Nonlinear system identification using neural state space models, applicable to robust control design [*]

Johan A.K. Suykens
Bart L.R. De Moor[†]
Joos Vandewalle

Katholieke Universiteit Leuven
Department of Electrical Engineering, ESAT-SISTA
Kardinaal Mercierlaan 94
B-3001 Leuven (Heverlee), Belgium
tel: 32/16/22 09 31   fax: 32/16/22 18 55
e-mail: Johan.Suykens@esat.kuleuven.ac.be

To appear in Int. J. Control

June 29, 1994

[*] This research work was carried out at the ESAT laboratory and the Interdisciplinary Center of Neural Networks ICNN of the Katholieke Universiteit Leuven, in the framework of a Concerted Action Project of the Flemish Community, entitled *Applicable Neural Networks* and within the framework of the Belgian programme on interuniversity attraction poles (IUAP-nr.17) initiated by the Belgian State - Prime Minister's Office - Science Policy Programming. The scientific responsibility is assumed by its authors.

[†] Research Associate of the Belgian National Fund for Scientific Research

Abstract

Prediction error learning algorithms for neural state space models are developed, both for the deterministic and the stochastic case with measurement and process noise. For the stochastic case a predictor with direct parametrization of the Kalman gain by a neural net architecture is proposed. Expressions for the gradients are derived by applying Narendra's sensitivity model approach. Finally a Linear Fractional Transformation representation is given for neural state space models, which makes it possible to use these models, obtained from input/output measurements on a plant, in a standard robust performance control scheme.

Keywords: Artificial neural networks, state space models, prediction error algorithms, Extended Kalman Filtering (EKF), sensitivity models, Linear Fractional Transformations (LFTs), robust control.

1 Introduction

The use of artificial neural networks in the context of control theory is motivated by the facts that any continuous nonlinear function can be approximated arbitrarily well on a compact interval by a multilayer feedforward neural net with one or more hidden layers, their ability of learning and adaptation, parallel distributed processing and possibility for efficient hardware implementation (Hunt *et al.*(1992)). In Chen *et al.*(1990a,b) identification methods are already proposed for input/output models, parametrized by multilayer feedforward neural networks. In the present paper prediction error algorithms are developed for general nonlinear state space models, parametrized by feedforward neural nets (simply called *neural state space models* here), for the deterministic identification case as well as for the stochastic case with process noise. In the predictor proposed for the latter a direct parametrization of the Kalman gain by a neural net architecture is made in innovations form. The advantage of a direct parametrization is revealed by the derivation of a Narendra's sensitivity model for generating the gradients of the cost function in the identification scheme (Narendra and Parthasarathy (1991)). These expressions are considerably simpler than for a parametrization based on the Extended Kalman Filter, which has the disadvantage of a complicated dependence on the parameter vector through a Riccati equation. The expressions for the stochastic case become then a natural extension for those of deterministic identification with a simulation model as predictor.

After discussing identification methods for neural state space models, it is shown how an LFT (Linear Fractional Transformation) representation can be derived for these models. Such LFTs are frequently used in modern robust control design (Dahleh and Khammash (1993), Doyle *et al.*(1991), Packard and Doyle (1993)). The neural state space models can be interpreted as a nominal linear system with bounded nonlinear feedback perturbation and in the stochastic case also corrupted by a white noise innovations sequence. The need for identification schemes that are able to estimate models which can be interpreted as such is e.g.expressed in Smith and Doyle (1992), Packard and Doyle (1993). Hence LFTs for neural state space models may bridge some of the existing gap between system identification and control design.

This paper is organized as follows: in Section 2 several neural state space models are introduced as parametrizations for nonlinear predictors for both the deterministic and stochastic case. Section 3 discusses prediction error learning algorithms for neural state space models, sensitivity models for generating the gradients of the cost function, together with some heuristics. In Section 4 LFTs for neural state space models are derived and finally in Section 5 an example is given on identification of a nonlinear interconnected system with hysteresis, corrupted by process and measurement noise.

2 Neural state space models

Nonlinear discrete time systems of the form

$$\left\{ \begin{array}{rcl} x_{k+1} & = & f_0(x_k, u_k) + v_k \\ y_k & = & g_0(x_k, u_k) + w_k \end{array} \right. \tag{1}$$

will be considered, with input vector $u_k \in \mathbb{R}^m$, output vector $y_k \in \mathbb{R}^l$ and state vector $x_k \in \mathbb{R}^n$. $v_k \in \mathbb{R}^n$, $w_k \in \mathbb{R}^l$ are respectively process noise and measurement noise and are assumed to be zero mean white Gaussian noise processes with covariance matrices

$$E\{\left[\begin{array}{c} v_k \\ w_k \end{array} \right] [v_s^t \; w_s^t]\} = \left[\begin{array}{cc} Q & S \\ S & R \end{array} \right] \delta_{ks}. \tag{2}$$

It is assumed that f_0 and g_0 are continuous nonlinear mappings.
Predictors for the deterministic as well as for the stochastic identification case will be discussed now, together with corresponding parametrizations by a neural network architecture in each of these cases.

2.1 Choice of predictors

In the case of *deterministic* identification ($v_k = 0$, $w_k = 0$) a simulation model (Ljung (1987) p.133) can be chosen as predictor

$$\mathcal{M}_d(\theta) : \left\{ \begin{array}{rcl} \hat{x}_{k+1} & = & f(\hat{x}_k, u_k; \theta) \; ; \; \hat{x}_0 = x_0 \text{ given} \\ \hat{y}_k & = & g(\hat{x}_k, u_k; \theta) \end{array} \right. \tag{3}$$

with model structure $\mathcal{M}_d$ and set of models $\mathcal{M}_d^* = \{\mathcal{M}_d(\theta) \mid \theta \in D_{\mathcal{M}_d}\}$. Given N input/output data Z^N a prediction error algorithm aims then at minimizing the cost function

$$V_N(\theta, Z^N) = \frac{1}{N} \sum_{k=1}^{N} l(\epsilon_k(\theta)) \tag{4}$$

with solution

$$\hat{\theta}_N = \arg \min_{\theta \in \mathrm{D}_{\mathcal{M}_\mathrm{d}}} \mathrm{V_N}(\theta, \mathrm{Z^N}). \tag{5}$$

Here $\epsilon_k(\theta) = y_k - \hat{y}_k(\theta)$ is the prediction error and $l(\epsilon_k)$ is a scalar valued positive function (typically $l(\epsilon_k) = \frac{1}{2}\epsilon_k^t \epsilon_k$).

In the *stochastic* case ($v_k \neq 0$, $w_k \neq 0$) one possibility for choosing a predictor is to consider the Extended Kalman Filter (EKF) for the system (1). The EKF will give an estimation of the state of system (1) by linearizing it around a reference trajectory. In fact it can be thought of as a restricted complexity filter which is constrained to have a similar format as that used for linear systems (Goodwin (1984) p.294). This filter is not an optimal state estimator in general, but nevertheless frequently used in many applications. Linearizing the nonlinear system (1) around $x_k = \hat{x}_k$, $v_k = 0$, $w_k = 0$ and applying the Kalman filter to the resulting time-varying linear system leads to the *extended Kalman filter* (Goodwin (1984) p.293,p.307):

$$\mathcal{M}_{s,ekf}(\theta) : \left\{ \begin{array}{rcl} \hat{x}_{k+1} & = & f(\hat{x}_k, u_k; \theta) + K_k(\theta)\epsilon_k \ ; \ \hat{x}_0 = x_0 \text{ given} \\ \hat{y}_k & = & g(\hat{x}_k, u_k; \theta) \\ K_k(\theta) & = & [F_k(\theta)\Sigma_k(\theta)H_k(\theta)^t + S(\theta)][H_k(\theta)\Sigma_k(\theta)H_k(\theta)^t + R(\theta)]^{-1} \\ \Sigma_{k+1} & = & F_k(\theta)\Sigma_k F_k(\theta)^t + Q(\theta) - K_k(\theta)[H_k(\theta)\Sigma_k(\theta)H_k(\theta)^t]K_k(\theta)^t, \end{array} \right. \tag{6}$$

where

$$F_k(\theta) = \frac{\partial f(x_k, u_k; \theta)}{\partial x_k} |_{x_k = \hat{x}_k} \ , \ H_k(\theta) = \frac{\partial g(x_k, u_k; \theta))}{\partial x_k} |_{x_k = \hat{x}_k} \tag{7}$$

and $K_k(\theta)$ is the Kalman gain. The covariance matrices Q, S, R are parametrized by θ. Σ_0 is given. In this case the optimal solution to the prediction error algorithm is:

$$\hat{\theta}_N = \arg \min_{\theta \in \mathrm{D}_{\mathcal{M}_{\mathrm{s,ekf}}}} \mathrm{V_N}(\theta, \mathrm{Z^N}). \tag{8}$$

Following the argumentation in Goodwin (1984) p.307,pp.366-367, in Ljung (1987) p.88 or in Ljung (1979) a feasible alternative to the EKF is a *direct* parametrization of the Kalman gain, rather than indirectly via the Riccati equation, which eliminates a great deal of complexity in fitting the predictor to the data. In that case we have a predictor in innovations form

$$\mathcal{M}_{s,direct}(\theta) : \left\{ \begin{array}{rcl} \hat{x}_{k+1} & = & f(\hat{x}_k, u_k; \theta) + K(\theta)\epsilon_k \ ; \ \hat{x}_0 = x_0 \text{ given} \\ \hat{y}_k & = & g(\hat{x}_k, u_k; \theta) \end{array} \right. \tag{9}$$

with as optimal solution to the prediction error algorithm

$$\hat{\theta}_N = \arg \min_{\theta \in \mathrm{D}_{\mathcal{M}_{\mathrm{s,direct}}}} \mathrm{V_N}(\theta, \mathrm{Z^N}). \tag{10}$$

2.2 Parametrizations by feedforward neural nets

For each of the model structures $\mathcal{M}_d$, $\mathcal{M}_{s,ekf}$ and $\mathcal{M}_{s,direct}$ parametrizations by neural nets will be proposed now. Such parametrizations make sense because any continuous nonlinear function can be approximated arbitrarily well on a compact interval by a multilayer feedforward neural network with one or more hidden layers (Cybenko (1989), Funahashi (1989), Hornik *et al.* (1989), Leshno *et al.* (1993)). The nonlinear mappings are parametrized now by multilayer feedforward neural networks with one hidden layer. The following choice of predictors is made:

$$\mathcal{M}_d(\theta) : \left\{ \begin{array}{rcl} \hat{x}_{k+1} & = & W_{AB} \tanh(V_A \hat{x}_k + V_B u_k + \beta_{AB}) \ ; \ \hat{x}_0 = x_0 \\ \hat{y}_k & = & W_{CD} \tanh(V_C \hat{x}_k + V_D u_k + \beta_{CD}) \end{array} \right. \tag{11}$$

with

$$\theta = [W_{AB}(:); V_A(:); V_B(:); \beta_{AB}; W_{CD}(:); V_C(:); V_D(:); \beta_{CD}]^1 \tag{12}$$

and

$$\mathcal{M}_{s,ekf}(\theta) : \left\{ \begin{array}{rcl} \hat{x}_{k+1} & = & W_{AB} \tanh(V_A \hat{x}_k + V_B u_k + \beta_{AB}) + K_k(\theta)\epsilon_k \ ; \ \hat{x}_0 = x_0 \\ \hat{y}_k & = & W_{CD} \tanh(V_C \hat{x}_k + V_D u_k + \beta_{CD}) \\ K_k(\theta) & = & [F_k(\theta)\Sigma_k(\theta)H_k(\theta)^t + S(\theta)][H_k(\theta)\Sigma_k(\theta)H_k(\theta)^t + R(\theta)]^{-1} \\ \Sigma_{k+1} & = & F_k(\theta)\Sigma_k F_k(\theta)^t + Q(\theta) - K_k(\theta)[H_k(\theta)\Sigma_k(\theta)H_k(\theta)^t]K_k(\theta)^t \end{array} \right. \tag{13}$$

with

$$\theta = [W_{AB}(:); V_A(:); V_B(:); \beta_{AB}; W_{CD}(:); V_C(:); V_D(:); \beta_{CD}; \theta_Q; \theta_S; \theta_R] \tag{14}$$

where θ_Q, θ_S, θ_R denote how Q,S,R are parametrized and finally

$$\mathcal{M}_{s,direct}(\theta) : \left\{ \begin{array}{rcl} \hat{x}_{k+1} & = & W_{AB} \tanh(V_A \hat{x}_k + V_B u_k + \beta_{AB}) + W_K \tanh(V_K \epsilon_k) \ ; \ \hat{x}_0 = x_0 \\ \hat{y}_k & = & W_{CD} \tanh(V_C \hat{x}_k + V_D u_k + \beta_{CD}) \end{array} \right. \tag{15}$$

with

$$\theta = [W_{AB}(:); V_A(:); V_B(:); \beta_{AB}; W_{CD}(:); V_C(:); V_D(:); \beta_{CD}; W_K(:); V_K(:)].$$

It will be shown later on in Section 4.2 in what sense precisely (15) can be interpreted as a direct parametrization of the Kalman gain in innovations form. The dimensions of the interconnection matrices and bias vectors are $W_{AB} \in \mathbb{R}^{n \times n_{hx}}$, $V_A \in \mathbb{R}^{n_{hx} \times n}$, $V_B \in \mathbb{R}^{n_{hx} \times m}$, $\beta_{AB} \in \mathbb{R}^{n_{hx}}$, $W_{CD} \in \mathbb{R}^{l \times n_{hy}}$, $V_C \in \mathbb{R}^{n_{hy} \times n}$, $V_D \in \mathbb{R}^{n_{hy} \times m}$, $\beta_{CD} \in \mathbb{R}^{n_{hy}}$, $W_K \in \mathbb{R}^{n \times n_{h\epsilon}}$, $V_K \in \mathbb{R}^{n_{h\epsilon} \times l}$, where the number of hidden neurons of the neural network architectures are n_{hx}, n_{hy} and $n_{h\epsilon}$. The predictors (11) and (15) are shown in Fig.1 and Fig.2.

Remarks:

- The parametrizations in (11)(13)(15) are not minimal. Like for linear state space models, which are only unique up to a similarity transformation, we have e.g. for $\mathcal{M}_d(\theta)$

$$\left\{ \begin{array}{rcl} \hat{z}_{k+1} & = & T W_{AB} \tanh(V_A T^{-1} \hat{z}_k + V_B u_k + \beta_{AB}) \ ; \ \hat{z}_0 = z_0 \\ \hat{y}_k & = & W_{CD} \tanh(V_C T^{-1} \hat{z}_k + V_D u_k + \beta_{CD}) \end{array} \right. \tag{16}$$

  with new state vector $\hat{z}_k = T\hat{x}_k$. If we assume that $n_{hx} \geq n$ this means that the total number of parameters in θ can be reduced by the amount n^2. Indeed if V_A is partitioned as

$$V_A = \left[\begin{array}{c} V_{A_1} \\ V_{A_2} \end{array} \right]$$

[1] As activation function we take the hyperbolic tangent function ($\tanh(x) = \frac{1-\exp(-2x)}{1+\exp(-2x)}$), which is applied elementwise to a vector or a matrix. The Matlab notations '(:)' means a columnwise scan of a matrix and '$x = [x_1; x_2]$' means a concatenation of the vectors x_1 and x_2 to the vector x.

and $T = V_{A_1} \in \mathbb{R}^{n \times n}$ is taken in (16), one obtains a new parametrization with

$$\theta = [W^*_{AB}(:); V^*_{A_2}(:); V_B(:); \beta_{AB}; W_{CD}(:); V^*_C(:); V_D(:); \beta_{CD}]$$

and $W^*_{AB} = TW_{AB}$, $V^*_{A_2} = V_{A_2}T^{-1}$ and $V^*_C = V_C T^{-1}$.
It is not investigated in this paper if it possible to reduce the number of parameters further by considering nonlinear transformations of the state space model. This aspect is important with respect to the identifiability concept, which concerns the unique representation of a given system description in a model structure (Ljung (1987) pp.100,105) and the fact whether two different values of θ can produce the same input/output behaviour or not. For linear state space models a minimal number of parameters is obtained by taking a canonical form, but on the other hand also non-minimal parametrizations are used in system identification practice such as e.g. for the algorithms proposed in (De Moor *et al.* (1991), Van Overschee and De Moor (1994)) which have advantages from a computational and numerical point of view.

- Instead of working with *full* parametrizations (11)(13)(15) for the nonlinear mappings f and g in (1) one can also take *partial* parametrizations if one has a priori knowledge on the structure of the state space model (1), e.g. from physical insight. Parametrizations by neural nets like

$$x_{k+1} = W_A \tanh(V_A x_k + \beta_A) + Bu_k \tag{17}$$

and

$$x_{k+1} = Ax_k + W_B \tanh(V_B u_k + \beta_B) \tag{18}$$

can be chosen respectively for

$$x_{k+1} = f_1(x_k) + Bu_k \tag{19}$$

and

$$x_{k+1} = Ax_k + f_2(u_k). \tag{20}$$

- The model structures (11)(13)(15)(17)(18) are called here briefly *neural state space models*. In neural networks terminology one would call this *recurrent neural networks* (see e.g. Zurada (1992) for an introduction), but from the viewpoint of identification theory these are state space models, parametrized by feedforward neural networks. In our opinion the term *neural state space models* is more suitable for the latter framework.

3 Learning algorithms for neural state space models

3.1 Optimization problems

Prediction error learning algorithms for neural state space models (11)(13)(15)(17)(18) will be discussed here. Computation of gradients of the cost function will be based on the framework of Narendra on gradient methods for the optimization of dynamical systems containing neural networks (Narendra and Parthasarathy (1991)). Only off-line algorithms will be discussed. The following nonlinear least squares (NLS) problems must be solved

1. Deterministic identification:

$$\min_{\theta \in D_{\mathcal{M}_d}} V_N(\theta, Z^N) = \frac{1}{N} \sum_{k=1}^{N} l(\epsilon_k(\theta)) \tag{21}$$

   subject to the dynamical model for $k = 0, ..., N$

$$\begin{cases} \hat{x}_{k+1} &= W_{AB} \tanh(V_A \hat{x}_k + V_B u_k + \beta_{AB}) \; ; \; \hat{x}_0 = x_0 \\ \hat{y}_k &= W_{CD} \tanh(V_C \hat{x}_k + V_D u_k + \beta_{CD}) \\ \epsilon_k &= y_k - \hat{y}_k \end{cases}$$

2. Stochastic case with EKF:

$$\min_{\theta \in D_{\mathcal{M}_{s,ekf}}} V_N(\theta, Z^N) = \frac{1}{N} \sum_{k=1}^{N} l(\epsilon_k(\theta)) \tag{22}$$

   subject to the dynamical model for $k = 0, ..., N$

$$\begin{cases} \hat{x}_{k+1} &= W_{AB} \tanh(V_A \hat{x}_k + V_B u_k + \beta_{AB}) + K_k(\theta)\epsilon_k \; ; \; \hat{x}_0 = x_0 \\ \hat{y}_k &= W_{CD} \tanh(V_C \hat{x}_k + V_D u_k + \beta_{CD}) \\ K_k(\theta) &= [F_k(\theta)\Sigma_k(\theta)H_k(\theta)^t + S(\theta)][H_k(\theta)\Sigma_k(\theta)H_k(\theta)^t + R(\theta)]^{-1} \\ \Sigma_{k+1} &= F_k(\theta)\Sigma_k F_k(\theta)^t + Q(\theta) - K_k(\theta)[H_k(\theta)\Sigma_k(\theta)H_k(\theta)^t]K_k(\theta)^t \\ \epsilon_k &= y_k - \hat{y}_k \end{cases}$$

3. Stochastic case with directly parametrized Kalman gain:

$$\min_{\theta \in D_{\mathcal{M}_{s,direct}}} V_N(\theta, Z^N) = \frac{1}{N} \sum_{k=1}^{N} l(\epsilon_k(\theta)) \tag{23}$$

   subject to the dynamical model for $k = 0, ..., N$

$$\begin{cases} \hat{x}_{k+1} &= W_{AB} \tanh(V_A \hat{x}_k + V_B u_k + \beta_{AB}) + W_K \tanh(V_K \epsilon_k) \; ; \; \hat{x}_0 = x_0 \\ \hat{y}_k &= W_{CD} \tanh(V_C \hat{x}_k + V_D u_k + \beta_{CD}) \\ \epsilon_k &= y_k - \hat{y}_k. \end{cases}$$

From the viewpoint of optimization theory there exist several methods for solving these problems. Either general purpose methods for unconstrained nonlinear optimization can be used or methods that take into account the particular structure of the NLS problem, which is the minimization of a sum of squared residuals. The simplest method is steepest descent. More advanced are Levenberg-Marquardt and Quasi-Newton methods (see Gill *et al.* (1981)), which are Newton-like methods that try to build up curvature information of the Hessian, based on gradient information only. For large scale problems conjugate gradient algorithms are to be preferred, because in this algorithm there is no need to store matrices. For each of these methods one needs to know the gradients of the performance index V_N with respect to the parameter vector θ. If we assume that $l(\epsilon) = \frac{1}{2}\epsilon^t\epsilon$ (for other choices see Ljung (1987)) the gradient becomes

$$\frac{\partial V_N}{\partial \theta} = \frac{1}{N}\sum_{k=1}^{N} \epsilon^t \frac{\partial \epsilon_k}{\partial \theta} = -\frac{1}{N}\sum_{k=1}^{N} \epsilon^t \frac{\partial \hat{y}_k}{\partial \theta}. \tag{24}$$

It will be shown now that the computation of the gradients is straightforward by applying Narendra's sensitivity model approach, but only in the case of deterministic identification and the stochastic case with directly parametrized Kalman gain. The derivation of the gradient for the stochastic case with EKF predictor is too complex because of the dependency on θ of the Kalman gain through the Riccati equation. In that case gradients can be calculated numerically or another optimization method, which is not gradient based, can be used then.

3.2 Sensitivity models

Given a predictor

$$\left\{ \begin{array}{rcl} \hat{x}_{k+1} & = & \Phi(\hat{x}_k, u_k; \alpha) \ ; \ \hat{x}_0 = x_0 \text{ given} \\ \hat{y}_k & = & \Psi(\hat{x}_k, u_k; \beta) \end{array} \right. \tag{25}$$

such as (11) or (15) where α, β are elements of the parameter vector $\theta \in \mathbb{R}^p$, a sensitivity model is then a model which generates the gradient vectors $\frac{\partial \hat{y}_k}{\partial \alpha}$ and $\frac{\partial \hat{y}_k}{\partial \beta}$.
According to Narendra and Parthasarathy (1990)(1991) such a sensitivity model can be obtained by taking the derivatives of (25) with respect to α and β

$$\left\{ \begin{array}{rcl} \frac{\partial \hat{x}_{k+1}}{\partial \alpha} & = & \frac{\partial \Phi}{\partial \hat{x}_k} . \frac{\partial \hat{x}_k}{\partial \alpha} + \frac{\partial \Phi}{\partial \alpha} \\ \\ \frac{\partial \hat{y}_k}{\partial \alpha} & = & \frac{\partial \Psi}{\partial \hat{x}_k} . \frac{\partial \hat{x}_k}{\partial \alpha} \\ \\ \frac{\partial \hat{y}_k}{\partial \beta} & = & \frac{\partial \Psi}{\partial \beta} \end{array} \right. \tag{26}$$

which is a dynamical model with state vector $\frac{\partial \hat{x}_k}{\partial \alpha}$ driven by the input vector consisting of $\frac{\partial \Phi}{\partial \alpha}$, $\frac{\partial \Psi}{\partial \beta}$ and at the output $\frac{\partial \hat{y}_k}{\partial \alpha}$, $\frac{\partial \hat{y}_k}{\partial \beta}$ are generated (Fig.3). A steepest descent learning algorithm which makes use of this sensitivity model for computation of the gradients is called by Narendra *dynamic backpropagation*, introduced as a complement to the original *backpropagation* algorithm (Rumelhart *et al.* (1986)), which is a learning rule for *static* nonlinear mappings. For nonlinear dynamical systems containing neural nets the dynamic backpropagation procedure must be applied instead.

The derivatives will be given now for the predictor with directly parametrized Kalman gain (which includes as a special case deterministic identification by setting $W_K = 0$, $V_K = 0$). An elementwise notation for (15) is

$$\left\{ \begin{array}{lcl} \hat{x}^i & := & \sum_j {w_{AB}}^i_j \tanh(\sum_r {v_A}^j_r \, \hat{x}^r + \sum_s {v_B}^j_s \, u^s + {\beta_{AB}}^j) + \sum_j {w_K}^i_j \tanh(\sum_r {v_K}^j_r \, \epsilon^r) \\ \hat{y}^i & = & \sum_j {w_{CD}}^i_j \tanh(\sum_r {v_C}^j_r \, \hat{x}^r + \sum_s {v_D}^j_s \, u^s + {\beta_{CD}}^j), \end{array} \right. \tag{27}$$

where $\{.\}^i$ and $\{.\}^i_j$ denote respectively the i-th element of a vector and the ij-th element of a matrix. The assignment operator ':=' is introduced here in order to make it possible to omit the time index k.

Defining

$$\begin{array}{lcl} \varphi^l & = & \sum_r {v_A}^l_r \, \hat{x}^r + \sum_s {v_B}^l_s \, u^s + \beta^l_{AB} \\ \psi^l & = & \sum_r {v_C}^l_r \, \hat{x}^r + \sum_s {v_D}^l_s \, u^s + \beta^l_{CD} \\ \rho^l & = & \sum_r {v_K}^l_r \, \epsilon^r \end{array} \tag{28}$$

one obtains the following derivatives

$$\frac{\partial \Phi}{\partial \alpha} : \left\{ \begin{array}{lcl} \frac{\partial \Phi^i}{\partial {w_{AB}}^j_l} & = & \delta^i_j \tanh(\varphi^l) \\ \frac{\partial \Phi^i}{\partial {v_A}^j_l} & = & {w_{AB}}^i_j \, (1 - \tanh^2(\varphi^j)) \, \hat{x}^l \\ \frac{\partial \Phi^i}{\partial {v_B}^j_l} & = & {w_{AB}}^i_j \, (1 - \tanh^2(\varphi^j)) \, u^l \\ \frac{\partial \Phi^i}{\partial {\beta_{AB}}^j} & = & {w_{AB}}^i_j \, (1 - \tanh^2(\varphi^j)) \\ \frac{\partial \Phi^i}{\partial {w_K}^j_l} & = & \delta^i_j \tanh(\rho^l) \\ \frac{\partial \Phi^i}{\partial {v_K}^j_l} & = & {w_K}^i_j \, (1 - \tanh^2(\rho^j)) \, \epsilon^l \end{array} \right.$$

$$\frac{\partial \Psi}{\partial \beta} : \left\{ \begin{array}{lcl} \frac{\partial \Psi^i}{\partial {w_{CD}}^j_l} & = & \delta^i_j \tanh(\psi^l) \\ \frac{\partial \Psi^i}{\partial {v_C}^j_l} & = & {w_{CD}}^i_j \, (1 - \tanh^2(\psi^j)) \, \hat{x}^l \\ \frac{\partial \Psi^i}{\partial {v_D}^j_l} & = & {w_{CD}}^i_j \, (1 - \tanh^2(\psi^j)) \, u^l \\ \frac{\partial \Psi^i}{\partial {\beta_{CD}}^j} & = & {w_{CD}}^i_j \, (1 - \tanh^2(\psi^j)) \end{array} \right. \tag{29}$$

$$\frac{\partial \Phi}{\partial \hat{x}_k} : \qquad \frac{\partial \Phi^i}{\partial \hat{x}^r} = \sum_j {w_{AB}}^i_j \, (1 - \tanh^2(\varphi^j)) \, {v_A}^j_r$$

$$\frac{\partial \Psi}{\partial \hat{x}_k} : \qquad \frac{\partial \Psi^i}{\partial \hat{x}^r} = \sum_j {w_{CD}}^i_j \, (1 - \tanh^2(\psi^j)) \, {v_C}^j_r .$$

In fact one can interpret the predictor with its corresponding sensitivity model as one *augmented* system, that generates $\hat{y}_k$ as well as $\frac{\partial \hat{y}_k}{\partial \theta}$ at its output. Such an extended network model was also defined in Chen *et al.* (1990a,b) in the context of input/output models parametrized by feedforward neural nets.

There is also the possibility for developing *parallel* training algorithms for the neural state space models because for each of the p parameters α, β of the parameter vector θ one has a simulation of the sensitivity model over N samples. These p simulations can be distributed over the available number of processors. Parallel algorithms for input/output models were also discussed in Chen *et al.* (1990b).

3.3 Heuristics

The nonlinear optimization problem (21)-(23) has in general many local minima. Hence one has to start from several initial parameter vectors in order to have some confidence in the quality of the obtained local optima. Two heuristics are proposed here how a priori knowledge can be used for generating meaningful starting points for the identification procedure: initializing neural state space models as linear state space models (which makes sense for weakly nonlinear systems) and learning complex neural state space models from lower complex ones.

3.3.1 Linear models as starting points

Suppose a linear state space model is available in innovations form (Ljung (1987) p.87 or Ljung (1979))

$$\left\{ \begin{array}{rcl} \hat{x}_{k+1} & = & A(\theta)\hat{x}_k + B(\theta)u_k + K(\theta)\epsilon_k \\ \hat{y}_k & = & C(\theta)\hat{x}_k + D(\theta)u_k \end{array} \right. \tag{30}$$

with $E\{\epsilon_k \epsilon_s^t\} = \Lambda \delta_{ks}$ (Λ diagonal) and $\epsilon_k = y_k - \hat{y}_k$. Taking the neural state space model (15) with directly parametrized Kalman gain initially as

$$\begin{array}{l} W_{AB} = \frac{1}{\alpha_1}[I_n \ R_1], \quad [V_A \ V_B] = \alpha_1 \left[\begin{array}{cc} A & B \\ 0 & 0 \end{array} \right], \ \beta_{AB} = 0; \ (n_{hx} \geq n) \\ \\ W_{CD} = \frac{1}{\alpha_2}[I_l \ R_2], \quad [V_C \ V_D] = \alpha_2 \left[\begin{array}{cc} C & D \\ 0 & 0 \end{array} \right], \ \beta_{CD} = 0; \ (n_{hy} \geq l) \\ \\ W_K = \frac{1}{\alpha_3}[I_n \ R_3], \quad V_K = \alpha_3 \left[\begin{array}{c} K \\ 0 \end{array} \right]; \ (n_{h\epsilon} \geq n), \end{array} \tag{31}$$

where $\alpha_1, \alpha_2, \alpha_3$ are small positive real numbers and R_1, R_2, R_3 are arbitrary matrices of appropriate dimension, (15) behaves as the linear model (30) for $\alpha_1, \alpha_2, \alpha_3 \to 0$. Indeed for bounded input and state vector sequences one obtains

$$\begin{array}{rcl}
\hat{x}_{k+1} & = & \frac{1}{\alpha_1}[I_n \; R_1] \tanh(\alpha_1 \left[\begin{array}{cc} A & B \\ 0 & 0 \end{array}\right] \left[\begin{array}{c} \hat{x}_k \\ u_k \end{array}\right] + \left[\begin{array}{c} 0 \\ 0 \end{array}\right]) + \frac{1}{\alpha_3}[I_n \; R_3] \tanh(\alpha_3 \left[\begin{array}{c} K \\ 0 \end{array}\right] \epsilon_k) \\
 & \overset{\alpha_1,\alpha_3 \rightarrow 0}{=} & A\hat{x}_k + Bu_k + K\epsilon_k \\
\hat{y}_k & = & \frac{1}{\alpha_2}[I_l \; R_2] \tanh(\alpha_2 \left[\begin{array}{cc} C & D \\ 0 & 0 \end{array}\right] \left[\begin{array}{c} \hat{x}_k \\ u_k \end{array}\right] + \left[\begin{array}{c} 0 \\ 0 \end{array}\right]) \\
 & \overset{\alpha_2 \rightarrow 0}{=} & C\hat{x}_k + Du_k,
\end{array} \tag{32}$$

which follows immediately from the Taylor expansion of the activation function

$$\tanh(x) = x - \frac{1}{3}x^3 + \frac{2}{15}x^5 - \frac{17}{315}x^7 + ... \qquad , (|x| < \frac{\pi}{2}), \tag{33}$$

where $\tanh(x) \simeq x$ for $|x| \rightarrow 0$. This means that results from linear system identification can be used to initialize neural state space models in the local optimization scheme.

3.3.2 Learning complex neural state space models from lower complex ones

Suppose some neural state space model is already available (e.g. a model with a low number of hidden neurons, characterized by $W_{AB}^{(1)}$, $V_A^{(1)}$, $V_B^{(1)}$, $\beta_{AB}^{(1)}$, $W_{CD}^{(1)}$, $V_C^{(1)}$, $V_D^{(1)}$, $\beta_{CD}^{(1)}$) and one would like to decrease the fitting error by introducing extra hidden neurons in the model. The more complex neural state space model ($W_{AB}^{(2)}$, $V_A^{(2)}$, $V_B^{(2)}$, $\beta_{AB}^{(2)}$, $W_{CD}^{(2)}$, $V_C^{(2)}$, $V_D^{(2)}$, $\beta_{CD}^{(2)}$) has then the same input/output behaviour as the lower complex one if one takes

$$\begin{array}{lll}
W_{AB}^{(2)} = [W_{AB}^{(1)} \; R_1] \;\; , & [V_A^{(2)} \; V_B^{(2)}] = \left[\begin{array}{cc} V_A^{(1)} & V_B^{(1)} \\ 0 & 0 \end{array}\right] \;\; , & \beta_{AB}^{(2)} = \left[\begin{array}{c} \beta_{AB}^{(1)} \\ 0 \end{array}\right] \\
W_{CD}^{(2)} = [W_{CD}^{(1)} \; R_2] \;\; , & [V_C^{(2)} \; V_D^{(2)}] = \left[\begin{array}{cc} V_C^{(1)} & V_D^{(1)} \\ 0 & 0 \end{array}\right] \;\; , & \beta_{CD}^{(2)} = \left[\begin{array}{c} \beta_{CD}^{(1)} \\ 0 \end{array}\right] \\
W_K^{(2)} = [W_K^{(1)} \; R_3] \;\; , & V_K^{(2)} = \left[\begin{array}{c} V_K^{(1)} \\ 0 \end{array}\right] &
\end{array} \tag{34}$$

with R_1,R_2,R_3 arbitrary matrices of appropriate dimension.

4 LFT representation for neural state space models

An LFT representation will be derived now for the neural state space model with directly parametrized Kalman gain, which is the most general case. The derivation of an LFT for the other neural state space models such as the predictor in the deterministic identification case (11) and partial parametrizations (17)(18) is straightforward then because they can be interpreted as special cases. LFT representations for neural state space models were first introduced in Suykens *et al.* (1993).

4.1 Motivation

Modern robust control theory makes frequently use of LFTs such as in the formulation of the standard robust stability and standard robust performance control problems, where uncertainties are pulled out of an augmented plant model for which a linear controller is designed, taking into account the structure in the uncertainty block and the nature of the uncertainty (such as time-invariant, time varying and nonlinear perturbations, unmodeled dynamics) (Dahleh and Khammash (1993), Doyle *et al.*(1991), Packard and Doyle (1993)). On the other hand there exist the well-known gap between identification and control that comes up if one has to design a controller for a plant, based on identification results from input/output measurements on that plant. The need for new identification schemes that can bring estimated models in the framework of μ-theory is e.g. expressed in Packard and Doyle (1993). According to Smith and Doyle (1992) the application of robust control methods is also hampered by the fact that most popular identification methods assume all uncertainty in the form of additive noise, while modern robust control synthesis techniques aim at providing robustness with respect to uncertainty in the form of both additive noise and plant perturbations.

A candidate method proposed in this paper is the following

1. First nonlinear system identification is done using neural state space models. Both measurement and process noise can be taken into account in the model structure. The number of models that can be represented in the model structures $\mathcal{M}_d$ and $\mathcal{M}_{s,direct}$ is large, which makes the assumption that the true plant can be represented in the model structure more reasonable (an assumption which is almost always violated in practice for linear models).

2. An LFT representation is derived for the obtained neural state space model. The nonlinear model is represented here as a nominal linear model with bounded nonlinear feedback perturbations (to be interpreted as a linear model with parametric uncertainties caused by nonlinear perturbations). In fact two levels of uncertainty are

important here: the first level is the uncertainty characterized by confidence intervals on the estimated parameter vector $\hat{\theta}_N$ for the true parameter vector θ_0; the second level is that the obtained model can be represented as the uncertain linear model. We will only focuss on the second level in this paper.

3. Finally the LFT representation is used in a standard robust performance control scheme, assuming the certainty equivalence principle holds. If not ok, go back to 1.

In fact it is assumed here that the obtained neural state space model is valid in the following senses

1. The prediction error must be unpredictable from all linear and nonlinear combinations of past inputs and outputs. Methods for checking this are discussed e.g. in Billings *et al.*(1992).

2. The obtained neural state space model must perform well on fresh data, that were not used for identification. This the most popular way of model validation in the field of neural networks, where the available data set is normally splitted into a training set and a test set and a performance index is defined on both sets, respectively called fitting error and generalization error (see Hammerstrom (1993) or Zurada (1992) for an introduction). In Ljung (1987) p.416 this is also considered to be a good and pragmatic way of model validation, because there is no need for any probabilistic arguments or assumptions on the true system in this case.

3. It is assumed that the obtained model is also valid in the sense of Smith which means that given experimental data and a model with both additive noise and norm-bounded perturbations, the model can produce the observed input-output data (Smith and Doyle (1992)).

Besides the possibility for using the LFTs in robust control design, the representation provides us with more insight in the neural state space models itself such as e.g. how the model (15) can be interpreted as a model with direct parametrization of the Kalman gain in innovations form.

4.2 Derivation of LFTs

Using the elementwise notation (27)(28) for the predictor (15)

$$\left\{ \begin{array}{rcl} \hat{x}_{k+1} & = & W_{AB} \tanh(V_A \hat{x}_k + V_B u_k + \beta_{AB}) + W_K \tanh(V_K \epsilon_k) \ ; \ \hat{x}_0 = x_0 \\ \hat{y}_k & = & W_{CD} \tanh(V_C \hat{x}_k + V_D u_k + \beta_{CD}) \end{array} \right.$$

we obtain

$$\left\{ \begin{array}{rcl} \hat{x}^i & := & \sum_j w_{AB}{}_j^i \tanh(\varphi^j) + \sum_j w_K{}_j^i \tanh(\rho^j) \\ \hat{y}^i & = & \sum_j w_{CD}{}_j^i \tanh(\psi^j). \end{array} \right. \tag{35}$$

This can be written as

$$\left\{ \begin{array}{rcl} \hat{x}^i & := & \sum_j w_{AB}{}_j^i \gamma_{AB}{}_j^j \varphi^j + \sum_j w_K{}_j^i \gamma_K{}_j^j \rho^j \\ \hat{y}^i & = & \sum_j w_{CD}{}_j^i \gamma_{CD}{}_j^j \psi^j, \end{array} \right. \tag{36}$$

where

$$\begin{array}{l} \gamma_{AB}{}_j^j = \left\{ \begin{array}{ll} \frac{\tanh(\varphi^j)}{\varphi^j} & , (\varphi^j \neq 0) \\ 1 & , (\varphi^j = 0) \ ; \ j = 1, ..., n_{hx} \end{array} \right. \\ \gamma_{CD}{}_j^j = \left\{ \begin{array}{ll} \frac{\tanh(\psi^j)}{\psi^j} & , (\psi^j \neq 0) \\ 1 & , (\psi^j = 0) \ ; \ j = 1, ..., n_{hy} \end{array} \right. \\ \gamma_K{}_j^j = \left\{ \begin{array}{ll} \frac{\tanh(\rho^j)}{\rho^j} & , (\rho^j \neq 0) \\ 1 & , (\rho^j = 0) \ ; \ j = 1, ..., n_{h\epsilon}. \end{array} \right. \end{array}$$

The fact that the γ elements are equal to 1 if the argument of the activation function becomes 0 is easily seen by applying de l' Hospital's rule or by using the Taylor expansion for tanh(.). These elements have the property that they belong to bounded intervals: $\gamma_{AB}{}_j^j \in (0, 1]$, $\gamma_{CD}{}_j^j \in (0, 1]$, $\gamma_K{}_j^j \in (0, 1]$. Turning back again to matrix-vector notation (36) can be written as

$$\left\{ \begin{array}{rcl} \hat{x}_{k+1} & = & W_{AB} \Gamma_{AB}(\hat{x}_k, u_k)(V_A \hat{x}_k + V_B u_k + \beta_{AB}) + W_K \Gamma_K(\epsilon_k) V_K \epsilon_k \\ \hat{y}_k & = & W_{CD} \Gamma_{CD}(\hat{x}_k, u_k)(V_C \hat{x}_k + V_D u_k + \beta_{CD}) \end{array} \right. \tag{37}$$

or

$$\left\{ \begin{array}{rcl} \hat{x}_{k+1} & = & A(\hat{x}_k, u_k)\hat{x}_k + B(\hat{x}_k, u_k)u_k + b_2(\hat{x}_k, u_k) + K(\epsilon_k)\epsilon_k \\ \hat{y}_k & = & C(\hat{x}_k, u_k)\hat{x}_k + D(\hat{x}_k, u_k)u_k + d_2(\hat{x}_k, u_k) \end{array} \right. \tag{38}$$

with diagonal matrices $\Gamma_{AB} = \text{diag}\{\gamma_{AB}{}_1^1, ..., \gamma_{AB}{}_{n_{hx}}^{n_{hx}}\}$, $\Gamma_{CD} = \text{diag}\{\gamma_{CD}{}_1^1, ..., \gamma_{CD}{}_{n_{hy}}^{n_{hy}}\}$, $\Gamma_K = \text{diag}\{\gamma_K{}_1^1, ..., \gamma_K{}_{n_{h\epsilon}}^{n_{h\epsilon}}\}$ and

$$\begin{array}{rcl} A(\hat{x}_k, u_k) = W_{AB} \Gamma_{AB}(\hat{x}_k, u_k) V_A & , & B(\hat{x}_k, u_k) = W_{AB} \Gamma_{AB}(\hat{x}_k, u_k) V_B \\ C(\hat{x}_k, u_k) = W_{CD} \Gamma_{CD}(\hat{x}_k, u_k) V_C & , & D(\hat{x}_k, u_k) = W_{CD} \Gamma_{CD}(\hat{x}_k, u_k) V_D \\ b_2(\hat{x}_k, u_k) = W_{AB} \Gamma_{AB}(\hat{x}_k, u_k) \beta_{AB} & , & d_2(\hat{x}_k, u_k) = W_{CD} \Gamma_{CD}(\hat{x}_k, u_k) \beta_{CD} \\ K(\epsilon_k) = W_K \Gamma_K(\epsilon_k) V_K. & & \end{array} \tag{39}$$

The representation (38)(39) for the model (15) explains in what sense we have a direct parametrization of the Kalman gain and how it serves as a straightforward extension of a linear model in innovations form (30) towards nonlinear systems, especially if $\beta_{AB} = 0$, $\beta_{CD} = 0$ making $b_2 = 0$, $d_2 = 0$.

The following step for representing (38)(39) as an LFT is to define a nominal model. Depending on the input and state vector sequences the elements $\gamma_{AB}{}^j_j$, $\gamma_{CD}{}^j_j$, $\gamma_K{}^j_j$ belong to certain intervals

$$\begin{array}{rclcl} \gamma_{AB}{}^j_j & \in & [\gamma^-_{AB}{}^j_j, \gamma^+_{AB}{}^j_j] & \subset & (0,1] \\ \gamma_{CD}{}^j_j & \in & [\gamma^-_{CD}{}^j_j, \gamma^+_{CD}{}^j_j] & \subset & (0,1] \\ \gamma_K{}^j_j & \in & [\gamma^-_K{}^j_j, \gamma^+_K{}^j_j] & \subset & (0,1]. \end{array} \qquad (40)$$

The nominal γ values will be defined now as the midpoint of these intervals. The choice of these intervals is a matter of degree of conservativeness that one wants to take into account and will depend on the input signals for which the predictor (35) is simulated. The most conservative LFT will be obtained by setting all nominal γ values equal to 0.5 as the midpoint of the intervals (0,1], ensuring independence of the input and state vector sequence. The following definitions are made (see also Steinbuch *et al.*(1992))

$$\begin{array}{rcl} \gamma^{nom\,j}_{AB\;\;j} = (\gamma^-_{AB}{}^j_j + \gamma^+_{AB}{}^j_j)/2 & , & s_{AB}{}^j_j = (\gamma^+_{AB}{}^j_j - \gamma^-_{AB}{}^j_j)/2 \\ \gamma^{nom\,j}_{CD\;\;j} = (\gamma^-_{CD}{}^j_j + \gamma^+_{CD}{}^j_j)/2 & , & s_{CD}{}^j_j = (\gamma^+_{CD}{}^j_j - \gamma^-_{CD}{}^j_j)/2 \\ \gamma^{nom\,j}_{K\;\;j} = (\gamma^-_K{}^j_j + \gamma^+_K{}^j_j)/2 & , & s_K{}^j_j = (\gamma^+_K{}^j_j - \gamma^-_K{}^j_j)/2 \end{array} \qquad (41)$$

and

$$\begin{array}{lcl} \Gamma^{nom}_{AB} = \mathrm{diag}\{\gamma^{\mathrm{nom1}}_{\mathrm{AB}\;\;1}, ..., \gamma^{\mathrm{nomn_{hx}}}_{\mathrm{AB}\;\;\mathrm{n_{hx}}}\} & , & S_{AB} = \mathrm{diag}\{\mathrm{s_{AB}}{}^1_1, ..., \mathrm{s_{AB}}{}^{\mathrm{n_{hx}}}_{\mathrm{n_{hx}}}\} \\ \Gamma^{nom}_{CD} = \mathrm{diag}\{\gamma^{\mathrm{nom1}}_{\mathrm{CD}\;\;1}, ..., \gamma^{\mathrm{nomn_{hy}}}_{\mathrm{CD}\;\;\mathrm{n_{hy}}}\} & , & S_{CD} = \mathrm{diag}\{\mathrm{s_{CD}}{}^1_1, ..., \mathrm{s_{CD}}{}^{\mathrm{n_{hy}}}_{\mathrm{n_{hy}}}\} \\ \Gamma^{nom}_{K} = \mathrm{diag}\{\gamma^{\mathrm{nom1}}_{\mathrm{K}\;\;1}, ..., \gamma^{\mathrm{nomn_{h\epsilon}}}_{\mathrm{K}\;\;\mathrm{n_{h\epsilon}}}\} & , & S_{K} = \mathrm{diag}\{\mathrm{s_{K}}{}^1_1, ..., \mathrm{s_{K}}{}^{\mathrm{n_{h\epsilon}}}_{\mathrm{n_{h\epsilon}}}\} \end{array} \qquad (42)$$

and

$$\begin{array}{lcl} \Gamma_{AB} = \Gamma^{nom}_{AB} + S_{AB}\,\Delta_{AB} & , & \Delta_{AB} = \mathrm{diag}\{\delta_{\mathrm{AB}}{}^1_1, ..., \delta_{\mathrm{AB}}{}^{\mathrm{n_{hx}}}_{\mathrm{n_{hx}}}\} \\ \Gamma_{CD} = \Gamma^{nom}_{CD} + S_{CD}\,\Delta_{CD} & , & \Delta_{CD} = \mathrm{diag}\{\delta_{\mathrm{CD}}{}^1_1, ..., \delta_{\mathrm{CD}}{}^{\mathrm{n_{hy}}}_{\mathrm{n_{hy}}}\} \\ \Gamma_{K} = \Gamma^{nom}_{K} + S_{K}\,\Delta_{K} & , & \Delta_{K} = \mathrm{diag}\{\delta_{\mathrm{K}}{}^1_1, ..., \delta_{\mathrm{K}}{}^{\mathrm{n_{h\epsilon}}}_{\mathrm{n_{h\epsilon}}}\} \end{array} \qquad (43)$$

with ${\delta_{AB}}_j^j \in [-1,1]$, ${\delta_{CD}}_j^j \in [-1,1]$, ${\delta_K}_j^j \in [-1,1]$, such that $\|\Delta_{AB}\| \leq 1$, $\|\Delta_{CD}\| \leq 1$, $\|\Delta_K\| \leq 1$. Hence the matrices in (38) can be written as

$$\begin{array}{lcl}
A(\hat{x}_k, u_k) = A^{nom} + A_\delta(\delta_{AB}) & , & B(\hat{x}_k, u_k) = B^{nom} + B_\delta(\delta_{AB}) \\[2ex]
C(\hat{x}_k, u_k) = C^{nom} + C_\delta(\delta_{CD}) & , & D(\hat{x}_k, u_k) = D^{nom} + D_\delta(\delta_{CD}) \\[2ex]
b_2(\hat{x}_k, u_k) = b_2^{nom} + b_{2_\delta}(\delta_{AB}) & , & d_2(\hat{x}_k, u_k) = d_2^{nom} + d_{2_\delta}(\delta_{CD}) \\[2ex]
K(\epsilon_k) = K^{nom} + K_\delta(\delta_K) & &
\end{array} \tag{44}$$

with

$$\begin{array}{lclclcl}
A^{nom} & = & W_{AB}\Gamma_{AB}^{nom}V_A & , & A_\delta(\delta_{AB}) & = & W_{AB}S_{AB}\Delta_{AB}V_A \\[1.5ex]
B^{nom} & = & W_{AB}\Gamma_{AB}^{nom}V_B & , & B_\delta(\delta_{AB}) & = & W_{AB}S_{AB}\Delta_{AB}V_B \\[1.5ex]
C^{nom} & = & W_{CD}\Gamma_{CD}^{nom}V_C & , & C_\delta(\delta_{CD}) & = & W_{CD}S_{CD}\Delta_{CD}V_C \\[1.5ex]
D^{nom} & = & W_{CD}\Gamma_{CD}^{nom}V_D & , & D_\delta(\delta_{CD}) & = & W_{CD}S_{CD}\Delta_{CD}V_D \\[1.5ex]
b_2^{nom} & = & W_{AB}\Gamma_{AB}^{nom}\beta_{AB} & , & b_{2_\delta}(\delta_{AB}) & = & W_{AB}S_{AB}\Delta_{AB}\beta_{AB} \\[1.5ex]
d_2^{nom} & = & W_{CD}\Gamma_{CD}^{nom}\beta_{CD} & , & d_{2_\delta}(\delta_{CD}) & = & W_{CD}S_{CD}\Delta_{CD}\beta_{CD} \\[1.5ex]
K^{nom} & = & W_K\Gamma_K^{nom}V_K & , & K_\delta(\delta_K) & = & W_K S_K \Delta_K V_K.
\end{array} \tag{45}$$

Finally an upper Linear Fractional Transformation (LFT) is obtained then for the neural state space model with directly parametrized Kalman gain (Fig.4)

$$\hat{y} = \mathcal{F}_u(G, \Delta) \begin{bmatrix} u \\ \epsilon \\ 1 \end{bmatrix} \tag{46}$$

with state space representation

$$\left\{ \begin{array}{llcl}
G: & \left[\begin{array}{c} \hat{x}_{k+1} \\ \hline \hat{y}_k \\ q_k \end{array} \right] & = & \left[\begin{array}{c|cccc}
A^{nom} & B^{nom} & K^{nom} & b_2^{nom} & [W_{AB}S_{AB}\ W_K S_K\ 0] \\ \hline
C^{nom} & D^{nom} & 0 & d_2^{nom} & [0\ 0\ W_{CD}S_{CD}] \\
\begin{bmatrix} V_A \\ 0 \\ V_C \end{bmatrix} & \begin{bmatrix} V_B \\ 0 \\ V_D \end{bmatrix} & \begin{bmatrix} 0 \\ V_K \\ 0 \end{bmatrix} & \begin{bmatrix} \beta_{AB} \\ 0 \\ \beta_{CD} \end{bmatrix} & 0
\end{array} \right] . \left[\begin{array}{c} \hat{x}_k \\ \hline u_k \\ \epsilon_k \\ 1 \\ p_k \end{array} \right] \\[4ex]
\Delta: & p_k & = & \Delta \,.\, q_k \quad , \Delta = \mathrm{diag}\{\Delta_{\mathrm{AB}}, \Delta_{\mathrm{K}}, \Delta_{\mathrm{CD}}\} \quad , \|\Delta\| \leq 1.
\end{array} \right. \tag{47}$$

An LFT representation for the neural state space model in the deterministic identification case (Fig.4)

$$\hat{y} = \mathcal{F}_u(G,\Delta) \begin{bmatrix} u \\ 1 \end{bmatrix} \tag{48}$$

can be seen as a special case of (46)-(47) ($W_K = 0$,$V_K = 0$), with state space representation

$$\left\{ \begin{array}{llll} G: & \left[\begin{array}{c} \hat{x}_{k+1} \\ \hline \hat{y}_k \\ q_k \end{array} \right] & = & \left[\begin{array}{c|ccc} A^{nom} & B^{nom} & b_2^{nom} & [W_{AB} S_{AB} \; 0] \\ \hline C^{nom} & D^{nom} & d_2^{nom} & [0 \; W_{CD} S_{CD}] \\ \begin{bmatrix} V_A \\ V_C \end{bmatrix} & \begin{bmatrix} V_B \\ V_D \end{bmatrix} & \begin{bmatrix} \beta_{AB} \\ \beta_{CD} \end{bmatrix} & 0 \end{array} \right] . \left[\begin{array}{c} \hat{x}_k \\ \hline u_k \\ 1 \\ p_k \end{array} \right] \\ & & & \\ \Delta: & p_k & = & \Delta \, . \, q_k \qquad , \Delta = \mathrm{diag}\{\Delta_{\mathrm{AB}}, \Delta_{\mathrm{CD}}\} \qquad , \|\Delta\| \leq 1. \end{array} \right. \tag{49}$$

At the component level the uncertainty is caused by the γ elements and corresponding δ elements which depend nonlinearily on the input and state vector. The interconnection matrices of the neural state space model are assumed to be exact, although there exist uncertainty on these matrices because of the identification procedure. At the system level this uncertainty becomes *structured* because the matrix Δ is diagonal (see also Doyle *et al.*(1991)). The uncertainty is *real* and the dimension of Δ does only depend upon the number of hidden neurons in the neural net architectures. Finally the LFT representations (46)-(49) can be used in a standard robust stability or robust performance control scheme (Fig.5).

Remarks:

- In the standard robust control scheme of Fig.6 with augmented plant P the exogenous input vector w and the regulated output z, related to the 1-DOF control schemes of Fig.5 consist respectively of the variables r, ϵ, 1 and $r - \hat{y}$ (tracking error), u (see e.g. Boyd and Barratt (1991)). The constant input 1, due to the bias vectors of the neural nets, can be interpreted as an extra disturbance signal. Measurement noise in w is characterized by the white noise innovations sequence ϵ for $\mathcal{M}_d$. For $\mathcal{M}_{s,direct}$ both process noise and measurement noise in w are related to ϵ because of the directly parametrized Kalman gain in innovations form.

- LFT representations for simulation model predictors related to the partial parametrizations (17) and (18) are a special case of (48)(49) and correspond respectively to

$$\left\{\begin{array}{llll} G: & \begin{bmatrix} \hat{x}_{k+1} \\ q_k \end{bmatrix} & = & \left[\begin{array}{c|ccc} A^{nom} & B & b_2^{nom} & W_A S_A \\ \hline V_A & 0 & \beta_A & 0 \end{array}\right] . \left[\begin{array}{c} \hat{x}_k \\ \hline u_k \\ 1 \\ p_k \end{array}\right] \\ & & & \\ \Delta: & p_k & = & \Delta_A . q_k \qquad , \|\Delta_A\| \leq 1 \end{array}\right. \tag{50}$$

with $b_2^{nom} = W_A \Gamma_A^{nom} \beta_A$ and to

$$\left\{\begin{array}{llll} G: & \begin{bmatrix} \hat{x}_{k+1} \\ q_k \end{bmatrix} & = & \left[\begin{array}{c|ccc} A & B^{nom} & b_2^{nom} & W_B S_B \\ \hline 0 & V_B & \beta_B & 0 \end{array}\right] . \left[\begin{array}{c} \hat{x}_k \\ \hline u_k \\ 1 \\ p_k \end{array}\right] \\ & & & \\ \Delta: & p_k & = & \Delta_B . q_k \qquad , \|\Delta_B\| \leq 1 \end{array}\right. \tag{51}$$

with $b_2^{nom} = W_B \Gamma_B^{nom} \beta_B$.

- Results based on the LFTs (46)(48) are only rigorous if a nominal model was defined by setting $\gamma_{AB}{}_j^j = 0.5$, $\gamma_{CD}{}_j^j = 0.5$, $\gamma_K{}_j^j = 0.5$. All other choices are heuristic because they depend on the input vector and state vector sequence but on the other hand may lead to less conservative results.

- The size of the intervals $[\gamma^{-}_{AB}{}^{j}_{j}, \gamma^{+}_{AB}{}^{j}_{j}]$, $[\gamma^{-}_{CD}{}^{j}_{j}, \gamma^{+}_{CD}{}^{j}_{j}]$, $[\gamma^{-}_{K}{}^{j}_{j}, \gamma^{+}_{K}{}^{j}_{j}]$ gives an indication of the 'hardness' of nonlinearity of the underlying nonlinear system: a maximal 'distortion' is obtained if these intervals coincide with $(0,1]$ and a minimal 'distortion' for the limiting case $\gamma^{+}_{AB}{}^{j}_{j} = 1$, $\gamma^{+}_{CD}{}^{j}_{j} = 1$, $\gamma^{+}_{K}{}^{j}_{j} = 1$ and $\gamma^{-}_{AB}{}^{j}_{j} \rightarrow 1$, $\gamma^{-}_{CD}{}^{j}_{j} \rightarrow 1$, $\gamma^{-}_{K}{}^{j}_{j} \rightarrow 1$, corresponding to a linear model. The larger the size of the intervals $[\gamma^{-}_{AB}{}^{j}_{j}, \gamma^{+}_{AB}{}^{j}_{j}]$, $[\gamma^{-}_{CD}{}^{j}_{j}, \gamma^{+}_{CD}{}^{j}_{j}]$, $[\gamma^{-}_{K}{}^{j}_{j}, \gamma^{+}_{K}{}^{j}_{j}]$, the higher the 'distortion' will be and the more difficult it will become to find a linear robustly stabilizing controller that can cope with the uncertainty characterized by these intervals.

5 Example

An example is given on nonlinear system identification using the neural state space model with directly parametrized Kalman gain (15). The system to be identified is an interconnected system, consisting of two dynamical subsystems and two static nonlinearities: a hysteresis curve $f_1(.)$ and a hyperbolic tangent function $f_2(.) = \tanh(.)$ (see Fig.7). The linear systems L and M are both SISO of order 2 and 1 with state vectors x_k and z_k respectively. The interconnected system with input u_k, output y_k and state vector $[x_k; z_k]$ has the following form

$$\left\{ \begin{array}{rcl} x_{k+1} & = & A_L x_k + b_L u_k + \begin{bmatrix} 1 \\ 0 \end{bmatrix} v_k \\ z_{k+1} & = & a_M z_k + b_M f_1(c_L^t x_k) \\ \\ y_k & = & f_2(c_M z_k + d_M f_1(c_L^t x_k)) + w_k \end{array} \right. \tag{52}$$

with v_k, w_k zero mean white Gaussian noise processes. The I/O data were generated by a random input signal, uniformly distributed in the interval [-1,1]. Process noise v_k and measurement noise w_k have both standard deviation 0.01. The system matrices for L and M are

$$A_L = \begin{bmatrix} 0.1 & -0.2 \\ 1 & 0.3 \end{bmatrix}, b_L = \begin{bmatrix} 0 \\ 1 \end{bmatrix}, c_L = \begin{bmatrix} 1 \\ 0 \end{bmatrix}, d_L = 0, a_M = 0.7, b_M = c_M = d_M = 1$$

The nonlinearity $f_1(.)$ is shown in Fig.8 and defined by the following table

	$x_2 > 0$	$x_2 \leq 0$
$-c-d \leq x_1 \leq -c+d$	$f_1 = -c$	-
$-c+d \leq x_1 \leq c+d$	$f_1 = x_1 - d$	-
$c-d \leq x_1 \leq c+d$	-	$f_1 = c$
$-c-d \leq x_1 \leq c-d$	-	$f_1 = x_1 + d$
$-c-d \leq x_1$	$f_1 = -c$	$f_1 = -c$
$x_1 \leq c+d$	$f_1 = c$	$f_1 = c$

which means that the right or left part of the curve is selected depending on the sign of x_2. In total 2000 data points were generated by (52) with $c = 1$, $d = 0.2$ in f_1. This data set is splitted into two parts: a training set containing the first 1000 data points ($N_{fit} = 1000$) and a test set consisting of the following 1000 data points ($N_{gen} = 1000$) which are fresh data to test the obtained models. Corresponding fitting error $V_{N_{fit}}$ and generalization error $V_{N_{gen}}$ are defined on these sets. As predictor a neural state space model (15) was taken with $n = 3$, $n_{hx} = n_{hy} = 7$, $n_{h\epsilon} = 2$. In order to minimize the

cost function (23) a quasi-Newton method with BFGS updating of the Hessian and a mixed quadratic and cubic line search was used (function fminu of Matlab's optimization toolbox) (Matlab User's Guide (1992)). Simulation of the neural state space model and its corresponding sensitivity model, needed to generate the gradient of the cost function, were both written in C code, making use of Matlab's *mex* facilty. The best local minimum after taking 100 different starting points (according to a random Gaussian distribution with standard deviation 0.5) was $V_{N_{fit}} = 6.3848\,e-04$. This model had also a minimal generalization error equal to $V_{N_{gen}} = 1.5803\,e-03$. Model validation tests were done according to Billings *et al.*(1992): residuals ϵ_k should be unpredictable from all linear and nonlinear combinations of past inputs and outputs. The following conditions should hold: $\phi_{\epsilon\epsilon}(\tau) = E[\epsilon_{k-\tau}\epsilon_k] = \delta_\tau$, $\phi_{u\epsilon}(\tau) = E[u_{k-\tau}\epsilon_k] = 0\ (\forall\tau)$, $\phi_{u^{2'}\epsilon}(\tau) = E[(u^2_{k-\tau} - \overline{u}^2_k)\epsilon_k] = 0\ (\forall\tau)$, $\phi_{u^{2'}\epsilon^2}(\tau) = E[(u^2_{k-\tau} - \overline{u}^2_k)\epsilon^2_k] = 0\ (\forall\tau)$, $\phi_{\epsilon(\epsilon u)}(\tau) = E[\epsilon_k\epsilon_{k-1-\tau}u_{k-1-\tau}] = 0\ (\tau \geq 0)$. Here $u^{2'} = u^2_k - \overline{u^2_k}$, where $\overline{u^2_k}$ denotes the mean of u^2_k. In practice normalized correlations are computed. The sampled correlation function between two sequences α_k and β_k is given by $\hat{\phi}_{\alpha\beta}(\tau) = (\sum_{k=1}^{N-\tau}\alpha_k\beta_{k+\tau})/[\sum_{k=1}^{N}\alpha^2_k\sum_{k=1}^{N}\beta^2_k]^{1/2}$. This normalization ensures that $-1 \leq \hat{\phi}_{\alpha\beta} \leq 1$. 95% confidence bands are defined as $1.96/\sqrt{N}$ (N is data length). These tests are shown in Fig.9 for the training data.
The intervals $[\gamma^{-\,j}_{AB\,j}, \gamma^{+\,j}_{AB\,j}]$, $[\gamma^{-\,j}_{CD\,j}, \gamma^{+\,j}_{CD\,j}]$, $[\gamma^{-\,j}_{K\,j}, \gamma^{+\,j}_{K\,j}]$ were calculated based on the training data and the optimal model with as result

$$\gamma^{+\,j}_{AB\,j} = 1, \gamma^{+\,j}_{CD\,j} = 1, \gamma^{+\,j}_{K\,j} = 1$$

and

$$\gamma^-_{AB} = [0.0562 \quad 0.8942 \quad 0.7313 \quad 0.7659 \quad 0.8904 \quad 0.8306 \quad 0.3841]^t$$

$$\gamma^-_{CD} = [0.1785 \quad 0.6271 \quad 0.5012 \quad 0.6022 \quad 0.0244 \quad 0.1877 \quad 0.2284]^t$$

$$\gamma^-_K = [0.9920 \quad 0.9997]^t$$

which indicates that the nonlinearity of the underlying system is rather 'hard' except for the neural net responsible for the Kalman gain: the $\gamma^{-\,j}_{K\,j}$ elements close to 1 indicate that the system dynamics depend linearily on the innovations input ϵ_k, which is indeed also the case for the true system (52). This is also seen in the plots of Fig.10: the elements of the matrices $A(\hat{x}_k, u_k)$, $B(\hat{x}_k, u_k)$, $C(\hat{x}_k, u_k)$, $D(\hat{x}_k, u_k)$, $K(\epsilon_k)$ are plotted with respect to time for a part of the training data. The variation on the elements of K is small.

6 Conclusions

A neural state space model framework for nonlinear system identification is proposed. Both models for the deterministic identification case and the stochastic case with process noise and measurement noise are treated. Prediction error algorithms are discussed where the gradients of the cost function are generated by a Narendra's sensitivity model. LFT representations are given which make it possible to interpret a given neural state space model as a nominal linear model with bounded nonlinear feedback perturbation and to use it in a standard robust performance control scheme. Further possible research directions are e.g.to look for parametrizations of neural state space models with a minimal number of parameters, the development of on-line and parallel learning algorithms and to work out real life examples on robust control, based on identification results from neural state space models.

References

Billings S.A., H.B. Jamaluddin, S. Chen (1992). Properties of neural networks with applications to modelling non-linear dynamical systems, *Int. J. Control*, Vol.55, No.1, pp. 193-224.

Boyd S., C. Barratt (1991). *Linear controller design, limits of performance*, Prentice-Hall.

Chen S., S. Billings, P. Grant (1990a). Nonlinear system identification using neural networks, *Int. J. Control*, Vol.51, No.6, pp. 1191-1214.

Chen S., C. Cowan, S. Billings, P. Grant (1990b). Parallel recursive prediction error algorithm for training layered neural networks, *Int. J. Control*, Vol.51, No.6, pp. 1215-1228.

Cybenko G. (1989). Approximations by superpositions of a sigmoidal function, *Mathematics of Control, Signals and Systems,* 2, pp.183-192.

Dahleh M.A., M.H. Khammash (1993). Controller design for plants with structured uncertainty, *Automatica, Special Issue on Robust Control*, Vol.29, No.1, pp.37-56.

De Moor B., P. Van Overschee, J. Suykens (1991). *Subspace algorithms for system identification and stochastic realization*, Recent Advances in Mathematical Theory of Systems, Control, Networks and Signal Processing, Proc. of the International Symposium MTNS-91, Kobe, Japan, MITA Press, pp. 589-595, June 17-21.

Doyle J., A. Packard, K. Zhou (1991). *Review of LFTs, LMIs, and μ*, Proc. of the 30th Conference on Decision and Control, Brighton, England, Dec, pp.1227-1232.

Fletcher R. (1987). *Practical methods of optimization*, second edition, Chichester and New York: John Wiley and Sons.

Funahashi K-I. (1989). On the Approximate Realization of continuous Mappings by Neural Networks, *Neural Networks*, Vol.2, pp 183-192.

Gill P.E., W. Murray, M.H. Wright (1981). *Practical Optimization*, London: Academic Press.

Goodwin G., K. Sin (1984). *Adaptive filtering, prediction and control*, Prentice-Hall.

Hammerstrom D. (1993). Working with neural networks, *IEEE Spectrum*, July, pp.46-53.

Hornik K., M. Stinchcombe, H. White (1989). Multilayer feedforward networks are universal approximators, *Neural Networks*, Vol.2, pp.359-366.

Hunt K.J., D. Sbarbaro, R. Zbikowski, P.J. Gawthrop (1992). Neural networks for control systems - a survey, *Automatica*, Vol.28, No.6, pp.1083-1112.

Leshno M., V.Y. Lin, A. Pinkus, S. Schocken (1993). Multilayer feedforward networks with a nonpolynomial activation function can approximate any function, *Neural Networks*, Vol.6, pp.861-867.

Ljung L. (1979). Asymptotic behavior of the Extended Kalman Filter as a parameter estimator for linear systems, *IEEE Trans. Automatic Control*, Vol.AC-24, No.1, pp.36-50.

Ljung L. (1987). *System Identification: Theory for the User*, Prentice-Hall.

Maciejowski J.M. (1989). *Multivariable feedback design*, Addison-Wesley.

Matlab User's Guide (1992). *Optimization Toolbox User's Guide*, The MathWorks, Inc., Version 4.1.

Narendra K.S., K. Parthasarathy (1990). Identification and control of dynamical systems using neural networks, *IEEE Trans. on Neural Networks*, Vol.1, No.1, pp. 4-27.

Narendra K.S., K. Parthasarathy (1991). Gradient methods for the optimization of dynamical systems containing neural networks, *IEEE Trans. on Neural Networks*, Vol.2, No.2, pp.252-262.

Packard A., J. Doyle (1993). The complex structured singular value, *Automatica, Special Issue on Robust Control*, Vol.29, No.1, pp.71-109.

Rumelhart D., G. Hinton, R. Williams (1986). Learning representations by back-propagating errors, *Nature*, 323, pp.533-536.

Smith R.S., J. Doyle (1992). Model validation: a connection between robust control and identification, *IEEE Trans. Automatic Control, Special issue on system identification for robust control design*, Vol.37,No.7, pp.942-952.

Steinbuch M., J. Terlouw, O. Bosgra, S. Smit (1992). Uncertainty modelling and structured singular-value computation applied to an electromechanical system, *IEE Proc-D*, Vol.139, No.3, pp.301-307.

Suykens J., B. De Moor, J. Vandewalle (1993). *Neural network models as linear systems with bounded uncertainty, applicable to robust controller design*, NOLTA 93 International Symposium on Nonlinear Theory and its Applications, Honolulu Hawaii, Dec., pp.419-422.

Van Overschee P., B. De Moor (1994). N4SID : Subspace Algorithms for the Identification of Combined Deterministic-Stochastic Systems, *Automatica, Special Issue on Statistical Signal Processing and Control*, Vol.30, No.1, pp.75-93.

Zurada J.M. (1992). *Introduction to Artificial Neural Systems*, West Publishing Company.

List of Captions

Figure 1. Neural state space model for deterministic system identification: simulation model parametrized by feedforward neural nets.

Figure 2. Neural state space model for the stochastic case with process noise: predictor with directly parametrized Kalman gain in innovations form, parametrized by feedforward neural nets.

Figure 3. A nonlinear dynamic model and its corresponding Narendra's sensitivity model for generating the gradient of the cost function with respect to the parameter vector θ in a prediction error learning algorithm $(\alpha, \beta \in \theta)$.

Figure 4. LFT representation of neural state space models for deterministic identification (left) and the stochastic case with process noise (right). G is related to a nominal linear system. Bounded uncertainty is pulled out in the block Δ, represented in feedback form. The uncertainty at this system level is *structured* because of the diagonal structure of Δ. The elements of Δ are real and of nonlinear nature.

Figure 5. Example of using neural state space models in LFT representation (of Fig.4) in a 1-DOF control scheme, assuming the certainty equivalence principle holds.

Figure 6. Standard robust performance control scheme with augmented plant P. Related to Fig.5 the exogenous input vector w consists of r, 1 and ϵ and the regulated output z to $r - \hat{y}$ and u. y is the sensed output and u the actuator input.

Figure 7. Nonlinear interconnected system consisting of two linear dynamic systems L and M, respectively of order 2 and 1, and two static nonlinearities: a hysteresis curve $f_1(.)$ and $f_2(.) = \tanh(.)$. The system is corrupted with process noise v and measurement noise w.

Figure 8. Hysteresis curve $f_1(x_1)$. Depending on the sign of x_2 the right or left part of the curve is selected.

Figure 9. Model validation: normalized correlation tests with 95% confidence intervals for the model with minimal fitting and generalization error, evaluated on the training set: a/ $\hat{\phi}_{\epsilon\epsilon}(\tau)$, b/ $\hat{\phi}_{u\epsilon}(\tau)$, c/ $\hat{\phi}_{u^{2'}\epsilon}(\tau)$, d/ $\hat{\phi}_{u^{2'}\epsilon^2}(\tau)$, e/ $\hat{\phi}_{\epsilon(\epsilon u)}(\tau)$.

Figure 10. Illustration of parametric uncertainties on the elements of the matrices $A(\hat{x}_k, u_k)$

(Fig.a), $B(\hat{x}_k, u_k)$ (Fig.b), $C(\hat{x}_k, u_k)$ (Fig.c), $D(\hat{x}_k, u_k)$ (Fig.d), $K(\epsilon_k)$ (Fig.e), evaluated on part of the training set. The variation on the elements of the Kalman gain is small.

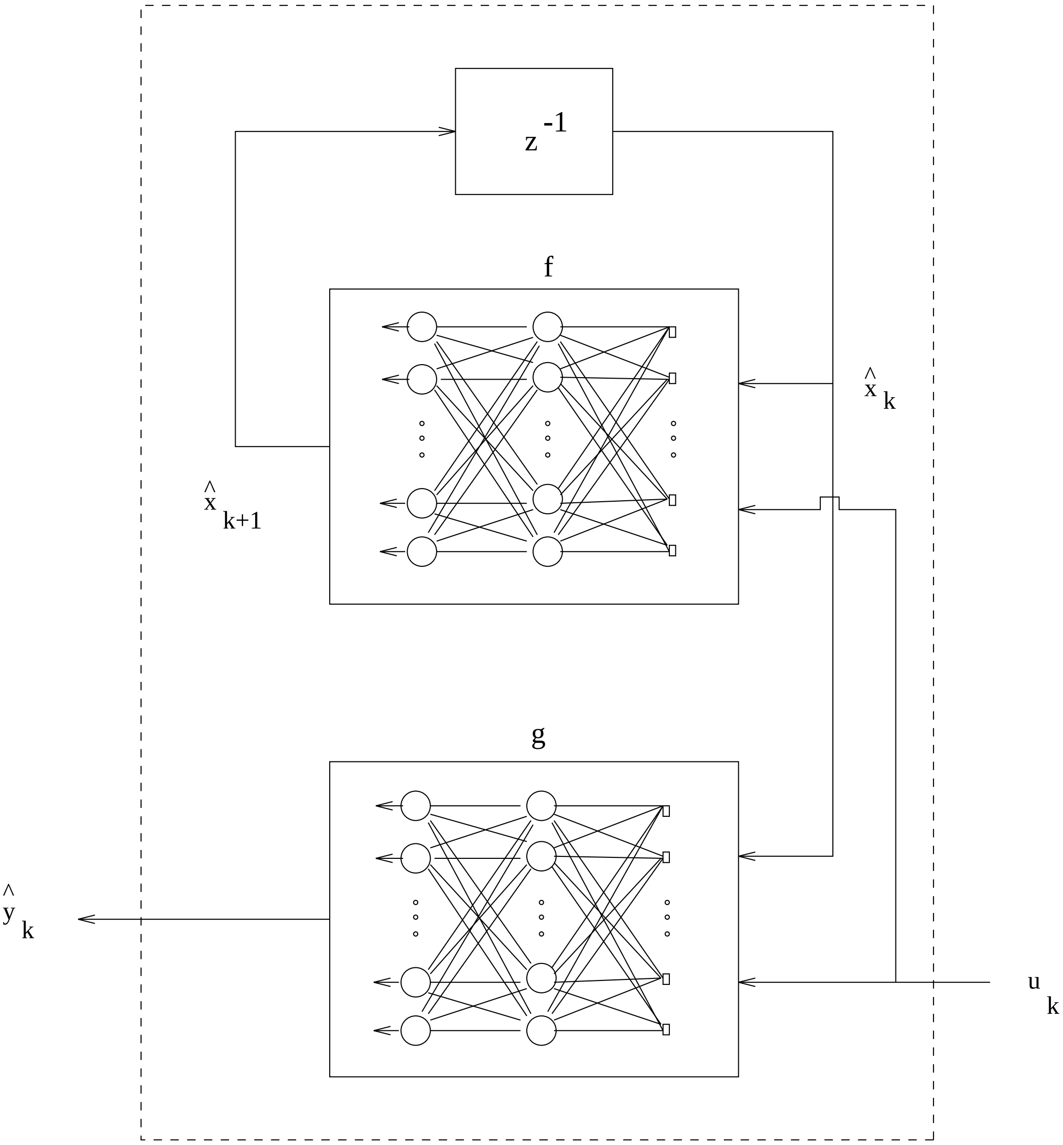

Figure 1:

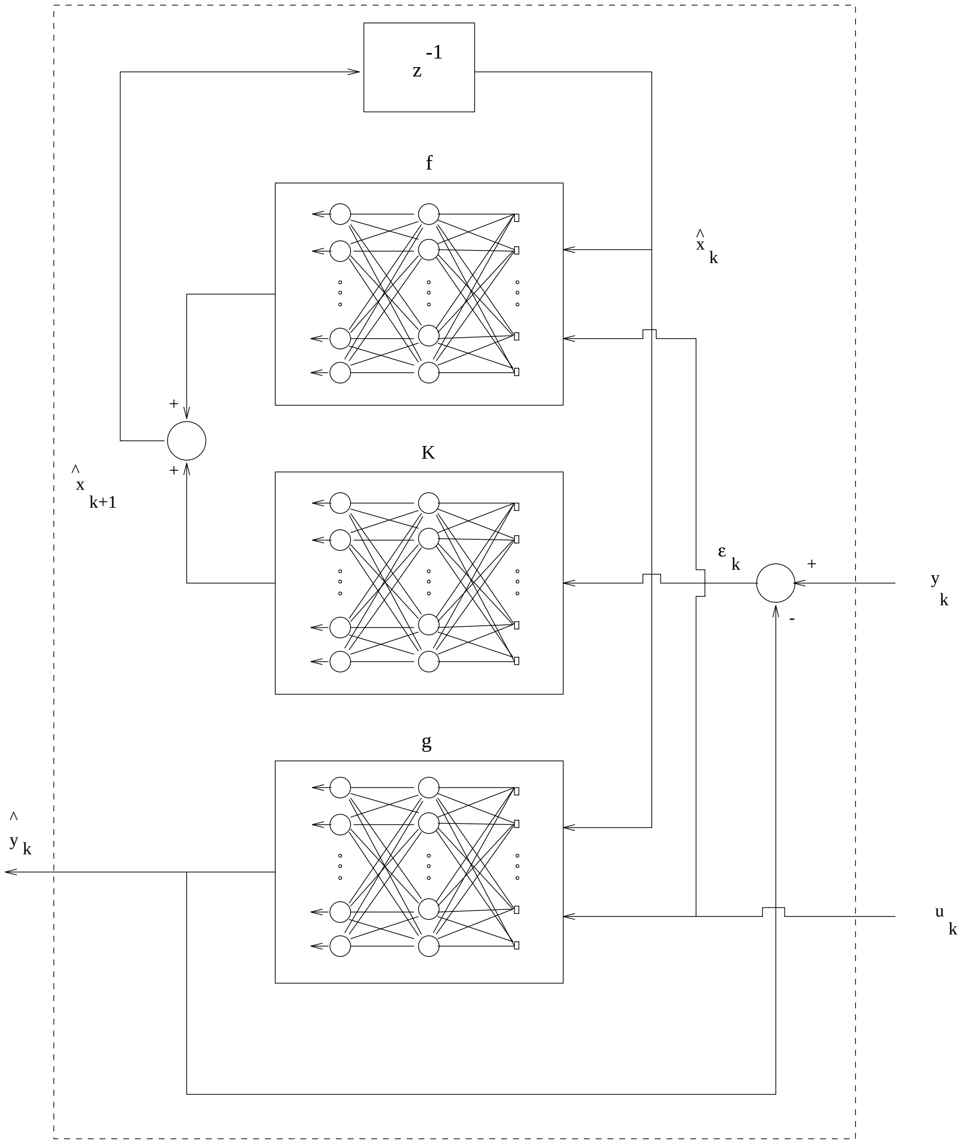

Figure 2:

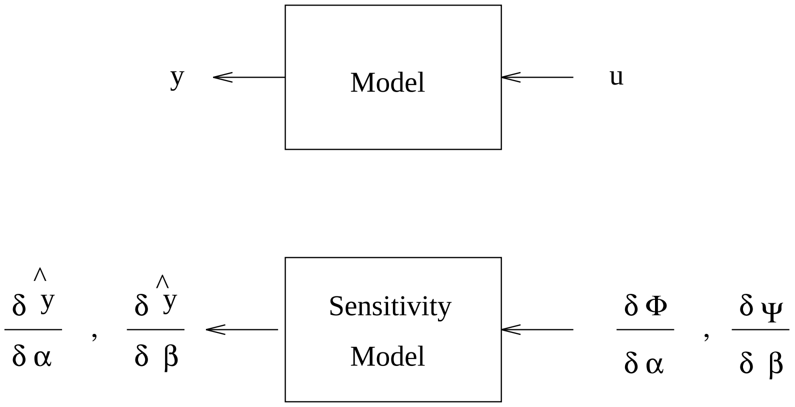
y
Model
u
$\frac{\delta \hat{y}}{\delta \alpha}$, $\frac{\delta \hat{y}}{\delta \beta}$
Sensitivity Model
$\frac{\delta \Phi}{\delta \alpha}$, $\frac{\delta \Psi}{\delta \beta}$

Figure 3:

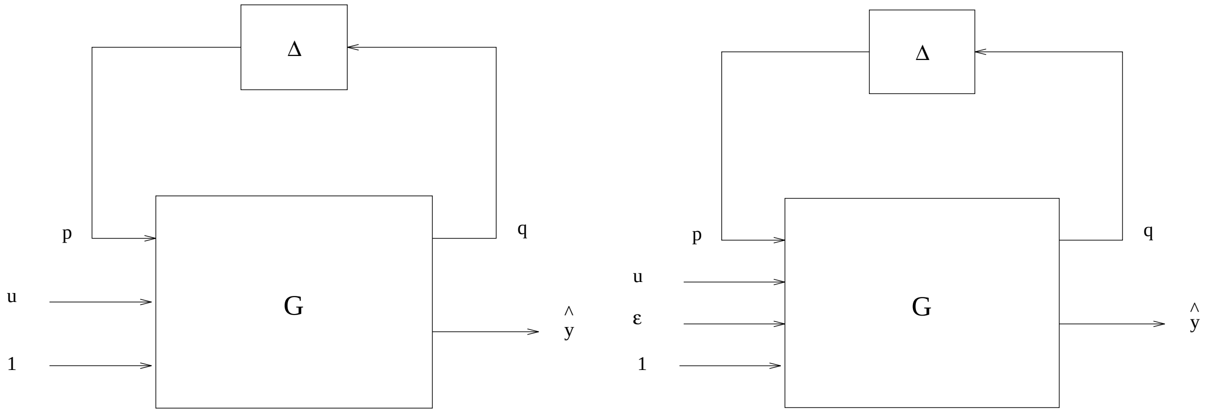

Figure 4:

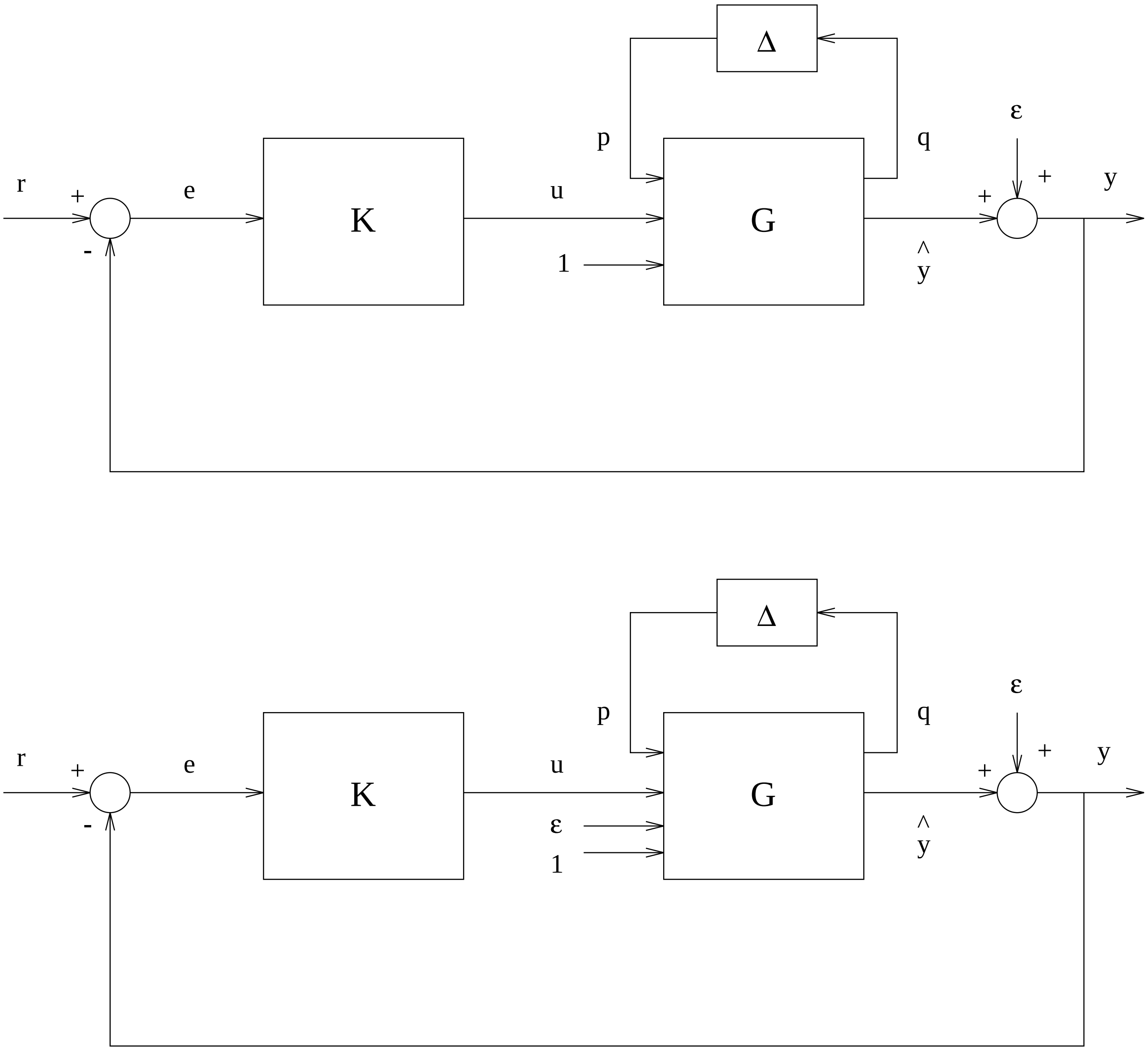

Figure 5:

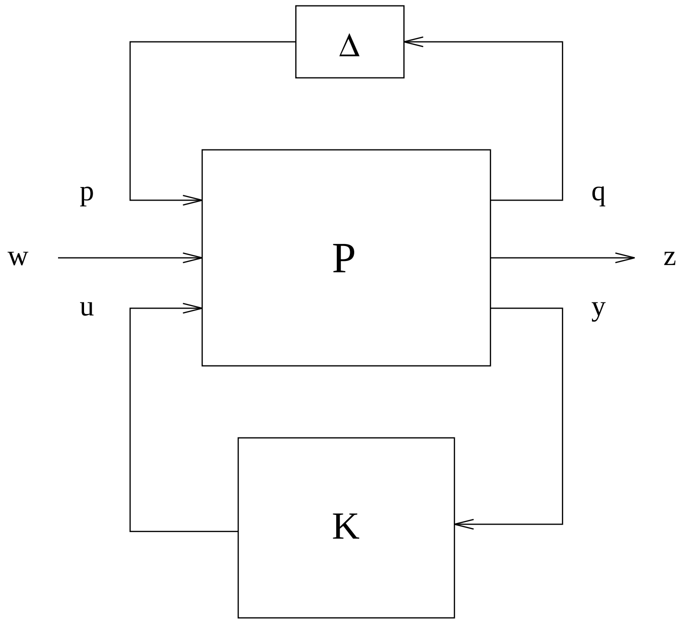

Figure 6:

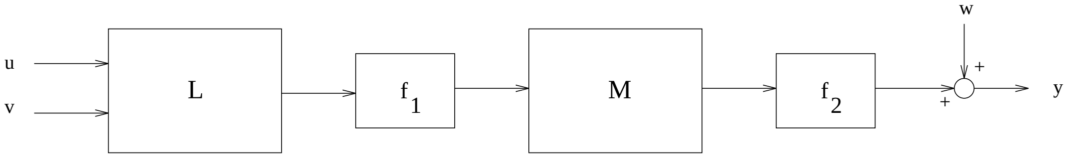

Figure 7:

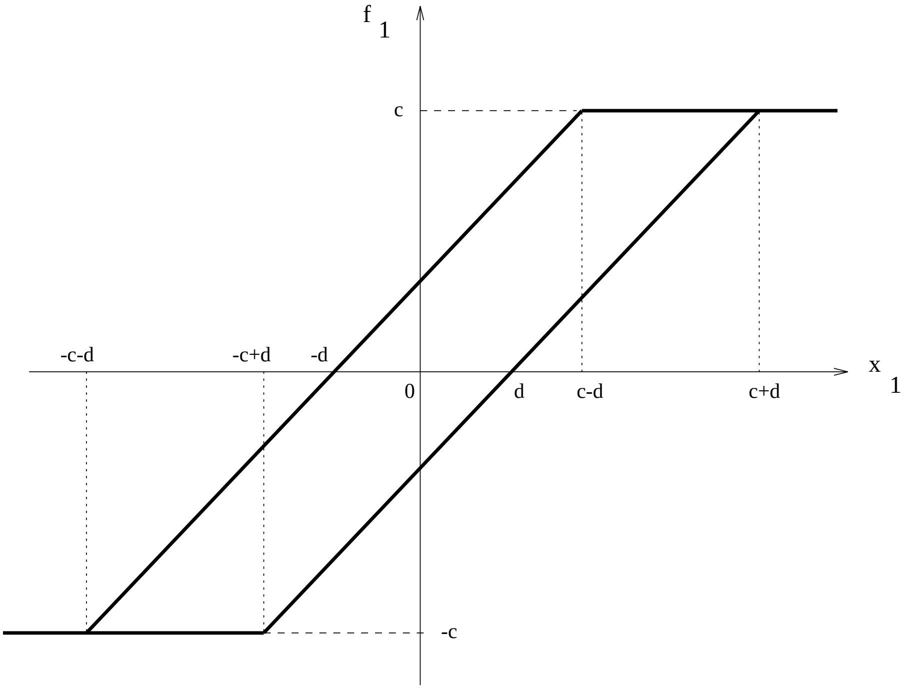

Figure 8:

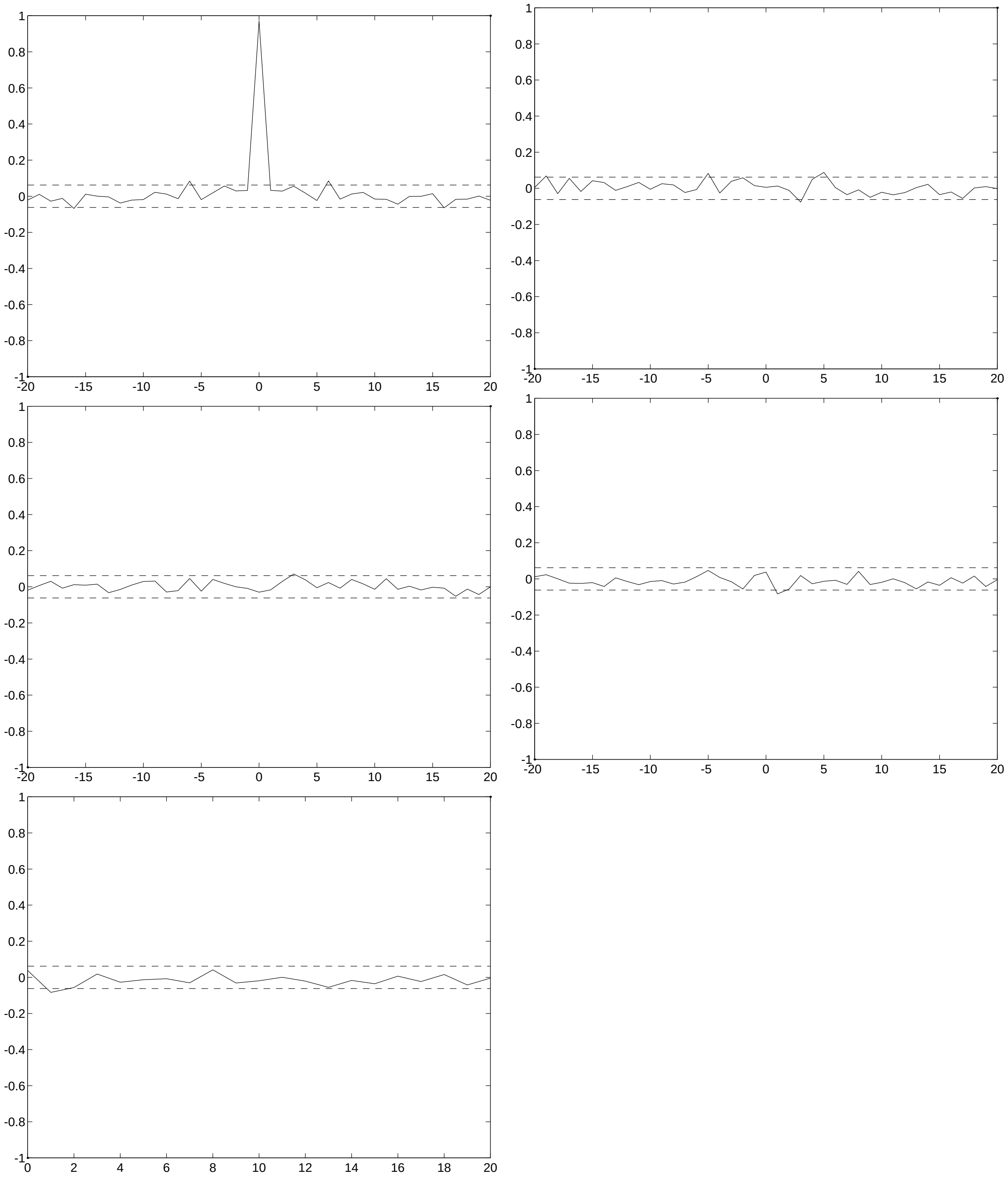
1
0.8
0.6
0.4
0.2
0
-0.2
-0.4
-0.6
-0.8
-1
-20
-15
-10
-5
0
5
10
15
20
0
2
4
6
8
10
12
14
16
18
20

Figure 9:

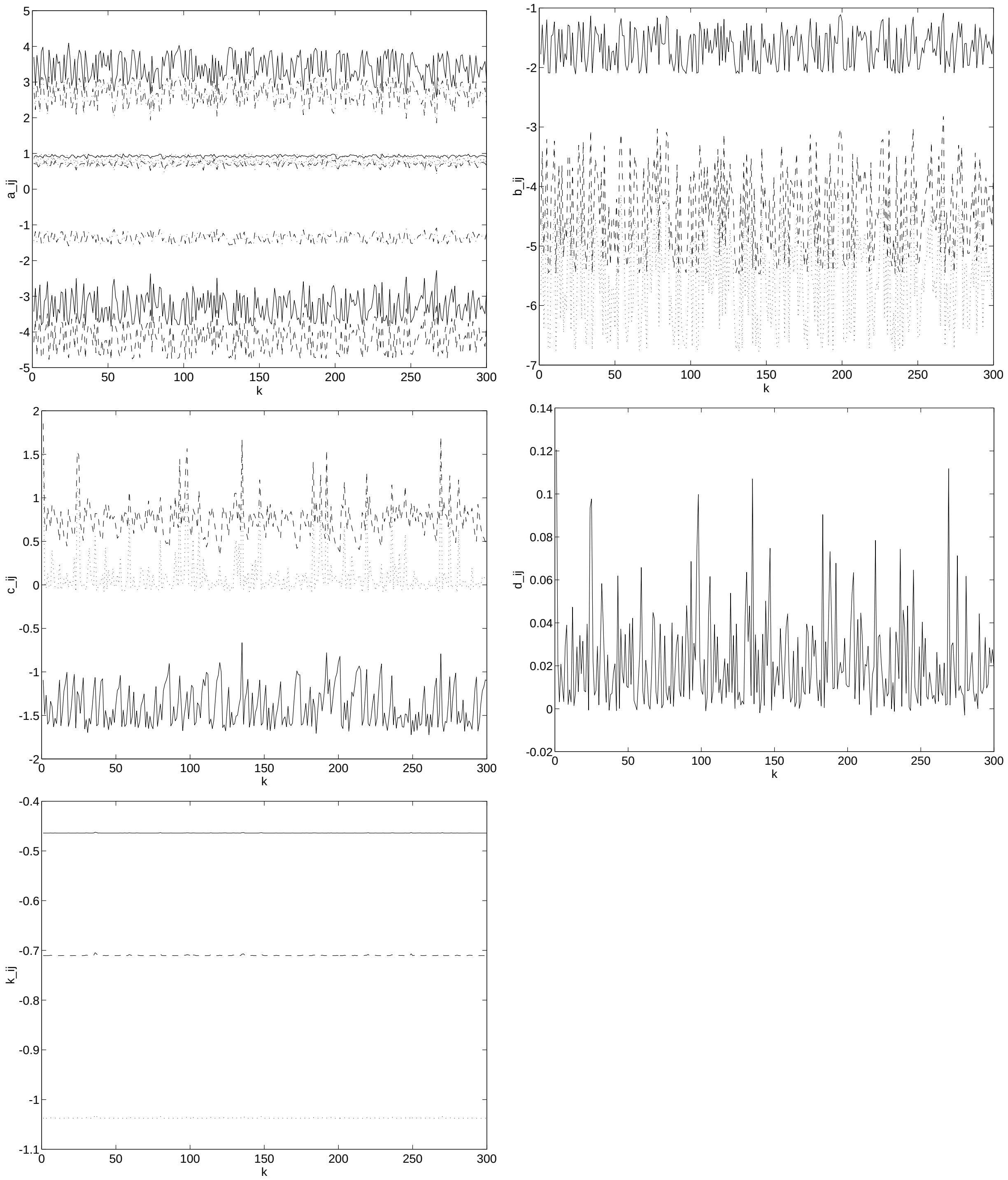
a_ij
5
4
3
2
1
0
-1
-2
-3
-4
-5
0
50
100
150
200
250
300
k
b_ij
-1
-2
-3
-4
-5
-6
-7
0
50
100
150
200
250
300
k
c_ij
2
1.5
1
0.5
0
-0.5
-1
-1.5
-2
0
50
100
150
200
250
300
k
d_ij
0.14
0.12
0.1
0.08
0.06
0.04
0.02
0
-0.02
0
50
100
150
200
250
300
k
k_ij
-0.4
-0.5
-0.6
-0.7
-0.8
-0.9
-1
-1.1
0
50
100
150
200
250
300
k

Figure 10: