International Journal of Fluid Power 2 (2001) No. 1 pp. 37-47

A FUZZY NEURAL NETWORK APPROACH TO MODEL HYDRAULIC COMPONENT FROM INPUT/OUTPUT DATA

Wei Xiang, Sai Cheong Fok and Fook Fak Yap

School of Mechanical & Production Engineering, *Nanyang Technological University, 50 Nanyang Avenue, Singapore 639798*
P145355261@ntu.edu.sg

Abstract

The knowledge of dynamics of hydraulic components are vital for the virtual prototyping of fluid power systems. This paper proposes a fuzzy neural network approach to model the behavior of a hydraulic component from its input-output data. The main advantage of this approach is that the network structure can be determined based on the analysis of the input variables to output response, without trial and error, network pruning or network growing techniques. The process involves resolving the significant inputs through an analysis of their effects with respect to the output. The number of fuzzy rules is determined based on partitioning of the input-output space. The number of significant inputs and the number of fuzzy rules together define the fuzzy neural network structure. A hydraulic pressure relief valve is used to demonstrate the proposed approach. The results indicate that the structure of the fuzzy neural network determined based on the proposed approach can effectively model the dynamics of the relief valve. This work constitutes initial effort towards determining the structure of neural networks based on the analysis of input-output data.

Keywords: fuzzy neural network (FNN), fluid power system, virtual prototyping

1 Introduction

A hydraulic system typically comprises of many standard components. Some examples of these standard components are the directional control valve, relief valve, cylinder, motor and variable displacement pump. After conceptualizing the hydraulic circuit, the designer will have to develop a physical prototype. The process involves the selection and assembly of commercially available components. Although the basic function of the same type of standard components from different manufacturers does not differ, their dynamic characteristics, sizes and shapes may not be similar due to design and manufacturing variation. Hence for a fixed hydraulic circuit, different combinations of parts from different manufacturers will affect not only the overall system performances but also its physical structure. Under such circumstances, design evaluation through physical prototyping can be time consuming and expensive. To overcome this problem, Xiang, Fok, and Yap (2000) had proposed to use virtual prototyping technology to analyze and assess the system design.

Virtual prototyping can be viewed as a part of the computer-aided design process, which employs modeling and simulating tools to address the broad issues of physical layout, operational concept, functional specifications and dynamics analysis under various operating environments (Drews and Weyrich, 1997). A virtual hydraulic system prototype must be able to provide to the designer the system behavior, structure and other associated product evaluation information. The modeling of the hydraulic components' dynamics and the encapsulation of these models with other useful design evaluation information are important issues that need to be addressed in the development of virtual prototypes. This paper focuses on the modeling of the system dynamics performance, which would depend on the dynamics of the constituent basic components and their interaction.

The dynamics of a hydraulic component are usually non-linear due to the physical behavior of the component. The conventional modeling approach requires complete understanding of the component and using fundamental physical principles to represent these characteristics. Transfer functions (for Single-Input-Single-Output systems) and state-space equations (for Multiple-Input-Multiple-Output systems) are often used to approximate these dynamics characteristics about nominal operating conditions. Power bond graph (Dransfield, 1981) is another useful approach for representing the dynamics of hydraulic systems. This is an

This manuscript was received on 9 October 2000 and was accepted after revision for publication on 2 March 2001

intuitive modeling approach designed specifically for the description of processes that transfer energy. It is based on the causal effects that describe the energy transformations and is directly applicable to both linear and non-linear systems. The main disadvantage of the mentioned approaches is that the derivation of the dynamics equations of a complicated fluid power system can become very tedious.

An alternative to the mentioned approach is to measure the component's input-output variables and identify the behavior through the data (Watton and Xue, 1995). There are many methods available for the identification of dynamics behavior using input-output data. Among these, artificial intelligence techniques like neural networks are becoming popular. The reason is that properly trained neural networks process good generalization capabilities and could predict the dynamics behavior under other operating conditions. Given a set of input-output data (with N inputs, $\{x_i \mid i = 1, 2 \ldots N\}$ and one output y), a multi-layered neural network can always map these data with a function $[f: \Re^N \rightarrow \Re]$ over the compact set (Poggio and Girosi, 1990).

The description of a component's dynamics in terms of input-output mapping has its advantages. It is quite natural for hydraulic system engineers to use input-output mapping to describe the behavior of a hydraulic component. The configuration design of a hydraulic system is often achieved through steps of function decomposition. The engineer often tries to decompose the functions and their requirements down to the component level. The information for the design at this level is basically an input to output description and the engineer has to search for a component that can map the function to the requirements. The input-output mapping approach is also becoming more appealing to the manufacturers because unlike other representations, the inherent details that characterize components' behaviors need not be disclosed and the parameters describing the dynamics are hidden within the weights of the neural-nets. This is important in virtual prototyping, as manufacturers do not need to reveal confidential data relating to the virtual component while allowing the engineers to access the dynamics for design evaluation purposes. To incorporate neural network models for virtual prototyping of fluid power systems, Fok, Xiang and Yap (2000) had established a framework, which can integrate the dynamics represented by neural networks with other structural and product attributes.

The major drawback of using artificial neural networks for modeling hydraulic components' performances lies in the determination of the network structure. In the work of Burton, Ukrainetz, Nikiforuk and Schoenau (1999), the backpropagation neural networks were used to map the input signal to control

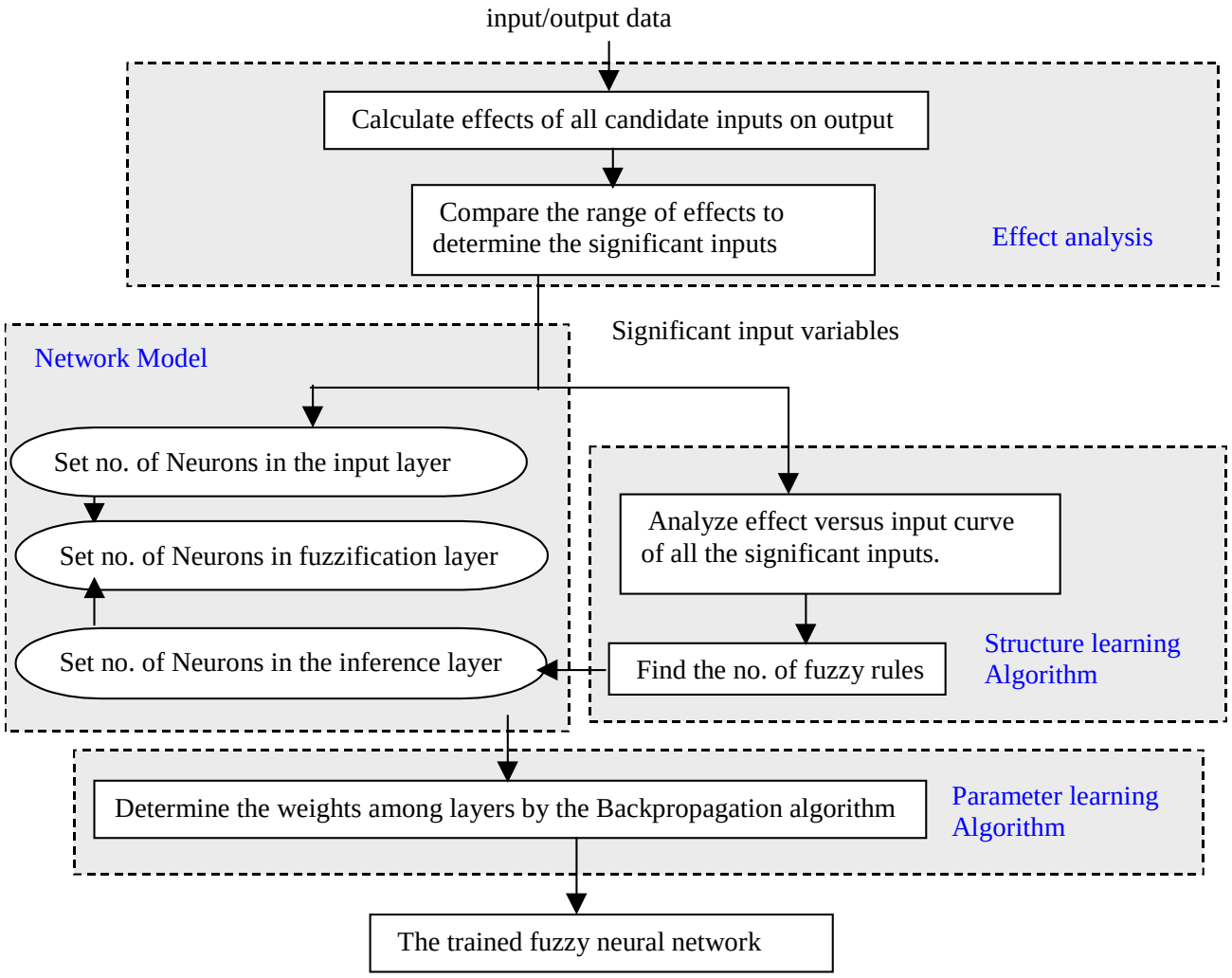


Fig. 1: *The fuzzy neural network approach*

action so as to control the performance of nonlinear hydraulic servo systems. The work indicated that the backpropagation neural networks have a tremendous potential for industrial application of hydraulic servo systems. Nevertheless, the main concern with is approach is that there is still an element of trial and error involved in the determination of the appropriate network structure that includes a number of hidden neurons.

To determine the network structure, researchers have resolved to use either of the two search methods. The first involved network pruning. For example, the self-organizing neural network proposed by Xue and Watton (1995) used the Group Method of Data Handling (GMDH) algorithm to model the dynamics of hydraulic components. The nonlinear mapping between the overall inputs and outputs is represented using a perceptron-type feed-forward neural network structure, where a polynomial function was applied in each layer. The GMDH algorithm can evolve the optimal network structure based on three parameters: the required RMSE (Root-Mean-Square Error), the maximum number of neurons in each layer, and the number of inputs. This network pruning approach essentially eliminates redundant nodes. The second method involved growing the network. For example, the self-organising radial basis function network using genetic algorithm by Xue and Watton (1998) is based on the network growing approach; i.e. the number of hidden neurons is increased to reduce the model error.

The results of the works by Burton, Ukrainetz, Nikiforuk and Schoenau (1999) and Xue and Watton (1995) (1998) indicated that the mapping of the input-output requires the consideration of two factors: identification of the significant input variables and the determination of the network structure. Although both network pruning and growing can be used to determine the neural-nets structure, these search techniques can be very time consuming. This paper proposes an alternative approach to determine the structure of a fuzzy neural network based on the analysis of the input-output data. The proposed fuzzy neural network approach is based on the concept of "fuzzy curve" proposed by Lin, Cunnihinghan III and Coggeshall (1997). An "effect" variable, which can be derived from the input/output data, is introduced to determine the significant inputs, estimate the number of rules needed in the fuzzy model, and determine the appropriate structure of the fuzzy neural network. Section 2 describes the fuzzy neural network approach with the identification of the network structure using the "effect" variables and the network training using the backpropagation algorithm. Section 3 presents the proposed approach to map a hydraulic pressure relief valve's performance in a hydraulic pressure control system. Results of the performance of the proposed mapping are included. Based on these results, conclusions and future work are summarized in the last section.

2 A Fuzzy Neural Network Modeling Approach

The proposed fuzzy neural network modeling approach is shown in Fig. 1.

2.1 Effect analysis to determine the significant inputs

The proposed fuzzy neural network aims to model the discrete time behavior of hydraulic components. The "effect analysis" is used to determine the significant inputs. To obtain the significant recurrent inputs in discrete time models, the extraneous inputs from the time history of the input/output data have to be eliminated. The identification of the significant inputs could be accomplished using a simplified analysis of the "effectiveness" of the sampled inputs to influence the output. The concept of "effectiveness" is based on the fuzzy logic curves proposed by Lin, Cunnihinghan III and Coggeshall (1997). Assume that the input/output data of a multi-input-single-output system contains the time history of "n" input candidates x_i(i = 1, 2, ..., n) and output y. For input x_i there are q sampled data points. The effect of the k^{th} sampled input x_i on output y can be defined as:

$$effect_k\left(x_i(k)\right) = \frac{\sum_{j=1}^{q} \exp\left(-\left(\frac{x_i(j)-x_i(k)}{\sigma}\right)^2\right)\cdot y(j)}{\sum_{j=1}^{q} \exp\left(-\left(\frac{x_i(j)-x_i(k)}{\sigma}\right)^2\right)} \quad (1)$$

$$\left(k = 1, \cdots q\right)$$

where $x_i(j)$ and $y(j)$ are the j^{th} samples of input x_i and output y respectively, and σ is typically taken as about 20% of the length of the interval of x_i. The $effect_k(x_i(k))$ can be viewed as the centroid of the input x_i to output y based on a Gaussian distribution function centered at $x_i(k)$. For proper comparison of the effects for different inputs, it is highly desirable to normalize all the sampled data to appropriate ranges before using Eq. 1.

The determination of the significant inputs can be based on the comparison of the range of variation of the effects of all candidate inputs. The larger the range, the larger the effect of that input on the output will be. This concept is analogous to examining the shift of the centroid of the input to output data. If the effect of an input variable remains relatively constant (i.e. the range of shift in the centroid is small), then that input variable should have very little effect on the output. The range of $effect_k(x_i(k))$ should be normalized with respect to $effect_k(y(k))$. The input variables with the relatively big ranges of effect values are identified as the significant input variables. The identified significant inputs will determine the number of input neurons in the input layer of the fuzzy neural network.

Fig. 2: *The structure of a fuzzy neural system*

2.2 The fuzzy neural network structure

The structure of the fuzzy neural system consists of components of a conventional fuzzy logic system and the neural network system as shown in Fig. 2. The fuzzy logic rule is represented as "if x_1 is A_1, …, and x_i is A_i, then y is B" where A_i is a vague or linguistic term. The fuzzification layer is used to build the antecedent of the fuzzy logic rule. The output of the fuzzification layer characterizes the possible distribution of the antecedent clause x_i is A_i". The inference layer calculates the certainty of each compound proposition "if x_1 is A_1, … , and x_i is A_i". This indicates how well the prerequisites of each fuzzy logic rule are satisfied. The defuzzification layer performs the rule evaluation. Figure 3 shows the structure of the fuzzy neural network derived from the system in Fig. 2. It is a four-layer network consisting of the input, fuzzification, inference and output/defuzzification layers.

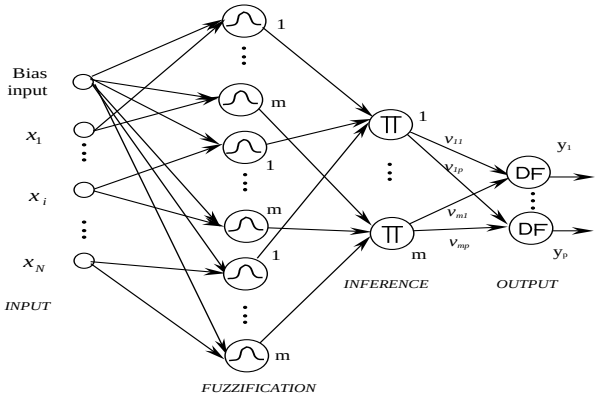


Fig. 3: *The structure of the fuzzy neural network*

The input layer of the fuzzy neural network has *N+1* input nodes, where *N* is the number of significant inputs. The bias input has a constant activation value of 1 and the bias neurons provide adjustable thresholds for each neuron connected to them. The number of output nodes corresponds to the number of output required in the model. The inference layer has *m* rule nodes. Each rule node represents one fuzzy logic rule, which describes the characteristics of a sub-set of input-output data. The number of fuzzy logic rules required will depend on the number of appropriate subsets in the input-output space. There are many methods to partition the input-output space into subsets (Juang and Lin, 1998), like Genetic Algorithm based partition (Sun and Jang, 1993), grid partition and cluster partition (Ruspini, 1982). A simplified cluster-based partition approach based on the "effect" is used. The "effects" of input-output data have been used to identify the significant inputs. From the variation of the "effect" of an input variable, the number of partitions required for mapping that input could be estimated. The process involves plotting $effect_k(x_i(k))$ against $x_i(k)$ for k = 1 to q. For any local or global maximum or minimum point in the "effect" versus input curve, one rule needs to be added. This is based on the idea that the fuzzy logic model will interpolate between the maximum and minimum points. In doing so, it essentially partitions the significant input to output space and assigns a fuzzy rule to each partition. The "fuzzy" rule has the form "if x_i is $\mu_{ik}(x_i)$, then y is $y(k)$", where $\mu_{ik}(x_i)$ is the Gaussian membership function for x_i.

Let the number of rules associated with significant input x_1 be R_1. For N significant inputs, each with a set of fuzzy rules, there will be N different sets of rules expressed as R_1, R_2, ..., R_N. The number of rules "m" needed in the overall fuzzy neural network is defined as $m = \max(R_1, R_2, ..., R_N)$.

Once the number of fuzzy logic rules for the network has been determined, the number of nodes in the fuzzification layer can be determined. The fuzzification layer has $N \cdot m$ nodes, where N is the number of significant inputs and m is the number of fuzzy rules. With the number of nodes in the input, fuzzification, inference and output layers determined, the network structure is fully defined.

The defined network will function as follow:

Layer 1: (Input Layer)

It has N significant input nodes (i.e. input number $\{i = 1, 2, ... N\}$) and a bias input of value 1.

$$net_i^{(1)} = x_i^{(1)} = x_i$$

$$y_i^{(1)} = f_i^{(1)}\left(net_i^{(1)}\right) = net_i^{(1)} \tag{2}$$

Layer 2: (Fuzzification Layer)

It has $N \cdot m$ nodes. m is the number of fuzzy logic rules. A Guassian membership function is employed in the neurons. The function can be represented by:

$$x_i^{(2)} = y_i^{(1)}$$

$$net_{ij}^{(2)} = -\frac{\left(x_i^{(2)} - m_{ij}\right)^2}{\left(\sigma_{ij}\right)^2}$$

$$y_{ij}^{(2)} = f_{ij}^{(2)}\left(net_{ij}^{(2)}\right) = \exp\left(net_{ij}^{(2)}\right) \tag{3}$$

where m_{ij} and σ_{ij} are respectively the center and the width of the Gaussian membership function of the j^{th} term of the i^{th} input variable x_i. In addition, m_{ij} and σ_{ij} form the weights and the bias between the input layer and the fuzzification layer. These parameters have to be learned through network training using the backpropagation algorithm.

Layer 3: (Inference Layer)

It has m fuzzy logic rule nodes (i.e. rule number $\{j = 1, 2, ... m\}$). The fuzzy AND operation used in this layer is the algebraic product (Tanaka, 1997). There are no weights to be estimated in this layer.

$$x_{ij}^{(3)} = y_{ij}^{(2)}$$

$$net_j^{(3)} = \prod_i^n x_{ij}^{(3)}$$

$$y_j^{(3)} = f_j^{(3)}\left(net_j^{(3)}\right) = net_j^{(3)} \quad (4)$$

Layer 4: (Defuzzification Layer)

It has p output nodes (i.e. output number $\{k = 1, 2, \ldots p\}$). The layer performs the Centre of Gravity (COG) defuzzification and gives the final network output.

$$x_j^{(4)} = y_j^{(3)}$$

$$net_k^{(4)} = \sum_{j=1}^{m} v_{jk} x_j^{(4)}$$

$$y_k^{(4)} = \frac{f_k^{(4)}\left(net_k^{(4)}\right)}{\sum_{j=1}^{m} x_j^{(4)}} = \frac{net_k^{(4)}}{\sum_{j=1}^{m} x_j^{(4)}} \quad (5)$$

where v_{jk} has to be learned through network training using the backpropagation algorithm.

2.3 A Parameter Learning Algorithm

Fuzzy neural networks can be trained using any of a number of training methods, such as the back propagation method (Horikawa, Furuhashi and Uchikawa, 1992), the conjugate gradient method (Leonard and Kramer, 1990), etc. In this work, the basic backpropagation algorithm is applied to learn the parameters of the network:

Normalize the sample input/output;
Initialize the proposed network parameters
Do {
Calculate the Fuzzy Neural Network output using to equation (2) to (5)
Calculate the error between the output $y_k^{(4)}$ *and the actual target* $d_k^{(4)}$*, i.e. find*

$$E = \frac{1}{2}\sum_{k=1}^{p}\left(d_k^{(4)} - y_k^{(4)}\right)^2$$

If (E < tolerance) or (epoch > Limit) {stop=true;}
else {
Calculate the back propagation of error in the layers 4/3/2. (see appendix A);
Calculate $V_{jk}, m_{ij}, \sigma_{ij}$*, and adjust the network parameters accordingly;*
}
} while(stop==false)

The backpropagation learning uses a gradient search procedure to find the weights of the network. The process can get trapped into local minima. This can be avoided using the momentum term (α in Appendix A). The convergence rate of the process can be controlled by η, the learning rate. Both the input and output training data for the network should be normalized. The fuzzification layer weights can be initialized based upon the distribution ranges of inputs and output. The algorithms for the fuzzy neural network were implemented in Java on the SGI-NT PIII542.

3 An Example

A circuit diagram of a hydraulic pressure control system using a direct acting pressure relief valve is shown in Fig. 4. The pump provides the pressure flow to the system and the pressure relief valve is used to control the pressure. Once the directional control valve is set to position 'open' (i.e. the situation shown in Fig. 4), the pressure in the system will vary accordingly to the changing load of the flow control valve. As soon as the direction control valve is closed, the system pressure will be limited to the pre-set pressure reference level of the pressure control valve.

The dynamics of the pressure relief valve was used to demonstrate the use of the proposed fuzzy neural network because the behavior can be easily and accurately modeled using bond graphs (Appendix B). The output was defined as present flow rate Q flowing into the pressure relief valve. The flow rate depends on the varying system load shown in Fig. 4. All the training data for the proposed fuzzy neural network model is obtained from the simulation result of the bond graph model. The pressure relief valve can be operated over a wide range of pressures and flow rates. In this example, it was set to operate at a pressure reference level of 30 bar. The fixed displacement pump delivered a flow rate of 27.6 l/min. The pressure P and flow rate Q were acquired through simulation at a sampling frequency of 500 Hz. This sampling frequency was derived based on the practical considerations by Watton and Xue (1995) in their identification of fluid power component behavior. Ten recurrent pressure responses (from P, P^{-1},…P^{-9}) and nine recurrent flow rate responses (Q^{-1},…Q^{-9}) were selected as possible candidate inputs. 960 data samples were used for training. Inputs were scaled to the range of [-1,1], while output was normalized to within [0,1].

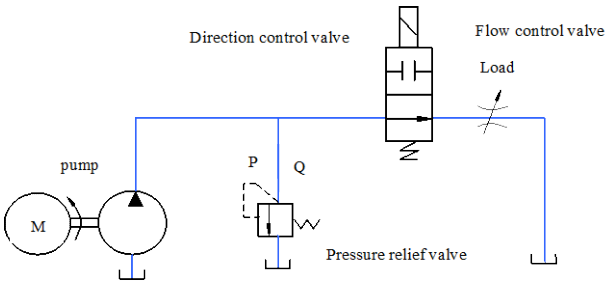


Fig. 4: *Pressure control system using pressure relief valve*

To find the set of significant inputs from the 19 possible candidate inputs, the effects of the sampled data were analyzed. Table 1 shows the normalized ranges of the effect analysis of all candidate inputs. The results of the analysis show that Q^{-1}, Q^{-2}, Q^{-3}, Q^{-4}, P^{-1}, P^{-2}, P^{-5}, P^{-6} (highlight in light gray in Table 1) have

Table 1: The normalized ranges of the effects of the 19 inputs with output Q

P	Q^{-1}	Q^{-2}	Q^{-3}	Q^{-4}	Q^{-5}	Q^{-6}	Q^{-7}	Q^{-8}	Q^{-9}
0.68	0.87	0.81	0.81	0.82	0.76	0.71	0.68	0.66	0.61
P^{-1}	P^{-2}	P^{-3}	P^{-4}	P^{-5}	P^{-6}	P^{-7}	P^{-8}	P^{-9}	Q
0.89	0.8	0.67	0.68	0.89	0.84	0.65	0.71	0.76	1

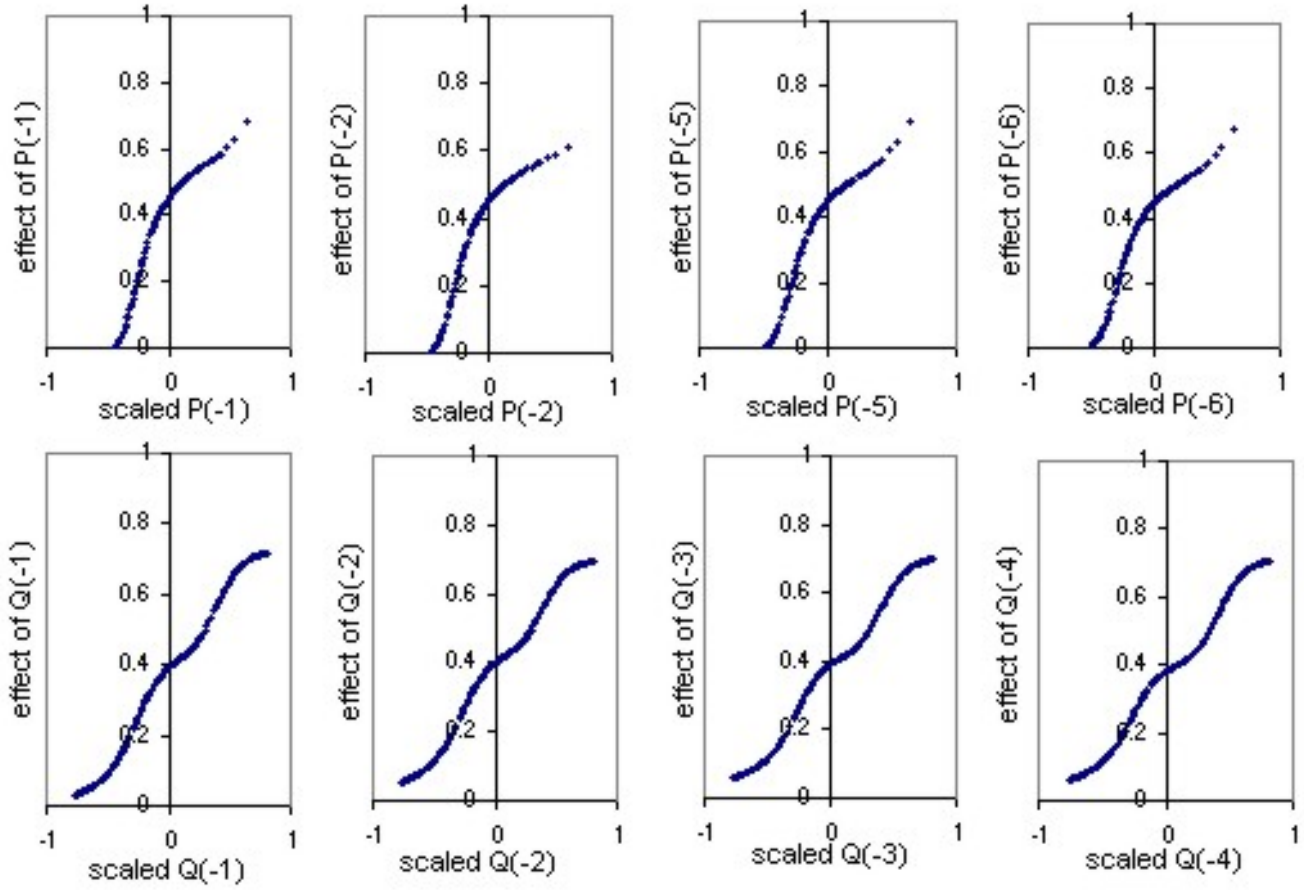


Fig. 5: *The effect vs. input curves*

relative high effects on the output Q. Figure 5 shows the "effect" versus input curves for Q^{-1}, Q^{-2}, Q^{-3}, Q^{-4}, P^{-1}, P^{-2}, P^{-5}, P^{-6}. From Fig. 5, the number of fuzzy logic rules for the selected recurrent flow rate inputs, i.e. Q^{-1}, Q^{-2}, Q^{-3}, Q^{-4} is $R_1 = 4$ and number of fuzzy logic rules for the selected recurrent pressure inputs, i.e. P^{-1}, P^{-2}, P^{-5}, P^{-6} is $R_2 = 3$. Therefore, the number of fuzzy rules required for the fuzzy neural network is $m = \max(R_i) = 4$ ($i = 1, 2$). The structure of the fuzzy neural network can be defined as follow:

Input layer has $N + 1 = 9$ nodes: i.e. nodes for Q^{-1}, Q^{-2}, Q^{-3}, Q^{-4}, P^{-1}, P^{-2}, P^{-5}, P^{-6}; and the bias input of value 1.

Inference layer has 4 rule nodes or neurons;

Fuzzification layer has $8 \cdot 4 = 32$ nodes or neurons;

Output layer has 1 node for Q;

In order to make a comparison of different fuzzy neural network structures, six models with a different number of fuzzy rules (i.e. 6, 5, 4, 3, and 2 rules) and with different selected inputs were trained using the same sampled data. The learning rate was set at $\eta = 0.008$ and the momentum rate was $\alpha = 0.4$. All the networks were trained for 6000 epochs and all parameters were found to have converged within the iteration. The average training time for the networks was 11 minutes on the Pentium III computer. This is considered adequate considering the low learning rate. A small learning rate will normally takes a longer time for the parameters to converge. The performances of the different fuzzy neural networks were compared using the Root-Mean-Square Error (RMSE):

$$RMSE = \sqrt{\frac{1}{n}\sum_{k=1}^{n}\left(Q_d(k) - Q_{fnn}(k)\right)^2} \qquad (6)$$

where, Q_d is the desired output flow rate generated from simulation, and Q_{fnn} is the actual output calculated by the network. The performance results are listed in Table 2.

Table 2: The comparison of fuzzy neural network models with different number of rules

Inputs	Number of Rules	RMSE
Q^{-1}, Q^{-2}, Q^{-3}, Q^{-4}, P^{-1}, P^{-2}, P^{-5}, P^{-6}	6	0.0310
	5	0.0315
	4	0.0317
	3	0.0355
	2	0.0462
Q^{-5}, Q^{-6}, Q^{-7}, Q^{-8}, Q^{-9}, P^{-3}, P^{-4}, P^{-7}, P^{-8},P^{-9}	4	0.0735

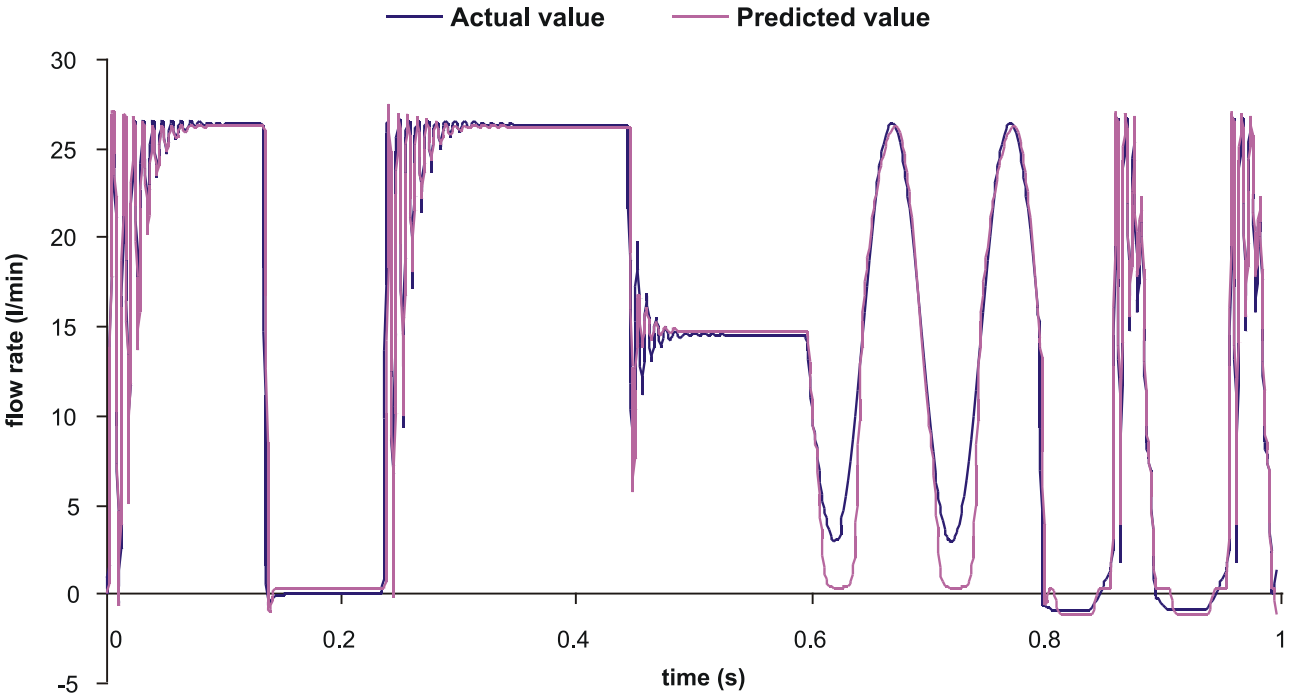


Fig. 6: *The flow rate response of the trained fuzzy neural network model*

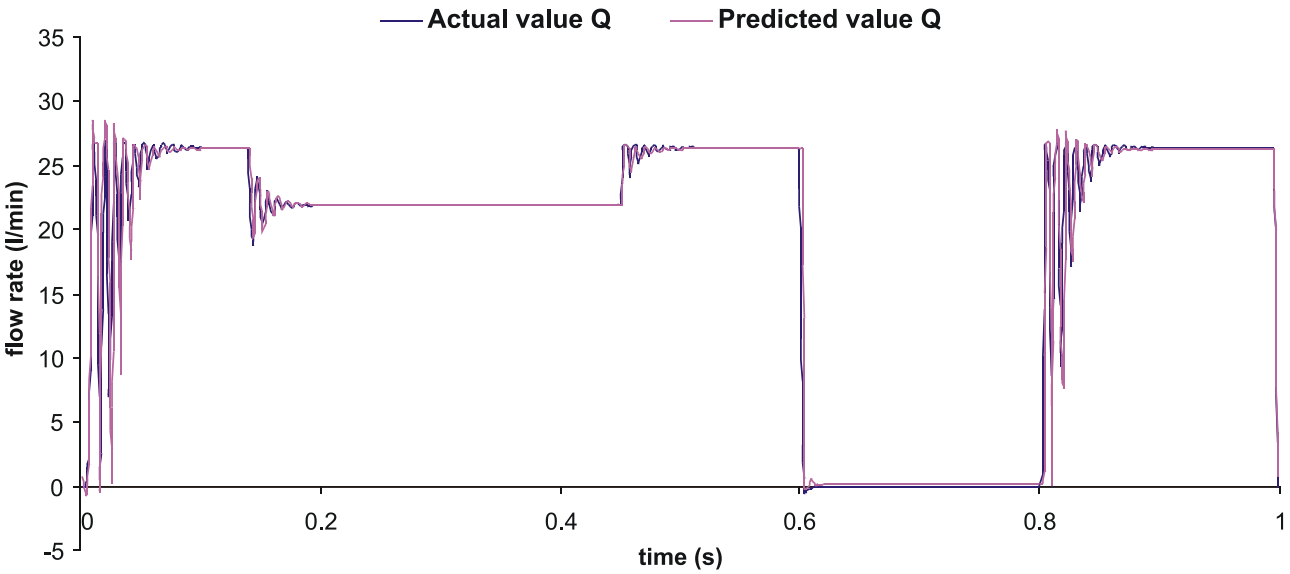


Fig. 7: *Validating the trained fuzzy neural network model under different operating condition*

The results indicated that there is little improvement in accuracy when the number of rules is increased from 4 to 5 and to 6. The accuracy using only 2 rules is un satisfactory. The performance of the fuzzy neural network with the set of selected inputs obtained from the effect analysis is better in terms of accuracy when compared to other inputs. Therefore, the results generally indicate that the fuzzy neural network model with Q^{-1}, Q^{-2}, Q^{-3}, Q^{-4}, P^{-1}, P^{-2}, P^{-5}, P^{-6} as inputs and 4 rules in the inference layer is sufficiently accurate to model the dynamics of the relief valve in Fig. 4. The recurrent inputs were not ordered due to the non-linearity of the discrete time model. Figure 6 shows the performance of this trained network. The results compared with the actual system's response show that this trained fuzzy neural network can memorize the training input-output data. The negative flow in Fig. 6 represents conditions when the relief valve closes, i.e. due to the reverse flow in Q_1 (Fig. B2 in Appendix B). Figure 7 shows the performance of the trained network when subjected to input data at other operating conditions. Compared with the simulated response, the results indicate that the trained network can also generalize well when given input data different from its training set. The results obtained so far mainly serve to illustrate the determination of the proposed network structure. The results indicate the potential of the proposed approach to develop a fuzzy neural network of determinable structure for input-output data mapping. As pressure and flow rate are commonly used in the dynamic performance descriptions and specification of hydraulic systems, the approach can be extended for the modeling of other fluid power components. However, it is anticipated that the network performance has to be further verified using real measured input-output data containing time delay effects and noises.

4 Conclusion

The dynamics of hydraulic components, expressed in terms of input to output mappings, are vital for the virtual prototyping of fluid power systems. This paper proposes a fuzzy neural network approach to model the dynamics of a hydraulic component from the input-output data. The proposed approach has advantages compared with other neutral network approaches in terms of identification of the significant input variables and the determination of the network structure from analysis of the input-output data.

The paper introduces an "effect" variable, which can be used to identify the significant inputs from the original input/output. The variation of the "effects" of the significant inputs can also be used to determine the number of fuzzy logic rules and the structure of the network. The fuzzy neural network can be trained using the back propagation learning algorithm. An example using the proposed approach to model the simulated dynamics of a hydraulic pressure relief valve is presented. The performances indicate that the proposed fuzzy neural network can be used to effectively map the simulated input to output behavior of the hydraulic component. Future work will involve:

The investigation of the performance of the network based on measured data with time delays and noise; Application of the concept to determine the structure of other types of neural networks from the analysis of the input-output data; Integration of the neural network model with 3-D geometrical data and product information for virtual prototyping.

Nomenclature

A	Spool's bottom area
C_p	Effective capacitance of the fluid and the pipe
C_s	Elastics of the spring in directory pressure relief valve
d_k	Desired output of the network
E	Error between desired output and neural network output value
$effect(x_i)$	effect of x_i on output
$I_{Flow\ coefficient}$	Mass of the pressure relief valve's spool
net_i, net_{ij}	Net input to the network layer
m_{ij}	Center of the Gaussian membership function.
P, P^{-d}	Pressure and it's discrete value
Q, Q^{-d}	Flow rate and its discrete value
N	The number of significant inputs
m	The number of fuzzy logic rules
R_h	Resistance along the alternative path in the relief valve.
R_l	Resistance of the leakage in the pump
R_s	System resistance caused by system load
R_v	Resistance of flow through the relief valve
S_e	Force source in Bond graph model
S_f	Flow source in Bond graph model
x_i, $x_i(k)$	Neuron's inputs, and k^{th} sample of neuron's input x_i
x_m	Spool's movement
x_{m0}	Initial overlap of the spool in pressure relief valve
y_i, $y_i(k)$	Neuron outputs and k^{th} sample of neuron's output y_i
V_{jk}	Weight between inference and defuzzification layer
v	Spool's velocity
σ, σ_{ij}	Width of the Gaussian membership function
δ	Back propagation of error
η	Learning rate of the neural network
α	Momentum of the neural network

References

Burton, R. T., Ukrainetz, P. R., Nikiforuk, P. N. and **Schoenau, G. J.** 1999. Neural networks and hydraulic control-from simple to complex applications. *Proc. Instn, Mech. Engrs. Part I*. Vol. 213, pp. 349-358.

Drews, P. and Weyrich, M. 1997. A system for Digital Mock-up's and Virtual Prototype Design in Industry: The Virtual Workbench. *Proc. IEEE International Symposium on Industrial Electronics*. Vol. 3, pp. 1292-1296.

Dransfield, P. 1981. *Hydraulic control Systems—Design and Analysis of Their Dynamics*. Springer-Verlag.

Fok, S. C., Xiang, W. and **Yap, F. F.** 2000. Feature-based Component Models for Virtual Prototyping of Hydraulic Systems. *International Journal of Advanced Manufacturing Technology*.

Horikawa, S., Furuhashi, T. and **Uchikawa, Y.** 1992. On fuzzy modelling using fuzzy neural networks with the back-propagation algorithm. *IEEE Transactions on Neural Networks*. Vol.3, No.5, pp. 801-806.

Juang, C. F. and **Lin, C. T.** 1998. An on-line self-constructing neural fuzzy inference network and its applications. *IEEE Transactions on Fuzzy systems*. Vol.6, No.1, pp. 12-32.

Leonard, J. A. and **Kramer, M. A.** 1990. Improvement of the back-propagation algorithm for training neural networks. *Computers Chem. Eng*. Vol. 14, pp. 337-341.

Lin, Y., **Cunnihingham III, G. A.** and **Coggeshall, S. V.** 1997. Using fuzzy partitions to create fuzzy systems from input-output data and set the initial weights in a fuzzy neural network. *IEEE Transactions on Fuzzy systems*. Vol. 5, No. 4, pp. 614-621.

Poggio, T. and **Girosi, F.** 1990. Networks for approximation and learning. *Proc. IEEE*. Vol. 78, No. 9, pp. 1481-1497.

Ruspini, E. H. 1982. Recent development in fuzzy clustering. *Fuzzy Set and Possibility Theory*. New York: North Holland, pp. 113-147.

Sun, C. T. and **Jang, J. S.** 1993. A neuro-fuzzy classifier and its applications. *Proc. IEEE Int. Conf. Fuzzy System*. San Francisco, CA, Mar. Vol I, pp. 94-98

Tanaka, K. 1997. *Introduction to fuzzy logic for practical application*. New York: Springer.

Watton, J. and **Xue, Y.** 1995. Identification of fluid power component behavior using dynamic flow rate measurement. *Proc. Instn Mech. Egnrs. Part C, Journal of Mechanical Engineering Science*. Vol. 209, No. 3, pp. 179-191.

Xiang, W., Fok, S. C. and **Yap, F. F.** 2000. Computational Tools for Fluid Power System Design: Towards distributed artificial intelligence and virtual reality. *International Journal of Computer Applications in Technology*. Vol. 13, No. 6, pp. 295-301.

Xue, Y. and **Watton, J.** 1995. A self-organizing neural network approach to data-based modelling of fluid power systems dynamics using the GMDH algorithm. *Proc. Instn. Mech. Engrs. Part I*. Vol. 209, pp. 229-240.

Xue, Y. and **Watton, J.** 1998. Dynamics modeling of fluid power systems applying a global error descent algorithm to a self-organising Radial Basis Function network. *Mechatronics*. Vol. 8, No. 7, pp. 728-745.

Appendix A: The backpropagation Algorithm

The basic back propagation algorithm is presented in this appendix. The main goal of the supervised learning is to minimize the error function, defined as:

$$E = \frac{1}{2}\sum_{k=1}^{p}\left(d_k^{(4)} - y_k^{(4)}\right)^2 \tag{A.1}$$

where, $d_k^{(4)}$ is the desired output and $y_k^{(4)}$ is the actual output of the k^{th} output node.

After random initialization of the network parameters, the training data are input into the network. The input signals are allowed to propagate through the network to obtain the calculated outputs. Based on the errors generated between the calculated and desired outputs, the back propagation process can be described as:

Backpropagation of error in output layer:

$$\delta_k^{(4)} = -\frac{\partial E}{\partial net_k^{(4)}} = -\frac{\partial E}{\partial y_k^{(4)}}\frac{\partial y_k^{(4)}}{\partial net_k^{(4)}} = \frac{d_k^{(4)} - y_k^{(4)}}{\sum_{j=1}^{m} y_j^{(3)}} \tag{A.2}$$

$$-\frac{\partial E}{\partial \nu_{jk}} = -\frac{\partial E}{\partial y_k^{(4)}}\frac{\partial y_k^{(4)}}{\partial net_k^{(4)}}\frac{\partial net_k^{(4)}}{\partial \nu_{jk}} = \delta_k^{(4)} \cdot y_j^{(3)} \tag{A.3}$$

$$\begin{aligned}\nu_{jk}(t+1) &= \nu_{jk}(t) + \eta \cdot \left(-\frac{\partial E}{\partial \nu_{jk}}\right) + \alpha\,\Delta\nu_{jk}(t) \\ &= \nu_{jk}(t) + \eta\,\delta_k^{(4)}(t) \cdot y_j^{(3)}(t) + \alpha\,\Delta\nu_{jk}(t)\end{aligned} \tag{A.4}$$

Backpropagation of error in inference layer:

$$\begin{aligned}\delta_j^{(3)} &= -\frac{\partial E}{\partial net_j^{(3)}} = -\frac{\partial E}{\partial y_j^{(3)}}\frac{\partial y_j^{(3)}}{\partial net_j^{(3)}} \\ &= -\sum_{k=1}^{p}\frac{\partial E}{\partial y_k^{(4)}}\frac{\partial y_k^{(4)}}{\partial y_j^{(3)}} = \sum_{k=1}^{p}\delta_k^{(4)}\left(\nu_{jk} - y_k^{(4)}\right)\end{aligned} \tag{A.5}$$

Backpropagation of error in fuzzification layer:

$$\delta_{ij}^{(2)} = -\frac{\partial E}{\partial net_{ij}^{(2)}} = -\frac{\partial E}{\partial y_{ij}^{(2)}}\frac{\partial y_{ij}^{(2)}}{\partial net_{ij}^{(2)}} = \delta_{j}^{(3)} \cdot y_{j}^{(3)} \qquad (A.6)$$

$$-\frac{\partial E}{\partial m_{ij}} = -\frac{\partial E}{\partial y_{ij}^{(2)}}\frac{\partial y_{ij}^{(2)}}{\partial net_{ij}^{(2)}}\frac{\partial net_{ij}^{(2)}}{\partial m_{ij}} = \delta_{ij}^{(2)} \cdot \left[\frac{2\left(x_i^{(2)} - m_{ij}\right)}{\sigma_{ij}^{\ 2}}\right] \qquad (A.7)$$

$$m_{ij}(t+1) = m_{ij}(t) + \eta\,\delta_{ij}^{(2)}(t) \cdot \left[\frac{2\left(x_i^{(2)} - m_{ij}\right)}{\sigma_{ij}^{\ 2}}\right] + \alpha\,\Delta m_{ij}(t) \qquad (A.8)$$

$$-\frac{\partial E}{\partial \sigma_{ij}} = -\frac{\partial E}{\partial y_{ij}^{(2)}}\frac{\partial y_{ij}^{(2)}}{\partial net_{ij}^{(2)}}\frac{\partial net_{ij}^{(2)}}{\partial \sigma_{ij}} = \delta_{ij}^{(2)} \cdot \left[\frac{2\left(x_i^{(2)} - m_{ij}\right)^2}{\sigma_{ij}^{\ 3}}\right] \qquad (A.9)$$

$$\sigma_{ij}(t+1) = \sigma_{ij}(t) + \eta\,\delta_{ij}^{(2)}(t) \cdot \left[\frac{2\left(x_i^{(2)} - m_{ij}\right)^2}{\sigma_{ij}^{\ 3}}\right] + \alpha\,\Delta\sigma_{ij}(t) \qquad (A.10)$$

where η is the learning rate, and α is the momentum parameter. Equations (A.4), (A.8) and (A.10) are used to update the weights in the fuzzy neural network.

Appendix B: Bond Graph model for the pressure control system using the direct control pressure relief valve

The example pressure control system is shown in Fig. 4. Figure B1 shows the Bond graph of the pressure control system used in the circuit of Fig. 4. The compressibility of the flow in the pipe is taken into consideration and is represented as C_p. R_l is the leakage in the pump, and R_v is the resistance of flow through the relief valve. R_h is the resistance along the alternative path in the relief valve. R_S is the system resistance caused by system load and is used to set the different operating conditions in the system. When the directional control valve is shut, R_s is regarded as ∞. S_e represents the preset spring force. S_f is the flow source and represents the constant flow from the pump. C_s represents the spring's

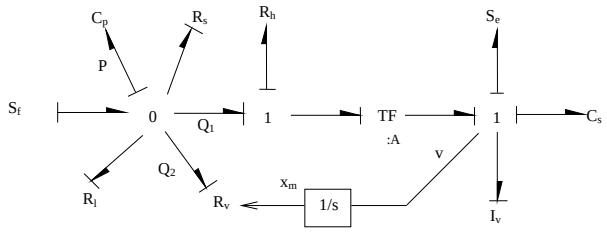


Fig B1: *The bond graph of the pressure control system*

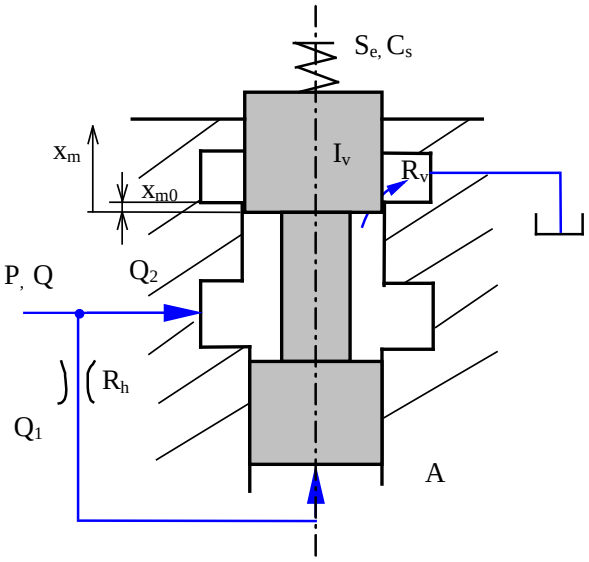


Fig B2: *Internal structure of direct pressure relief valve*

elasticity and I_v is the mass of the relief valve's spool. The internal structure of the direct pressure relief valve is given in Fig. B2. The flow rate passing through the relief valve consists of two parts: Q_1, Q_2. Q_1 is the flow that passes through a resistant hole located at the bottom of the valve's spool. Q_2 is the main flow that passes through the valve.The equations generated from the bond graph model are:

$$\dot{v} = -\frac{A^2 R_h}{I_v}v - \frac{1}{C_s I_v}x_m + \frac{A}{I_v}P - \frac{1}{I_v}S_e$$

$$\dot{x}_m = v$$

$$\dot{P} = -\frac{A}{C_p}v - \left(\frac{1}{R_l} + \frac{1}{R_s}\right)\frac{1}{C_p}P + \frac{1}{C_p}S_f \quad (x_m \le x_{m0})$$

$$\dot{P} = -\frac{A}{C_p}v - \left(\frac{1}{R_l} + \frac{1}{R_s}\right)\frac{1}{C_p}P + \frac{1}{C_p}S_f - K \cdot \pi \cdot d\sqrt{2/\rho} \cdot \frac{x_m - x_{m0}}{C_p} \cdot \sqrt{P} \quad (x_m > x_{m0})$$

$$Q = Q_1 + Q_2$$

where, $Q_1 = A\,v$

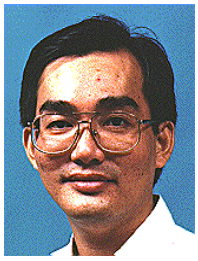
$$Q_2 = K \cdot \pi \cdot (x_m - x_{m0})\sqrt{2P/\rho} \quad (x_m > x_{m0})$$

$$Q_2 = 0 \quad (x_m \le x_{m0})$$

v is the spool's speed, x_m is the spool's movement, P and Q are the pressure relief valve's pressure and flow rate respectively. The following values were used in the simulation: $x_{m0} = 0.14$ (cm); $C_p = 0.8 \cdot 10^{-12}$ (m^3/Pa); $S_f = 4.6 \cdot 10^{-4}$ (m^3/s); $d = 1.2$ (cm); $I_v = 6.15 \cdot 10^{-2}$ (Kg); $R_h = 3.92 \cdot 10^{10}$ (Pa s/m^3); $R_l = 1.47 \cdot 10^{11}$ (Pa s/m^3); $C_s = 2 \cdot 10^{-5}$ (m/N); and $K = 0.7$.

To simulate the relief valve's dynamics at different work conditions, the different values for R_s were used.

Wei Xiang
Received the M.Sc degree in Mechanical Engineering from Dalian University of Technology, China, in 1996. She worked as a research assistant in Dept. of Mechanical Engineering of Hongkong Polytechnic University from 1997-1998. She is currently the Ph.D. candidate in School of Mechanical & Production Engineering at Nanyang Technological University, Singapore.



Sai-Cheong Fok
Received the B.A.Sc. degree in engineering from University of Ottawa, Canada in 1985 and the Ph.D. degree in mechanical engineering from Monash University, Australia in 1990. He has worked as an engineer in the aircraft industry. He is currently an Associate Professor in the Sch. of Mechanical & Production Engineering at Nanyang Technological University, Singapore. His current research interests are in virtual prototyping, machine learning, and intelligent modeling and control.

Fook-Fak Yap
Received the B.A.Sc. degree and Ph.D. degree in engineering from University of Cambridge, UK. in 1990 and 1994. He is currently an Associate Professor in the Sch. of Mechanical & Production Engineering at Nanyang Technological University, Singapore. His current research interests are in Dynamics, Statistical Energy Analysis, Vibro-acoustic Analysis, Virtual Dynamic Prototyping .