ENG-GENES: A NEW GENETIC MODELLING APPROACH FOR NONLINEAR DYNAMIC SYSTEMS

Kang Li

School of Electrical & Electronic Engineering, Queen's University Belfast
Ashby Building, Stranmillis Road, Belfast BT9 5AH, UK

Abstract: A novel neural modelling method, namely 'eng-genes', is proposed for complex nonlinear dynamic engineering systems. This method performs system modelling by first establishing the 'eng-genes' – some fundamental engineering functions from 'a priori' engineering knowledge, which are then constructed and coded into appropriate chromosome representations. Given a suitable fitness function, using evolutionary approaches such as the genetic algorithms, a population of chromosomes evolves for a certain number of generations to finally produce a neural model best-fitting the system data. In this paper, the eng-genes genetic modelling framework is discussed in detail and it is then applied to model two nonlinear engineering systems to confirm the effectiveness. *Copyright @2005 IFAC*

Keywords: nonlinear systems, modelling, genetic algorithms, neural networks, chemical process, power plant.

1. INTRODUCTION

Nonlinearity widely exists in fundamental engineering equations, for example, the Bernoulli equation for non-turbulent, compressible, and barotropic fluid undergoing steady motion; Navier-Stokes equations for incompressible, Newtonian fluids in both laminar and turbulent flow; and the Arrhenius equation for chemical reactions. For complex dynamic engineering systems exhibiting non-linear behaviour, to obtain a simple yet meaningful model for real-time system operation and control is a pressing problem. Ideally, such model can be easily built and updated with desired generalisation performance, and it can also retain physically interpretable information to help control engineers and human operators to gain a deep knowledge about the system under control.

Current system modelling techniques more or less fall into three general categories, namely 'white-box', 'black-box' and 'grey-box' methods, though the boundaries between different categories are getting 'grey'. For engineering systems, transparent modelling often results in a set of partial differential equations (PDEs) and ordinary differential equations (ODEs). To solve these equations may require the use of numerical tools, i.e. the computational fluid dynamics (CFD) method, which sometimes is quite demanding in computing and is time-consuming for complex engineering systems, therefore not suitable for the real-time operation and control, other than for plant design (Thompson and Li, 2003).

Neural models and data-driven regression models may require little physical insight, and the model quality then largely relies on the training data. In practice, the experimental data or field operational data is often limited, and the resultant models can be poor in interpreting unseen data/phenomena, and they may also lack transparency. Grey-box approach is used when physical knowledge is partially incorporated into the modelling (Bohlin, 1994; Pearson and Pottmann, 2000; Sohlberg, 2003; Tulleken, 1993). Depending on how, and how much, ‘a priori’ engineering knowledge can be used, there exist different grey-box methods. There are two general categories among which one method, based on the regression model structure, uses ‘a priori’ knowledge to obtain a set of constraints on model parameters or variables, such as constraints on the static gain in a linear model. The others start with a model originating from the mathematical relations, which describe the behaviour of the system. In the later approach, physical modelling and system identification form two interacting paths, and most methods would assume that the model structure is known 'a priori', and the major modelling task is then to identify unknown parameters and unmodelled dynamics. These two grey-box approaches have limitations when the engineering systems are either too complex to derive a simplified model or that our physical knowledge about the process is incomplete.

The eng-genes method proposed in this paper takes a different approach in incorporating system physical knowledge into a neural network structure. It starts system modelling by establishing some fundamental static engineering functions from 'a priori' engineering knowledge. The fundamental functions reflect the nonlinearity uniquely identifiable from the fundamental physical or chemical laws of engineering system and may be regarded by analogous to the genes in the human neural system. It is then natural to construct and code these functions and their associated structure into appropriate chromosome representations. Given a suitable fitness function, using evolutionary approach such as the genetic algorithm (Goldberg, 1989), a population or multiple populations of chromosomes will evolve for a certain number of generations to finally produce a neural model best-fitting the system.

This paper is organised as follows. In section 2, the ‘eng-genes’ method and its framework is discussed in detail. Then two case studies are presented in section 3 to confirm the effectiveness. Section 4 is the concluding remarks.

2. THE ‘ENG-GENES’ METHOD

The solutions of a set of partial and/or ordinary differential equations for an engineering system are some multivariate continuous real-valued functions. To obtain these multivariate functions however can be difficult and computationally intensive. Kolmogorov proved that these multivariate functions could be represented by superposition and composition of continuous functions of only one variable- for each $n \geq 2$ (n is the dimension of the variable space) there exist continuous functions φ_q - $[0,1] \rightarrow \Re$, $q = 0, 1, ..., 2n$ and constants $\lambda_p \in \Re$, $p=1,...,n,$ such that the following holds true: for each continuous function $f: [0,1]^n \rightarrow \Re$, there exists a continuous function $g: [0,1] \rightarrow \Re$, such that

$$f(x_1,...,x_n) = \sum_{q=0}^{2n} g\left(\sum_{p=1}^{n} \lambda_p \varphi_q(x_p) \right) \qquad (1)$$

where φ_q and λ_p are independent of f, but g is dependent on f.

Although the Kolmogorov superposition theorem does not intend to solve multivariate engineering equations, it indeed has suggested the neural network structure where different system independent basis functions can be used to approximate the original system. However, despite the fact that neural networks provide a universal structure for modelling nonlinear functions, conventional neural models carry little physical knowledge about the system.

To incorporate ‘a priori’ engineering knowledge into modelling while taking advantage of the neural structure, a number of researches have been carried out. Most methods employ a hybrid modelling or semi-physical modelling method (Psichogios, and Ungar, 1992), i.e. the clearly defined physical equations remain in the model, while neural networks are used to approximate unknown parameters or unknown dynamics, etc. This method has limitations when the engineering system is too complex to derive a simplified model. Literature survey shows little has been done so far to incorporate the physical knowledge into the neural structure, i.e. to select activation functions for the neural model from 'a priori' engineering knowledge to approximate the continuous function g in equation (1). Although each type of neural networks, e.g. MLP, RBF, Wavelet, etc is claimed to be successful in some application domains, little has been done so far to study why one paradigm has worked well in some applications, but failed in others.

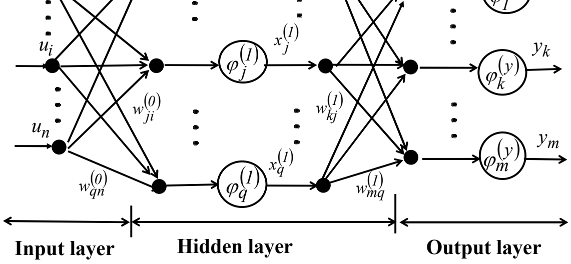

Fig. 1. ‘Eng-genes’ network with one hidden layer

In the proposed eng-genes method, some fundamental static functions – so called ‘eng-genes’, abbreviation of ‘engineering genes’, are extracted from 'a priori' knowledge. The ‘a priori’ knowledge can be first principle laws specified in ODEs or PDEs for the system, or in more general sense, some fundamental chemical or physical laws for a wide range of problems. These fundamental functions are then used as the activation functions in a multiplayer neural structure.

A mathematical formulation of an ‘eng-genes’ model for a MISO (multiple input single output) system is given by

$$\left.\begin{aligned} x_i^{(k)}(t) &= \phi_i^{(k)}\left(w_{0\sim i}^{(k)} + \sum_{j=1}^{n_k} w_{j\sim i}^{(k)} x_j^{(k-1)}(t) \right) \\ y(t) &= \phi^{(y)}\left(w_0 + \sum_{i=1}^{n_h} w_i x_i^{(h)}(t) \right) \\ k &= 1,\cdots,h \end{aligned}\right\} \quad (2)$$

where $[x_1^{(0)},\cdots,x_{n_0}^{(0)}]$ denotes the inputs to the ‘eng-genes’ network; n_k, $k=1,\cdots,h$ denotes the number of neurons in the *k*’th hidden layer, *h* is the number of hidden layers; $x_i^{(k)}$, $i=1,\cdots,n_k$ is the output of the *i*’th neuron in the *k*’th hidden layer; w_i, $i=1,\cdots,n_h$ is the output weight from the *i*’th neuron of the last (the *h*’th) hidden layer to the output and w_0 is the output bias; $w_{j\sim i}^{(k)}$, $i=1,\cdots,n_k$, $j=1,\cdots,n_{k-1}$, are the weights from *j*’th neuron of the (*k*-1)’th hidden layer to *i*’th neuron of the *k*’th hidden layer and $w_{0\sim i}^{(k)}$ is the bias; $\phi_i^{(k)}$, $i=1,\cdots,n_k$, denote the eng-gene for the *i*’th neuron of the *k*’th hidden layer as its activation function; $\phi^{(y)}$ is the eng-gene for the output neuron.

Fig.1 illustrates such an eng-genes neural structure with only hidden layer in the network, where $\varphi_i^{(1)}, j=1,...,q$, $\varphi_i^{(y)}, j=1,...,m$ are the ‘eng-genes’ in the first hidden layer or the output layer.

2.1 Extraction of ‘eng-genes’

For engineering systems where ‘*a priori*’ fundamental knowledge or first-principal laws are available, it is possible to establish some fundamental engineering functions or ‘eng-genes’. These fundamental functions are appropriate static non-linear functions that are uniquely identifiable from the ‘a priori’ engineering knowledge, and they can be exponential, power, trigonometric, or rational functions, etc. However, the exact function form depends on the application, or more specifically the mathematical equations describing the fundamental laws governing the behaviour of the system.

It is interesting to notice that for these fundamental engineering functions, the manipulation on them by operators like superposition and subtraction, multiplication and division, or differentiation and integration, may produce functions of imilar type. These properties allow one to separate and to refine these functions. A simplified and realistic model can then be produced to represent the original system through appropriate composition and combination. This is analogous to producing a simplified equivalent circuit with fewer inductors, capacitors and resistors from a complex electrical circuit.

Since these fundamental functions or ‘eng-genes’ reflect physical reality in one way or another, they are useful in helping operators and control engineers to gain some physical insight into the system. For example, in modelling many chemical, biological or mechanical processes, $\varphi_1 = e^{(c_1/(x+b_1))}$ and $\varphi_2 = (x+b_2)^{c_2}$ and are the two types of ‘eng-genes’, where x is a system variable like mass flow of materials, chemical concentrations, or environmental temperature, and *b₁, b₂, c₁, c₂* are parameters. φ_1 is related to the Arrhenius behaviour in kinetics of chemical or biological processes. According to the well-known Arrhenius equation, the reaction rate *k* is expressed as $Ae^{(\text{-E}_\text{a}/RT)}$ where is the reaction rate, *A* is a constant, E_a is the activation energy, *R* is the universal gas constant, and *T* is the temperature. φ_2 appears in many industrial processes in relation to non-Arrhenius behaviour of the chemical or biological reactions, or to the thermal or fluid dynamics.

In case that the complete set of PDEs or ODEs for complex engineering systems is unavailable, it is always possible to examine a set of general chemical or physical laws applicable to the specific system. Therefore it is feasible to build an eng-genes library for engineering systems of different physical natures, and genetic algorithms can then be used to select appropriate eng-genes from the library and to optimise the neural model for the specific system under study.

2.2 Optimisation in eng-genes based neural modelling

Given a performance index, optimisation of the neural model (2) is a mixed integer non-linear hard problem taking into account the following issues:

Selection of neural inputs - In neural modelling of non-linear complex systems, the neural inputs may include any system input variable of interest and system outputs (for recurrent neural network) with time delays, and the number of candidate neural inputs can be extremely large. To select a subset of inputs is a combinational problem, and the selection process can be quite time-consuming.

Selection and optimisation of 'eng-genes' - For a specific engineering system, there may exist a set of 'eng-genes' but usually only a few are dominant and distinctive. To select which eng-genes being used in the neural model and to optimise parameters in these eng-genes are part of the whole optimisation problem.

Optimisation of the neural structure - Once the neural inputs and dominant 'eng-genes' are selected and optimised, the next step is to optimise the neural structure in (2), including the number of hidden layers and the number of hidden nodes in each layer.

Network training: Finally to identify the optimal weights and biases.

It is difficult to deal with the above optimisation issues effectively using conventional analytic tools. As adaptive stochastic optimisation tools, evolutionary algorithms such as genetic algorithms (GAs) have been shown effective for complex optimisation problems. Prototype software coded in C++ has been developed by the authors as an integrated optimisation platform for neural modelling, dealing with all the above optimisation issues. First-order or second-order training algorithms coded in Matlab programs for eng-genes networks have also been developed.

2.3 'Eng-genes' based neural modelling framework

The 'eng-genes' based neural modelling procedure can be summarised as follows together with Fig. 2:
Step 1 Establish fundamental mechanism of the engineering system based on first principle laws.
Step 2 From these 'a priori' knowledge to extract fundamental functions or 'eng-genes'.
Step 3 Data acquisition and pre-processing, generally though experimental tests or selection from real-time operational data from the plants.
Step 4 Optimisation of the 'eng-genes' based neural model using genetic algorithms, or for simple applications using LM method for neural training.
Step 5 Model validation.

3. CASE STUDIES

Case 1- Continuous Stirring Tank Reactor (CSTR)

Fig. 3 shows a schematic representation of a chemical system common to many chemical processing plants, known as a Continuously Stirred Tank Reactor (CSTR). Within a CSTR two chemicals are mixed and react to produce a product compound at a concentration $C_a(t)$, with the temperature of the mixture being $T(t)$.

The differential equations representing the CSTR reaction are shown as follows.

$$\dot{C}_a(t) = \frac{q}{v}(C_{ao} - C_a(t)) - k_o C_a(t) exp\left(-\frac{E}{R\cdot T(t)}\right) \quad (3)$$

$$\dot{T}(t) = \frac{q}{v}\cdot(T_o - T(t)) + k_1 \cdot C_a(t)\cdot exp\left(-\frac{E}{R\cdot T(t)}\right)$$

$$+ k_2 \cdot q_c(t)\cdot\left(1 - exp\left(-\frac{k_3}{q_c(t)}\right)\right)\cdot(T_{co} - T(t)) \quad (4)$$

where C_a is product concentration, C_{ao} is the inlet feed concentration, q the process flow-rate, T_o and T_{co} the inlet feed and coolant temperatures respectively. k_o, E/R, v, k_1, k_2 and k_3, are thermodynamic and chemical constants relating to this particular problem.

A MLP model and an eng-genes model were developed to model the above system. Firstly, the system (3) and (4) was simulated at a sampling period of 0.2 seconds, and subjected to an input consisting of uniformly distributed random values of the input $q_c(t)$ in the range [-10, 10] from the operating point. 3000 data points were are obtained, among which 120 samples were used for training, and the rest 2880 samples were used for validation. The statistics of product concentration C_a in the two data sets is illustrated in Table 1.

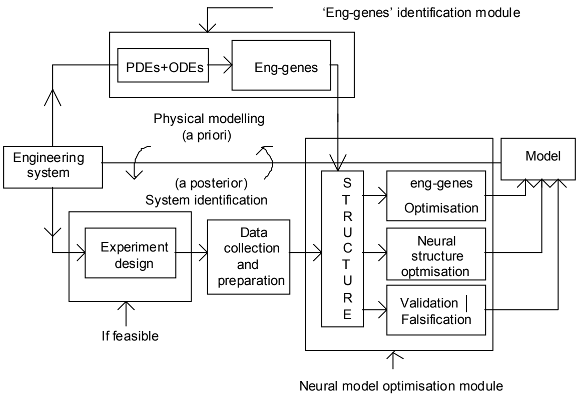

Fig. 2. 'Eng-genes' framework

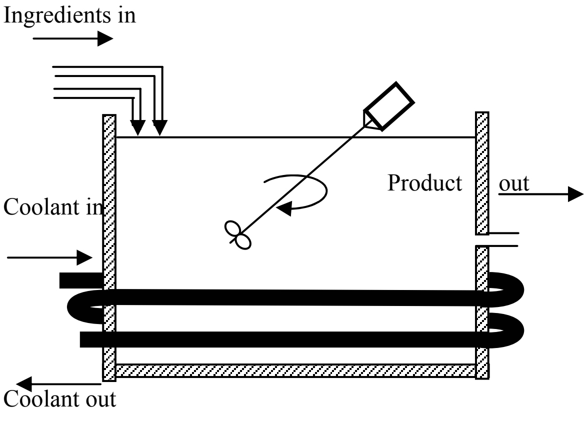

Fig. 3. Schematic representation of CSTR

Table 1 Statistics of C_a in the two data sets (mol/l)

	Min	Max	Mean	STD
Training	0.102	2.0	1.051	1.342
Validation	0.086	2.0	1.043	1.353

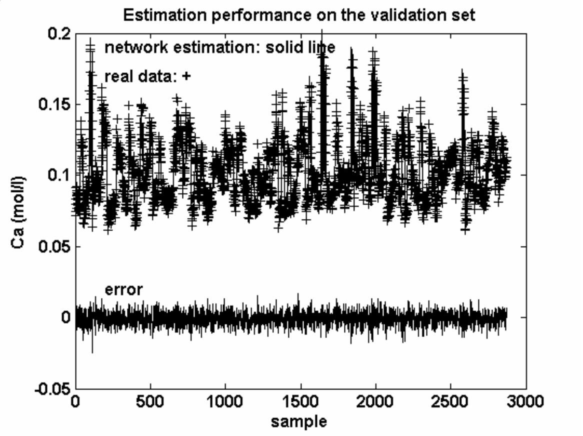

Fig.3. Long-term estimation performance of the eng-genes model on the validation data

- *MLP model* - A 5-3-1 multilayer perceptron (MLP) was first developed. The GA based optimisation platform was used to select the network inputs, to optimise the network structure as well as to identify the optimal network weights and biases. The population size in the genetic algorithms was 200. An elitist scheme was employed in the genetic optimisation and, in each generation, the 8 best chromosomes were retained. The long-term prediction performance of the MLP model over the two data sets when the network evolved for 100 generations and 150 generations is listed in table 2. The selected neural inputs are *$q_c(t\text{-}2)$, $q_c(t\text{-}3)$, $q_c(t\text{-}5)$, $C_a(t\text{-}1)$, $C_a(t\text{-}6)$.*

- *'Eng-genes' model*- According to (3) and (4), the nonlinearity originates from the Arrhenius equation, and therefore only one 'eng-genes' were extracted, that is $\varphi_1 = e^{(c_1/(x+b_1))}$ where x is the activation potential for the hidden neurons, b_1, c_1 are parameters to be optimised using genetic algorithms.

Similarly a 5-3-1 'eng-genes' model with the same neural inputs was developed and the same GA based optimisation platform with same GA operation parameters were used. Table 2 compares long-term prediction performance of the eng-genes model with that of the MLP model.

Table 2 Long-term prediction performance (RMSE) of the 'eng-genes' model and the MLP model

		100 gen	150 gen
MLP	Modelling	0.0066	0.0066
	Validation	0.0063	0.0063
Eng-genes	Modelling	0.0051	0.0049
	Validation	0.0054	0.0050

Fig. 3 illustrates the long-term prediction of the eng-genes model over the validation dataset when the model was evolved for 150 generations.

Case 2 - NOx emissions in a coal-fired power plant

Nitrogen oxides (NO_x) emissions from combustors and engines have many harmful effects on both the environment and human health. In coal-fired power generation plant, the major part of NOx (nitrogen oxides) emission has been found to be NO. To model and predict the NOx emission level in power plants is the first step for optimisation and control to reduce the overall pollutant emissions (Thompson, and Li, 2003). According to De Soete (1975), there are three main sources of NO in combustion, namely thermal NO, Prompt NO and Fuel NO. The thermal, temporal and fuel NO formation mechanism can be found in (Thompson, and Li, 2003). According to these mechanism, two 'eng-genes' were selected as the activation function in 'eng-genes' based neural modelling, $\varphi_1 = e^{(c_1/(x+b_1))}$ and $\varphi_2 = (x+b_2)^{c_2}$. Again, φ_1 is an 'Arrhenius' type 'eng-gene', φ_2 is a 'non-Arrhenius' type 'eng-gene'.

In this paper, a coal-fired power plant in Northern Ireland was studied with a full load of 300 MWe with oil firing or 200 MWe with coal firing. According to the plant operation mechanism, the following operational variables are identified to have contributions to the overall NOx emission (Li, Thompson & Peng, 2003): Mass flow of fuel m_f (Kg/s); Mass flow of air m_a (Kg/s), specifically, mass flow of primary air m_{pa} and mass flow of secondary air m_{sa} (Kg/s); Tilting position of burners θ (degree).

Three data sets with 7000 samples in total was examined, among which first data set with 2300 data points were used for modelling, and data set 2 with 2500 data points and data set 3 with 2500 data points were used for validation. The statistics of the NOx emissions in the three data sets is listed in Table 3.

Again eng-genes models and MLP models were developed using the GA-based optimisation software with the same GA operation parameters. The long-term prediction performance of a 6-3-1 MLP model and a 6-3-1 eng-genes model is shown in table 4 with both networks evolved for 50 generations and 100 generations. The computation time for optimising the two networks was almost the same.

Table 3 Statistics of NOx in the two data sets (ppm)

	Min	Max	Mean	STD
Data set 1	252.90	452.5	306.8	21.9
Data set 2	198.6	375.0	296.9	20.9
Data set 3	199.8	342.6	298.7	17.4

Table 4 Long-term prediction performance (RMSE) of the 'eng-genes' model and the MLP model

		50 gen	100 gen
MLP	Modelling	15.50	14.50
	Valid-1	16.18	22.32
	Valid-2	12.34	16.55
Eng-genes	Modelling	15.64	14.35
	Valid-1	13.82	17.67
	Valid-2	11.91	13.43

The selected neural inputs are θ(t-1), θ(t-2), θ(t-4), m_f(t-2), m_{sa}(t-1), NO$_x$(t-1). The eng-genes models optimised by the GAs used $\varphi_1 = e^{(c_1/(x+b_1))}$ as the activations function for all the three hidden neurons in the network, and non-Arrhenius' type 'eng-gene' $\varphi_2 = (x+b_2)^{c_2}$ was not selected. Fig. 4 and 5 show the long-term predictions of the 6-3-1 'eng-genes' model over the modelling data set and the validation data set 2 when the model was evolved for 50 generations.

4. CONCLUSION

In this paper, a detailed discussion has been made on the motivation behind the 'eng-genes' genetic modelling approach for nonlinear engineering systems where the system behaviour is either too complex to build a simple model or our knowledge about the system is only partially known. The eng-genes genetic modelling framework has been proposed. This modelling method has then been used to model two different processes, and it has been shown that even simple 'eng-genes' models can be built to approximate complex nonlinear systems with desired long-term prediction performance. Moreover, in these two case studies the eng-genes used for modelling are all Arrhenius' type exponential functions which are physically meaningful therefore the model transparency has been significantly improved. Future work will include the development of various neural network structures for general engineering problems, and application of these networks to real time system operation and control.

ACKNOWLEDGEMENTS

The author wishes to acknowledge the financial support of the UK Engineering and Physical Sciences Research Council (EPSRC Grant GR/S85191/01).

REFERENCES

Bohlin, T., (1994). Case study of grey box identification. *Automatica*, **30**, pp. 307-318.

De Soete, G.G., (1975). Overall reaction rates of NO and N$_2$ formation from fuel nitrogen. 15th Symposium (international) on Combustion, The Combustion Institute, 1093-1102.

Goldberg, D. E. (1989). *Genetic Algorithms in Search, Optimization, and Machine Learning*. Addison-Wesley Publishing Company, Inc.

Pearson RK, Pottmann M (2000). Gray-box identification of block-oriented nonlinear models. *Journal of Process Control*, **4**, pp. 301-315.

Psichogios, D.C., and L.H. Ungar (1992). A Hybrid Neural Network – First Principles Approach to Process modelling. AiChE Journal, 38, pp. 1499-1511.

Sohlberg, B (2003). Grey box modelling for model predictive control of a heating process. *Journal of Process Control*, **13**, pp. 225-238.

Li, K., S. Thompson and J. Peng (2004). Modelling and prediction of NO$_x$ emission in a coal-fired power generation plant. *Control Engineering Practice*. **12**, 707-723.

Thompson, S. and K. Li (2003). Modelling of NOx emission, In: *Thermal power plant simulation, monitoring and control.* (D. Flynn. (Ed)) pp: 243-268. IEE, London.

Tulleken, H. J.A.F. (1993). Grey-box modelling and indentification using physical knowledege and Bayesian techniques. *Automatica*, **29**, pp. 285-308.

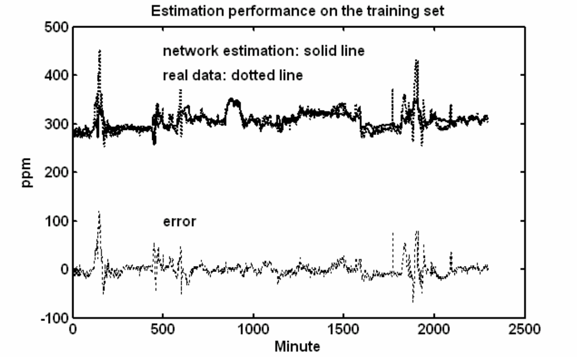

Fig. 4. Long-term estimation performance of the eng-genes model on the modelling data

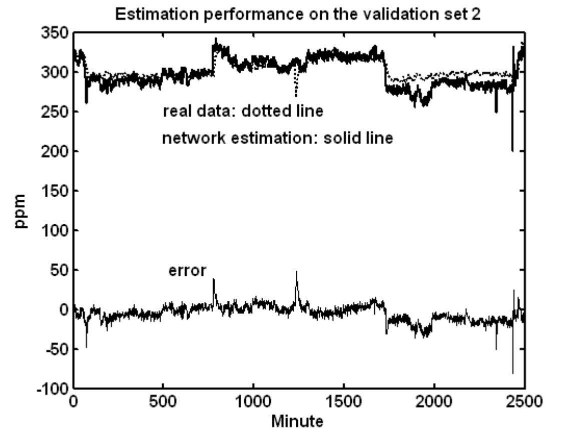

Fig. 5. Long-term estimation performance of the eng-genes model on the validation data 2