SPATIO-TEMPORAL CORRELATED NOISE IN MULTI-OUTPUT NEURAL NETWORKS

Shahram Hosseini and Christian Jutten

LIS, INPG, 46 Av. Felix Viallet
Grenoble, France
hosseini or chris@lis-viallet.inpg.fr

ABSTRACT

In this paper, supposing a Gaussian noise model, we study the problem of multi output nonlinear regression using MLP when the noise in each output is a correlated autoregressive time series and there is a spatial correlation between different output noise sources. We show that using a maximum likelihood approach, the noise parameters can be determined simultaneously with the network weights and used to improve the network generalization performance. For two special cases of first and second order AR noise, the appropriate cost functions to minimize are derived.

1. INTRODUCTION

When multi layer perceptrons are used for non linear regression, it is generally supposed that output data are corrupted by independent and identically distributed (i.i.d.) noises and the ordinary least square method is usually used to estimate the network weights. However, in many practical situations, the noise may be correlated [1]. Two kinds of correlation can be investigated: correlation between the successive noise samples in each output, and correlation between the different outputs. In a time-series estimation problem, the first kind may be due to a filtering effect (temporal correlation) and the second type may be the effect of a common noise source on the different outputs (spatial correlation).

We proposed recently a maximum likelihood framework to improve the estimation of a single output multi layer perceptron when the output noise is a correlated autoregressive time series [2]. In this paper, we generalize the method for multi-output MLPs when there is also some correlation between the different output noises. The paper is organized as follows: in section 2, the problem will be formulated for a general case. In sections 3 and 4, two special cases of first and second order noises are investigated. Simulation results are given in section 5, before the conclusion.

2. STATEMENT OF THE PROBLEM

Suppose we have a P input, M output system which can be represented using the following equations:

$$\mathbf{y}(t) = \mathbf{g}(\mathbf{x}(t), \mathbf{W}^*) + \boldsymbol{\epsilon}(t) \tag{1}$$

$$\boldsymbol{\epsilon}(t) = \sum_{i=1}^{q} \boldsymbol{\Phi}_i \boldsymbol{\epsilon}(t-i) + \mathbf{n}(t) \tag{2}$$

$$\mathbf{n}(t) \sim \mathcal{N}(\mathbf{0}, \boldsymbol{\Sigma}) \tag{3}$$

where $\mathbf{y}(t) = [y_1(t), ..., y_M(t)]^T$ are the system outputs, $\mathbf{x}(t) = [x_1(t), ..., x_P(t)]^T$ are the system inputs, $\mathbf{g}(\mathbf{x}(t), \mathbf{W}^*) = [g_1(\mathbf{x}(t), \mathbf{W}^*), ..., g_M(\mathbf{x}(t), \mathbf{W}^*)]^T$ are the neural network outputs, $\boldsymbol{\epsilon}(t) = [\epsilon_1(t), ..., \epsilon_M(t)]^T$ represent q-th order autoregressive stationary colored noises and $\mathbf{n}(t) = [n_1(t), ..., n_M(t)]^T$ are jointly Gaussian white noises with $E[n_i(t)n_i(t-j)] = \sigma_i^2\delta(j)$ and $E[n_i(t)n_j(t)] = \sigma_{ij}$ where $\delta(.)$ is the Dirac distribution. $\boldsymbol{\Sigma}$ is the covariance matrix of $\mathbf{n}(t)$, $\boldsymbol{\Phi}_i$ are the diagonal matrices, modeling the autoregressive structure of $\boldsymbol{\epsilon}(t)$ and $\mathbf{W}^*$ is the network optimal weights matrix.

Suppose we have N samples of the system inputs and outputs, collected at regular time intervals, and we want to find the best approximation of $\mathbf{W}^*, \boldsymbol{\Phi} = [\boldsymbol{\Phi}_1, ..., \boldsymbol{\Phi}_q]$ and $\boldsymbol{\Sigma}$. As $\boldsymbol{\Phi}_i$ are $M \times M$ diagonal matrices and $\boldsymbol{\Sigma}$ is an $M \times M$ symmetric matrix, there are $dim(\mathbf{W}^*) + qM + \frac{M^2+M}{2}$ independent parameters to estimate. Let's write the likelihood function, decomposed in two parts:

$$\begin{aligned} f(\mathbf{y}(1), \mathbf{y}(2), ..., \mathbf{y}(N); \mathbf{W}, \boldsymbol{\Phi}, \boldsymbol{\Sigma}) = \\ f(\mathbf{y}(1)...\mathbf{y}(q); \mathbf{W}, \boldsymbol{\Phi}, \boldsymbol{\Sigma}). \\ f(\mathbf{y}(q+1)...\mathbf{y}(N)|\mathbf{y}(1), ..., \mathbf{y}(q); \mathbf{W}, \boldsymbol{\Phi}, \boldsymbol{\Sigma}) \end{aligned} \tag{4}$$

The second part can be easily determined using:

$$\begin{aligned} f(\mathbf{y}(q+1)...\mathbf{y}(N)|\mathbf{y}(1), ..., \mathbf{y}(q); \mathbf{W}, \boldsymbol{\Phi}, \boldsymbol{\Sigma}) = \\ \prod_{i=q+1}^{N} f(\mathbf{y}(i)|\mathbf{y}(i-1), ..., \mathbf{y}(i-q)); \mathbf{W}, \boldsymbol{\Phi}, \boldsymbol{\Sigma}) \end{aligned} \tag{5}$$

Conditioning on a random variable means treating it as a deterministic constant. So, using (1), (2) and (3):

$$(\boldsymbol{\epsilon}(i)|\boldsymbol{\epsilon}(i-1),...,\boldsymbol{\epsilon}(i-q)) \sim \mathcal{N}(\sum_{j=1}^{q} \boldsymbol{\Phi}_j \boldsymbol{\epsilon}(i-j), \boldsymbol{\Sigma}) \quad (6)$$

$$(\mathbf{y}(i)|\mathbf{y}(i-1),...,\mathbf{y}(i-q)) \sim \mathcal{N}(\mathbf{g}(\mathbf{x}(i),\mathbf{W}) + \sum_{j=1}^{q} \boldsymbol{\Phi}_j \boldsymbol{\epsilon}(i-j), \boldsymbol{\Sigma}) \quad (7)$$

where $\mathbf{g}(\mathbf{x}(i),\mathbf{W})$ is the network output for the i-th observation and $\mathbf{W}$ is the network actual weights matrix. Denoting the estimation residue by $\boldsymbol{\epsilon}(i,\mathbf{W}) = \mathbf{y}(i) - \mathbf{g}(\mathbf{x}(i),\mathbf{W})$, we obtain:

$$f(\mathbf{y}(i)|\mathbf{y}(i-1),...,\mathbf{y}(i-q);\mathbf{W},\boldsymbol{\Phi},\boldsymbol{\Sigma}) = \frac{1}{\sqrt{2\pi}|\boldsymbol{\Sigma}|^{1/2}} exp(\frac{-1}{2}[\boldsymbol{\epsilon}(i,\mathbf{W}) - \sum_{j=1}^{q} \boldsymbol{\Phi}_j \boldsymbol{\epsilon}(i-j,\mathbf{W})]^T \boldsymbol{\Sigma}^{-1}[\boldsymbol{\epsilon}(i,\mathbf{W}) - \sum_{j=1}^{q} \boldsymbol{\Phi}_j \boldsymbol{\epsilon}(i-j,\mathbf{W})]) \quad (8)$$

Thus, (5) and (8) give the second right side term of (4). The first term is more difficult to obtain. The solution consists in decomposing it to the conditional likelihood terms and determining each term separately:

$$f(\mathbf{y}(1),\mathbf{y}(2),...\mathbf{y}(q);\mathbf{W},\boldsymbol{\Phi},\boldsymbol{\Sigma}) = f(\mathbf{y}(1)).f(\mathbf{y}(2)|\mathbf{y}(1);\mathbf{W},\boldsymbol{\Phi},\boldsymbol{\Sigma}). ...f(\mathbf{y}(q)|\mathbf{y}(1),...,\mathbf{y}(q-1);\mathbf{W},\boldsymbol{\Phi},\boldsymbol{\Sigma}) \quad (9)$$

In this paper, we study two special cases q=1 and q=2. Higher order structures may be identified similarly.

3. AR1 MODEL

In this case, (9) reduces to $f(\mathbf{y}(1))$. Using (2) and noting that $\boldsymbol{\epsilon}(t-1)$ is independent of $\mathbf{n}(t)$, we can write:

$$E[\boldsymbol{\epsilon}(t)\boldsymbol{\epsilon}(t)^T] = \boldsymbol{\Phi}_1 E[\boldsymbol{\epsilon}(t-1)\boldsymbol{\epsilon}(t-1)^T]\boldsymbol{\Phi}_1^T + E[\mathbf{n}(t)\mathbf{n}(t)^T] \quad (10)$$

Denoting $E[\boldsymbol{\epsilon}(t)\boldsymbol{\epsilon}(t)^T]$ by $\boldsymbol{\Lambda}$ and using the stationary properties of $\boldsymbol{\epsilon}(t)$:

$$\boldsymbol{\Lambda} = \boldsymbol{\Phi}_1 \boldsymbol{\Lambda} \boldsymbol{\Phi}_1^T + \boldsymbol{\Sigma} \quad (11)$$

Using matrix to vector operator $vec(.)$ and the Kronecker product defined respectively by:
$vec(\mathbf{X}) = [x_{11}, x_{21}, ..., x_{m1}, x_{12}, ..., x_{m2}, ..., x_{1m}, ..., x_{mm}]^T$
and $\mathbf{A} \otimes \mathbf{B} = \begin{bmatrix} a_{11}\mathbf{B} & \cdots & a_{1m}\mathbf{B} \\ \vdots & \ddots & \vdots \\ a_{m1}\mathbf{B} & \cdots & a_{mm}\mathbf{B} \end{bmatrix}$
and using the following property:
$vec(\mathbf{AXB}) = (\mathbf{B}^T \otimes \mathbf{A})vec(\mathbf{X})$
we can obtain the vectorial equivalent of (11):

$$vec(\boldsymbol{\Lambda}) = (\boldsymbol{\Phi}_1 \otimes \boldsymbol{\Phi}_1)vec(\boldsymbol{\Lambda}) + vec(\boldsymbol{\Sigma}) \quad (12)$$

which can be solved by:

$$vec(\boldsymbol{\Lambda}) = (\mathbf{I} - \boldsymbol{\Phi}_1 \otimes \boldsymbol{\Phi}_1)^{-1} vec(\boldsymbol{\Sigma}) \quad (13)$$

Since $\boldsymbol{\Phi}_1$ is a diagonal matrix with diagonal entries ϕ_{1ii} and $\boldsymbol{\Lambda}$ and $\boldsymbol{\Sigma}$ are the symmetric matrices with entries λ_{ij} and σ_{ij} respectively [1], it can be easily shown that:

$$\lambda_{ij} = \frac{\sigma_{ij}}{1 - \phi_{1ii}\phi_{1jj}} \quad (14)$$

Then, using:

$$\mathbf{y}(1) \sim \mathcal{N}(\mathbf{g}(\mathbf{x}(1),\mathbf{W}), E[\boldsymbol{\epsilon}(1,\mathbf{W})\boldsymbol{\epsilon}(1,\mathbf{W})^T]) \quad (15)$$

$f(\mathbf{y}(1))$ can be computed:

$$f(\mathbf{y}(1);\mathbf{W},\boldsymbol{\Phi},\boldsymbol{\Sigma}) = \frac{1}{\sqrt{2\pi}|\boldsymbol{\Lambda}|^{1/2}} exp(-\frac{1}{2}\boldsymbol{\epsilon}(1,\mathbf{W})^T \boldsymbol{\Lambda}^{-1} \boldsymbol{\epsilon}(1,\mathbf{W})) \quad (16)$$

Taking the logarithm of (4), and using (5), (8) and (16), the log-likelihood function can be computed:

$$\mathcal{L}_Y(y) = cte - \frac{1}{2}log|\boldsymbol{\Lambda}| - \frac{1}{2}\boldsymbol{\epsilon}(1,\mathbf{W})^T \boldsymbol{\Lambda}^{-1}\boldsymbol{\epsilon}(1,\mathbf{W}) - \frac{N-1}{2}log|\boldsymbol{\Sigma}| - \frac{1}{2}\sum_{i=2}^{N}\{[\boldsymbol{\epsilon}(i,\mathbf{W}) - \boldsymbol{\Phi}_1\boldsymbol{\epsilon}(i-1,\mathbf{W})]^T \boldsymbol{\Sigma}^{-1}[\boldsymbol{\epsilon}(i,\mathbf{W}) - \boldsymbol{\Phi}_1\boldsymbol{\epsilon}(i-1,\mathbf{W})]\} \quad (17)$$

By maximizing this function, the unknown matrices $\mathbf{W}$, $\boldsymbol{\Phi}$ and $\boldsymbol{\Sigma}$ may be estimated.

4. AR2 MODEL

Denoting $\boldsymbol{\Lambda}_0 = E[\boldsymbol{\epsilon}(t)\boldsymbol{\epsilon}(t)^T]$, $\boldsymbol{\Lambda}_1 = E[\boldsymbol{\epsilon}(t)\boldsymbol{\epsilon}(t-1)^T]$ and $\boldsymbol{\Lambda}_2 = E[\boldsymbol{\epsilon}(t)\boldsymbol{\epsilon}(t-2)^T]$, we try to compute (9). Using the previous section results, one can write:

$$f(\mathbf{y}(1);\mathbf{W},\boldsymbol{\Phi},\boldsymbol{\Sigma}) = \frac{1}{\sqrt{2\pi}|\boldsymbol{\Lambda}_0|^{1/2}} exp(-\frac{1}{2}\boldsymbol{\epsilon}(1,\mathbf{W})^T \boldsymbol{\Lambda}_0^{-1} \boldsymbol{\epsilon}(1,\mathbf{W})) \quad (18)$$

To compute $f(\mathbf{y}(2)|\mathbf{y}(1);\mathbf{W},\boldsymbol{\Phi},\boldsymbol{\Sigma})$, we can compute $f(\boldsymbol{\epsilon}(2,\mathbf{W})|\boldsymbol{\epsilon}(1,\mathbf{W});\mathbf{W},\boldsymbol{\Phi},\boldsymbol{\Sigma})$ and replace $\boldsymbol{\epsilon}(1,\mathbf{W})$ and $\boldsymbol{\epsilon}(2,\mathbf{W})$ by $\mathbf{y}(1) - g(\mathbf{x}(1),\mathbf{W})$ and $\mathbf{y}(2) - g(\mathbf{x}(2),\mathbf{W})$ respectively. For this purpose, we must determine the

[1] $\sigma_{ii} = \sigma_i^2$.

density function of the part of $\boldsymbol{\epsilon}(2,\mathbf{W})$ which is not predictable from $\boldsymbol{\epsilon}(1,\mathbf{W})$:

$$f(\boldsymbol{\epsilon}(2,\mathbf{W})|\boldsymbol{\epsilon}(1,\mathbf{W});\mathbf{W},\boldsymbol{\Phi},\boldsymbol{\Sigma}) = f(\boldsymbol{\epsilon}(2,\mathbf{W})-\boldsymbol{\Gamma}\boldsymbol{\epsilon}(1,\mathbf{W});\mathbf{W},\boldsymbol{\Phi},\boldsymbol{\Sigma}) \qquad (19)$$

where $E[(\boldsymbol{\epsilon}(2,\mathbf{W})-\boldsymbol{\Gamma}\boldsymbol{\epsilon}(1,\mathbf{W}))\boldsymbol{\epsilon}(1,\mathbf{W})^T]=\mathbf{0}$, *i.e.* :

$$\boldsymbol{\Gamma} = E[\boldsymbol{\epsilon}(2,\mathbf{W})\boldsymbol{\epsilon}(1,\mathbf{W})^T][E[\boldsymbol{\epsilon}(1,\mathbf{W})\boldsymbol{\epsilon}(1,\mathbf{W})^T]]^{-1} = \boldsymbol{\Lambda}_1\boldsymbol{\Lambda}_0^{-1} \qquad (20)$$

$\boldsymbol{\eta}(\mathbf{W}) = \boldsymbol{\epsilon}(2,\mathbf{W})-\boldsymbol{\Gamma}\boldsymbol{\epsilon}(1,\mathbf{W})$ is a Gaussian random vector: $\boldsymbol{\eta}\sim\mathcal{N}(\mathbf{0},E[\boldsymbol{\eta}(\mathbf{W})\boldsymbol{\eta}(\mathbf{W})^T])$. Its covariance matrix can be computed as following:
$E[\boldsymbol{\eta}(\mathbf{W})\boldsymbol{\eta}(\mathbf{W})^T] = E[(\boldsymbol{\epsilon}(2,\mathbf{W})-\boldsymbol{\Gamma}\boldsymbol{\epsilon}(1,\mathbf{W}))(\boldsymbol{\epsilon}(2,\mathbf{W})^T-\boldsymbol{\epsilon}(1,\mathbf{W})^T\boldsymbol{\Gamma}^T)] = \boldsymbol{\Lambda}_0-\boldsymbol{\Lambda}_1\boldsymbol{\Gamma}^T-\boldsymbol{\Gamma}\boldsymbol{\Lambda}_1^T+\boldsymbol{\Gamma}\boldsymbol{\Lambda}_0\boldsymbol{\Gamma}^T$

Using (20) and considering that $\boldsymbol{\Lambda}_0$ is symmetric, a simplified notation can be obtained:

$$E[\boldsymbol{\eta}(\mathbf{W})\boldsymbol{\eta}(\mathbf{W})^T] = \boldsymbol{\Lambda}_0-\boldsymbol{\Lambda}_1\boldsymbol{\Lambda}_0^{-1}\boldsymbol{\Lambda}_1^T \qquad (21)$$

Finally:

$$f(\mathbf{y}(2)|\mathbf{y}(1);\mathbf{W},\boldsymbol{\Phi},\boldsymbol{\Sigma}) = \frac{1}{\sqrt{2\pi}|\boldsymbol{\Lambda}_0-\boldsymbol{\Lambda}_1\boldsymbol{\Lambda}_0^{-1}\boldsymbol{\Lambda}_1^T|^{1/2}} exp(-\frac{1}{2}(\boldsymbol{\epsilon}(2,\mathbf{W})-\boldsymbol{\Lambda}_1\boldsymbol{\Lambda}_0^{-1}\boldsymbol{\epsilon}(1,\mathbf{W}))^T (\boldsymbol{\Lambda}_0-\boldsymbol{\Lambda}_1\boldsymbol{\Lambda}_0^{-1}\boldsymbol{\Lambda}_1^T)^{-1}(\boldsymbol{\epsilon}(2,\mathbf{W})-\boldsymbol{\Lambda}_1\boldsymbol{\Lambda}_0^{-1}\boldsymbol{\epsilon}(1,\mathbf{W}))) \qquad (22)$$

Once more, using the matrix to vector conversion and the Kronecker product, $\boldsymbol{\Lambda}_0$ and $\boldsymbol{\Lambda}_1$ satisfy (Annex 1) :

$$vec(\boldsymbol{\Lambda}_0) = \mathbf{D}^{-1}vec(\boldsymbol{\Sigma}) \qquad (23)$$

$$vec(\boldsymbol{\Lambda}_1) = \mathbf{C}vec(\boldsymbol{\Lambda}_0) \qquad (24)$$

where:

$$\mathbf{C} = (\mathbf{I}-\boldsymbol{\Phi}_2\otimes\boldsymbol{\Phi}_2)^{-1}(\mathbf{I}\otimes\boldsymbol{\Phi}_1+\boldsymbol{\Phi}_1\otimes\boldsymbol{\Phi}_2) \qquad (25)$$

$$\mathbf{D} = \mathbf{I}-\boldsymbol{\Phi}_1\otimes\boldsymbol{\Phi}_1-\boldsymbol{\Phi}_1^2\otimes\boldsymbol{\Phi}_2-\boldsymbol{\Phi}_2\otimes\boldsymbol{\Phi}_2-(\boldsymbol{\Phi}_2\otimes\boldsymbol{\Phi}_1+\boldsymbol{\Phi}_2\boldsymbol{\Phi}_1\otimes\boldsymbol{\Phi}_2)\mathbf{C} \qquad (26)$$

Using (5), (8), (18) and (22), taking the logarithm of (4) and eliminating the constants, the ML estimation of the unknown matrices $\mathbf{W}$, $\boldsymbol{\Phi}$ and $\boldsymbol{\Sigma}$ is obtained by minimizing the following objective function:

$$\begin{aligned}\mathcal{E} = {} & log|\boldsymbol{\Lambda}_0|+\boldsymbol{\epsilon}(1,\mathbf{W})^T\boldsymbol{\Lambda}_0^{-1}\boldsymbol{\epsilon}(1,\mathbf{W}) + log|\boldsymbol{\Lambda}_0-\boldsymbol{\Lambda}_1\boldsymbol{\Lambda}_0^{-1}\boldsymbol{\Lambda}_1^T| + (\boldsymbol{\epsilon}(2,\mathbf{W})-\boldsymbol{\Lambda}_1\boldsymbol{\Lambda}_0^{-1}\boldsymbol{\epsilon}(1,\mathbf{W}))^T \\ & (\boldsymbol{\Lambda}_0-\boldsymbol{\Lambda}_1\boldsymbol{\Lambda}_0^{-1}\boldsymbol{\Lambda}_1^T)^{-1}(\boldsymbol{\epsilon}(2,\mathbf{W})-\boldsymbol{\Lambda}_1\boldsymbol{\Lambda}_0^{-1}\boldsymbol{\epsilon}(1,\mathbf{W})) + (N-2)log|\boldsymbol{\Sigma}| + \\ & \sum_{i=3}^{N}(\boldsymbol{\epsilon}(i,\mathbf{W})-\boldsymbol{\Phi}_1\boldsymbol{\epsilon}(i-1,\mathbf{W})-\boldsymbol{\Phi}_2\boldsymbol{\epsilon}(i-2,\mathbf{W}))^T \\ & \boldsymbol{\Sigma}^{-1}(\boldsymbol{\epsilon}(i,\mathbf{W})-\boldsymbol{\Phi}_1\boldsymbol{\epsilon}(i-1,\mathbf{W})-\boldsymbol{\Phi}_2\boldsymbol{\epsilon}(i-2,\mathbf{W}))\end{aligned} \qquad (27)$$

5. SIMULATION RESULTS

To test our method, we try to estimate two functions: $s_1 = 2\tanh(x+1)-2\tanh(2x-1.5)$ and $s_2 = 3\tanh(2x-1)+5\tanh(-3x+1)$; $x\in[-5,5]$, from 100 samples corrupted by spatio-temporal Gaussian noise:

$$\begin{cases} y_1(i) = s_1(i)+\epsilon_1(i) \\ y_2(i) = s_2(i)+\epsilon_2(i) \end{cases} \text{where:}$$

$$\begin{cases} \epsilon_1(i) = \phi_1\epsilon_1(i-1)+n_1(i) \\ \epsilon_2(i) = \phi_2\epsilon_2(i-1)+n_2(i) \end{cases} \text{and}$$

$$(n_1(i),n_2(i))\sim N(0,\Sigma) \quad \text{with } \Sigma = \begin{pmatrix} \sigma_1^2 & \sigma_{12} \\ \sigma_{12} & \sigma_2^2 \end{pmatrix}$$

where $\phi_1 = 0.95$, $\phi_2 = 0.9$, $\sigma_1 = 0.2$, $\sigma_2 = 0.2$ and $\sigma_{12} = 0.024$. The signals are chosen so that a 4 hidden units network (with tangent hyperbolic activation) can surely approximate them. The jointly Gaussian white noise samples (n_1,n_2) are produced by generating two independent Gaussian sequences with mean 0 and variance 1: $\alpha\sim\mathcal{N}(0,1)$, $\beta\sim\mathcal{N}(0,1)$ and using $n_1 = 0.09(2\alpha+\beta)$, $n_2 = 0.09(2\alpha-\beta)$.

The used network contains 4 hidden units with tangent hyperbolic activation function and two linear output units. The ML estimation of 23 parameters (18 network weights+5 noise parameters) is obtained by maximizing (17) using a gradient ascent algorithm. Two important problems should be considered here:
1) The ascent trajectory may pass the regions of the parameters space where the arguments of the logarithmic terms of (17) are negative so that the search procedure crashes. A solution is a variable changing which guarantees that the numerical search always stays within certain specified boundaries. In our problem, the following variable changing provides this guarantee:
$\sigma_{12} = \sigma_1\sigma_2\frac{a}{1+|a|}$, $\phi_1 = \frac{b}{1+|b|}$, $\phi_2 = \frac{c}{1+|c|}$
It can be easily shown that the above changes lead to: $\sigma_1^2\sigma_2^2 \geq \sigma_{12}^2$, $(1-\phi_1^2)(1-\phi_2^2)\geq 0$ and $(1-\phi_1\phi_2)^2 \geq (1-\phi_1^2)(1-\phi_2^2)$ so that $|\boldsymbol{\Lambda}|$ and $|\boldsymbol{\Sigma}|$ in (17) are positive. Thus, the new parameters vector to estimate contains the network weights plus σ_1,σ_2,a,b and c.
2) In general, the gradient values for the network weights and the noise parameters are not in the same order of magnitude and choosing the same learning rate for all the parameters may slow down the algorithm or lead to divergence. To avoid this problem, we used the conjugate gradient method which finds the optimal value of the learning rate using a line minimization algorithm.

The experiment consists of 10 runs, each run corresponding to a different noise seed value. To evaluate the results, another network is trained using the Ordinary Least Square (OLS) algorithm. The results are given in Table 1, where performance is evaluated as

	ML		OLS	
	mean	STD	mean	STD
Performance	63.11	24.90	73.98	35.00
ϕ_1	0.83	0.13	-	-
ϕ_2	0.81	0.07	-	-
σ_1	0.24	0.08	-	-
σ_2	0.24	0.06	-	-
σ_{12}	0.05	0.04	-	-

Table 1: The simulation results for our algorithm (ML) and Ordinary Least Square algorithm (OLS).

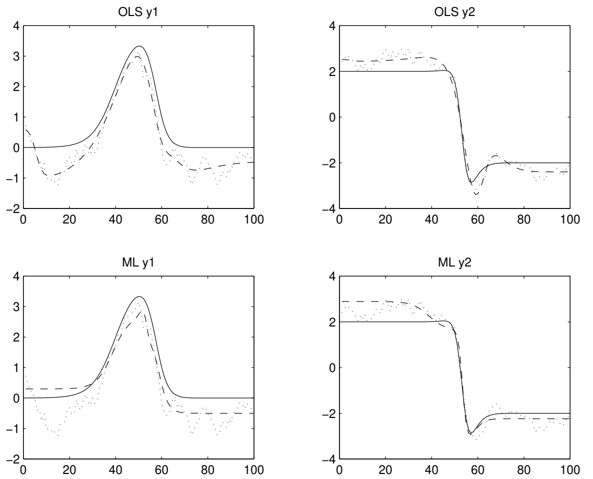

Figure 1: The result of a sample run for a two-output system. Target functions (solid lines), noisy samples (dots) and the network approximations (dashed lines) by OLS and ML algorithm.

the distance between the noise-less target signal and the estimations, $\sum_i (\hat{y}_1(i) - s_1(i))^2 + (\hat{y}_2(i) - s_2(i))^2$, which may be used as a generalization index. As it can be seen, the networks trained by our algorithm have a better performance in generalization, thanks to the noise modeling. To understand better, a sample result is shown in Figure 1. It is clear that the OLS estimation has a tendency to memorize the data, which leads to a good performance on the training data but a weak generalization capacity.

6. CONCLUSION

The case of spatio-temporal correlated noise in a multi-output neural network model is investigated and it is shown that the generalization performance can be improved by taking into account the noise parameters estimation using a maximum likelihood framework.

REFERENCES

[1] M. Viberg, "Sensitivity of parametric direction finding to colored noise fields and undermodeling,", *Signal Processing*, vol. 34, pp. 207-222, 1993.

[2] Sh. Hosseini and Ch. Jutten, "Simultaneous estimation of signal and noise in constructive neural networks," in *Proc. of NC'98*, Austria, pp. 412-417, 1998.

7. ANNEX 1: COMPUTATION OF Λ_0 AND Λ_1 FOR AR2 MODEL

Multiplying the main autoregressive equation:

$$\boldsymbol{\epsilon}(t) = \boldsymbol{\Phi}_1 \boldsymbol{\epsilon}(t-1) + \boldsymbol{\Phi}_2 \boldsymbol{\epsilon}(t-2) + \mathbf{n}(t) \tag{28}$$

by $\boldsymbol{\epsilon}(t)^T$, $\boldsymbol{\epsilon}(t-1)^T$ and $\boldsymbol{\epsilon}(t-2)^T$, and taking the mathematical expectation of the results, we obtain the following system of equations:

$$\begin{aligned} \boldsymbol{\Lambda}_0 &= \boldsymbol{\Phi}_1 \boldsymbol{\Lambda}_1^T + \boldsymbol{\Phi}_2 \boldsymbol{\Lambda}_2^T + \boldsymbol{\Sigma} \\ \boldsymbol{\Lambda}_1 &= \boldsymbol{\Phi}_1 \boldsymbol{\Lambda}_0 + \boldsymbol{\Phi}_2 \boldsymbol{\Lambda}_1^T \quad \boldsymbol{\Lambda}_2 = \boldsymbol{\Phi}_1 \boldsymbol{\Lambda}_1 + \boldsymbol{\Phi}_2 \boldsymbol{\Lambda}_0 \end{aligned} \tag{29}$$

Using the symmetry of $\boldsymbol{\Lambda}_0$, $\boldsymbol{\Sigma}$, $\boldsymbol{\Phi}_1$ and $\boldsymbol{\Phi}_2$:

$$\boldsymbol{\Lambda}_1^T = \boldsymbol{\Lambda}_0 \boldsymbol{\Phi}_1 + \boldsymbol{\Lambda}_1 \boldsymbol{\Phi}_2 \quad \boldsymbol{\Lambda}_2^T = \boldsymbol{\Lambda}_1^T \boldsymbol{\Phi}_1 + \boldsymbol{\Lambda}_0 \boldsymbol{\Phi}_2 \tag{30}$$

Replacing in two first equations of (29), $\boldsymbol{\Lambda}_2$ is eliminated and after simplifying, we obtain:

$$\begin{aligned} \boldsymbol{\Lambda}_0 = {} & \boldsymbol{\Phi}_1 \boldsymbol{\Lambda}_0 \boldsymbol{\Phi}_1 + \boldsymbol{\Phi}_1 \boldsymbol{\Lambda}_1 \boldsymbol{\Phi}_2 + \boldsymbol{\Phi}_2 \boldsymbol{\Lambda}_0 \boldsymbol{\Phi}_1^2 + \\ & \boldsymbol{\Phi}_2 \boldsymbol{\Lambda}_1 \boldsymbol{\Phi}_2 \boldsymbol{\Phi}_1 + \boldsymbol{\Phi}_2 \boldsymbol{\Lambda}_0 \boldsymbol{\Phi}_2 + \boldsymbol{\Sigma} \\ \boldsymbol{\Lambda}_1 = {} & \boldsymbol{\Phi}_1 \boldsymbol{\Lambda}_0 + \boldsymbol{\Phi}_2 \boldsymbol{\Lambda}_0 \boldsymbol{\Phi}_1 + \boldsymbol{\Phi}_2 \boldsymbol{\Lambda}_1 \boldsymbol{\Phi}_2 \end{aligned} \tag{31}$$

Using matrix to vector operations and the Kroneker product properties, we can write:

$$\begin{aligned} vec(\boldsymbol{\Lambda}_0) = {} & (\boldsymbol{\Phi}_1 \otimes \boldsymbol{\Phi}_1) vec(\boldsymbol{\Lambda}_0) + (\boldsymbol{\Phi}_2 \otimes \boldsymbol{\Phi}_1) vec(\boldsymbol{\Lambda}_1) + \\ & (\boldsymbol{\Phi}_1^2 \otimes \boldsymbol{\Phi}_2) vec(\boldsymbol{\Lambda}_0) + (\boldsymbol{\Phi}_2 \boldsymbol{\Phi}_1 \otimes \boldsymbol{\Phi}_2) vec(\boldsymbol{\Lambda}_1) \\ & + (\boldsymbol{\Phi}_2 \otimes \boldsymbol{\Phi}_2) vec(\boldsymbol{\Lambda}_0) + vec(\boldsymbol{\Sigma}) \\ vec(\boldsymbol{\Lambda}_1) = {} & (\mathbf{I} \otimes \boldsymbol{\Phi}_1) vec(\boldsymbol{\Lambda}_0) + (\boldsymbol{\Phi}_1 \otimes \boldsymbol{\Phi}_2) vec(\boldsymbol{\Lambda}_0) + \\ & (\boldsymbol{\Phi}_2 \otimes \boldsymbol{\Phi}_2) vec(\boldsymbol{\Lambda}_1) \end{aligned} \tag{32}$$

which can be simplified to:

$$\begin{aligned} & (\mathbf{I} - \boldsymbol{\Phi}_1 \otimes \boldsymbol{\Phi}_1 - \boldsymbol{\Phi}_1^2 \otimes \boldsymbol{\Phi}_2 - \boldsymbol{\Phi}_2 \otimes \boldsymbol{\Phi}_2) vec(\boldsymbol{\Lambda}_0) - \\ & \quad (\boldsymbol{\Phi}_2 \otimes \boldsymbol{\Phi}_1 + \boldsymbol{\Phi}_2 \boldsymbol{\Phi}_1 \otimes \boldsymbol{\Phi}_2) vec(\boldsymbol{\Lambda}_1) = vec(\boldsymbol{\Sigma}) \\ & (\mathbf{I} - \boldsymbol{\Phi}_2 \otimes \boldsymbol{\Phi}_2) vec(\boldsymbol{\Lambda}_1) = (\mathbf{I} \otimes \boldsymbol{\Phi}_1 + \boldsymbol{\Phi}_1 \otimes \boldsymbol{\Phi}_2) vec(\boldsymbol{\Lambda}_0) \end{aligned} \tag{33}$$

and leads to:

$$vec(\boldsymbol{\Lambda}_0) = \mathbf{D}^{-1} vec(\boldsymbol{\Sigma}) \quad vec(\boldsymbol{\Lambda}_1) = \mathbf{C} vec(\boldsymbol{\Lambda}_0) \tag{34}$$

with $\mathbf{C}$ and $\mathbf{D}$ as defined by (25) and (26).