w_{ij} are determined to minimize this error function.

Another aspect that has to be considered is the number of neurons and the number of layers in the network. The difficulty of this step should not be underestimated. Enough neurons may be introduced to achieve sufficient accuracy but too many neurons leads to over-fitting problems. Overfitting is a generic problem encountered in automatic learning. It generally appears when the model extracted is too complex with respect to the information provided in the learning set, resulting in a mapping with dramatic oscillations between the measured data.

The procedure followed in this paper is to increase the number of neurons and the number of hidden layers until an optimal structure has been reached. The optimal structure is the one which provides the best results for data not belonging to the learning set, providing the best generalization. One way to achieve such a verification is to split the database in two sets: the *learning set* which is used to train the network and the *test set* which is used to test the network on unseen data. The number of neurons and layers is increased until a minimum on the test set error is observed. A complete description of this technique can be found in [2].

Other optimization methods are available based on a bayesian inference of regularization parameters to determine the optimal geometry (see [12] and [13]). Even though these approaches are supposed to perform better, they require more coding work and therefore will not be considered here.

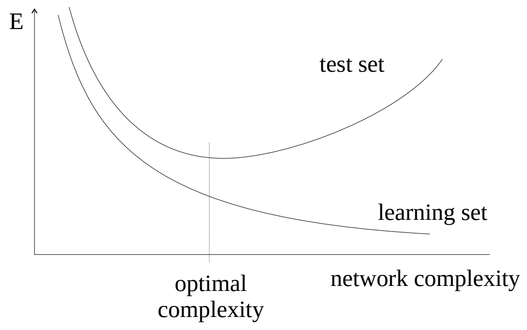

Figure 4: Neural Network Generalization

Robust Validation Procedure

Now that a mathematical representation of the engine is available, we will concentrate on another aspect of the measurement validation, namely the development of an efficient test to compare the measurements performed on the engine to the outputs of its model.

This step is performed through the determination of the input parameters ($\tilde{\theta}$).

According to the principle of maximum likelihood the probability $P(Y, \tilde{\theta})$ is maximized by finding the parameters leading to the most likely solution. As for the training of the neural network, this is done by minimizing an error function. A usual approach is to consider that measurement noise is gaussian, leading to the least square method which gives very poor results when measurements faults are encountered.

Some improvements are brought by using *robust* estimators. An efficient approach has been proposed by Huber and later Poljak & Tsypkin, based on the *a priori* information contained in the noise distribution, and has led to a new error function. We will only mention here the expression of this function. More details can be found in [1], [7], [8] and [9].

The error function is:

$$E = \sum_{j=1}^{c} E_j(\epsilon_j) \qquad \text{where} \tag{10}$$

$$E_j(\epsilon_j) = \begin{cases} -\ln(k_1) + \dfrac{\epsilon_j^2}{2\sigma_j^2} & \text{if } |\epsilon_j| \leq \Delta_j \\ -\ln(k_2) + \dfrac{|\epsilon_j|}{k_3} & \text{if } |\epsilon_j| > \Delta_j \end{cases} \tag{11}$$

where k_1, k_2, k_3 and Δ are positive constants functions of σ and satisfying the continuity of $E(\epsilon)$ and $E'(\epsilon)$.

This distribution is very suitable for modeling measurement noise where the variance is limited for valid measurements but may take any value in the case of faulty ones.

Parameter observability and measurement criticity

Before any parameter identification, it is worth it

1. to verify that the number and the type of measurements allow the determination of the parameters (observability of the parameters),
2. to investigate how each measurement contributes to the estimation of these parameters (criticity of the measurements).

Let the function f represents the non linear model of the process

$$\tilde{Y} = f(\tilde{\theta}) \tag{12}$$

and J the jacobian of the system

$$J_{jk} = \frac{\partial \tilde{Y}_j}{\partial \tilde{\theta}_k} \Rightarrow \Delta\tilde{Y} = J\Delta\tilde{\theta} \tag{13}$$

The parameters are observable if the relation (13) can be inverted, i.e. the rank of J must be equal to the number of parameters p. When the model is inferred from the measurements, it is clear that input parameters can always be determined by the set of output parameters.

Besides, a measurement is said to be critical if removing this measurement reduces the rank of J by 1, i.e. the line of J corresponding to a critical measurement is linearly independent of the other lines (see [4, 5]). The criticity of a measurement is affected by the standard deviation on the latter. By specifying a standard deviation on each measurement and on each parameter (through *a priori* information), constraints are added to the identification problem.

A criticity of 1 means that the corresponding measurement is used without any redundancy to determine one parameter and therefore this measurement cannot be validated. On the contrary the less the criticity is, the more the corresponding measurement can be validated. The evaluation of the criticities is mandatory for the validation procedure and is performed through a simple computation of the influence matrix G defined as

$$\Delta\tilde{Y} = G\Delta Y \tag{14}$$

Assuming a quadratic form of the objective function in the vicinity of the optimum $(\tilde{\theta_0})$ the matrix G is computed by:

$$G = J(J^T\Sigma J)^{-1}J^T\Sigma^T \tag{15}$$

where the $(c \times c)$ diagonal matrix Σ is defined as

$$\Sigma_{jj} = \begin{cases} \frac{1}{\sigma_j^2} & \text{if } |\epsilon_j| \leq \Delta_j \\ 0 & \text{if } |\epsilon_j| > \Delta_j \end{cases} \tag{16}$$

The diagonal terms of G reflect the criticity of each measurement Y_j. An interesting property of this influence matrix is that the trace of G is equal to the number of parameters p (see [11] and [12]).

$$\text{trace }(G) = p \tag{17}$$

The standard deviation $\tilde{\sigma_j}$ associated with validated values $\tilde{Y_j}$ depends on the influence of measured values on identified values and on standard deviations of each measurements by:

$$\tilde{\sigma_j} = \sqrt{\sum_{i=1}^{c}\left(\frac{\partial\tilde{Y_j}}{\partial Y_i}\sigma_i\right)^2} = \sqrt{\sum_{i=1}^{c}(G_{ji}\sigma_i)^2} \tag{18}$$

Information on the accuracy of identified values is of great interest for testing whether the accuracy of the results is compatible with their utilization (predictive maintenance, engine fault detection, ...).

Application

It is proposed to test the ability of a neural network to achieve a real time robust validation on unseen measurements. This test implies the following operations:

- generate a database with a physical model (1000 items),
- add gaussian noise on each measurement,
- split up the database into a learning set (700 items) and a test set (300 items). The learning set is used to train the network and the test set is used to test the generalization of the network,
- use a robust estimator based on (10) with the ANN model to validate the measurements.

The results are also compared to those obtained by a least squares approach in order to highlight the need of a robust estimator.

The measurements considered are summarized on table 1. Input parameters are T2, P2, T4, A8.

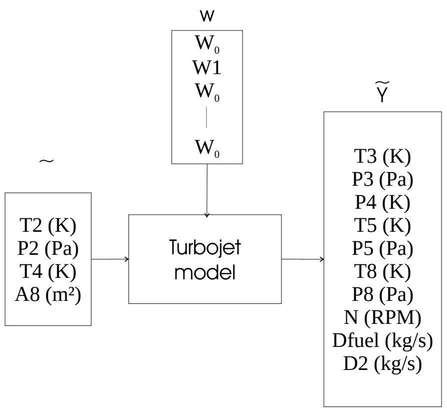

Figure 5: Identification applied to SR30

It is important to mention that the procedure is made up of two distinctive identifications:

- the first one, which is performed off-line as a preliminary phase, is the identification of the network weights (w) according to the database and minimizing (9),
- the second one is the measurement validation of one sample at a time by identifying the input parameters θ of the network. This step can be performed on-line provided that the validation procedure is fast enough, namely the minimization of 10.

Three different situations will be considered in this test case:

1. no sensor faults,
2. a large sensor fault on T_4, the turbine entry temperature,
3. 4 sensor faults - in order to test the robustness of the method.

Results

The neural network has been trained using the database, and the number of units and layers has been increased until an optimal structure was found. This structure is made of two hidden layers with 10 neurons each, meaning 270 weights to be determined. The accuracy of the network is close to the measurements accuracy, and it is assumed that noise introduced by the network does not spoil the results. This assumption will be confirmed during the tests.

When no engine faults and no sensor faults occur, there are 14 measurements, 4 parameters and $14 - 4 = 10$ degrees of freedom. A measurement is rejected if $|Y_i - \tilde{Y}_i| > 3\sigma_i$, corresponding to a confidence interval of 99% (the risk of concluding falsely that Y_j is faulty is 1%).

Results for the 3 situations are proposed in table 2. For the sake of simplicity, only reduced variables are mentioned, 3 types of reduced variables are used:

$$\begin{aligned}
\chi_{ANN} &= \frac{Y - \tilde{Y}_{ANN}}{\sigma} && \text{neural network used} \\
\chi_{mod} &= \frac{Y - \tilde{Y}_{mod}}{\sigma} && \text{physical model used} \\
\chi_{true} &= \frac{Y - Y_{true}}{\sigma} && \text{true value}
\end{aligned}$$

From table 2 it can be seen that the neural network identifies clearly the sensor faults and does not introduce any false alarm. Values identified by

Measurement	Unit	standard deviation	Description
T2	K	$0,5K$	inlet temperature
P2	bar	$50Pa$	inlet pressure
T4	K	$1K$	turbine inlet temperature
A8	m^2	$1.10^{-5}m^2$	nozzle area
T3	K	$1K$	compressor outlet temperature
P3	bar	$200Pa$	compressor outlet pressure
P4	bar	$200Pa$	turbine inlet pressure
T5	K	$1K$	turbine outlet temperature
P5	bar	$100Pa$	turbine outlet pressure
T8	K	$1K$	nozzle outlet temperature
P8	bar	$100Pa$	nozzle outlet pressure
N	RPM	$100RPM$	rotational speed
Dfuel	g/s	$0,5g/s$	fuel flow
D2	kg/s	$3.10^{-3}kg/s$	air flow

Table 1: Measurements available on the SR30 engine

the network ($\tilde{Y}$) are not exactly the same as those identified by the physical model but the difference remains within the noise. The identified input parameters ($\tilde{\theta}$) are correctly estimated compared to the real value (generated by the physical model) and sensor fault isolation remains very effective. Indeed in this particular case, the physical model is the true generator of the data and therefore allows an easier identification. Generally, a physical model has also to be fitted to a training set of data and the difference between the physical model and the mathematical representation (the network) would disappear.

In [1], a physical model has been used together with a robust validation strategy. Several test cases have shown the numerous advantages of such a method on the more common Least Squares Method. Here we see that a neural network instead of a physical model does not introduce any significant decrease in sensor fault isolation efficiency.

	Test case 1		Test case 3	
Var.	χ_{ANN}	χ_{true}	χ_{ANN}	χ_{true}
T2	0,74	-0,80	**8,92**	-0,80
P2	-0,32	-0,94	-2,62	-0,94
T4	0,12	-0,63	**-41,60**	**-60,00**
A8	0,00	1,00	-1,00	1,00
T3	0,41	-1,31	**6,49**	-1,31
P3	-1,06	0,41	-2,32	0,41
D2	0,24	0,01	-1,10	0,01
N	0,93	-0,51	**7,06**	-0,51
Dfuel	-0,14	0,02	0,02	0,02
P4	1,38	-0,21	**3,33**	-0,21
T5	-1,30	-0,34	**15,52**	-0,34
P5	0,13	-0,20	**3,08**	-0,20
P8	0,68	-0,43	2,79	-0,43
T8	-0,12	0,71	**17,11**	0,71

Table 3: Identification results using a least square method

Table 3 summarizes the results obtained with a usual least square identification. It means that the input parameters are determined by minimizing the sum of square error (SSE). Test case 1 shows that the robust approach reduces to a least square approach when no fault occurs. Test case 3 shows that the least square method only identifies a fault in the sample but cannot locate it.

Comparing these results to those of table 2, it is obvious that robust identification performs much better, identifying the parameters, isolating the fault quite clearly and rejecting only faulty measurements (no false alarm).

The analysis of criticity and standard deviation associated to the validated values $\tilde{Y}_j$ provides a good overview of how the robust identification performs. The advantage of using robust identification is clearly shown by test case 3 in table 4.

Robust estimation sets criticity of rejected measurements to zero. This means that they are not used by the identification procedure. As a consequence the criticity of the other measurements changes (since trace$(G) = p$) and so the standard deviation associated to each validated value. The least square approach does not show such behavior, and the criticity is not modified when faults occur.

Least square methods do not take into account faulty measurements and additional tests are mandatory to achieve fault isolation. One of the

	Test case 1			Test case 2			Test case 3		
Var.	χ_{ANN}	χ_{mod}	χ_{true}	χ_{ANN}	χ_{mod}	χ_{true}	χ_{ANN}	χ_{mod}	χ_{true}
T2	0,73	-0,38	-0,80	1,27	-0,07	-0,80	1,64	0,25	-0,80
P2	-0,28	0,12	-0,94	-0,56	0,04	-0,94	0,58	0,28	-0,94
T4	0,20	-0,09	-0,63	**-58,13**	**-58,91**	**-60,00**	**-58,04**	**-58,62**	**-60,00**
A8	0,00	1,00	1,00	0,00	1,00	1,00	0,00	1,00	1,00
T3	0,39	-1,06	-1,31	0,78	-0,84	-1,31	**-5,46**	**-7,10**	**-7,76**
P3	-1,18	0,55	0,40	-1,51	0,19	0,40	-1,29	-0,35	0,40
D2	0,22	-0,84	-0,81	0,12	-0,91	-0,81	0,14	-0,97	-0,81
N	0,87	-0,40	-0,51	1,25	-0,26	-0,51	**12,31**	**10,72**	**10,40**
Dfuel	-0,14	0,00	0,00	-0,14	0,00	0,00	-0,14	0,00	0,00
P4	1,36	-0,04	-0,21	1,25	-0,31	-0,21	1,41	-0,75	-0,21
T5	-1,20	0,17	-0,34	-0,20	0,68	-0,34	-0,17	0,96	-0,34
P5	0,22	0,01	-0,40	0,28	-0,08	-0,40	0,26	-0,37	-0,40
P8	0,77	-0,44	-0,85	0,78	-0,53	-0,85	**25,21**	**48,13**	**48,10**
T8	-0,02	1,23	0,71	1,01	1,73	0,71	1,05	2,01	0,71

Table 2: Identification results

Var.	crit (SSE)	crit (Huber)	χ_{true}
T2	0,21	0,25	-0,80
P2	0,95	0,95	-0,94
T4	0,30	**0,00**	**-60,00**
A8	0,12	0,12	1,00
T3	0,12	0,13	-1,31
P3	0,65	0,65	0,41
D2	0,01	0,01	0,01
N	0,20	0,21	-0,51
Dfuel	0,00	0,00	0,02
P4	0,35	0,35	-0,21
T5	0,28	0,39	-0,34
P5	0,27	0,27	-0,20
P8	0,25	0,25	-0,43
T8	0,30	0,41	0,71

Table 4: Criticity of faulty measurements

main advantages of the robust estimator is that the true value of the parameters, the influence of each measurement on the result, the number of good measurements available and the accuracy (standard deviation) of each identified parameter are obtained in one step.

Conclusions

It has been shown in this paper that neural networks are sufficiently smooth to model a jet-engine.

The robust identification method based on Huber's error function appears to give very good results in the case of measurement validation and sensor faults detection on a turbojet. The parameters are correctly determined and the sensor faults are easily detected and located.

Such a robust estimator can be very useful in a real time measurement validation process since a few measurements are needed to detect a fault. Only one sample at a time is used, that should provide fast results right after the acquisition.

The introduction of neural networks provides a great computational speed to the identification process since no non-linear system of equations has to be solved, resulting in a very short computational time for one measurement validation. Several tests have been made to assess the gain in computational time. When using the physical model only 1 validation per second (1 Hz) can be achieved (on an Intel PIII 650MHz) whereas a neural network enables validation frequency of 500 Hz to 1000 Hz. Such a property can be very useful for on-line validation.

The tests have shown that even though the identified parameters were not exactly the same, no loss of fault detection efficiency was observed. Therefore measurement validation can be achieved by a neural network without any loss of efficiency. Moreover this procedure may be applied to any engine even if no physical model is available.

More study are necessary to develop an on-line robust learning of the neural network together with an on-line measurement validation that should provide an automatic procedure (since no database would have to be stored as a preliminary step).

Acknowledgements

This work is funded by the Walloon Region of Belgium in the frame of its Program FIRST and is supported by Techspace Aero, Snecma Group.

References

[1] P. Dewallef, O. Léonard (2001), *Robust Validation of Measurements on Jet Engines*, Fourth European Conference on Turbomachinery.

[2] Louis Wehenkel, *Automatic Learning Techniques in Power Systems*, Kluwer Academic Publishers.

[3] P.P. Walsh, P. Fletcher (1998), *Gas Turbine Performance*, Blackwell Science.

[4] A. Navez (1993), *Développement de nouveaux modèles de simulation de turbine à gaz et de turboréacteurs d'avions*, Université de Liège, Rapport de Projet FIRST.

[5] P. Camus (1997), *Contribution à l'utilisation de modèles mathématiques pour l'analyse des essais de propulseurs aérospatiaux: application aux moteurs fusées cryotechniques*, Université de Liège, Thèse de doctorat.

[6] Christopher M. Bishop (1995), *Neural Networks for Pattern Recognition*, Clarendon Press - Oxford.

[7] Peter J. Huber (1996), *Robust Statistics*, Society for industrial and applied Mathematics.

[8] Peter J. Huber (1982), *Robust Statistics*, Wiley series in probability and statistics.

[9] B.T. Poljak, JA.Z. Tsypkin (1980), *Robust Identification*, Automatica, Vol. 16, pp.53-63.

[10] R. Douglas Martin, C.J. Masreliez (1975), *Robust Estimation via Stochastic Approximation*, IEEE Transactions on information Theory, Vol 21 n°3.

[11] John E. Moody (1992), *The effective Number of Parameters: An Analysis of Generalization and Generalization in Nonlinear Learning Systems*, Advances in Neural Information Processing Systems 4, pp. 847-854.

[12] David J.C. MacKay (1995), *Bayesian Methods for Neural Networks: Theory and applications*, University of Cambridge.

[13] Peter M. Williams (1994), *Bayesian Regularisation and Pruning using Laplace Prior*, School of Cognitive and Computing Sciences - University of Sussex, Technical Report CSRP-312.

[14] Ben Kröse and Patrick Van Der Smagt (1996), *An Introduction to Neural Networks*, University of Amsterdam.

[15] J. Henseler (1995), *Back Propagation*, Lecture Notes in Computer Science, vol. 931 p37.