System Identification: Resolving the Conflict Between Fault Detection and Control

Fahmida N. Chowdhury, Member, IEEE

Electrical and Computer Engineering

University of Louisiana at Lafayette, LA 70504-3890

Fax: (318) 482-6687, Email: fnchowdh@louisiana.edu

Abstract

In this paper, we explore the issue of fault tolerance vs. fault detection in engineering systems. We argue that system identification for controller design should be adaptive and therefore fault tolerant, while system identification for fault detection should be non-adaptive. We propose a combined strategy for maximizing robustness and safety of the system. The combined strategy, assuming that the system is nonlinear, includes a neural network for the non-adaptive modeling and a Kalman filter for the adaptive modeling.

Keywords: ARMA model, Neural network, Kalman filter, Time-varying, Fault detection, Parameter estimation

1 Introduction

System identification can be described as the technique of building a model of an unknown dynamic system (plant) using experimental input-output data. The field has enormous practical importance, is vigorously active, and a rich and extensive literature is available. In this paper, we do not attempt to give a comprehensive list: we cite only the most relevant ones (Leontaritis, et al, 1985; Mosca, 1995). The motivation behind trying to "identify" a system could be either of the following: 1) Understand the physical nature of the system, that is, build a model that has physical meaning, 2) Build a black-box model in order to design a controller, 3) Build a model in order to generate residuals that can be used as signatures of normal/abnormal behavior of the system. The first one of these goals is beyond the scope of this paper. We focus on the second and third goals.

1.1 Fault Tolerance vs. Fault Detection

Despite the large literature on system identification, little attention has been paid to the fundamental conflict between system identification geared toward designing adaptive controllers and that geared toward fault detection. The goal of this paper is to point out that conflict, and to propose a closed-loop configuration in which the conflict is resolved.

In adaptive controller desin, we want the system identifier to be able to track parameter variations quickly so that the controller can be made robust to them. If this is done successfully, then we expect small residuals despite changes inside the plant. This can ap-

propriately be termed "fault-tolerance". On the other hand, for fault-detection we want the identifier to **not** track the changing parameters, thus generating large residuals that are used in detecting system abnormalities. This is the fundamental conflict between robustness and fault-detection. In practical engineering systems, the best approach would be to combine the two aspects: (i) robustness to parameter variations and (ii) sensitivity to faults. This would require a parallel combination of two different types of system identification. Such a combination, a unified scheme for control and monitoring of engineering plants, is presented in this paper.

1.2 Robustness and Fault-detection: A Combined Strategy

To achieve a high degree of reliability, some researchers have proposed that "supervision and coordination" loops be implemented in the conventional parameter-adaptive control systems. For example, some authors (Knapp and Isermann, 1990) propose to add a step called "Detection of a process model mismatch" to the parameter estimation and controller design steps of the adaptive control scheme. However, there is only one parameter estimation block in this scheme, which is used both for controller design and for detecting the "model mismatch". In this paper, we argue that *one parameter estimator should not be expected to handle both tasks successfully, since the requirements are fundamentally conflicting.* To achieve the best possible combination of robustness and safety, we should use two parallel models of the system: one to produce adaptive parameter estimates, and the other to produce residuals that are sensitive to system faults. The proposed scheme is shown in Figure 1. As long as the parameter variations are small, the controller should be able to "absorb" the fluctuations;

but when there is some drastic change in the system, the fault-detection loop should produce an alarm so that appropriate action can be taken by the plant operators. During this "alarm" phase, the robust controller will still be performing satisfactorily, thus essentially "hiding" the fact that a fault has occurred. This could spell disaster for the plant operation, since eventually the controller will fail. Hence a separate fault-detection loop that uses an independent system identifier with fundamentally different properties.

Although we propose using two different system identifiers, both of them are based on the same type of input-output model. The difference between the two identifiers lies in their parameter estimation techniques. We demonstrate through some simulation experiments that in our dual-identification method, system faults can be detected well in advance of actual performance degradation.

2 Input-output Models for System Identification

Assume a linear, discrete-time state space model

$$x_{k+1} = Fx_k + Gu_k \tag{1}$$

$$y_k = Hx_k, \tag{2}$$

where F, G and H are constant matrices, y_k is the output, and u_k is the input sequence. It is known that the relationship between the output and input sequences of the above system

can be written as an AutoRegressive Moving Average (ARMA) model:

$$y_k = \sum_{i=1}^{n} a(i) y_{k-i} + \sum_{i=1}^{n} b(i) u_{k-i}, \tag{3}$$

where $a(i)$ and $b(i)$ are coefficients that can be computed from the knowledge of matrices F, G, H. If the output and input sequences are known (can be measured), then the coefficients $a(i)$ and $b(i)$ can be estimated by many different techniques.

The input-output relationship when the output measurements are corrupted by noise is given by:

$$y_k = \sum_{i=1}^{n} a(i) y_{k-i} + \sum_{i=1}^{n} b(i) u_{k-i} + \sum_{i=1}^{n} c(i) e_{k-i} + e_k, \tag{4}$$

which has the structure of the ARMAX model. In system identification, F, G, H are not known; the ARMAX coefficients can be estimated from the measured values of y_k, u_k and e_k. But the true observation noise terms e_k are not known and cannot be measured; so we replace them with the residuals obtained from the previous steps of estimation, as in (Mosca, 1995). This modified version still represents the original dynamic system: we simply estimate a different set of coefficients $\{d(i)\}$ instead of the set $\{c(i)\}$. Thus, the ARMAX model that can be implemented in real time is the following:

$$y_k = \sum_{i=1}^{n} a(i) y_{k-i} + \sum_{i=1}^{n} b(i) u_{k-i} + \sum_{i=1}^{n} d(i) r_{k-i} + e_k, \tag{5}$$

where r_k are the residuals of the coefficient estimation process up to the time instant k.

The input-output models described above can represent linear deterministic or stochastic plants. In addition, we show below that the introduction of a slowly time-varying (random

walk) model of the parameter vector allows a class of nonlinear plants to be represented by locally linear models. Thus, these recursive models are suitable for input-output representation of arbitrary unknown plants.

2.1 Nonlinear Systems: Time-varying ARMA Approach

For nonlinear systems, the conjecture is that at any discrete time step, there exists a linear system that can match the input-output data of the nonlinear plant. This gives rise to a family of ARMA models, each with its own set of coefficients:

$$y_k = \sum_{i=1}^{n} a_k(i) y_{k-i} + \sum_{i=1}^{n} b_k(i) u_{k-i}. \tag{6}$$

The above "time-varying ARMA" (Ben Mrad, 1998; Chowdhury, 2000) model will be called TVARMA in this paper.

2.2 Nonlinear Systems: Neural Network Approach

In addition to ARMA and TVARMA, there is also the NARMA (nonlinear ARMA) model, estimated by time-delay neural networks (Narendra, 1990; Werbos, 1992). It is well known that an arbitrary function $f(x) \in C^k(S)$ can be written as

$$f(x) = C^T \sigma_h(W^T x) + \epsilon(k) \tag{7}$$

wheres $\epsilon(k)$ is the function reconstruction error, and W and C are weighting matrices. If we interpret eq. 7 as a representation of a neural network with one hidden layer, then C would be the weights between the hidden layer and the output layer, and W would be the weights

between the input layer and the hidden layer. The function σ_h indicates the nonlinear activation function used in the hidden layer neurons. The activation functions are maps from real numbers onto the closed interval [0, 1], and are measurable, bounded, and nondecreasing. In the context of function approximation we will assume that the output neuron is a linear combiner.

A generic nonlinear version of the ARMA model is obtained naturally from the formulation and structure of the time-delay neural network. Consider a three-layer feedforward net with an input vector p_k that consists of past outputs $\{y_{k-i}\}$ and inputs $\{u_{k-i}\}$ of the dynamic system:

$$p_k = [y_{k-i}, \ldots, u_{k-i}], \tag{8}$$

where $i = 1, ..., n$ are indices of the past data points. Then, the scalar output of the net is a nested function of the input vector p_k:

$$\hat{y}_k^{nn} = \sum_{i=1}^{n} c_i q_i = \sum_{i=1}^{n} c_i \sigma_h (\sum_{j=1}^{m} w_{ij} p_{k,j}), \tag{9}$$

where q_i are the outputs of the hidden layer neurons, c_i are the weights of the output neuron, w_{ij} are the weights of the hidden layer neurons, m is the length of the input vector, and $p_{k,j}$ is the $j-th$ component of the input vector at time instant k. The model is estimated by training the neural network, that is, finding appropriate values for the weights C and W so that the function reconstruction error is minimized.

3 System Identification for Control: Kalman Filter

In the control loop, we use a Kalman filter as the estimator of the input-output model (Chowdhury, 2000). Briefly, to build a Kalman filter to estimate the coefficients $a(i), b(i)$ of Eq. 6, define the "state vector" as:

$$\theta_k = [a(1), ..., a(n), b(1), ..., b(n)]^T,$$

where T indicates transpose, and the set $\theta_k \in \Theta \subset R^{2n}$ contains the model coefficients to be estimated. In the proposed approach, the evolution of the state vector θ_k is assumed to follow a random walk model:

$$\theta_{k+1} = \theta_k + w_k. \tag{10}$$

The observed output of this *fictitious* system is given by:

$$y_k = \Phi_k^T \theta_k + e_k, \tag{11}$$

where

$$\Phi_k = [y_{k-1}, ..., y_{k-n}, u_{k-1}, ..., u_{k-n}]^T. \tag{12}$$

In Eq. (10), w_k (the random walk term, acting as process noise in the Kalman filter) and e_k are zero-mean noise processes with known or assumed covariances. Note that $\Phi_k \in R^{2n}$ is the measurable regressor in the correspondinging regression-like formulation. Together, Eq. (10) and Eq. (11) represent a *fictitious* discrete-time linear system with a time-varying observation matrix Φ_k, a system matrix that is Identity, a process noise w_k, and an observation noise e_k. The process noise is not a real noise - it is a design parameter that makes the estimator

adaptive. This allows us to build a Kalman filter to estimate θ_k in real time, and generate the required residual:

$$r_k = y_k - \Phi_k^T \hat{\theta}_k.$$

The residuals are used to update the parameter estimates; these parameters are then used in the adaptive controller. In this paper, we do not discuss the controller design issue, except to mention that at each time step, the controller parameters are functions of the estimated plant parameters. Moreover, if the plant parameters happen to be slowly time-varying, the adaptive estimator described above is well-equipped to handle that, as long as the parameter variations do not make the plant unstable. In our adaptive controller, the updated plant papameter estimates from the Kalman filter are used at each time-step to update the control signal. The Kalman filter continuously updates the parameter estimates using closed-loop data.

4 System Identification for Fault Detection: Neural Network

For the purpose of fault detection, we need an estimator that is **not** adaptive. That is, once we have settled on a model and have estimated its parameters, we should define any deviation from it as a "fault", and our goal would be to detect these faults as quickly as possible. Most off-line system identification methods would be well-suited to this purpose. In this paper, since we wish to be able to handle a large class of plants including some possibly non-linear ones, we use a neural network to estimate a NARMA model of the plant. In current literature on neural network based system identification, the typical goal is to accomplish

some control task; for this reason, many researchers are trying to develop adaptive training rules so that neural networks could estimate system parameters adaptively. However, it is the contention of this author that the already-existing (non-adaptive) neural net methods are ideally suitable for fault detection.

A well-established method of detecting faults is to use statistical hypothesis tests on the residuals of the system. Denoting the residual generated by the neural net model as r_k^{nn}, we write:

$$r_k^{nn} = y_k - \hat{y}_k^{nn}. \tag{13}$$

Ideally, the residual would be normalized so as to yield a zero-mean, unit-variance random variable under "normal" operating conditions. When linear models are used for system identification, we can get an estimate of the covariance of the estimation error, from which we can compute an estimate of the covariance of the residual. This covariance is then used to normalize the residual. However, one of the deficiencies at the current stage of development in neural net modeling is that it does not provide an estimate of the covariance; therefore, we have to use the unnormalized residual. Under these circumstances, we establish the thresholds for the failure of the test of hypothesis by observing the range of residual values during "fault-free" operation.

5 A Multi-loop System for Safe Operation of Engineering Systems

The overall scheme proposed here (see Fig. 1) is a practical way of ensuring safe operation of engineering systems. Small fluctuations of system parameters are absorbed by the adaptive estimator (based on the Kalman filter), while large, sudden changes (faults) are detected by the fault-detector (based on a neural network). *In case of a detected fault, an alarm is to be sounded even though the controller may still be working well; this is the most useful aspect of the proposed scheme.* This allows us to detect faults before any significant degradation occurs in the performance of the overall system.

In the system described in this paper, three different sequences of residuals are generated. First, the one produced by the Kalman filter that is estimating the system parameters adaptively ($r_k = y_k - \Phi_k^T \hat{\theta}_k$); second, by the neural network monitoring the plant ($r_k^{nn} = y_k - \hat{y}_k^{nn}$). There is also the "performance residual", that is, the difference between the plant output and the desired response ($r_k^d = y_k^d - y_k$). The Kalman filter residuals are used for adaptive parameter estimates; these estimates are used by the controller to update the control signal. The neural net residuals are used for fault detection. Depending on the result of the test of hypothesis, the Kalman filter covariance matrix may be adjusted, which results in a changed rate of parameter adaptation. In this way, all the blocks in the system interact with one another. The two system identifiers complement each other for fault tolerance and fault detection.

6 Simulation Experiments

In order to demonstrate the closed-loop operation of the proposed scheme, we assume a nonlinear plant given by:

$$y_k = \frac{y_{k-1}y_{k-2}u_{k-2}(y_{k-3}-1)+u_{k-1}}{1+y_{k-3}^2+y_{k-2}^2}. \tag{14}$$

The following assumption are used:

1. The plant is initially unknown, all inputs and outputs are measurable, and the measurement noise is small ($\sigma = 0.001$).

2. The control task is to produce a desired (target) output. In this example, the target output is a sinusoid, y_k^d.

3. A neural network model is built off-line, using random inputs to the open-loop plant.

4. The controller is built on-line, using closed-loop input-output data in a time-varying ARMA model estimated by a Kalman filter. The initial guess for the ARMA parameters is zero.

Three scenarios are simulated: a small fault, a large fault, and a time-varying fault. Our goal is to demonstrate that the combination of two parallel system identification schemes is an excellent idea because it produces a much safer closed-loop system of operation. In the first experiment, at time step $k = 220$, a "permanent but small fault" is introduced, and the output is given by:

$$y_k = \frac{2.0y_{k-1}y_{k-2}u_{k-2}(y_{k-3}-1)+u_{k-1}}{1+y_{k-3}^2+y_{k-2}^2}. \tag{15}$$

In this case, the controller is able to "absorb" the fault, although the neural network residuals indicate some amount of model mismatch. Note the near-perfect performance of the adaptive controller, and the very small "performance residuals" in Fig. 2 and Fig. 3.

In the next experiment, a large fault is simulated: beginning at time-step $k = 220$, the plant output is given by:

$$y_k = 1.5 y_{k-1} y_{k-2} u_{k-2} (y_{k-3} - 1) + u_{k-1}. \tag{16}$$

In this case, the neural network model and its residuals clearly show a big mismatch, and this fault can be detected very easily, although the adaptive controller is so robust that the performance returns to normal after a short period of time. If the neural network-based fault detection were not present, we could have come to the entirely wrong conclusion that there was only a temporary fault. This scenario is an example of a combination of fault-tolerance and fault detection occurring at the same time. The results are shown in Fig. 4 and Fig. 5.

In the last experiment, we introduce a time-varying (exponentially increasing) fault from the beginning of the plant operation. For this, the plant equation is assumed to be:

$$y_k = \frac{e^{k/150} y_{k-1} y_{k-2} u_{k-2} (y_{k-3} - 1) + u_{k-1}}{1 + y_{k-3}^2 + y_{k-2}^2}. \tag{17}$$

In this case, the adaptive controller is able to handle the fault only upto a certain time: after that, a catastrophic failure occurs. In our simulation experiment, we let the plant reach this point, and then shut it down. The plots in Fig. 6 and Fig. 7 clearly show this situation.

Note that the neural network output is showing a *gradually growing mismatch* from about time step $k = 100$, although the performance of the controller or the Kalman filter based system identifier does not indicate that anything might be wrong until much later; in fact, when the controller fails, it is a sudden and irreversible damage in the system, since at this time the plant output "explodes". This situation could be prevented by using the neural network residuals as the fault-indicator; the fault could be detected (and perhaps fixed) without having to allow performance degradation or failure.

7 Conclusion

In this paper, we have shown the effectiveness of a dual scheme for system identification, control, and fault detection. The system identification based on a time-varying ARMA model, estimated by a Kalman filter, is used for adaptive controller design. This identifier is highly adaptive, and therefore the controller is able to drive the plant to produce the desired output. However, this system identifier is not suitable for generating residuals for fault detection. A neural network model is used for the purpose of fault detection. We demonstrate that by using this dual scheme, a combination of robustness and safety can be achieved.

Acknowledgment

This work was supported by US National Science Foundation Grant No. ECS-9753084. An earlier version of was presented at the IFAC SAFEPROCESS 2000 Symposium at Budapest,

Hungary, in June 2000. The author thanks Mr. Joel Derouen, an undergraduate senior at the University of Louisiana Lafayette, for carrying out computer simulations for this work.

REFERENCES

Ben Mrad, R., S.D. Fassois and J. A. Levitt (1998). A Polynomial Algebraic Method for Non-Stationary TARMA Signal Analysis. *Signal Processing*, **Vol. 65**, Part I, 1-19 and Part II, 21-38

Chowdhury, F.N. (2000). Input-output Modeling of Nonlinear Systems with Time-varying Linear Models. *IEEE Trans. Automat. Contr.*, Vol. 45, No. 7, 1355-1358

Knapp, T and R. Isermann (1990). Supervision and Coordination of Parameter-Adaptive Controllers. *Proc. Automat. Contr. Conf.*, 1632-1637

Leontaritis, I.J and S.A. Billings (1985). Input-output Parametric Models for Non-linear Systems, Part II: Stochastic Non-linear Systems. *Int. Journal of Control*, **Vol. 41**, No. 2, 329-344

Mosca, E. (1995). *Optimal, Predictive, and Adaptive Control*, Ed. Thomas Kailath, Prentice Hall Information and System Sciences Series

Narendra, K.S. and K. Parthasarathy (1990). Identification and Control of Dynamic Systems using Neural Networks. *IEEE Trans. Neural Networks*, **Vol. 1**, No. 1, 4-27

Werbos, P.J., T. McAvoy, and T. Su (1992). Neural Networks, System Identification, and

Control in the Chemical Process Industries. In: *Handbook of Intelligent Control*, Ed. White and Sofge, Van Nostrand Reinhold, 283-356, New York

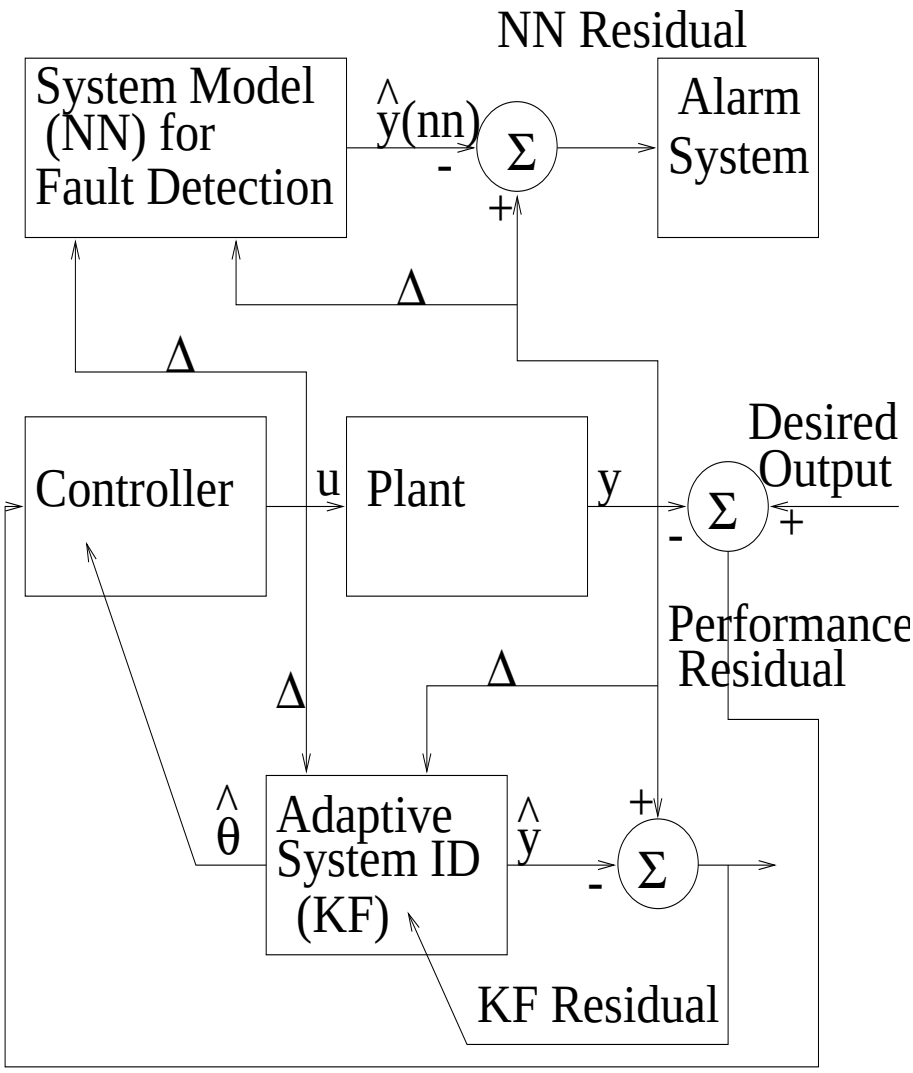

Figure 1: Simultaneous Control and Fault Detection: Δ denotes the backward time-shift operator.

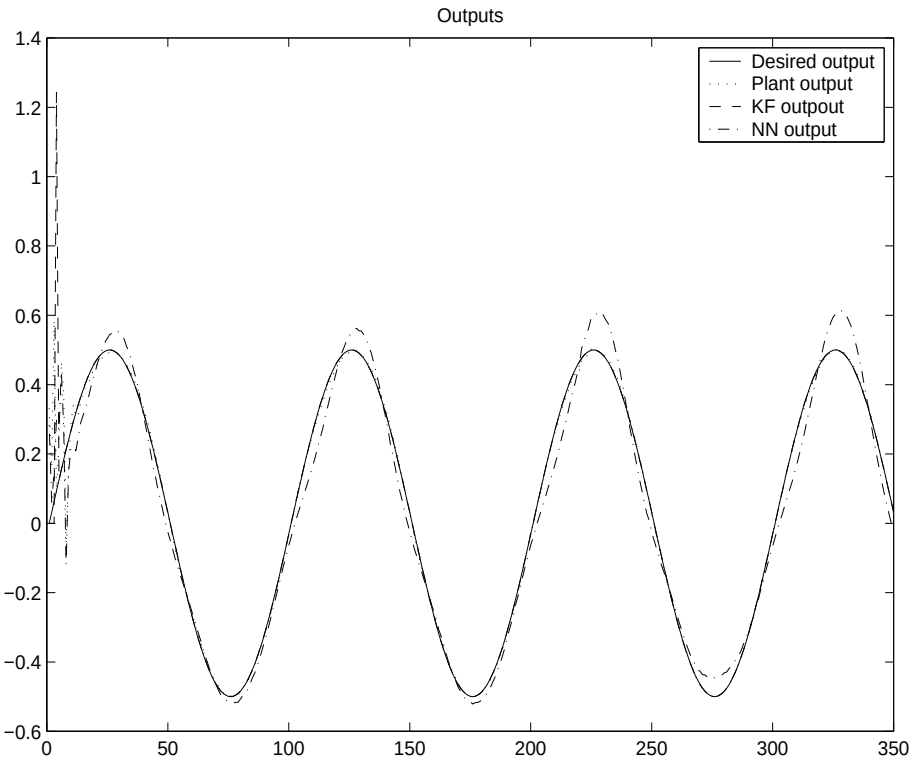

Figure 2: Simultaneous Control and Fault Detection: Small Fault, Outputs.

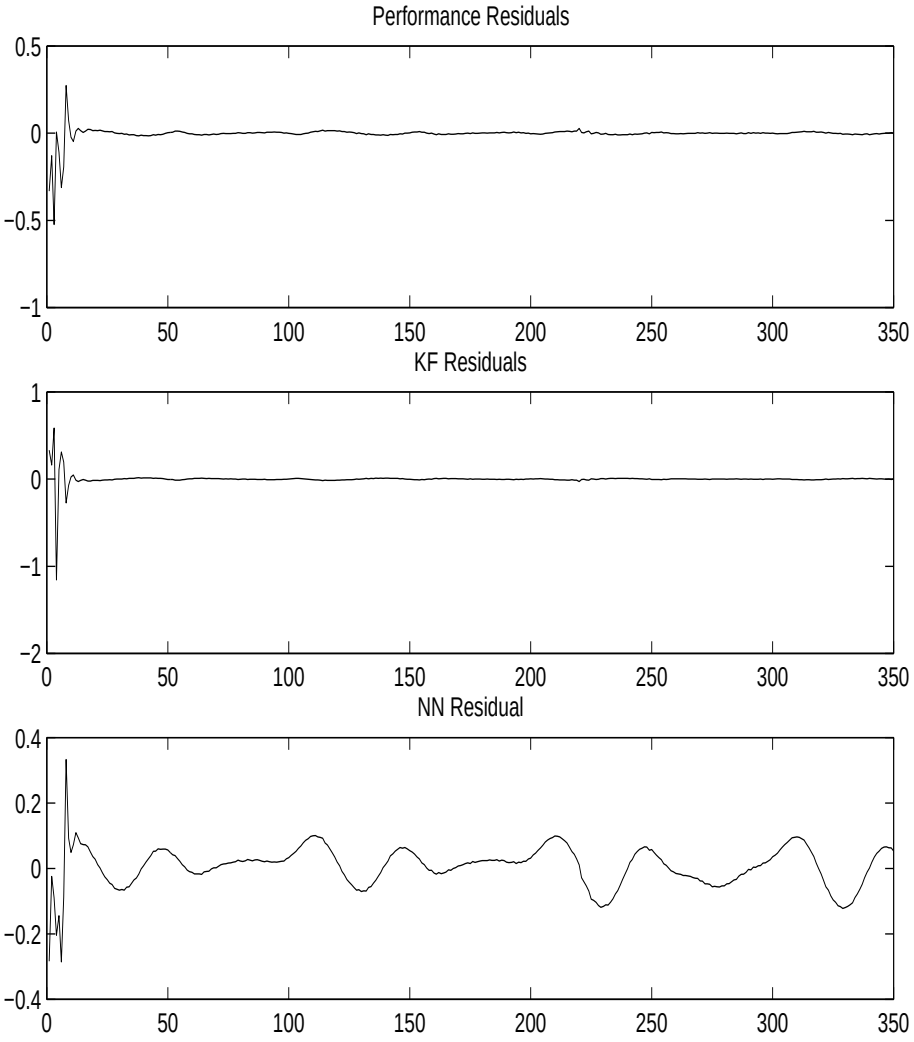

Figure 3: Simultaneous Control and Fault Detection: Small Fault, Residuals.

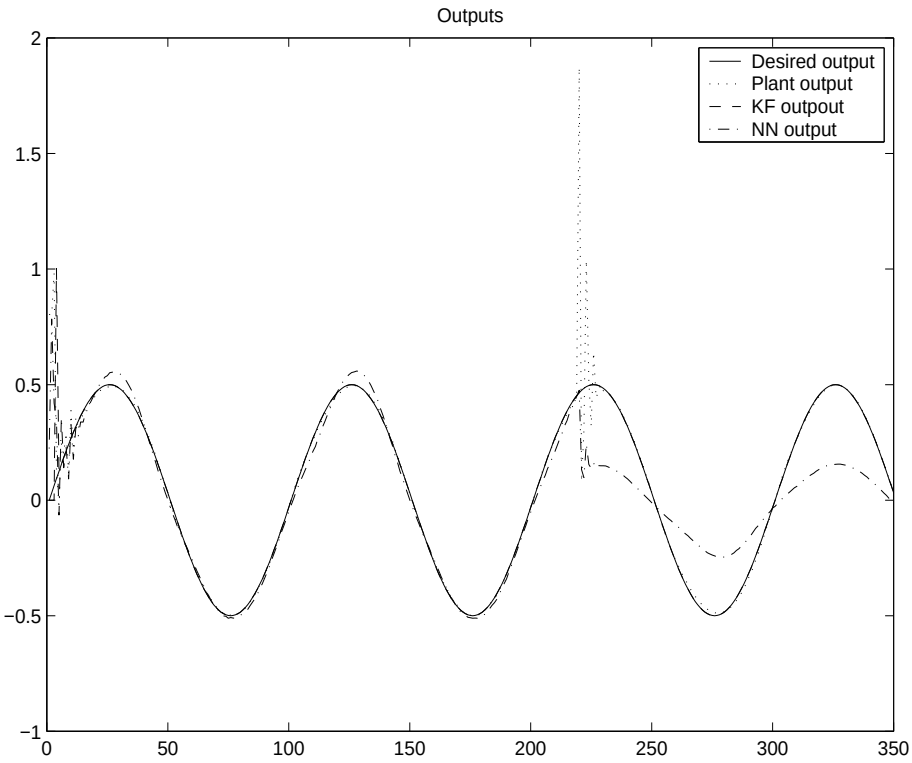

Figure 4: Simultaneous Control and Fault Detection: Large Fault, Outputs.

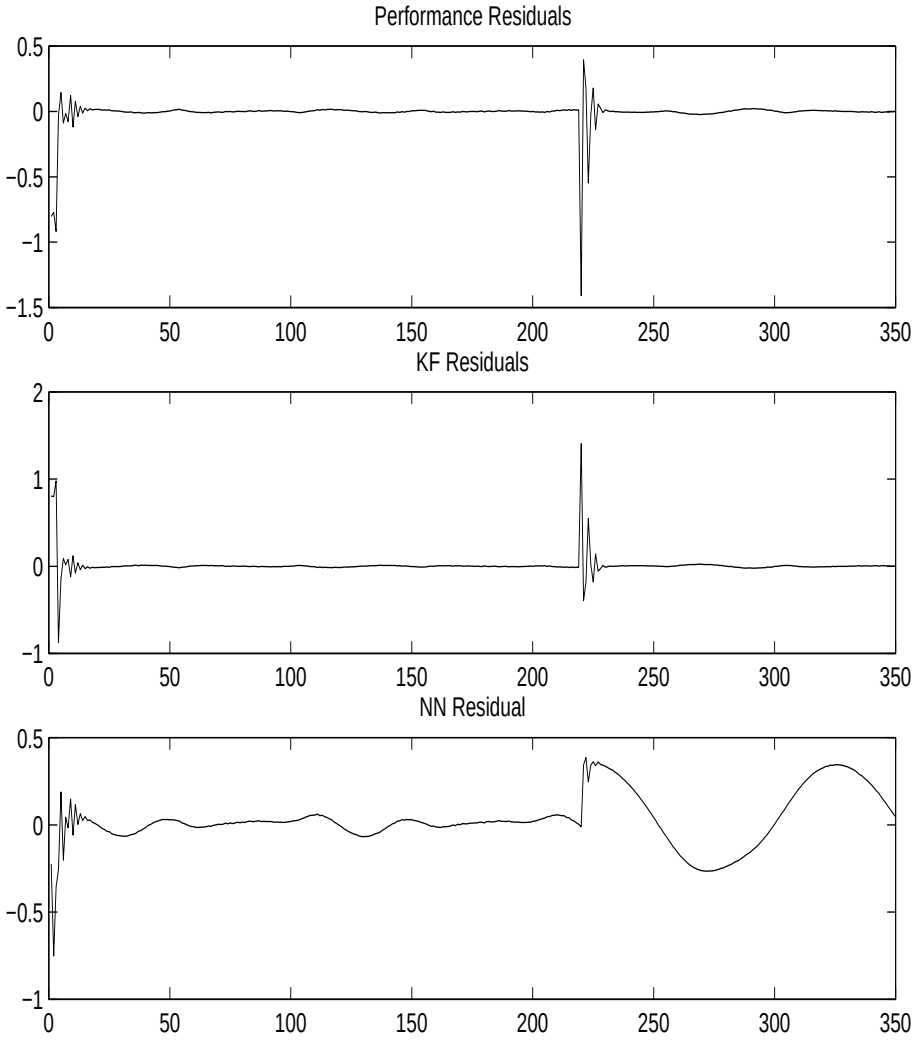

Figure 5: Simultaneous Control and Fault Detection: Large Fault, Residuals.

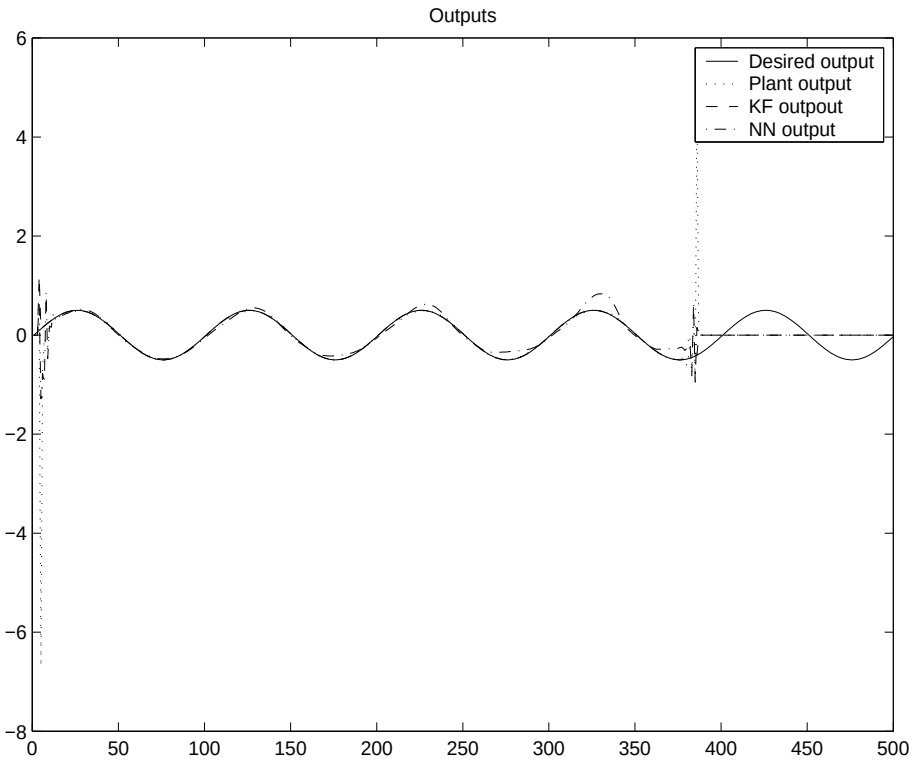

Figure 6: Simultaneous Control and Fault Detection: Time-varying Fault, Outputs.

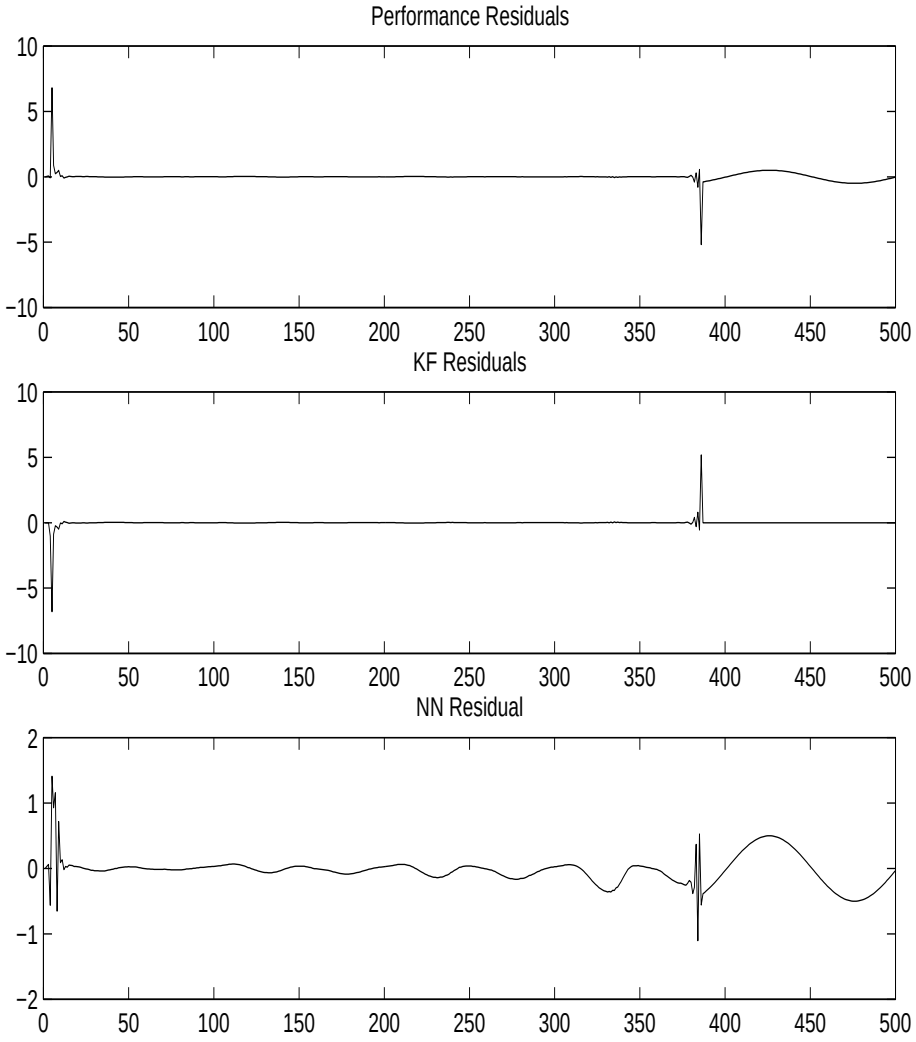

Figure 7: Simultaneous Control and Fault Detection: Time-varying Fault, Residuals.

Figure Captions

1. Simultaneous Control and Fault Detection; Δ denotes the backward time-shift operator

2. Simultaneous Control and Fault Detection: Small Fault, Outputs

3. Simultaneous Control and Fault Detection: Small Fault, Residuals

4. Simultaneous Control and Fault Detection: Large Fault, Outputs

5. Simultaneous Control and Fault Detection: Large Fault, Residuals

6. Simultaneous Control and Fault Detection: Time-varying Fault, Outputs

7. Simultaneous Control and Fault Detection: Time-varying Fault, Residuals