ÉCOLE DE TECHNOLOGIE SUPÉRIEURE
UNIVERSITY OF QUEBEC

THESIS SUBMITTED TO
ÉCOLE DE TECHNOLOGIE SUPÉRIEURE

IN FULFILLMENT OF THE REQUIREMENTS
FOR THE DEGREE OF MASTER
IN ENGINEERING MECHANICS
M. Eng

BY
NAM TUAN PHUONG LE

MODELLING OF CALIBRATION STEAM FLOWMETER TEST BENCH

MONTREAL, JANUARY 15, 2004

THIS THESIS IS EVALUATED

BY THE JURY COMPOSED OF :

M. Christian Masson, president of jury
Department of Mechanical Engineering, École de technologie supérieure

M. Louis Lamarche, director of thesis
Department of Mechanical Engineering, École de technologie supérieure

M. Stanislaw Kajl, co-director of thesis
Department of Mechanical Engineering, École de technologie supérieure

M. Gérald Bénard, engineer
Preston Phipps Inc

THIS THESIS WAS PRESENTED IN FRONT OF JURY AND PUBLIC

ON DECEMBER 17, 2003

AT ÉCOLE DE TECHNOLOGIE SUPÉRIEURE

MODÉLISATION D'UN BANC D'ESSAI DE CALIBRATION DES DÉBITMÈTRES DE VAPEUR

Nam Tuan Phuong Le

SOMMAIRE

Ce présent projet développe des méthodes de simulation pour le banc d'essai utilisé pour calibrer le débitmètre de vapeur en utilisant la vapeur comme le fluide de travail. Ce banc d'essai est équipé avec un condenseur de vapeur. Le condensât mesuré dans une période de temps donnée en régime permanent est employé pour calibrer le débitmètre.

Deux modèles mathématiques sont développés pour étudier le transfert de chaleur dans le condenseur. Il s'agit d'un échangeur thermique de type tube et calandre. Basé sur la théorie du réseau de neurones, le premier modèle est pour prédire la température, la pression et le débit du vapeur et la hauteur du condensât. Les paramètres d'entrée pour ce modèle sont la position d'ouverture des valves de contrôle et la température d'entrée de l'eau de refroidissement.

Le second modèle (modèle physique) est basé sur la conservation de la masse et de l'énergie. Pour augmenter la précision, le condenseur est divisé en volumes de contrôle auxquels l'équilibre de la masse et de l'énergie sont appliqué. Le modèle décrit la réponse du système en différentes variables, y compris le niveau du condensât et la pression de la vapeur dans l'échangeur thermique. Les valeurs d'entrée de ce modèle sont la position d'ouverture des valves, le débit et la température d'entrée de l'eau de refroidissement.

Ces deux modèles donnent de bons résultats comparés aux données expérimentales. Cependant, le modèle physique est préféré grâce à sa capacité de construire des relations entre des paramètres physiques du problème. Il est suggéré que dans le futur travail, ce modèle soit employé pour modéliser un système de contrôle à multivariables, qui est pour la régulation du niveau du condensât et la pression de la vapeur.

MODELLING OF CALIBRATION STEAM FLOWMETER TEST BENCH

Nam Tuan Phuong Le

ABSTRACT

The present project develops simulation methods for the test bench that serves to calibrate the steam flowmeters by using steam as working fluid. The test bench contains a condenser, the purpose of which is to condense the steam. By maintaining a steady-state regime, the weight of condensate measured during a given period of the time is used to calibrate the steam flowmeter.

Two mathematical models are developed in this project for the heat transfer process within the condenser, which is a shell-and-tube heat exchanger. The first model is based on neural network theory for predicting steam temperature, pressure, flow and the height of condensate. The input parameters to this model include the opened-closed position of the control valves and the inlet temperature of the cooling water.

The second model (physical model) is based on the mass and energy conservation principles. For increasing the accuracy, the condenser is divided into control volumes to which the mass and energy balances are applied. The model describes the dynamic response of the system at different locations including the condensate level and the steam pressure within the heat exchanger. The input values are the opened-closed position of the valves, the flow rate and inlet temperature of cooling water.

Both of these models give satisfactory results compared with experimental data. However, the physical model is preferred by its capability of explaining the relationships between parameters. It is suggested for a future work that physical model is applied to a multivariable control system for the regulation of the level of condensate and the steam pressure.

RÉSUMÉ

Le but de ce projet est le développement un modèle mathématique pour un échangeur thermique au Centre de technologie thermique - École de technologie supérieure à Montréal. Le modèle sert à la calibration d'un débitmètre de vapeur. La connaissance du comportement dynamique de cet échangeur est important pour connaître la réponse du système au changement de débit, de température ou de pression. Ces informations sont vraiment nécessaires pour concevoir des contrôleurs d'un tel système.

Ce travail va présenter deux approches pour la modélisation de cet échangeur thermique. Le rapport comprend cinq chapitres.

Le premier chapitre fait une revue de la littérature portant sur la modélisation et simulation d'échangeur thermique de type '*shell* et *tube*' et de l'application du réseau de neurones dans la modélisation et la simulation.

Le deuxième chapitre décrit le système de calibration du débitmètre de vapeur et de l'opération du système de calibration suivant : la vapeur est alimentée par source de vapeur extérieure au système de condensateur à la pression 862 kPa. Le débit de vapeur est commandé par une vanne de contrôle. La vapeur après la vanne de contrôle est dirigée vers l'échangeur du type "*shell* et *tube*" et suivant un échange thermique avec l'eau froide à l'intérieur des tuyaux. L'eau qui se forme pendant la condensation est stockée en bas de l'échangeur et est ensuite amenée dans un réservoir et mesurée pour calibrer le débitmètre. La construction de l'échangeur thermique est présentée avec des dimensions géométriques.

La modélisation de notre échangeur thermique a été réalisée en utilisant deux approches différentes : une est basée sur les réseaux de neurones et l'autre sur le principe de conservation de matière et d'énergie.

Le troisième chapitre décrit le développement du modèle en utilisant le réseau de neurones. Ce modèle est capable de prédire le débit, la température et la pression de vapeur ainsi que le niveau d'eau condensée dans l'échangeur thermique. Cette nouvelle approche est largement utilisée dans plusieurs domaines : en finance, en ingénierie... et elle est maintenant utilisée dans notre travail. La procédure de modélisation passe par les étapes suivantes :

1. Détermination des *inputs & outputs*
2. Choix des données expérimentales
3. Division des données
4. Normalisation de données
5. Détermination de l'architecture de réseau
6. Application

Pendant la première étape, les paramètres d'entrée et de sortie du réseau de neurones doivent être déterminés.

Les entrées sont :

- Les positions d'ouverture des trois valves VR2-1V; VR1-1V; VR1-1C
- La température de l'eau refroidissant CT1-1E

Les sorties sont :

- La pression, la température, le débit de vapeur.
- Le niveau de condensât dans l'échangeur thermique.

Les données expérimentales mesurées au laboratoire sont utilisés pour déterminer les entrées et sorties afin de former le réseau de neurones. Les données expérimentales

contiennent toujours du bruit. Par conséquent, les données stables sont choisies pour l'entraînement (*training*) le réseau de neurones et les données stables sont divisées en sous-groupes.

Selon la pratique courante, les données disponibles sont divisées en deux sous-groupes : un groupe pour l'entraînement (*training*) et un autre groupe indépendant pour la validation. Mais les données stables (après le filtrage du bruit) sont divisées en trois sous-groupes : un pour l'entraînement, un autre pour la validation et le troisième pour le test. Le sous-groupe pour le test est utilisé pour vérifier la performance du modèle à différentes étapes du processus d'apprentissage. Les erreurs détectées ici ne sont pas utilisées pendant l'entraînement mais plutôt pour comparer la performance de différents modèles.

Les données sont divisées en trois sous-groupes mentionnés plus haut de la façon suivante :

1. Un quart est utilisé pour la validation.
2. Un autre quart pour le test.
3. La moitié qui reste est utilisée pour le traitement.

Après la division des données, les *inputs* et *outputs* sont normalisées à l'aide des fonctions **premnmx**, **postmnmx** disponibles dans Matlab avec les critères *Max* et *Min* de telle façon qu'elles se trouvent dans l'intervalle [-1;1].

Dans cinquième étape, il faut déterminer l'architecture du réseau de neurones. C'est une tâche très importante dont le but est de déterminer le nombre de couches dans le réseau de neurones, le nombre de neurones dans chaque couche, la fonction de transfert et l'algorithme mathématique.

Pour ce projet, un réseau comportant quatre couches avec une couche pour l'entrée de données (*inputs*), deux couches cachées et une couche pour la sortie de données (*outputs*). La première couche cachée contient 16 nœuds et la deuxième en contient 40, donc le rapport des nœuds de ces deux couches cachées est 1 : 2.5. Ce rapport a été obtenu par la méthode "*trial and error*". Le nombre de nœuds dans la couche *outputs* est égale au nombre de sorties (4) et le nombre de nœuds dans la couche d'entrée de données *(inputs)* est aussi égale au nombre des entrées (4). La fonction de transfert : la fonction **log-sigmoid** pour toutes les couches cachées et la fonction **purelin** pour la couche de sortie

L'algorithme '*back-propagation*' qui utilise la méthode de SCG (*scaled conjugate gradient*) développée par Moller [11] est choisi pour calculer des vecteurs de gradient pour mettre à jour les degrés de liberté.

1. Les degrés de liberté sont initialisés à zéro.
2. La taille d'Epoch (*mode Batch*) est égale à 75.
3. Le taux d'apprentissage est égal à 0.5
4. Critère d'arrêt : critère de 'cross-validation' est utilisé.

Finalement, après l'arrêt de processus d'apprentissage, un ensemble de données indépendantes sont utilisés pour valider le modèle comme c'est montré dans le rapport à la page 25 et 26.

Dans la section " Normalisation de données ", les données sont normalisées. De cette façon, pour valider le modèle avec un ensemble de données indépendantes, il faut vérifier que les *inputs* se trouvent dans l'intervalle [-1;1] avec les critères *Max* et *Min* définis plus haut. Cela signifie que les valeurs maximales et minimales de l'ensemble des données indépendantes ne dépassent pas les valeurs maximales et minimales des données d'entraînement.

Maintenant la capacité générale du réseau de neurones est améliorée et les intervalles d'opération suivante du modèle :

- Pour la valve de vapeur : 5% - 100 % (VR2-1V)
- Pour la valve de l'eau condensée : 5.2% - 6.5 % (VR1-1C)
- Pour la température d'entrée de l'eau froide : 119^0F – 131^0F (CT1-1E)

Dans le quatrième chapitre, un modèle physique est développé pour le même système de condensateur de vapeur précédemment décrit. Cette approche est basée sur la conservation de masse et d'énergie. Le modèle physique a même l'objectif comme le modèle de réseau de neurones, qui est de prédire des quantités de sorties (la pression de vapeur et le niveau d'eau condensée) en fonction des entrées. Les entrées du modèle physique comprend les paramètres : La position de l'ouverture de valve, la température d'entrée de l'eau froide et le débit de l'eau froide. De plus, le modèle physique décrit dynamiquement le système d'une condition initiale dans tout régime transitoire jusqu'à ce que la condition du régime permanent soit atteinte, alors que le modèle de réseau de neurones traite juste des états d'état d'équilibre.

L'échangeur thermique est divisé en trois volumes de contrôle (V.C.) où les deux premiers V.C. pour le condensât se trouve en bas et le troisième V.C. pour la vapeur se trouve en haut du *shell*. Pour le faire, les hypothèses suivantes sont considérées :

- La température de l'eau condensée T_{cj} dans chaque V.C. est uniforme.
- Les hauteurs de première et deuxième V.C. sont égales à la moitié de niveau du fluide dans l'échangeur ($H_1 = H_2 = H/2$).

L'application des bilans de matière et d'énergie est présentée à chacune de ces V.C. Le transfert de chaleur est calculé selon les formules présentées dans [13]. De plus, il y a deux équations caractéristiques pour la valve de vapeur et la valve condensât. Dans ces

deux équations, il faut trouver les valeurs K_s, K_c correspondantes aux positions fermée et ouverte de la vanne de contrôle et elles sont déterminées à partir des expériences.

Finalement, un système d'équations algébriques différentielles (EAD) est obtenu, qui serait résolu comme un ensemble des équations différentielles ordinaires (EDO) dans Matlab.

La validation de ce modèle a été réalisée avec les données expérimentales et les résultats sont présentés à la page 42 et 43 de ce rapport.

Dans le cinquième chapitre, la conclusion générale pour cette étude est présentée. Tous les deux modèles donnent des résultats satisfaisants comparés aux données expérimentales. Cependant, le modèle physique est préféré pour ses possibilités d'expliquer les relations entre les paramètres. On suggère pour des travaux futurs que le modèle physique soit appliqué à un système de contrôle *multivariable* pour la régulation du niveau du condensât et de la pression de vapeur.

Avec les résultats étant obtenus à partir de cette étude, ces modèles semblent maintenant être robustes et simuler qualitativement le phénomène physique.

ACKNOWLEDGMENTS

I would like to express my deeply gratitude to my director of this thesis professor Louis Lamarche for his supervision and guidance. I also thank my co-director of thesis professor Stanislaw Kajl for all the help as well as useful instructions that he brought me. A lot of technical meetings have been held between them and me, where they have provided me with knowledge and book titles making it possible to proceed with the progress. This thesis would not have been possible without their help.

I would also like to thank Mr. Christian Masson, professor of the department of mechanical engineering ÉTS, for his informed advices that were very useful for me.

I would also like to thank Mr. Van Ngan Le, professor of the department of mechanical engineering ÉTS for many good discussions and advice.

I would also like to thank the all Vietnamese overseas students at ÉTS for their help and support throughout my study at ÉTS.

Finally, I am anxious to send warm thanks to my parents, my brother and sister, my relatives and all my friends for their moral support.

CONTENTS

LIST OF TABLES

Page

LIST OF FIGURES

Page

NOMENCLATURE

A	Net cross-section area of shell side, m^2
C_{pj}	Specific heat of condensate water in control volume j (j = 1,2), J/kg.K
C_{ps}	Specific heat of steam, J/kg.K
D_h	Hydraulic diameter, m
D_s	Diameter of shell side, m
D_t	Inside diameter of tube, m
D_{to}	Outside diameter of tube, m
h_{cond}	Enthalpy of condensate water at control volume 2, J/kg
h_{fg}	Heat of vaporization J/kg.
h_i	Convection heat transfer coefficient at tube side, $W/m^2.K$
h_o	Convection heat transfer coefficient at shell side, $W/m^2.K$
h_i	Height of control volume i (i = 1,2,3), m
h_s	Enthalpy of steam, J/kg
h_{valve}	Enthalpy of condensate water at control volume 1, J/kg
K	Thermal conductivity W/m.K
K_s	Proportionality constant at steam valve, $kg/s/Pa^{1/2}$
K_c	Proportionality constant at condensate water valve, $kg/s/Pa^{1/2}$
L	Effective height of heat exchanger, m
L_{tot}	Total length of a tube, m
m_i	Weight of liquid in control volume i (i = 1,2,3), kg
$\dot{m}_c$	Mass inflow rate of control volume 1, kg/s
$\dot{m}_{cond}$	Mass inflow rate of control volume 2, kg/s
$\dot{m}_s$	Mass flow rate of steam, kg/s
$\dot{m}_{valve}$	Mass outflow rate of heat exchanger, kg/s
N	Number of tube

P_s	Steam Pressure, kPa
P_w	Pressure of condensate water in heat exchanger, kPa
q_k	Heat flux (k = 1,2…,5), W/m
Re	Reynolds number
T_i	Temperature inside the tube (i = 1,2,3,4,5,6), ^oK
T_{ci}	Temperature of condensate water at control volume (i = 1,2), ^oK
T_s	Temperature of steam, ^oK
UA	Overall transfer coefficient
ρ_i	Specific volume at control volume i (i = 1,2), kg/m^3
ρ_s	Specific volume of steam, kg/m^3
μ	Absolute viscosity, $N.s/m^2$

CHAPTER 1

LITERATURE REVIEW ON MODELING AND SIMULATION OF HEAT EXCHANGER

The shell and tube heat exchanger has been commonly used in the industry. It is important to know the dynamic behaviour of the heat exchanger in order to predict how the system will respond to a change in flow rate, temperature, pressure, etc.... Modeling and simulation of the dynamic behaviour of a shell-and-tube heat exchanger are presented in [1,2,3].

In [1], an experimental data was presented on the transient behaviour of an industrial scale shell and tube heat exchanger condensing a mixture of steam and air under reduced pressure. The system was subjected to step changes in steam flowrate, air flowrate, coolant flow rate and coolant inlet temperature. An understanding of the process is a necessary pre-cursor to modelling and simulation. These data were used to valid the mathematical model developed in [3].

In [2], a horizontal shell and tube heat exchanger was studied with both parallel and counter flow cases. It is assumed that the fluids are incompressible single phase, the temperature variation on both sides is relatively small, the density is constant, and there is no mass storage on either side of the heat exchanger. Energy balance equations at each well-mixed node were used to carry out the modelling. For each node, three energy balance equations are developed with the shell side fluid, the tube metal and the tube side fluid. Heat transfer correlations of shell side and tube side were taken from the paper of Kutbi. The goal of this model is to predict temperature distribution of the cold and hot fluids. The heat exchanger model developed in [2] has enough flexibility to allow both qualitative and quantitative comparisons of different designs for varying design goals. The precision of model is increased with increasing the number of well-

mixed nodes set by user. The restriction of this model is that we cannot predict the flow rate and the pressure of fluids. The model developed in [3] overcomes this problem.

In [3], the studied heat exchanger is also of horizontal shell and tube type but with the use of steam as the hot fluid and water as the cold fluid. In this study, a model has been developed that is able to simulate the steady state and dynamic behaviour of a condenser under conditions that can be found in chemical processes. Modelling was also done based on the conservation principle for each control volume, which is bounded by the baffles in the condenser. Material and energy balance equations are used for each phase in each control volume and the heat and material fluxes between the phases are determined using local transfer coefficients that are calculated for each baffle space. By comparing with the model in [2] which can just predict the temperature distribution of fluids, the model in [3] is able to predict steam and condensate flow rates, pressure drop, and temperature of steam, condensate, wall and coolant. Moreover, it also includes the determination of the pressure and temperature profile along the heat exchanger.

In this thesis, mass and energy balances are used to develop a mathematical model for a vertical shell-and -tube heat exchanger with one shell pass, two tube passes and no baffle. The hot fluid is steam and the cold fluid is water. The objective of this model is to predict steam and condensate flow rate, steam pressure and temperature, and condensate temperature as in [3]. Moreover, condensate level is also simulated for it is important in a vertical shell and tube heat exchanger.

In addition, another approach, namely artificial neural network is also used to model our system. A review in [4] of modelling issues and applications by using neural network was described for prediction and forecasting of water resources variables. The steps that should be followed in the development of such models are outlined, which include the choice of performance criteria, the division and pre-processing of the available data, the determination of appropriate model inputs and neural network, optimization of the

connection weights and model validation. These networks are trained using a back-propagation algorithm. Issues in relation to optimal division of the available data, data pre-processing and the choice of appropriate model inputs are considered. Moreover, choosing appropriate stopping criteria, optimal network geometry and internal network parameters are presented. In the present thesis, these same steps are used to develop a model for our system. This neural network model also predicts steam flow rate, pressure and temperatures and condensate level when we know the closed-opened position of control valves and the coolant inlet temperature.

CHAPTER 2

CALIBRATION SYSTEM OF STEAM FLOWMETERS

A test bench has been built in the CTT laboratory (Centre de technologie thermique) of École de technologie supérieure with the purpose of calibrating industrial steam flowmeters.

2.1 Description of the system

With the aim to calibrate steam flowmeters based on weight measurement of condensate, this system consists of a steam flow circuit and a cooling water circuit linked together by a shell and tube heat exchanger (see figure 1). The main components and operations of the system are as follows.

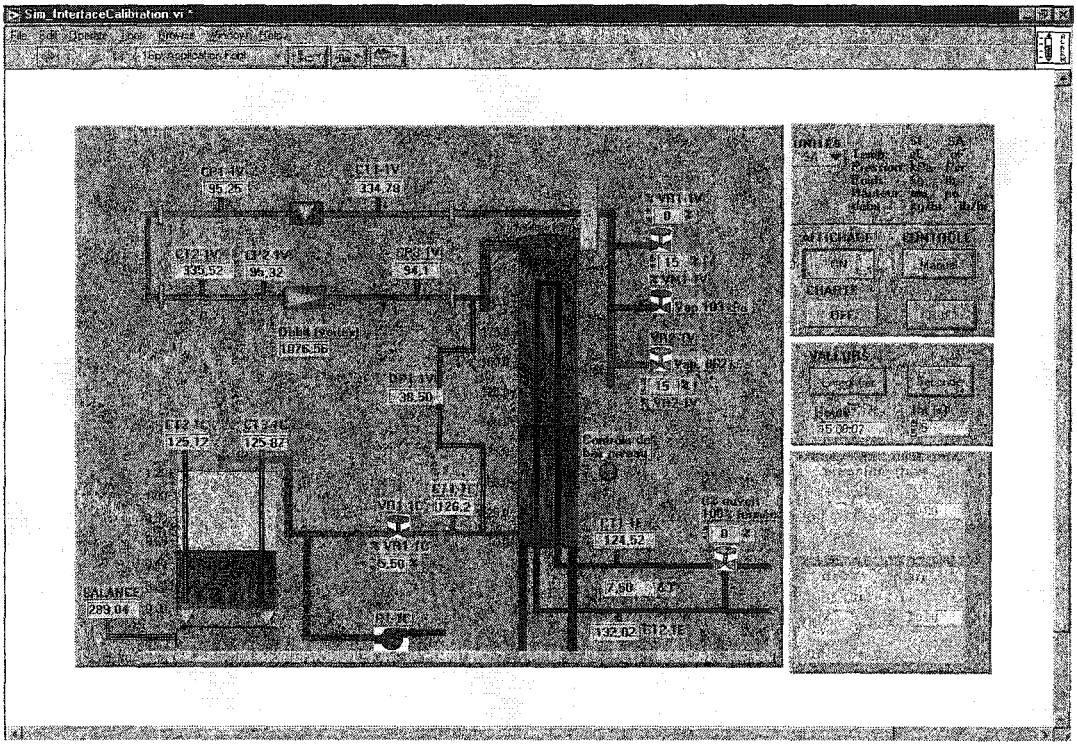


Figure 1 Calibration system of steam flowmeter

The steam is fed into the system by opening one of three valves shown by VR1-1V, VM1-1V and VR2-1V in the figure. This steam is in a saturated state the pressure (or temperature) of which is controlled by the opening position of the valve. The steam passes through the flowmeter to be calibrated shown by the symbol ▶ before entering the shell side at the upper end of the heat exchanger, which is a vertical vessel. By transferring heat to the cooling water, the steam condenses in this shell side. The condensate drops to the bottom of heat exchanger and is discharged by the shell side outlet pipe and collected into a reservoir (see bottom left of figure 1).

Cooling water enters the lower end of the heat exchanger from an inlet pipe by opening the valve installed on it. Inside the heat exchanger, the cooling water splits and flows upward into 24 small tubes (not shown in details) then downward into 24 other tubes, connect together and leaves the vessel by the outlet pipe.

A scale is used to measure the weight of collected condensate as a function of time in order to calibrate the condensate flow rate. The height of the condensate within the heat exchanger is indicated by a sight glass (a vertical transparent tube shown on left side of vessel) and is controlled by the opening position of the valve VR1-1C, installed on the condensate outlet pipe. When this height becomes steady, the condensate outflow measured by the scale determines also the steam flow rate.

An integrated data acquisition system allows to record numerical values of selected operating parameters (such as pressure, temperature, condensate level, etc) at regular time interval. In our experiments, this time interval is set to 5 seconds (see on the right side of figure 1) that give a reasonable number of sample data points linear fitting regression in deciding steady regimes.

2.2 Geometric parameters of the heat exchanger

Following parameters are needed for mathematical development of heat transfer process for controlling the steam pressure and the condensate level in the system (see figure 2).

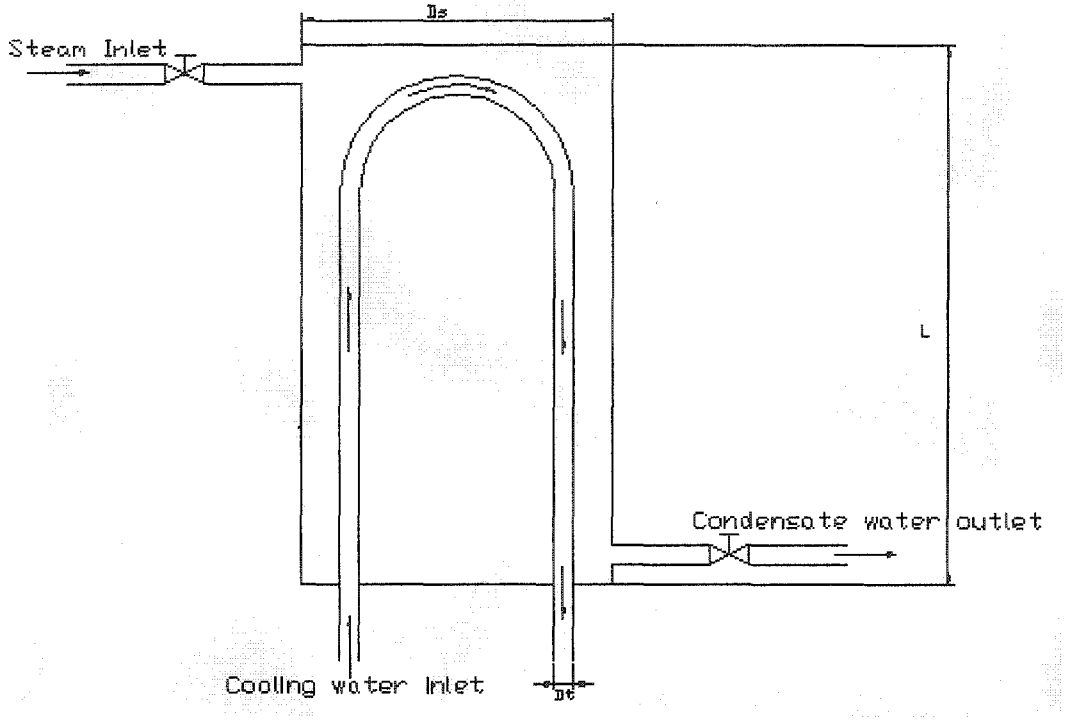


Figure 2 Shell and tube heat exchanger

- Shell diameter $D_s = 0.3048$m
- Diameter of tubes $D_t = 0.01905$m
- Effective height of heat exchanger $L = 2.5$m
- Number of tubes $N = 48$.

CHAPTER 3

DEVELOPMENT OF NEURAL NETWORK MODEL FOR A CONDENSER SYSTEM

3.1 Introduction to Neural Networks

Artificial Neural Networks is a system loosely modeled on the human brain. The field goes by many names, such as connectionism, parallel distributed processing, neuro-computing, natural intelligent systems, machine learning algorithms, and artificial neural networks. It is an attempt to simulate within specialized hardware or sophisticated software, the multiple layers of simple processing elements called neurons. Each neuron is linked to certain of its neighbours with varying coefficients of connectivity that represent the strengths of these connections. Training is accomplished by adjusting these strengths to cause the overall network to output appropriate results.

3.1.1 Basic unit of neural networks

The basic building block (unit) of neural networks, the artificial neuron, simulates the basic functions of natural neurons. The figure below shows the basics of an artificial neuron.

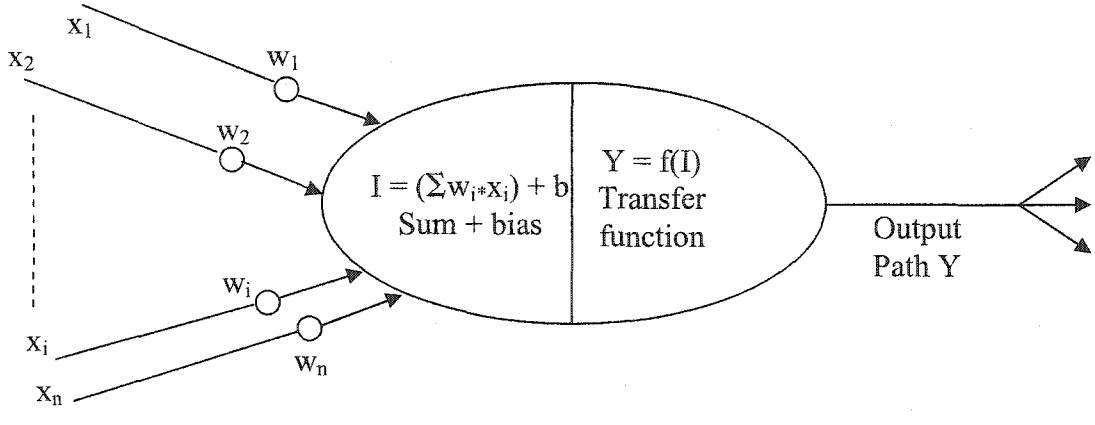
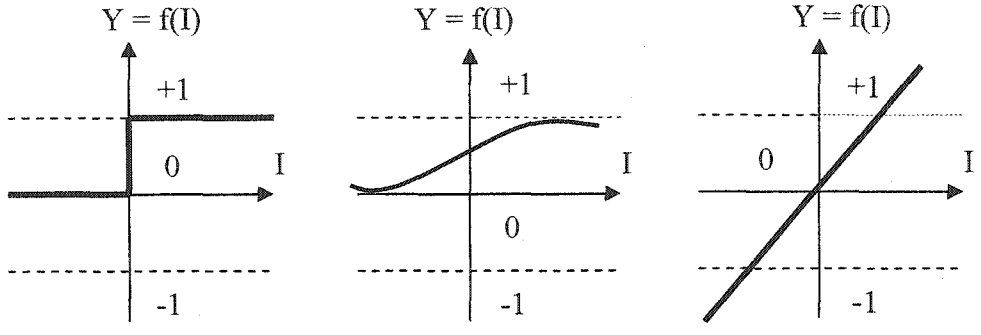


Figure 3 Artificial neuron, a unit of neuron networks

Each neuron receives various signals x_i coming from other neurons or directly from external inputs. These signals are processed within the actual neuron by a weighted summation $I = (\Sigma w_i x_i) + b$ where w_i is a connection weight of the signal x_i, b is called the bias constant of the neurons. The weight coefficient w_i and the bias constant b of a neural network are the degree of freedom, to be determined by scaled conjugate gradient algorithm (see section 3.3.2). This sum is fed through a transfer function to generate a result $Y = f(I)$ for the output path.

Three basic transfer functions are (1) all-or-nothing, (2) sigmoid and (3) linear. Only sigmoid and linear transfer functions are used for our heat exchanger process.



(1) all-or-nothing response (2) sigmoid response (3) linear response

Figure 4 Three basic transfer functions

3.1.2 General structure of artificial neural networks

The developer must go through a period of trial and error in the design decisions before coming up with a satisfactory structure of artificial neural network for a particular system.

In a biological brain, each neuron can connect with up to 200 000 other neurons and the total number of connection is almost infinite. In an artificial neural network however, the connection structure is much simpler by grouping neurons into one input layer, one

or several hidden layers and one output layer. The figure 5 shows a simple structure of a neural network.

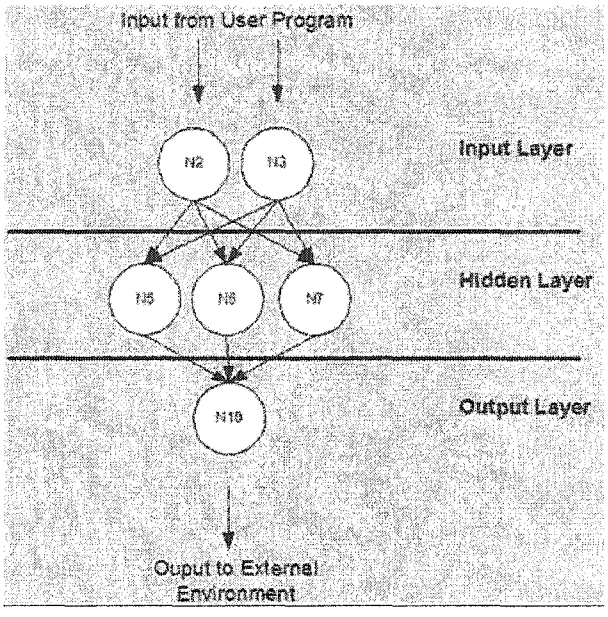


Figure 5 Simple structure of a neural network

The input layer consists of neurons that receive input form the external environment. The output layer consists of neurons that communicate the output of the system to the user or external environment. There are usually a number of hidden layers between the input and output layers; the figure above shows a simple structure with only one hidden layer.

3.1.3 Training a neural network

Similarly to the brain that is trained from experience, each artificial neural network must also be trained before being used. The figure 6 shows a simple diagram for training a neural network. Training a neural network consists of giving several input data sets and the corresponding target output sets that are real output data obtained from experiments. A training method, which consists of a mathematical algorithm, is then

applied to adjust the weight matrices iteration by iteration until the network outputs satisfactory converge to the targets.

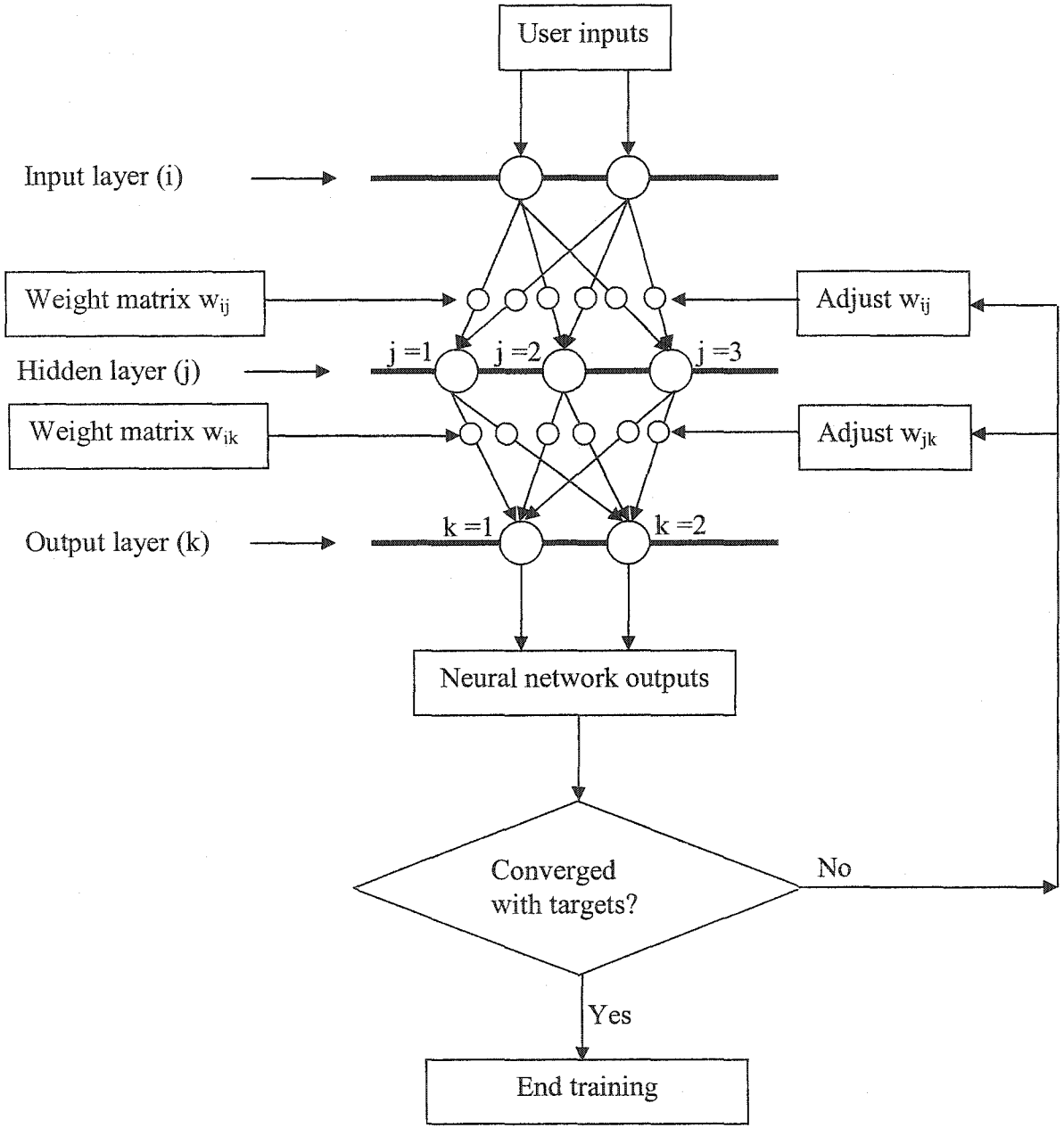


Figure 6 A simple diagram for training a neural network

3.2 Preliminary analyses of the condenser system

The diagram of the condenser system in CTT laboratory of École de Technologie Supérieure has been shown in chapter 2 and reproduced in figure 1 for easy follow up purpose. Some preliminary analyses are needed before constructing appropriate neural networks. These analyses include (1) determination of input and output parameters; (2) choice of experimental data sets; (3) division of data sets into subsets for training, validating and testing the model; and (4) normalization of data.

3.2.1 Input and output parameters

Four input parameters of the actual condenser system and their limit operating are shown in table I.

Table I

Input parameters of condenser system

Input parameters	Minimum and maximum values
Steam valve (VR1-1V)	Always closed (0%)
Steam valve (VR2-1V)	5% - 100% opening
Condensate valve (VR1-C1)	5.2% - 6.5% opening
Inlet temperature of cooling water (CT1-1E)	119^0F - 131^0F

There are also four output parameters which are :

-Steam pressure (indicated CP1-1V)

-Steam temperature (indicated CT2-1V)

-Steam flow rate (shown by the symbol ▶)

-Condensate level (indicated DP1-1V)

3.2.2 Choice of experimental data

During system operation, each control valve (for steam and condensate) can be set up and changed manually or remotely from the computer. Each change of control valve openings takes about 10 minutes for the system to reach a new steady state regime. The transient or steady state regimes are simply observed and judged by watching recorded values appearing on the monitor of the computer.

Several experiments have been carried out on the condenser system, each one lasting for 2 to 6 hours. During each experiment, the opening position of the steam and condensate control valves are randomly changed many times after 5 to 60 minute intervals and data are recorded every 5 seconds for each measured quantity.

A Matlab program is created (see annex 2) to linearize each of four selected output quantities in function of time for every data segment of 50 consecutive sample points (i.e every 250 seconds). The linearized functions are in the form

$$y = At + B$$

where t is the time, y is one of the four output quantities (steam flow, steam temperature, steam pressure and condensate level). A and B are the best fit constants for each data segment. The criteria used for deciding whether a data segment is steady, is the slope of the fitting line must be small :

- Unsteady if $|A| > A_0$
- Steady if $|A| \leq A_0$

where $|A|$ is the absolute value of the slope and A_0 is a given small value (see figure 7). The slope coefficient A_0 that can be considered horizontal (steady) for the four output quantities are presented in the table II

Table II

Maximum slope coefficient for steady data (A_0)

Quantity	A_0
Steam flow	0.04536 (lb/h/s)
Steam temperature	0.007 (K/s)
Steam pressure	0.007 (psia/s)
Condensate level	0.00522 (in/s)

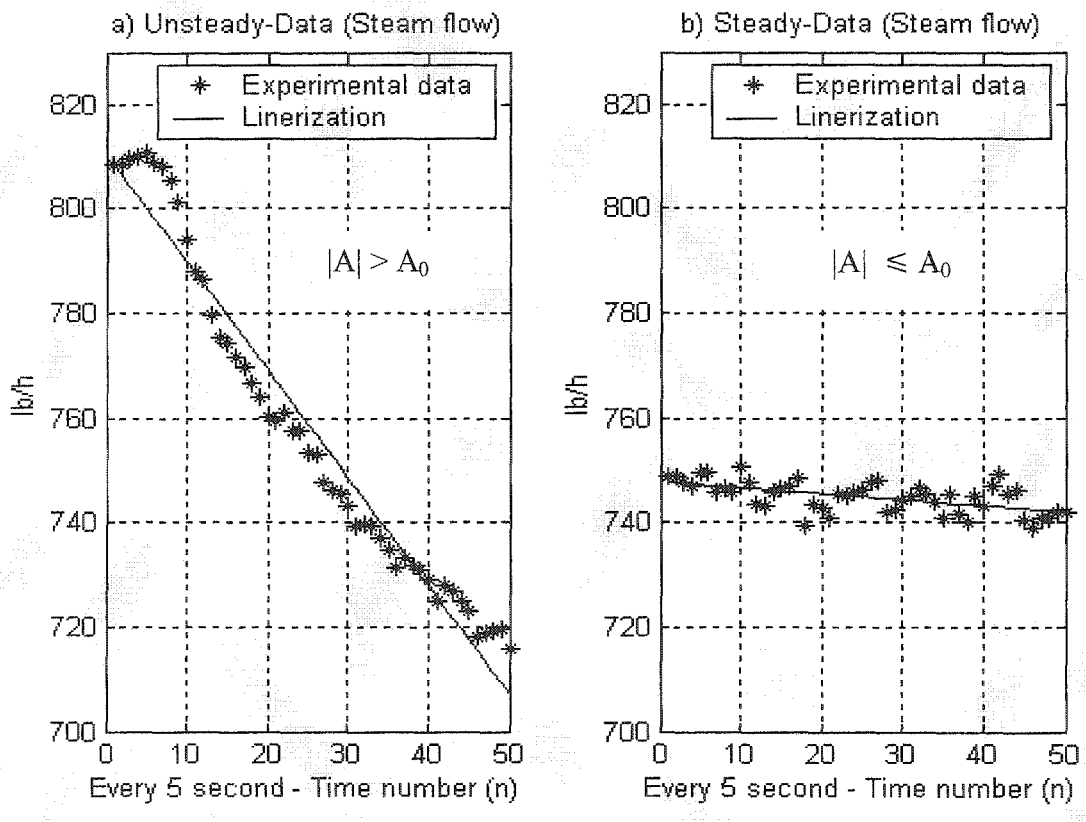


Figure 7 Examples of unsteady and steady data

The selection procedure for steady data is repeated until all experimental data are treated. The figure 8 shows an example of steady and unsteady data determined by the previous procedure applied to the steam flow during a 6-hour experiment.

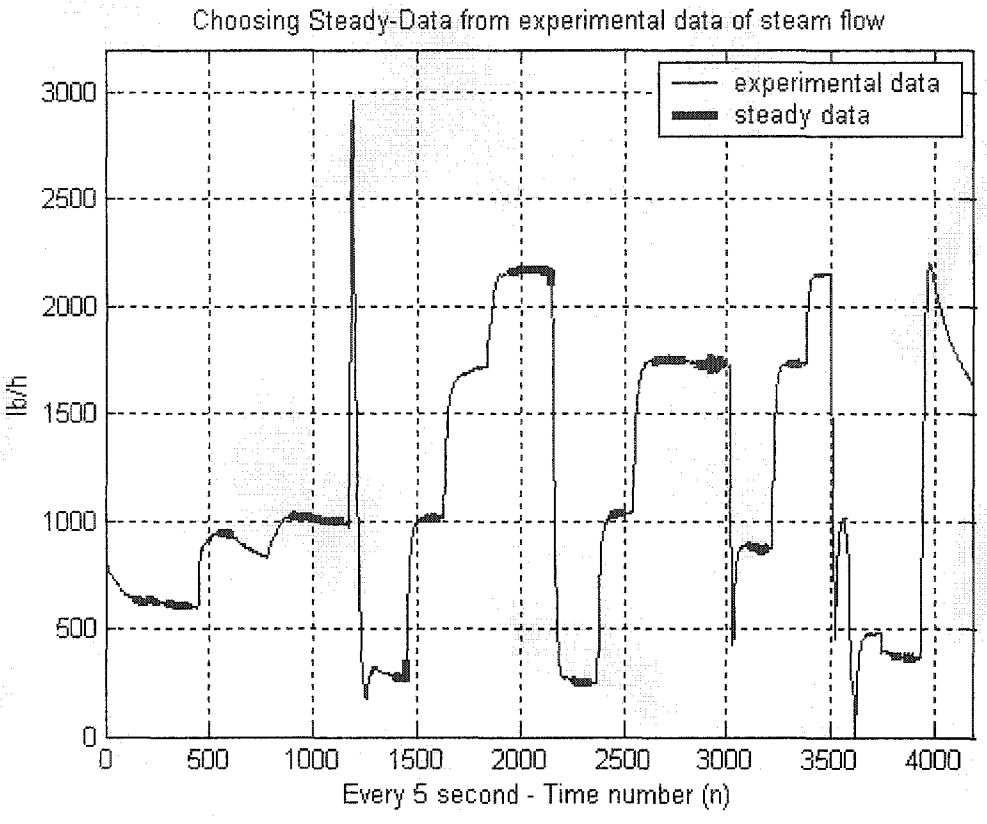


Figure 8 Steady and unsteady data for steam flow during a 6-hour experiment

Since the linearized procedure is independently applied to each one of the four output quantities, their steady state data may not occur at the same periods of time. By comparing four individual steady data sets (steam flow, steam temperature, steam pressure and condensate level), further elimination is finally done so as to keep only the time sets for which all quantities are simultaneously steady.

The total number steady data set is about 1130 which are consecutively renumbered. Some of these sets are shown in table III for three different combinations of steam and

condensate valve openings. Note that the inlet coolant temperature (second column of the table III) shows some fluctuations. However, it is always very small during any time segment of 250 seconds so that is always considered steady.

Table III

Steady state data sample points

Time number	Coolant Temperature (^{0}K)	Steam vavle (%)	Condensate vavle (%)	Steam flow (lb/h)	Steam Temperature (^{0}K)	Steam Pressure (psia)	Condensate level (in)
1	123,079	5	5,5	609,035	325,735	85,754	75,927
2	123,079	5	5,5	608,976	325,764	85,754	75,984
3	123,051	5	5,5	607,623	325,764	85,814	75,984
4	123,022	5	5,5	607,033	325,821	85,806	75,984
5	123,022	5	5,5	609,507	325,879	85,814	75,984
,,,	,,,	,,,	,,,	,,,	,,,	,,,	,,,
,,,	,,,	,,,	,,,	,,,	,,,	,,,	,,,
,,,	,,,	,,,	,,,	,,,	,,,	,,,	,,,
501	121,118	50	5,2	251,798	347,595	119,355	91,81
502	121,061	50	5,2	250,467	347,566	119,302	91,81
503	121,032	50	5,2	251,564	347,595	119,415	91,81
504	120,974	50	5,2	250,155	347,566	119,482	91,81
505	120,945	50	5,2	249,149	347,595	119,482	91,81
,,,	,,,	,,,	,,,	,,,	,,,	,,,	,,,
,,,	,,,	,,,	,,,	,,,	,,,	,,,	,,,
,,,	,,,	,,,	,,,	,,,	,,,	,,,	,,,
1126	119,648	5	5,3	363,35	334,041	97,387	93,987
1127	119,648	5	5,3	363,744	334,041	97,447	93,987
1128	119,648	5	5,3	365,71	334,041	97,447	93,987
1129	119,648	5	5,3	366,7	334,07	97,507	93,987
1130	119,676	5	5,3	365,519	334,07	97,447	93,987

3.2.3 Data division for testing neural network model

It is a common practice to split the available data into three sub-sets for training and validating each neural network model and for comparing different models. The basic idea is to withhold a small 'validation subset' to be used as a convergence criteria for each ANN model; (2) another small 'test subset' to be used to compare different models [4]; (3) the remaining subset for training each model.

In this study, the complete data is splited into three subsets in the following manner : (1) the validation subset takes 25% of the data set and is composed of every fourth line of the overall set, i.e lines 4,8,12,...,1128; (2) the test set takes another 25% of the data set and is composed of lines 2,6,10,...,1130; and (3) the training set takes the remaining 50% of the overall set.

3.2.4 Data normalization

In any model development process, familiarity with the available data is of the utmost importance. ANN is no exception and data pre-processing can have a significant effect on model performance. Thus, before training, it is often to scale the inputs and targets so that they always fall within a specified range. In addition, the variables have to be scaled in such a way as to be commensurate with the limits of the activation functions used in the output layer. In Matlab, there are approaches to normalize original inputs and outputs as Min and Max, Mean and standard deviation and Principal Component Analysis. [6]

The functions **premnmx, postmnmx** in Matlab are used to normalize the original inputs and outputs so that they fall within the [-1,1] range. For each input or output quantity, designated by y, the minimum and maximum values are sorted from the data table (designated by y_{min} and y_{max}) and the normalized value for each sample point is determined by [6] :

$$\bar{y} = 2\frac{(y - y_{min})}{(y_{max} - y_{min})} - 1$$

In the reverse manner, any quantity can be converted back to is ordinary values by :

$$y = 0.5\left(\bar{y} + 1\right)(y_{max} - y_{min}) + y_{min}$$

3.3 Determination of ANN architecture

The architecture of a ANN consists of the number of layers, the number of neurons in each layer, the basic transfer function (sigmoid or linear) and how the layers connect to each other. Generally, the sigmoid type is used for transfer functions of all neurons on hidden layers and the linear type is applied to neurons on the output layer.

3.3.1 General approach

There are two systematic approaches for determining optimal network architecture which are pruning and constructive approaches. [4]

Pruning approach : The basic idea of pruning is to start with a network that is 'large-enough' to capture the desired input-output relationship and to subsequently remove or disable unnecessary neurons

Constructive approach : This approach optimizes the number of hidden layer from the opposite direction to pruning approach, starting with zero hidden layer. Hidden layer, their neurons and connections are then added one at a time in an attempt to improve model performance.

In this project, the constructive approach is selected for it reaches the optimal network with a relatively number of trial and error iterations.

3.3.2 Mathematical training algorithm

In the mathematical point of view, training an neural network (NN) implies determining degree of freedom (D_i) so that the errors between NN outputs and the targets are as small as possible. This is thus an optimization problem. The objective function for optimizing a NN problem is named the performance function p defined by the 'mean square error' of a certain number of a sample points.

$$p([D])=\frac{1}{4Q}\sum_{k=1}^{Q}\sum_{j=1}^{4}\left(\bar{y}_{jk}([D])-\bar{e}_{jk}\right)^2$$

where [D] the degree-of-freedom vector, $\bar{y}_{jk}$ represents normalized output quantity j at a sample point k, (j = 1 for steam flow, j = 2 for steam temperature, j = 3 for steam pressure and j = 4 for condensate level), Q is called 'epoch size', i.e the number of randomly picked sample points within training data subset to be used to update the degree-of-freedom [D] and $\bar{e}_{jk}$ represents the corresponding experimental value (target). If this performance function is applied to all sample points of the validation data subset, the result is denoted by $p_V[D]$ which is useful for cross-validation purposes to be explained later on.

If this performance function is applied to all sample points of the test data subset, the result is denoted p_T which is useful for comparing different NN models.

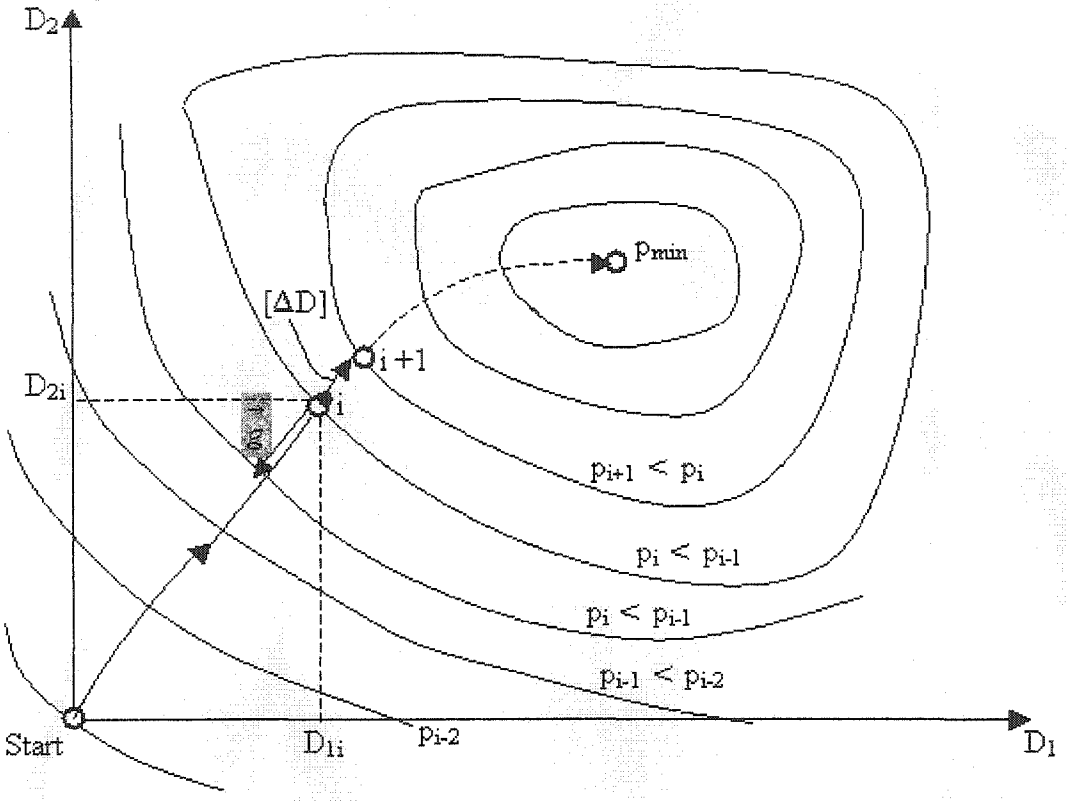


Figure 9 Illustration of optimizing two variables (D_1,D_2)

To start the optimization procedure, an initial point for degree of freedom must be given. In the present study, the first point is set to be all zeros: $[D_0] = [\text{zeros}]$.

At the beginning of iteration i, the degree of freedom $[D_i]$ are all known, and the training algorithm consists of choosing another set of degree of freedom $[D_{i+1}]$ so as give a smaller error :

$$p([D_{i+1}]) < p([D_i])$$

The back-propagation algorithm is the most commonly used for determining the next set $[D_{i+1}]$. This algorithm is shown in figure 10 and explained as follows :

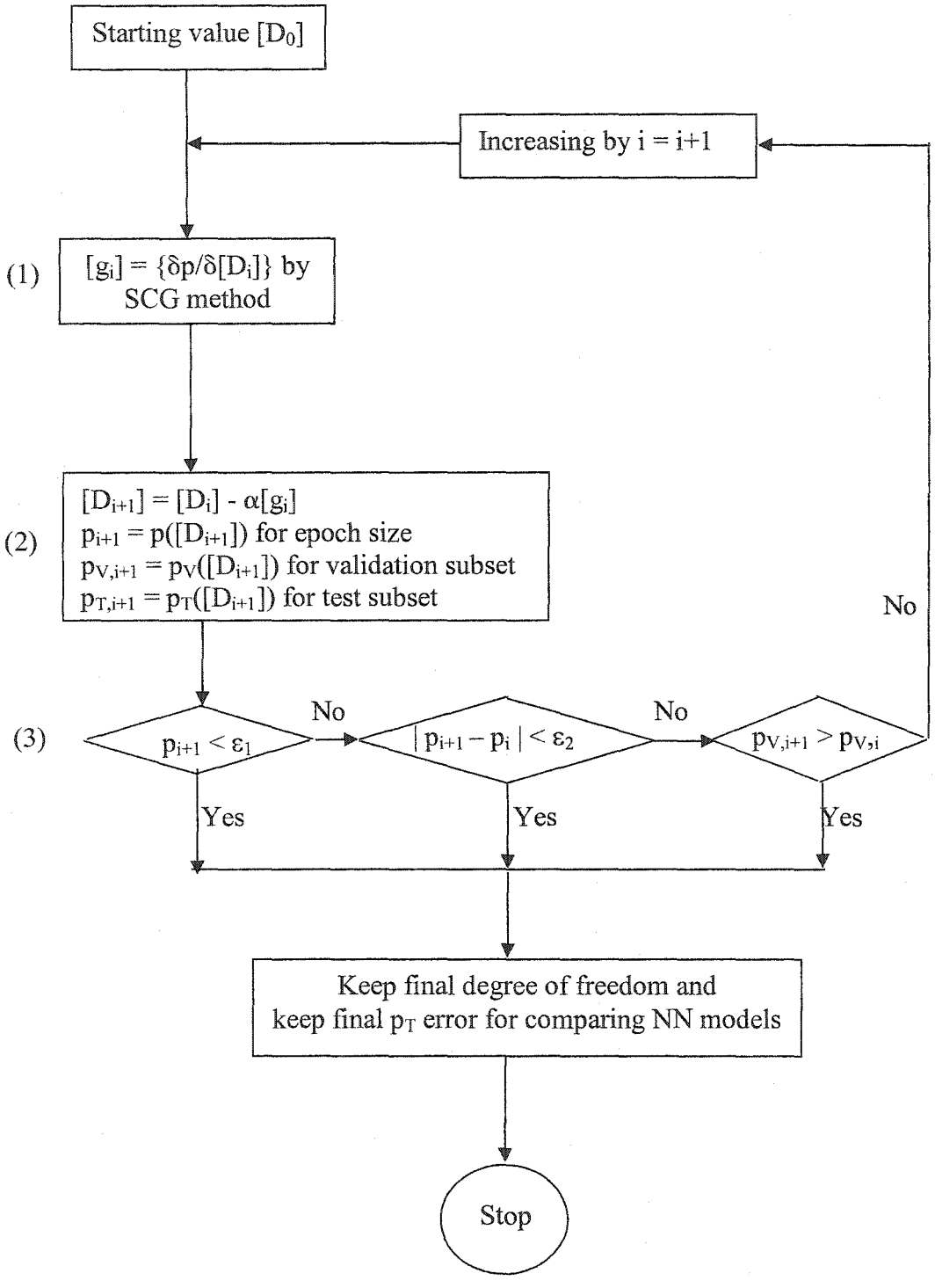


Figure 10 Back-propagation algorithm

(1) Calculating the gradient vector $[g_i]$ (see figure 9) at point $[D_i]$, i.e $[g_i] = \{\delta p/\delta[D_i]\}$ at point $[D_i]$. The present study uses a scaled conjugate gradient (SCG) method developed by Moller [11] to calculate gradient vectors.

(2) Calculating the next point $[D_{i+1}]$ on the negative gradient direction by equation

$$[D_{i+1}] = [D_i] - \alpha[g_i]$$

where α is a constant determining size of the next step ($\alpha = 0.5$ in this study). A smaller the size step gives the better accuracy but takes more iterations and more calculation time (see figure 10).

(3) Testing convergence criteria :

Three convergence criteria are sequentially used in order to stop the iterations.

- when training error has reached sufficiently small value

$p_{i+1} < \varepsilon_1$ (for each epoch size Q = 75 sample points)

- or when training error is not changed significantly :

$| p_{i+1} - p_i | < \varepsilon_2$ (for each epoch size Q = 75 sample points)

- or when the validation error starts to rise

$p_{V,i+1} > p_{V,i}$ (for all validation subset, about 280 sample points), the later is called the 'cross-validation' criteria, which is useful for cases in which the first two criteria can not be met.

3.3.3 Resulting NN for steam condenser system

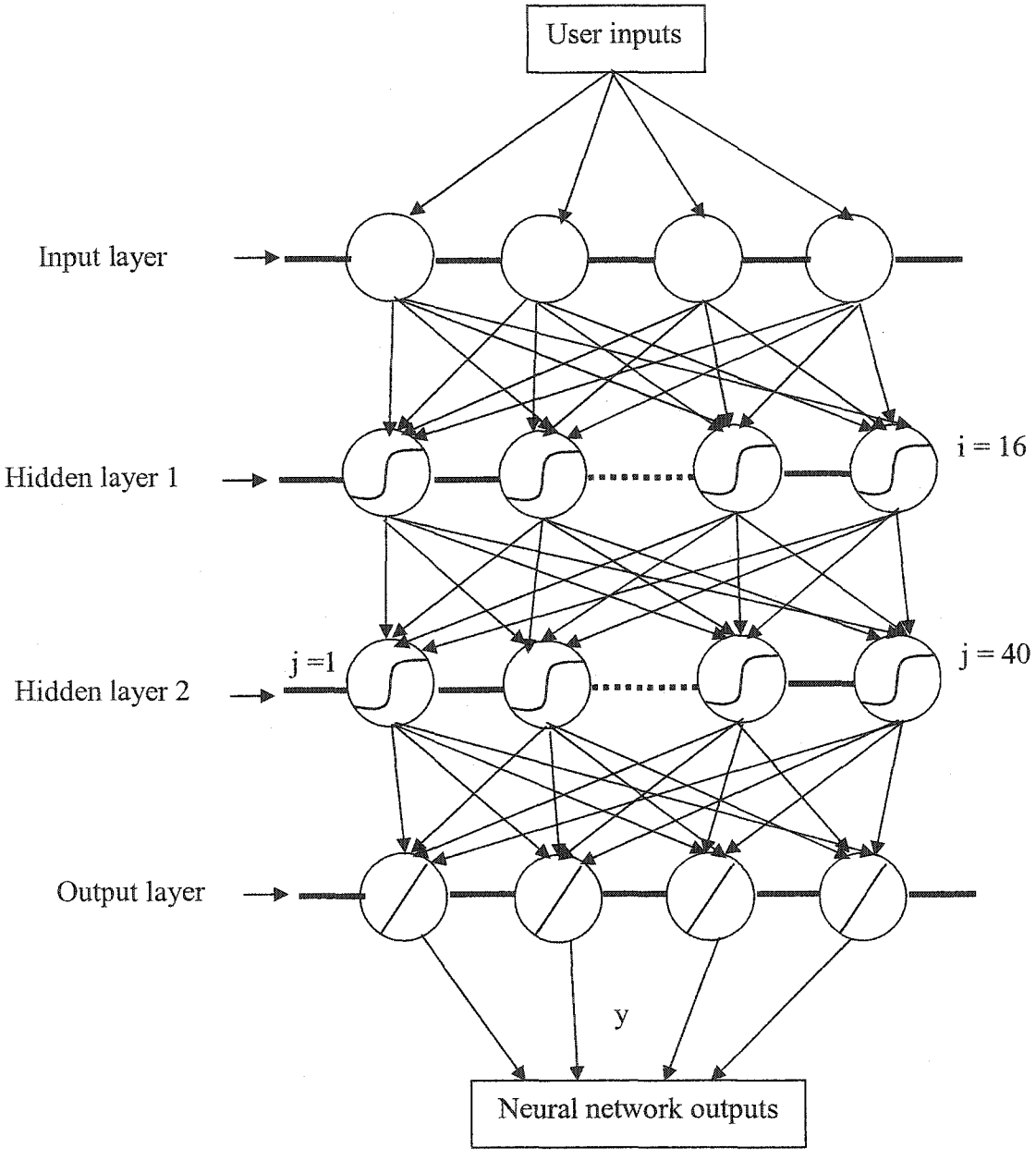


Figure 11 Neural network model of condenser system

A Matlab program is written for training NN models for our steam condenser system (see annex 2). The program contains the function '**trainscg**' already developed (by Matlab)

using the previous described algorithm. It also contains four manually given parameters which are (1) epoch size Q, (2) learning rate α, (3) number of hidden layers and (4) number of neurons in hidden layers.

Each combination of these parameters takes about 1 to 2 hours computer time before meeting the convergence stopping criteria (using a 900Mhz Pentium III computer). This program is applied several times by changing one of these parameters each time before obtaining the most acceptable NN model for the condenser system. The retained NN model is summarized as follows: (see figure 11)

Epoch size Q = 75 sample points which are repeated with the training data set until the convergence criteria are met;

- Learning rate: $\alpha = 0.5$;

- Number of hidden layers: 2;

- Number of neurons: 4 for input layer, 16 for the first hidden layer; 40 for the second hidden layer and 4 for output layers.

- Transfer function: sigmoid function for two hidden layers and linear function for output layer

The best values of weight matrices and bias constants are presented in Annex 1, which comprise a 16x4 weight matrix $[w_1]$ and a 16x1 bias vector $[b_1]$ for information going from the input layer to the first hidden layer, a 40x16 weight matrix $[w_2]$ and a 40x1 bias vector $[b_2]$ by going from the first hidden layer to the second hidden layer, a 4x40 weight matrix $[w_3]$ and a 4x1 bias vector $[b_3]$ by going the last layer to the output layer.

3.4 Application of retained neural network.

All the parameters of the retained NN model (weight matrices w_{ij} and bias vectors b_i) have been kept in text files w_1.txt, w_2.txt, w_3.txt, b_1.txt, b_2.txt, b_3.txt. In order to allows users to predict output results of the condenser system (i.e steam flow, pressure, temperature and condensate height) for a certain given input condition (i.e valve openings), another program is written (see annex 2) to execute following straight forward steps:

(1) Reading files for weight matrices w_i and bias vector b_i.

(2) Reading a set (x) of 4 input quantities to be treated (2 steam valve openings, condensate valve opening and coolant temperature). Each input must fall with the operating range of the system, already mentioned in Table 1.

(3) Calculating output values by transfer functions of the retained NN model shown in figure 11. The four quantities of the output vector y are un-normalized back to their real units (lb/h for steam flow, psia for pressure, Kelvin for temperature and inch for condensate height) and their calculation can be represented by the form

$$y = f_3(\Sigma w_3 * f_2(\Sigma w_2 * f_1(\Sigma w_1 * x + b_1) + b_2) + b_3)$$

where f_1, f_2 and f_3 are transfer functions of the first hidden layer, the second layer and the output layer of the NN model. To implement application example using this program, another experiment has been carried out independently from the training data and the program is applied for calculating predicted outputs for steady data of this experimental outputs and those given by NN model is presented in figure 12 for several input conditions (steam valve opening indicated by SV, condensate valve opening

indicated by CV and the average temperature of cooling water indicated by T_m). The results show good agreement between computed and measured values.

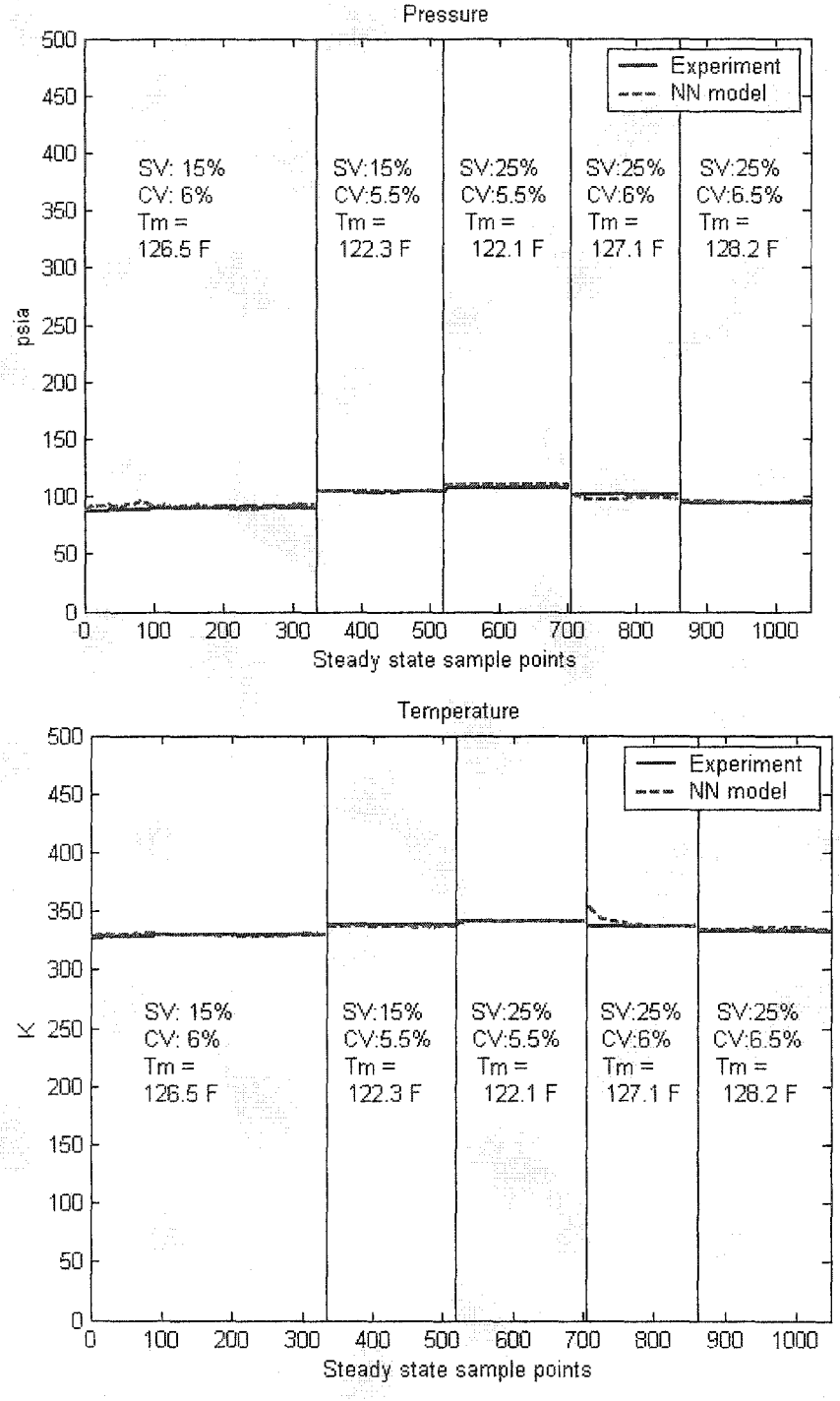


Figure 12(a) Results for steam pressure and temperature

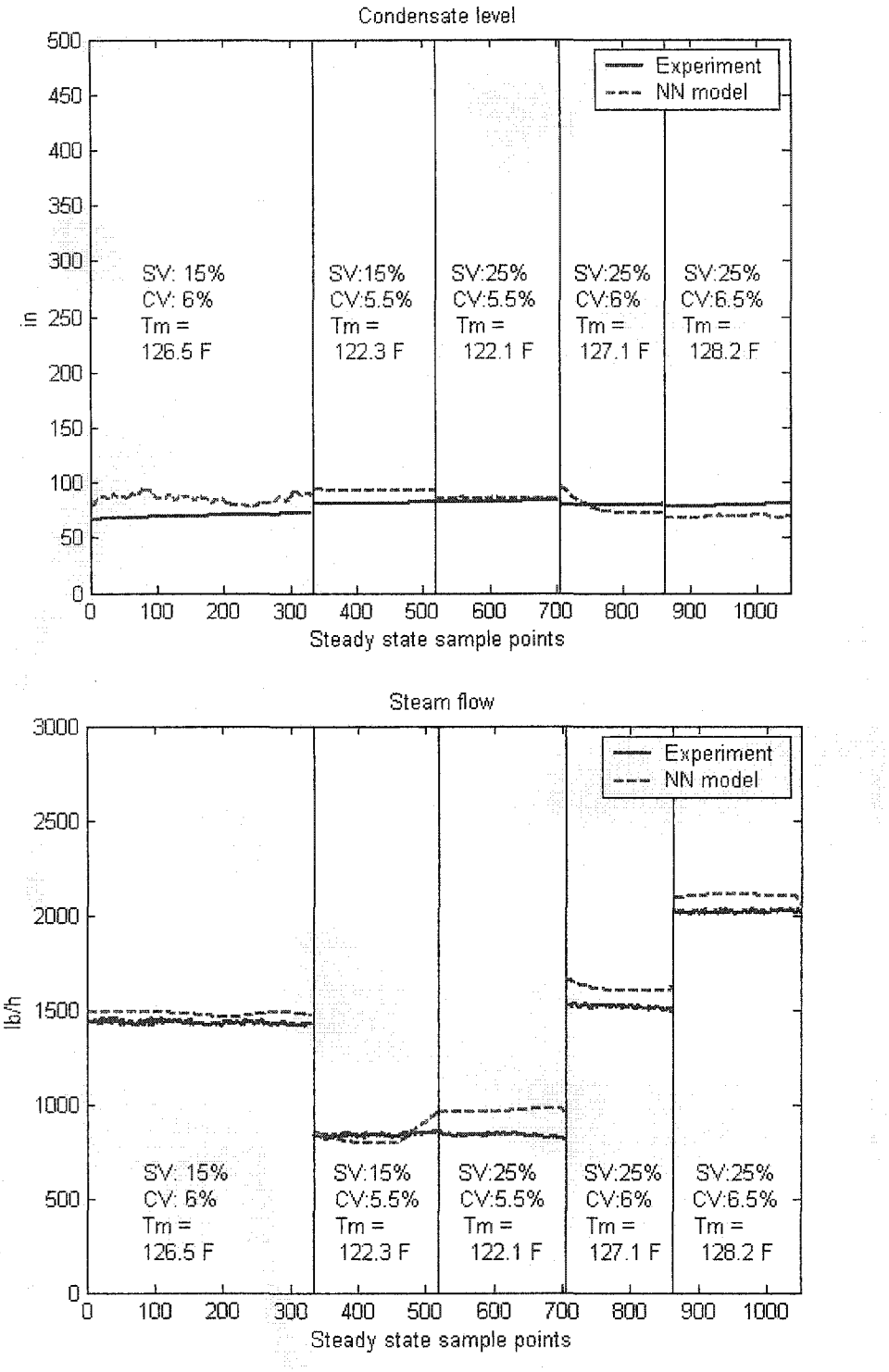


Figure 12(b) Results for condensate level and steam flow

3.5 Conclusion

According to the results shown in the previous numerical application, it is seen that the artificial neural network (ANN) model presented in this study performs successfully for the condenser system. The main advantage of the ANN approach is the automatic selection for the algorithm that relates the inputs to the outputs. Also, the approach could adapt itself to assemble the desired output with a training input sample. It allows multiple input versus multiple output relationship. The main drawback of the ANN model is that it can not explain how the quantities are related to each other

A way of overcoming this drawback is to develop a physical model for the condenser system, which is the subject of the next chapter.

CHAPTER 4

A PHYSICAL MODEL FOR THE STEAM CONDENSER SYSTEM

4.1 Objective

A physical model is herein developed for the same steam condenser system previously described. This approach is based on the conservation of mass and energy. The physical model has the same objective as the NN model, which is to predict output quantities in function of inputs. It differs from the NN model by several aspects including the following

(1) The physical model describes the response of the system by using equations derived from physical laws.

(2) The physical model directly takes real output quantities such as pressure, temperature, heat transfer, mass flow, etc... as degrees of freedom (DOF), which NN model use meaningless quantities such as weight coefficients, bias constants or transfer functions.

(3) The physical model dynamically describes the system from an initial condition through out transient states until a new steady state condition is reached, while the NN model just treats steady state conditions

(4) The number of DOF of the physical model depends on the number of discretion 'control volumes', such as described in the next section.

(5) The coolant water flow rate, $\dot{m}_w$ is not considered in NN model for it is practically unchanged during all operating conditions but is now considered in physical model for it simply appears in conservation equations.

(6) The physical model takes saturated steam conditions into account so that the steam pressure P_s and temperature T_s are related to each other by the saturated steam table, while NN model treats them like two independent quantities.

4.2 Modelling

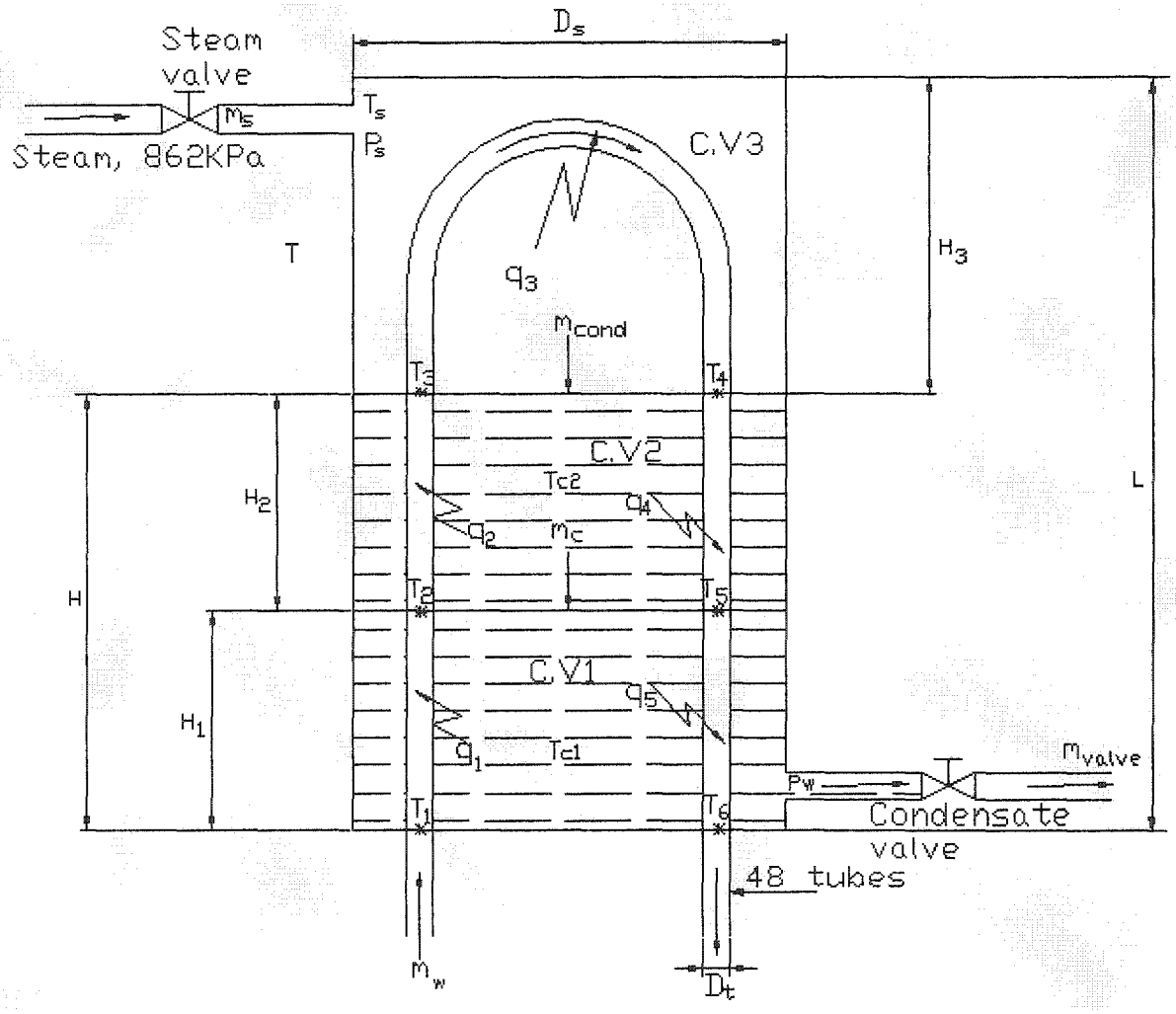


Figure 13 Discretion of control volumes of condenser system

The condenser system is divided into control volumes (C.V) and equations are developed for physical quantities at boundaries of these volumes. In general, increasing the number of CVs gives better results but involves more equations to be developed and more calculation time. In the present study, condensate volume is divided into two equal control volumes (C.V1 and C.V2) and the steam above the condensate is taken as one control volume (C.V3 in figure 13). There are thus five corresponding CVs on tube side denoted by six boundary temperatures T_1 to T_6 in figure 13.

In developing mathematical equations for this model, the following parameters are needed, some of them being constant and the other being variables in function of time.

The constant parameters are composed of geometric quantities and input quantities of the system. The geometric quantities are shell side and tube side dimensions such as inside diameter and total effective length of shell (D_s and L in figure 13), number of tubes (N = 48) and their inside and outside diameters (D_t and D_{to}). The input parameters are defined as the physical quantities that can be directly adjusted. In this model, the input quantities are :

- K_s...steam valve proportionality constant corresponding to a valve opening, ($kg/s/Pa^{1/2}$)
- K_c...condensate valve proportionality constant corresponding to a valve opening,
- T_1...inlet coolant temperature, and
- $\dot{m}_w$ for flow rate of cooling water on tube side.

The variable parameters in function of time include all quantities that are not directly adjusted but related input quantities via physical laws. They are :

- $\dot{m}_s$ for incoming steam flow rate at top of shell, (kg/s), the dot symbol above letter m being used to designated the derivative dm/dt with respect to time,

- P_s and T_s for saturated steam pressure (Pa) and temperature (^{0}K) in C.V3,

- $\dot{m}_{cond}$ and $\dot{m}_c$ for condensation rate at boundary C.V3 to C.V2 and intermediate flow rate from C.V2 to C.V1, respectively

- $\dot{m}_{valve}$ for outflow of condensate at bottom of shell

- T_{c1} and T_{c2} for average temperatures of condensate in C.V1 and C.V2, respectively

- H_1, H_2 and H_3 for heights of corresponding control volumes (Noted that $H_1 + H_2 + H_3 = L$ is constant, and $H_1 = H_2 = H/2$, half of condensate height, variable with time).

- T_1, T_2,..., T_6 for temperature of cooling water inside the reversed U tubes at boundary levels of control volumes

- q_1, q_2 , ..., q_5 for heat transfer rates from shell side CVs into tube side CVs (see figure ...)

- Physical properties such as enthalpies of steam and condensate (h_s, h_{cond}, ...W/m^2/K), specific heats (C_p, J/kg/K), and densities of steam, condensate and water (ρ, kg/m^3).

4.3 Mathematical equations of the model

The mathematical equations of the previously described model are presented in this section under 5 categories : - continuity equations (or conservation of mass) in control volumes of shell side; - energy equations in CVs of shell side; - energy equations in CVs of tube side; -heat transfer equations across tube wall; and miscellaneous relationships.

4.3.1 Continuity equations

The general principle of the continuity of mass in a control volume states that :

$$\begin{pmatrix}\text{Rate of input}\\ \text{mass}\end{pmatrix}-\begin{pmatrix}\text{Rate of output}\\ \text{mass}\end{pmatrix}=\begin{pmatrix}\text{Rate of accumulation}\\ \text{mass}\end{pmatrix}$$

For each of two control volumes of condensates (C.V1 and C.V2), this principle gives following equations.

$$\dot{m}_c - \dot{m}_{valve} = \frac{\rho_1 A dH_1}{dt} \tag{4.1}$$

$$\dot{m}_{cond} - \dot{m}_c = \frac{\rho_2 A dH_2}{dt} \tag{4.2}$$

where $A = \pi\frac{D_s^2}{4} - N * \pi\frac{D_{to}^2}{4}$ which means the net cross-section area of the shell side

For control volume 3 (C.V3), the fluid is saturated steam, so that the continuity equation of saturated steam must be used [14], which is

$$\dot{m}_s - \dot{m}_{cond} = \frac{Ad(P_s H_3)}{dt} \qquad (4.3)$$

On the tube side, the cooling water flow rate ($\dot{m}_w$) is practically unchanged for all operating conditions. Furthermore, water is treated as incompressible so that it is reasonable to assume that this flow rate is constant over the entire volume on tube side. This value is $\dot{m}_w = 0.65$ kg/s which is determined from experimental data and applies for all test conditions in this study.

4.3.2 Energy equations of control volumes on shell side

The general principle of the energy balance in a control volume states that (see figure 14)

$$\begin{pmatrix}\text{rate of}\\ \text{inflow enthalpy}\end{pmatrix} - \begin{pmatrix}\text{rate of}\\ \text{outflow enthalpy}\end{pmatrix} - \begin{pmatrix}\text{rate of heat loss}\\ \text{to tube side}\end{pmatrix} = \begin{pmatrix}\text{rate of}\\ \text{stored energy}\end{pmatrix}$$

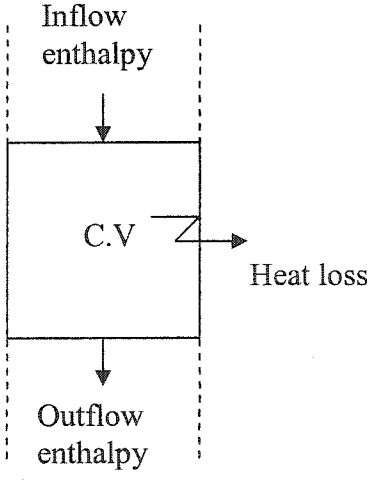


Figure 14 Typical control volume on shell side.

For control volumes 1 and 2 in which the following mater is condensate, this principle gives two following equations :

$$\dot{m}_c * h_c - \dot{m}_{valve} * h_{valve} - q_1 - q_5 - C_{p1} T_{c1} * (\dot{m}_c - \dot{m}_{valve}) = \frac{\rho_1 A H_1 C_{p1} d(T_{c1})}{dt} \quad (4.4)$$

$$\dot{m}_c * h_c - \dot{m}_{valve} * h_{valve} - q_2 - q_4 - C_{p2} T_{c2} * (\dot{m}_{cond} - \dot{m}_c) = \frac{\rho_2 A H_2 C_{p2} d(T_{c2})}{dt} \quad (4.5)$$

For control volume 3 (saturated steam), a simpler energy equation is obtained by using the heat of vaporization h_{fg} (J/kg), property of saturated steam, an intrinsic [13] :

$$\dot{m}_{cond} h_{fg} - q_3 = 0 \quad (4.6)$$

4.3.3 Energy equations of control volumes on tube side

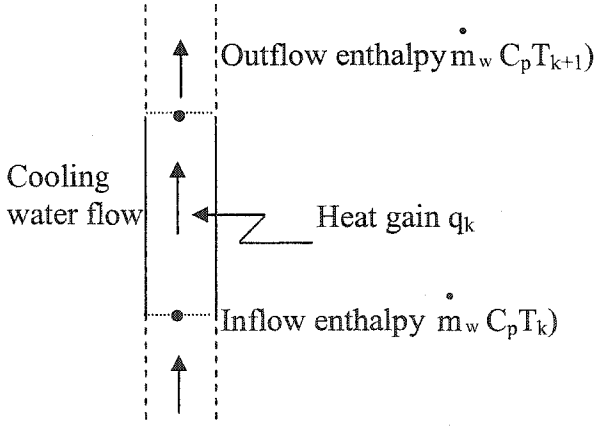


Figure 15 Typical control volume k on tube side (k = 1, to 5)

Applying the general principle of energy balance for control volumes on tube side gives much simpler equations because the tube side flow is considered constant (steady). The five equations for control volumes on tube side are represented by :

$$q_k = \dot{m}_w C_p (T_{k+1} - T_k), \; k = 1, 2, \ldots, 5. \qquad (4.7 \text{ to } 4.11)$$

Note that for simplification purposes, the specific heat of cooling water (C_p) is considered constant in each control volume.

4.3.4 Heat transfer equations across tube walls

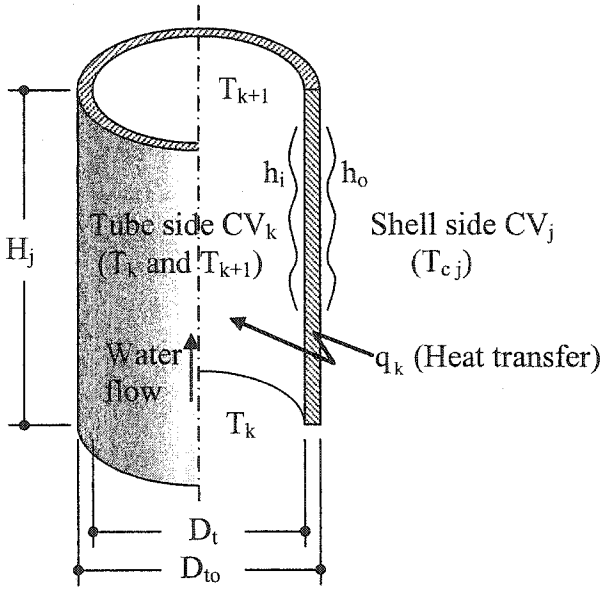


Figure 16 Heat transfer across wall from shell CV_j to tube CV_k

The previous equations are not sufficient to describe the relationships between shell side and tube side because the heat transfer coefficients across tube wall are not taken into account. This section completes this kind of relationships.

The figure 16 shows the main parameters involved in the heat transfer phenomenon across tube walls between a tube control volumes (CV_k with k =1, 2, ..., 5) and the corresponding shell control volume (CV_j with j =1, 2 or 3).

Applying the 'log-mean temperature difference' heat transfer equation developed for shell-and-tube heat exchanger [13], gives five equations of the same following form.

$$q_k = UA_j \frac{T_{k+1} - T_k}{\ln \frac{T_{cj} - T_k}{T_{cj} - T_{k+1}}} \qquad (4.12 \text{ to } 4.16)$$

where k designates tube control volume number (k = 1, 2, ..., 5) and j designates corresponding shell control volume (for example, if tube side k =1 then shell side j = 1, if tube side k = 4 then shell side j = 2, etc..., see model shown in figure 12), and UA_j is 'the overall heat transfer coefficient' across 48 tube walls in the control volume CV_j.

The overall heat transfer coefficient (UA_j) depends on the number of tubes (N=48), tube length in the control volume (H_j), tube diameters (D_t and D_{to}), convection coefficients h_o and h_i at outside and inside of tube wall, ($W/m^2/^0K$) and thermal conductivity K of tube material ($W/m/^0K$).

More precisely [13] :

$$UA_1 = \frac{N}{\frac{1}{h_{o1}\pi D_{to} H_1} + \frac{\ln(D_{to}/D_t)}{2\pi K H_1} + \frac{1}{h_{i1}\pi D_t H_1}}$$

$$UA_2 = \frac{N}{\frac{1}{h_{o2}\pi D_{to} H_2} + \frac{\ln(D_{to}/D_t)}{2\pi K H_2} + \frac{1}{h_{i2}\pi D_t H_2}}$$

$$UA_3 = \frac{N}{0 + \frac{\ln(D_{to}/D_t)}{2\pi K(2H_3)} + \frac{1}{h_{i3}\pi D_t(2H_3)}}$$

It is noted that in the expression of coefficient UA_3 for shell control volume 3, the convection h_{o3} due to steam condensation on outside tube wall is very large so that the ratio $1/h_{o3}$ is neglected, and the tube length is approximately equal to twice the height H_3 due to U bend.

4.3.5 Miscellaneous relationships

To complete the required number of equations for the unknown parameters in above equations, following miscellaneous equations are needed.

- Relationship between water flow and convection coefficient h_i inside tubes [13]: This relationship has two forms depending on whether the flow is laminar or turbulent, indicated by the Reynolds number Re_k :

$$Re_k = \frac{4 * \dot{m}_w}{\pi * D_t * \mu_k}$$

where μ_k is the absolute viscosity ($N.s/m^2$) of cooling water inside CV_k,

- If $Re_k < 2300$: the flow is laminar, then $\boxed{h_{ik} = 3.66K/D_t}$

- If $Re_k > 2300$: the flow is turbulent, then $h_{ik} = \frac{0.023K Re_k^{(4/5)} Pr^{0.4}}{D_t}$

where Pr is the Prandtl number of the cooling water at the corresponding temperature

- Relationship between condensate flow and convection outside tube [13]: The Reynolds number of condensate flow outside the tubes is calculated by

$$Re_j = \frac{4 * \dot{m}_j * D_h}{\pi * (D_s^2 - ND_{to}^2) * \mu_j} \quad j = 1 \text{or } 2, \text{ shell } CV_j$$

where $\dot{m}_j$ is the average condensate flow rate in the control volume j; and D_h is the 'hydraulic diameter' of shell side, defined by

$$D_h = \frac{D_s^2 - ND_{to}^2}{D_s + ND_{to}}$$

The condensate flow is always laminar, the convection coefficient h_o is given by

$$h_{oj} = (0.664 * Re_j^{0.5} * Pr_j^{(1/3)})/D_h$$

- Relationship between saturated pressure and temperature

For saturated steam, the pressure P_s and the temperature T_s are dependent to each other. This relationship can be represented by the general form.

$$T_s = f(P_s) \tag{4.17}$$

However, this function is practically replaced by a steam table.

- Height relationships

$$H_1 = H/2 \tag{4.18}$$

$$H_2 = H/2 \tag{4.19}$$

$$H_3 = L - H \tag{4.20}$$

- Characteristics equations of condensate valve and steam valve

The relationships between a valve opening condition (e.g 5.5% steam valve with 15% condensate valve) and other parameters of the model (P_o, P_s, P_c, K_s and K_c) are introduced under the characteristics equations of valves as follows :

$$\text{Steam valve : } \dot{m}_s = K_s\sqrt{P_o - P_s} \tag{4.21}$$

$$\text{Condensate valve : } \dot{m}_{valve} = K_c\sqrt{P_c - P_{atm}} \tag{4.22}$$

where P_o is the pressure of steam source outside the condenser system (P_o = 862kPa, constant), P_{atm} is the atmospheric pressure (the condensate tank is open to atmosphere), and P_c is the pressure of condensate at bottom of heat exchanger which is slightly higher than the steam pressure by the height of condensate :

$$P_c = P_s + (\rho_1 H_1 + \rho_2 H_2)g$$

K_s and K_c are the valve coefficient (kg/s/$Pa^{1/2}$), experimentally determined

The numerical values of these coefficients for nine combinations of valve openings are shown in table IV.

Table IV

Valve coefficients (K_s and K_c) corresponding to opening conditions

Steam valve / Condensate valve	15%	25%	50%
6.5%	K_s =12.18 K_c =10.01	K_s =17.45 K_c =9.9	K_s =23.45 K_c =10.2
6%	K_s =11.52 K_c =7.24	K_s =14.98 K_c =7.26	K_s =17.49 K_c =7.48
5.5%	K_s =8.72 K_c =3.93	K_s =9.625 K_c =3.85	K_s =10.49 K_c =3.92

4.4 Equation system solving

The previous section gives a set 22 differential-algebraic equations (DAEs), five of which are ordinary differential equations for the parameters T_{c1}, T_{c2}, P_s, H_1 and H_2. (see equations (4.1) to (4.5) and the remaining are algebraic equations). These equations are sufficient for determining 22 unknowns, which are 4 flow rates ($\dot{m}_s$, $\dot{m}_{cond}$, $\dot{m}_c$ and $\dot{m}_{valve}$), 8 temperatures (T_s, T_{c1}, T_{c2}, T_2 to T_6), 2 pressures (P_s and P_c), 5 heat transfer rates (q_1 to q_5) and 3 heights (H_1, H_2 and H_3).

A program has been written to solve this DAE system using the function '**ode15s**' of Matlab (see annex 2). This function already includes an algorithm for automatically adjusting the time interval Δt so as to give satisfactory convergence from an initial condition until the corresponding state condition is reached.

Using this program implies simply giving an input condition (K_s, K_c, T_1 and $\dot{m}_w$) and initial values for five parameters governed by differential equations which are T_{c1o}, T_{c2o}, P_{so}, H_{1o} and H_{2o}.

4.5 Model validation

Two numerical examples using the previous physical model are presented below to show predicted results in comparison with experiments during transient states. The numerical values of input parameters and initial condition are summarized in the table V for two different examples.

Table V

Data of two numerical examples

	Example 1	**Example 2**
Input condition	- $K_s = 0.012$; $K_c = 0.01$; - $T_1 = 325^0K$; - $\dot{m}_w = 0.65$ (kg/s)	- $K_s = 0.0175$; $K_c = 0.0075$; - $T_1 = 324^0K$; - $\dot{m}_w = 0.65$ (kg/s)
Initial condition	- $P_s = 69.8$ psia; - $T_{c1} = 330^oK$; $T_{c2} = 360^oK$; - $H_1 = H_2 = 0.5715$ m	- $P_s = 95.67$ psia; - $T_{c1} = 330^oK$; $T_{c2} = 360^oK$; - $H_1 = H_2 = 1.04$ m

The results for steam pressure P_s and condensate level H in function of time are shown in figures 17 and 18 for numerical examples 1 and 2 respectively. The solid lines in these figures represent results given by physical model (simulation) and the dashed lines represent experimental data.

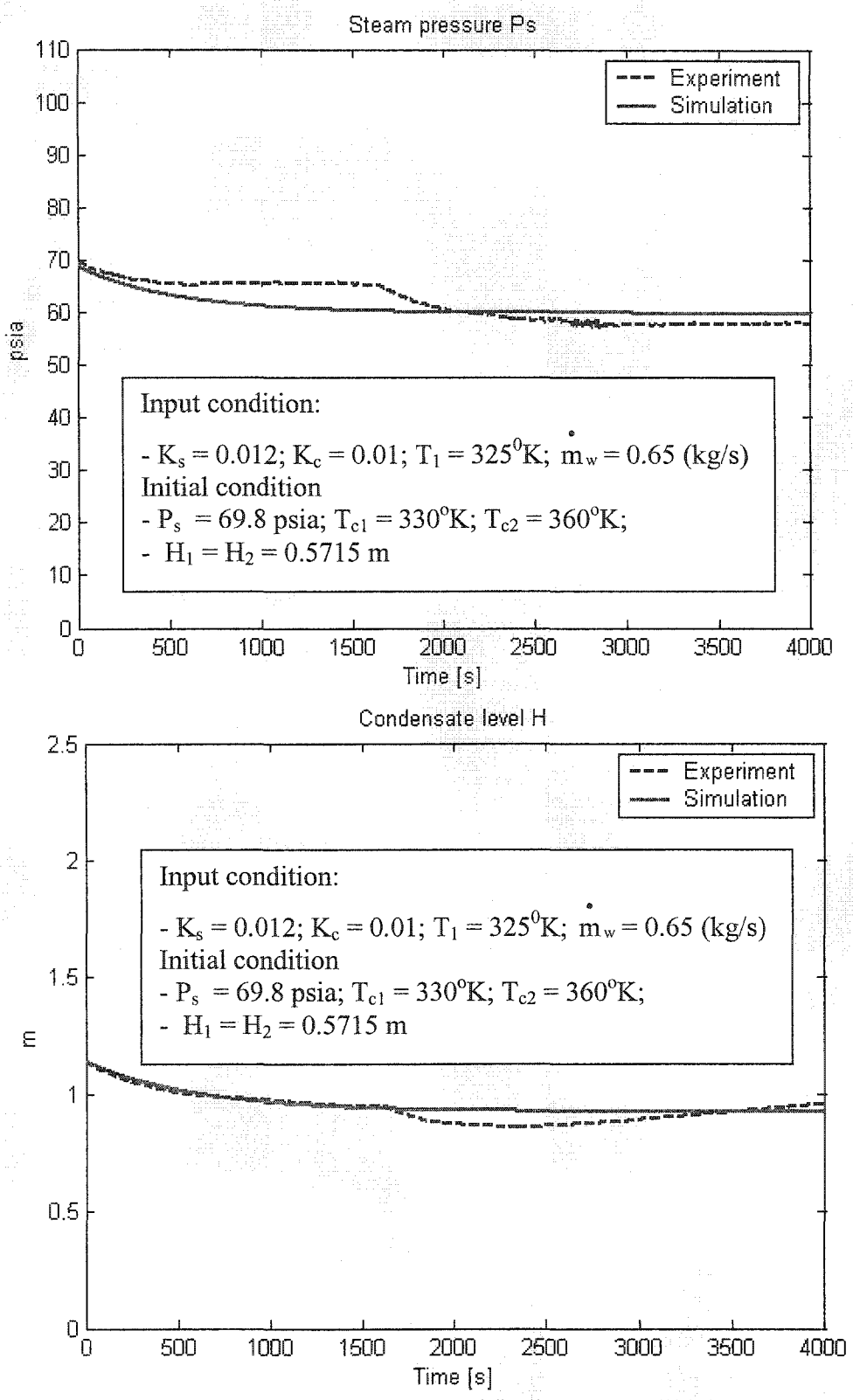


Figure 17 Representative results of numerical example 1

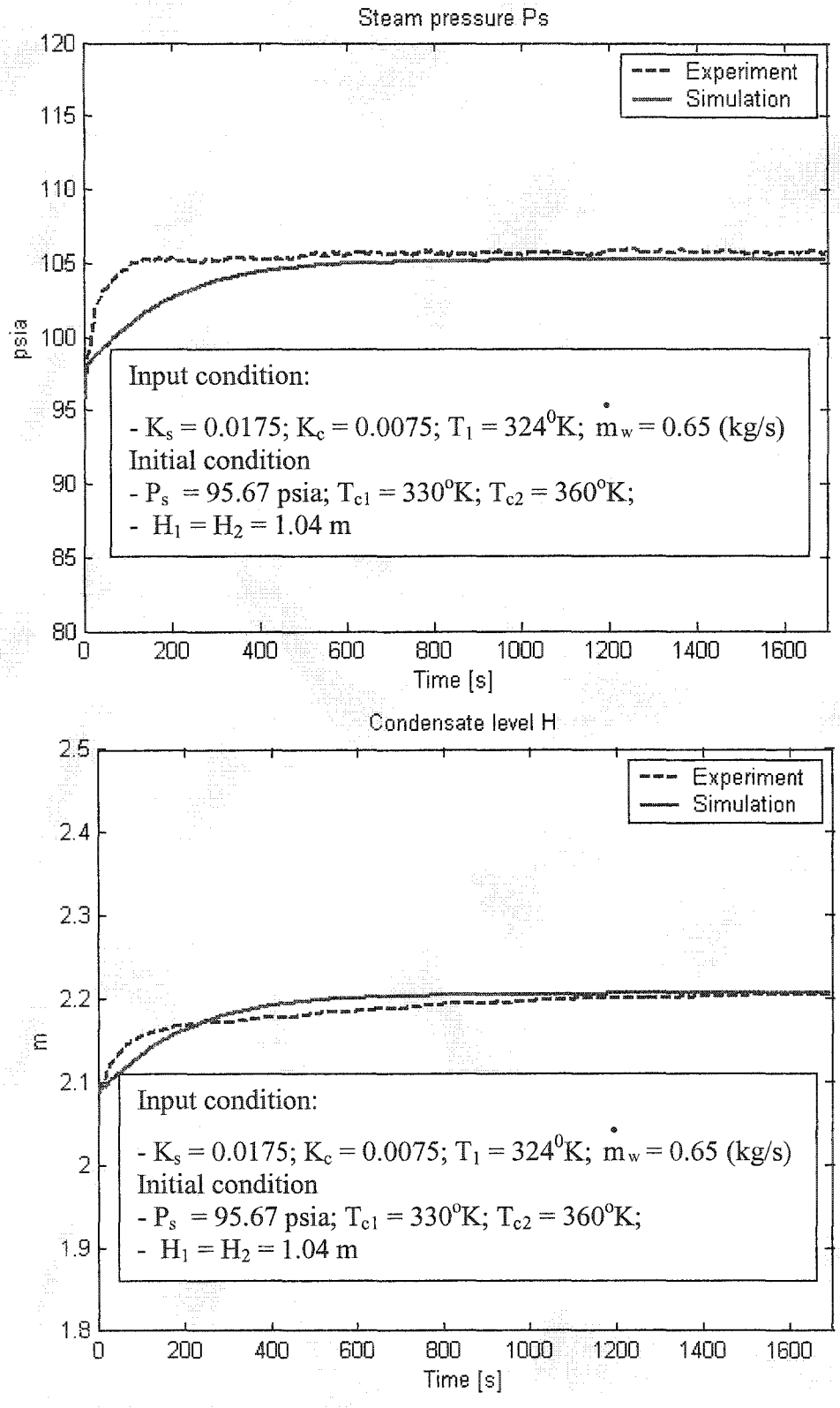


Figure 18 Representative results of numerical example 2

These figures show good agreement between simulated and experimental results. However, it appears that the condensate height is better predicted than the steam pressure. It can be explained by the fact that the condensate is almost incompressible and has much inertia against changes, while steam is more sensitive to external changes such as the action of opening the valve.

CHAPTER 5

CONCLUSION

The steam condenser system in the CTT (Centre de technologie thermique) of École de technologie supérieure has been successfully simulated using ANN (artificial neural network). The optimal NN model has a 4-neuron input layer, two hidden layers with 16 neurons in the 1st and 40 neurons in the 2nd, and a 4-neuron output layer.

Although better and more complicated NN models may be obtained if more time is spent for examining other cases with smaller trial-and-error increments of parameters. However, the model found in this study gives already satisfactory results.

The main drawback of the NN model is that it just applies for prediction of steady state conditions and it does not explain how the parameters of the system are related together.

The steam condenser system has also been successfully simulated using physical laws applied to control volumes. A model of 3 control volumes on shell side and 5 corresponding control volumes on tube side yields 22 DAEs (differential-algebraic equations) which give good predicted response of the system in the dynamic regimes in comparison with experiments.

The mathematical model presents a great advantage over the NN model by two aspects, the first being the capability of describing the dynamic response at different locations of the system, and the second being the meaningful relationships between parameters. These relationships allow some judgements about the rationality of the response. These judgements may be valuable in the application of the simulation to specific purposes, such as calibrating measure apparatus, optimizing a control system, etc.

Suggestion for future work : Design of a multivariable control system to regulate the condensate level and steam pressure using the physical model developed in this study.

ANNEX 1

VALUES OF THE WEIGHT MATRICES w_1, w_2, w_3 AND BIAS b_1, b_2, b_3

Weight coefficients $w_1(j,i)$ and bias constant $b_1(j)$ for 16 neurons of first hidden layer

(i = input neuron; j = 1st hidden layer neuron)

j \ i	1	2	3	4	
	$w_1(j,i)$				b_1
1	-1,82	-1,97	0,36	0,91	2,82
2	-0,74	-2,05	0,64	1,06	2,84
3	1,74	0,28	1,33	-0,03	-2,35
4	0,70	-2,65	-0,19	-0,17	-1,74
5	-1,71	-0,21	1,61	-0,73	1,12
6	0,31	-1,76	-0,29	-2,09	-0,97
7	0,09	1,60	-1,52	-1,64	0,45
8	1,67	0,57	1,39	-1,70	0,77
9	-0,99	-2,05	0,45	-1,59	-0,03
10	-2,30	1,79	0,36	-0,61	-0,47
11	0,00	1,24	1,50	1,85	-0,70
12	2,01	1,58	1,75	-0,75	1,61
13	2,02	1,18	1,44	-1,55	2,01
14	0,45	-0,59	2,28	1,86	2,56
15	0,15	-0,65	2,02	-1,35	2,49
16	2,61	0,04	-0,05	0,78	2,88

Weight coefficients $w_2(k,j)$ and bias constant $b_2(k)$ for 40 neurons of 2nd hidden layer

(j = 1st hidden layer neuron; k = 2nd hidden layer neuron)

j / k	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	
	$w_2(k,j)$																b_2
1	-0,05	-0,02	-0,43	-0,12	-1,22	-0,48	0,02	-0,54	0,69	0,00	-0,49	0,30	-0,08	-0,07	-0,46	-0,62	-1,95
2	0,13	-0,42	0,67	0,44	-0,15	0,45	0,23	-0,52	-0,56	-0,10	0,29	0,64	-0,39	0,66	0,23	-0,17	-1,78
3	-0,39	0,41	-0,55	0,76	0,12	-0,17	-0,26	0,21	0,05	0,07	-0,51	-0,15	-0,03	-0,75	-0,57	-0,21	1,59
4	-0,40	-0,56	-0,20	-0,60	0,65	0,70	-0,01	-0,11	0,04	-0,32	-0,02	0,08	-0,25	0,73	-0,03	-0,86	1,55
5	-0,15	-0,23	-0,50	-0,61	0,45	-0,75	-0,40	0,39	-0,27	1,14	0,77	0,76	0,57	0,15	-0,08	-0,30	1,43
6	-0,12	-0,39	-0,59	-0,47	-0,29	-0,50	0,55	0,50	0,07	-0,09	0,54	0,55	-0,28	0,96	0,28	-0,28	1,48
7	0,31	-0,04	0,26	-0,77	-0,27	0,01	0,25	0,31	-0,70	-0,09	0,48	-0,66	0,24	0,37	0,22	0,77	-1,13
8	0,77	-0,25	0,15	-0,29	-0,43	-0,71	-0,01	-0,16	0,06	0,61	0,51	0,15	0,53	-0,25	-0,39	0,61	-1,04
9	-0,04	0,68	0,47	-0,73	-0,29	-0,11	0,27	0,04	0,69	0,09	0,58	0,62	0,00	-0,84	-0,08	-0,20	1,20
10	0,12	0,63	-0,49	0,53	-0,18	-0,43	0,01	0,09	-0,62	-0,64	0,00	-0,09	-0,23	-0,04	-0,84	-0,43	-0,98
11	-0,29	-0,27	0,25	0,54	-0,07	-0,66	-0,85	-0,51	-0,61	0,04	-0,54	0,42	0,70	0,34	-0,38	-0,39	0,89
12	-0,56	-0,57	0,47	-0,43	-0,32	-0,46	-0,63	-0,58	-0,26	-0,15	0,65	-0,74	0,39	0,15	0,44	-0,33	0,67
13	-0,44	-0,91	0,38	0,27	0,38	-0,10	0,11	0,17	0,65	0,13	0,59	0,33	0,34	0,06	0,15	0,44	0,62
14	-0,02	0,01	0,53	-0,49	0,78	0,68	-0,56	0,02	-0,25	0,69	-0,54	0,12	-0,51	-0,86	-0,09	-0,11	-0,67
15	0,07	0,47	0,23	-0,33	0,19	0,57	-0,42	-0,57	0,47	-0,32	-0,79	-0,49	0,57	0,39	-0,66	0,20	0,57
16	0,66	0,44	-0,18	-0,52	0,56	-0,70	-0,30	0,28	0,08	0,50	0,01	-0,06	0,64	0,33	-0,03	-0,77	-0,46
17	-0,33	0,33	0,23	-0,46	-0,63	-0,06	0,24	-0,45	0,23	0,10	-0,42	-0,05	-0,04	0,66	0,57	0,54	0,44
18	-0,07	-0,38	-0,23	-0,06	-1,21	0,57	-0,27	-0,20	-0,82	0,17	0,09	0,10	0,66	0,39	0,50	-0,20	0,19
19	0,11	-0,51	0,35	-0,39	0,01	0,61	-0,36	0,27	-0,34	-0,22	0,43	-0,53	0,09	0,48	0,32	-0,21	-0,15
20	0,51	0,31	-0,51	-0,35	0,21	0,62	0,22	0,21	-0,31	-0,01	-0,33	0,36	-0,68	-0,49	0,65	0,66	0,06
21	-0,39	0,68	0,24	0,53	0,48	0,10	0,47	-0,46	-0,26	0,82	0,03	-0,32	-0,41	0,16	0,42	0,63	0,01
22	0,55	0,42	0,26	-0,07	0,02	0,27	-0,64	-0,64	-0,22	-0,45	-0,59	-0,61	0,08	0,75	-0,06	0,74	0,05
23	0,88	0,90	-0,65	-0,13	-0,12	-0,47	-0,17	0,51	-0,33	-0,24	0,01	-0,28	0,29	-0,45	0,67	-0,15	0,47
24	-0,61	-0,27	-0,62	-0,57	0,43	0,74	0,03	0,05	-0,05	0,30	-0,13	-0,91	-0,80	-0,07	-0,76	0,63	-0,29
25	0,44	-0,41	-0,18	-0,74	0,53	-0,35	-0,08	-0,67	0,73	-0,10	-0,60	-0,02	-0,41	-0,55	0,05	0,14	0,38
26	0,59	0,07	0,63	-0,44	0,28	-0,60	-0,32	-0,58	0,24	0,07	-0,17	0,97	0,90	0,16	0,67	-0,52	0,50
27	0,32	-0,52	0,48	0,59	-0,02	0,26	0,38	0,10	0,42	0,67	0,44	0,75	-0,12	-0,16	-0,86	-0,75	0,47
28	0,65	-0,42	-0,23	0,57	0,29	0,73	0,49	0,13	-0,37	-0,14	-0,71	0,02	-0,42	-0,64	-0,79	0,20	0,69
29	-0,24	-0,25	0,29	-0,51	0,66	-0,14	0,45	-0,83	0,41	-0,55	-0,42	0,45	-0,49	-0,29	0,53	0,65	-0,61
30	0,41	0,35	0,34	1,04	-0,11	-0,22	-0,10	0,30	-0,04	0,51	0,49	0,19	0,04	1,35	0,28	-0,24	0,90
31	0,49	-0,60	-0,28	0,47	0,66	-0,65	0,35	-0,24	-0,06	0,62	0,15	-0,70	0,26	0,32	-0,28	-0,17	0,83
32	-0,19	-0,35	0,12	0,47	0,23	-0,07	-0,47	-0,60	-0,58	0,60	0,20	0,38	-0,66	-0,62	0,52	-0,36	-1,02
33	0,14	0,24	0,02	0,03	-0,37	0,55	-0,92	-0,27	0,12	0,30	0,68	0,77	-0,62	0,99	-0,49	-0,09	1,03
34	0,23	-0,31	-0,52	-0,03	0,27	-0,24	-0,28	-0,42	-0,22	-0,65	-0,15	-0,64	0,71	-0,51	0,73	0,26	1,25
35	-0,88	-0,43	-0,47	-0,22	-0,11	-0,03	-0,52	0,65	0,69	0,32	-0,01	0,39	0,28	0,98	0,12	-0,16	-1,48
36	0,70	0,29	-0,51	-0,68	0,75	-0,15	-0,15	0,40	0,53	0,77	-0,04	-0,42	0,05	0,13	0,37	-0,11	1,34
37	0,56	0,64	-0,69	-0,33	-0,02	0,34	0,66	0,54	-0,17	-0,19	0,29	0,21	0,26	0,49	0,66	0,00	1,49
38	0,50	-0,31	0,33	-0,67	-0,49	0,17	-0,29	-0,02	-0,36	-0,46	-0,40	-0,12	0,65	-0,60	-0,37	-0,53	1,80
39	-0,18	-0,71	-0,64	0,67	0,14	0,57	-0,34	0,27	-0,34	0,31	0,44	0,50	-0,03	0,11	-0,63	0,28	-1,77
40	0,31	-0,59	0,51	-0,54	0,02	0,57	0,03	-0,05	0,72	-0,28	-0,12	-0,69	0,32	-0,65	0,14	-0,77	1,69

Weight coefficients $w_3(m,k)$ and bias constant $b_3(m)$ for 4 neurons of output layer

($k = 2^{nd}$ hidden layer neuron, m = output neuron)

k / m	1	2	3	4	5	6	7	8	9	10
	$w_3(m,k)$									
1	-0,39	-0,19	0,07	0,24	0,64	0,50	0,12	-0,03	0,50	0,16
2	0,33	-0,14	0,72	0,36	1,01	-0,04	0,52	0,41	-0,69	-0,32
3	-0,42	0,00	-0,50	-0,33	-0,20	0,66	-0,06	-0,75	-0,98	0,21
4	-1,07	0,16	-0,16	0,63	0,20	0,89	-0,24	-0,94	-0,62	0,77

k / m	11	12	13	14	15	16	17	18	19	20
	$w_3(m,k)$									
1	-0,25	0,46	-0,42	0,15	-0,48	-0,66	0,85	-0,36	-0,26	-0,45
2	-0,73	0,31	-0,31	0,46	0,43	-0,37	-0,60	0,00	0,26	-0,72
3	-0,28	0,33	0,49	0,81	-0,31	0,25	-0,54	0,16	-0,77	-0,33
4	0,73	0,09	-0,02	0,69	-0,03	0,81	0,03	1,02	0,71	0,18

k / m	21	22	23	24	25	26	27	28	29	30
	$w_3(m,k)$									
1	-0,49	-0,25	0,82	-0,49	-0,67	-0,56	-0,76	-0,51	0,63	0,00
2	0,71	-0,47	-0,03	-0,27	-0,18	0,11	-0,25	-0,76	0,43	0,33
3	-0,02	0,47	-0,46	-0,35	0,32	0,80	-0,46	-0,27	-0,55	-0,19
4	0,39	1,10	-1,03	-0,94	-0,55	0,61	-0,67	0,68	-0,44	-1,28

k / m	31	32	33	34	35	36	37	38	39	40
	$w_3(m,k)$									
1	0,71	0,79	-0,35	-0,24	0,04	0,40	-0,09	-0,11	0,65	-0,27
2	-0,76	0,29	-0,63	0,54	0,59	0,61	-0,82	0,73	-0,76	-0,45
3	-0,03	0,53	-0,13	0,53	-0,25	0,76	0,71	-0,51	0,32	0,10
4	0,97	1,08	0,15	-0,54	1,10	-0,30	0,19	-0,21	-0,38	0,97

m	b_3
1	-1,02
2	0,64
3	0,24
4	0,79

ANNEX 2

PROGRAMS IN MATLAB

```
%%%%%%%%%%%%%%%%%%%%%%%%
% The program for modelling the system %
%%%%%%%%%%%%%%%%%%%%%%%%
function [t,y] =lscript
clear;close;clc;
global zz  k Ks Kw
M = [1 0 0 0 0 0 0 0
    0 1 0 0 0 0 0 0
    0 0 1 0 0 0 0 0
    0 0 0 1 0 0 0 0
    0 0 0 0 0 0 0 0
    0 0 0 0 0 0 0 0
    0 0 0 0 0 0 0 0
    0 0 0 0 0 0 0 0];
% Use an inconsistent initial condition to test initialization.
zz = ones(8,1);
Ps = 511.2;
Ts=inter_temp('T',Ps);
%Ks=input('Enter Ks: ');
Ks = 0.01152;
%Kw = input('Enter Kw: ');
Kw = 0.00724;
pros = steam_props('ro',Ts);
ms = Ks*sqrt(362);
mcond = ms;
mc = mcond;
Pw = Ps + 1.6*9.8;
%mval = Kw*1000*sqrt(Pw);
mval = mc;
yo = [1.428  340 330 Ps mcond mval mc ms];
tspan = [0:5:2400];
% Use the LSODI example tolerances. The 'MassSingular' property
% is left at its default 'maybe' to test the automatic detection
% of a DAE.
options = odeset('Mass',M,'Vectorized','off','RelTol',1e-2);
k = 1;
[t,y]=ode15s(@fct,tspan,yo,options);

function z = fct(t,y)
global zz k Ks Kw
%yyy(:,k) = y;
ncv = 3; Twi=322.6;mwi=0.6548;
Do=0.3048;
Dt=0.01906;
```

```
Ai =(3.14*(Dt^2)/4);
A =(3.14*(Do^2)/4);
L = 2.5;
h = y(1);
T1 = y(2);
T2 = y(3);
Ps = y(4);
mcond = y(5);
mval = y(6);
mc = y(7);
ms = y(8);
h1 = h/2;
h2 = h/2;
pw1 = water_props('ro',T1);
pw2 = water_props('ro',T2);
Ts=inter_temp('T',Ps);
pros = steam_props('ro',Ts);
Pw = ((pw1*h2+pw2*h1)*9.8)/1000 + Ps;
 % Properties values of steam
 pros = steam_props('ro',Ts);
 hs = enthalpy_props('s',(Ts - 273.15));
 Cps =steam_props('cp',Ts);
 % Properties values of condensate water at the first C.V
 pw1 = water_props('ro',T1);
 Cp1 = water_props('cp',T1);
  % Properties values of condensate water at the second C.V
  pw2 = water_props('ro',T2);
  Cp2 = water_props('cp',T2);
  % Enthalpy
  %??
  hcond =  enthalpy_props('w',(T2 - 273.15));
  hc    =  enthalpy_props('w',(T2-273.15));
  hfg = enthalpyfg_props('hfg',Ts);
  hval =  enthalpy_props('w',(T1-273.15));
  %
  %********
  yy = [h/2 h/2 T2 T1];
  ym = [ms mval mcond mc];
  [q,UA,T] = compute2(Ts,ncv,Dt,Do,yy,ym,Twi,mwi);
  %********
  %******
  z(1) = (mcond-mval)*2/((pw1+pw2)*A);
  z(2) = (mc*hc - mval*hval - q(1)- q(5) - (Cp1*(T1-273.15)*(mc-
mval)))/(pw1*A*(h/2)*Cp1);
```

```
  z(3) = (mcond*hcond - mc*hc - q(2) - q(4) - (Cp2*(T2-273.15)*(mcond-
mc)))/(pw2*A*(h/2)*Cp2);
  z(4) = (ms-mcond+pros*A*z(1))*Ps/(pros*A*(L-h));
  z(5) = mcond - q(3)/hfg;
  z(6) = mval - Kw*(Pw^0.5);
  z(7) = mc-mcond+pw2*A*z(1)/2;
  z(8) = ms - Ks*sqrt(862-Ps);
  z = z';
  zz(k) = pros;
  k = k+1;

 function [q,UA,T] = compute2(Ts,ncv,Dt,Do,yy,ym,Twi,mwi)
%%%%%%
j=1;
 Ltot = 5.5;
for d = 1: ncv
% Calculating the value UA
% For the shell side
% Calculating value hoAo
% FOR STEAM
% Choosing properities value from temp Tc(i)
% Repair Tm
                mys = steam_props('my',Ts);% Change y(t)
      pros = steam_props('ro',Ts);
      Kws = steam_props('k',Ts);
      Prs =steam_props('pr',Ts);
                Cps =steam_props('cp',Ts);
                Prwi =steam_props('pr',Twi);
      hfg = enthalpyfg_props('hfg',Ts);
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%For saturated water
% Choosing properties value from temp Tc(i)
% Repair Tm
       mywo = water_props('my',yy(5-d));
       prowo = water_props('ro',yy(5-d));
       Kwo = water_props('k',yy(5-d));
       Prwo =water_props('pr',yy(5-d));
       Cpwo =water_props('cp',yy(5-d));
       Prwi =water_props('pr',Twi);
       %Transfer unit of Temp K -> C
       Tsc = Ts - 273.15;
       %%%%%Hydraulic diameter
       Dh=(((Do^2))-(48*(Dt^2)))/((Do)+48*Dt);
       % Nu = 3.66 for the flow is laminar
```

```
      if d < ncv
   Rewo = (4*ym(5-d)*Dh)/(3.14*(((Do^2))-(48*(Dt^2)))*mywo);
   Nuo=0.664*Rewo^0.5*Prwo^(1/3);%(Eq 7.25)
   ho = Kwo*Nuo/Dh;
   else
      Rewo = (4*ym(1)*Dh)/(3.14*(((Do^2))-(48*(Dt^2)))*mys);
   Nuo=0.664*Rewo^0.5*Prs^(1/3);%(Eq 7.25)
   ho = Kws*Nuo/Dh;
   end

   HH(j)= ho;
   j=j+1;
end % loop for
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%% For tube side
% Choosing properities value from temp Tc(i)
          mywi = water_props('my',Twi);
      prowi = water_props('ro',Twi);
      Kwi = water_props('k',Twi);
      Prwi =water_props('pr',Twi);
      Cpwi =water_props('cp',Twi);
   Rewi = (4*mwi)/(3.14*Dt*mywi);
%  if Rewi > 2300
%     f = (0.79*log(Rewi)-1.64)^(-2);
%     hi = (Kwi*(f/8)*(Rewi - 1000)*Prwi)/(Dt*(1+12.7*((f/8)^0.5)*(Prwi^(2/3)-1)));
   Nu = 0.023*Rewi^(4/5)*Prwi^(0.4);
   hi = Kwi*Nu/Dt;
   Dt2 = Dt+0.003;
   %else
%     hi = 3.66*Kwi/Dt;
% end
 % Value UA
 jj=1;
 for ii = 1 : (ncv - 1)
   if ii < ncv
     UA(jj) = (1/((1/(HH(ii)*pi*(Dt+0.003)*yy(3-ii)))+(1/(hi*pi*Dt*yy(3-ii)))));
   end
     jj=jj+1;
 end
 L3 = Ltot - 2*(yy(1)+yy(2));
 Rcc = log(Dt2/Dt)/(2*pi*300*L3);
UA(ncv) = 1/(1/(hi*pi*Dt*L3)+Rcc);
% UA(ncv) = 1/(1/(hi*pi*Dt*L3)+Rcc+(1/HH(3)*pi*Dt*L3));
```

```
%%%%%%%%%%%%%%%%%%%%

 qq=zeros((2*ncv-1),1);
%  qq(ncv)=yy(7)*hfg;
 % Calculating Temperature in Tube
  T(1) = Twi ;
  for i = 2: (2*ncv)
    Cpwic = water_props('cp',T(i-1));
    if i <= ncv
     T(i) = (exp((-UA(i-1))/(1*mwi*Cpwic)))*(T(i-1)-yy(2*ncv-i)) + yy(2*ncv-i);  %
Replace Tc(i)=yy(t)
    elseif i == (ncv + 1)
     aaa = -UA(ncv)/(1*mwi*Cpwic);  % Replace Tc(i)=y(t)
     T(i) = (exp((-UA(ncv))/(1*mwi*Cpwic)))*(T(i-1)-Ts) + Ts;  % Replace Tc(i)=y(t)
    else
      T(i) = (exp((-UA(2*ncv-i+1))/(1*mwi*Cpwic)))*(T(i-1)-yy(i-2)) + yy(i-2);
                end
    end
    T;
  % Calculating heat transfer q
 for dd = 1 : (2*ncv-1)
      T(1) = Twi ;
Cpwid = water_props('cp',T(dd));
   qq(dd) = mwi*Cpwid*(T(dd+1)-T(dd));
 end
 q = 24*qq;
  %  q = abs(qq);


%%%%%
%%Function plo
%%%
h=y(:,1);
T1=y(:,2);
T2=y(:,3);
P=y(:,4);
mco=y(:,5);
mva=y(:,6);
mc=y(:,7);
ms=y(:,8);
figure(1)
plot(t,P/6.895);title('P');grid;
figure(2)
plot(t,h);title('h');grid;
```

```
figure(3)
plot(t,mc,t,ms,t,mva,t,mco);grid;
legend('mc','ms','mva','mco');
figure(4)
plot(t,T1,t,T2);grid;
legend('T1','T2');
%title('The reults')
%%THE END


%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% The program determine the values Ks and Kw %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
clear;close;clc;

load data2111.txt

data=data2111(:,:);
n=length(data);
for i=1:(n-60)
        dcm(i)=(data(i+60,13)-data(i,13))/(0.08333);
  xx(i)=i;
end
%%%%Change dcm from lb/h -> kg/s
dcmm =dcm*1.26e-4;
ndata = [dcmm' data((1:1:length(dcm)),4) data((1:1:length(dcm)),3)
data((1:1:length(dcm)),8) data((1:1:length(dcm)),11) data((1:1:length(dcm)),12)
data((1:1:length(dcm)),16) data((1:1:length(dcm)),17)];
[mm,nn]=size(ndata);
%%%
k=1;
for i = 1:mm
  if (ndata(i,1) > 0)
    nd(k,:)=ndata(i,:);
    k=k+1;
  end
end

% Increment to consider for choosing steady data
c = 25;x=1:1:length(nd(:,1));l=1;
% Preprocessing
for i = 1: c : length(nd(:,1))
  b=1;
  for j = i : (i+c-1)
```

```
        if j <= length(nd(:,1))
        t(b)=x(j);
        v(b)=length(nd(i,1));
        b=b+1;
      end
    end
     P=polyfit(t,v,1);
 alpha = (atan(P(1)))*(180/(3.14));
          if (-5 < alpha)  & (alpha < 5)
       for j = i : (i+c-1)
        if j <= length(nd(:,1))
          m(l,:)= nd(j,:);
           l=l+1;
        end
       end
    end
end
m;
[mmm,nnn]=size(m);
CV =zeros(mmm,2);
MW=zeros(mmm,1);
for i = 1:(mmm-1)
   %if m(i+1,7)== m(i,7)
   Tsc = (5/9)*(m(i,2)-32);
   Twic = (5/9)*(m(i,5)-32);
   Twoc = (5/9)*(m(i,6)-32);
   Tcond = (5/9)*(m(i,4)-32);
   Tsk = Tsc + 273.15;
   Twik = Twic + 273.15;
   Twok = Twoc + 273.15;
   pros = steam_props('ro',Tsk);
   prow = water_props('ro',Twik);
   Cpw = water_props('cp',Twik);
   hs =  enthalpy_props('s',Tsc);
   hcond =  enthalpy_props('w',Tcond);
   Ks = m(i,1)/((sqrt((862 - (m(i,3)*6.895)))));
    CV(i,1)=Ks;
   Kw = m(i,1)/((sqrt(m(i,3)*6.895)));
   CV(i,2)=Kw;
   %Pressure Drop
   CV(i,3)=(125 + 14.7 - m(i,3))*6.895;
   mwi = (((m(i,1))*(hs-hcond))/(Cpw*(((m(i,6)-32)*(5/9)+273.15)-((m(i,5)-
32)*(5/9)+273.15))));
   MW(i,1)= mwi;
```

```
end
ldata = [m CV MW];%kk=1;hh=1;
[dd,ddd]=size(ldata);
q=1:1:dd;
figure(1);
plot (q,ldata(:,1)*100,'b-',q,ldata(:,9)*10^3,'r-',q,ldata(:,10)*10^3,'g-',q,ldata(:,12),'m-
',q,ldata(:,7),'c-',q,ldata(:,8),'k-',q,ldata(:,5)/10,'y-',q,ldata(:,11)/5,'k-',q,ldata(:,2)/10,'b');
      axis ([200 dd 0 140]);
      legend('Flow *100(kg/s)','Ks*10^3','Kw*10^3','mwi(kg/s)','% Opening valve of
steam','% Opening valve of water','Temp inlet of cooling water/10','Pressure
Drop/5(Kpa)','Temp of steam/10',1);
      title(' test 21/11')
      grid;
 figure(2);
 plot (q,ldata(:,9)*10^3,'r-',q,ldata(:,7),'b-',q,ldata(:,11)/5,'m-',q,ldata(:,2)/10,'g');
 axis ([200 dd 0 102]);
 legend('Ks*10^3','% Opening valve of steam','Pressure Drop/5(Kpa)','Steam
Temperature/10 (K)',1);
 title(' Ks and % Opening valve')
 grid;
figure(3);
 plot (q,ldata(:,10)*10^3,'r-',q,ldata(:,8),'b-');
 axis ([200 dd 0 10]);
 legend('Kw*10^3','% Opening valve of water',1);
 title(' Kw and % Opening valve')
 grid;
  fl = fopen('nle.txt','a');
 fprintf(fl,' Flow    Ts(F)    Ps    Tcond    Tinlet    Toulet    Valsteam    Valwater
Cvs    Cvw    Pres_Drop    mwi(kg/s)\n');
 for i = 1:dd
   for j = 1 : ddd
     if (j < ddd)
        fprintf(fl,'%f\t ',ldata(i,j));
       else
        fprintf(fl,'%f',ldata(i,j));
       end;
     end;
  if (i < dd)
    fprintf(fl,'\n');
    end ;
 end;
 fclose(fl);
%%%%%%%
 kkk = [ldata(:,9)*10^3 ldata(:,7) ldata(:,11)/5 ldata(:,2)/10 ldata(:,10)*10^3 ldata(:,8)];
```

```
[kr,kc]=size(kkk);
f2 = fopen('KvKs.txt','a');
fprintf(f2,'Ks*10^3  %Opening valve of steam   Pressure Drop/5(Kpa)  Steam
Temperature   Kw*10^3  %Opening valve of water \n');
for i = 1:kr
   for j = 1 : kc
     if (j < kc)
         fprintf(f2,'%f\t ',kkk(i,j));
       else
         fprintf(f2,'%f',kkk(i,j));
       end;
     end;
  if (i < kr)
    fprintf(f2,'\n');
    end ;
end;
fclose(f2);
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% The program for modelling the system with neural network %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
clear;close;clc;
load dataair; % Temp1-Val-Vr21v-Val_Con-Debit-Temp2-Press-Hauter
                                          % variable d

tempin = d(:,1);[tempn,mintempin,maxtempin]=premnmx(tempin);
val = d(:,2);%[valn,minval,maxval]=premnmx(val);
vr21v = d(:,3);[vr21vn,minvr21v,maxvr21v]=premnmx(vr21v);
valcon = d(:,4);[valconn,minvalcon,maxvalcon]=premnmx(valcon);
debit = d(:,5);[debitn,mindebit,maxdebit]=premnmx(debit);
tempout = d(:,6);[tempoutn,mintempout,maxtempout]=premnmx(tempout);
press = d(:,7);[pressn,minpress,maxpress]=premnmx(press);
hauter = d(:,8);[hautern,minhauter,maxhauter]=premnmx(hauter);
%
iput = [tempn val vr21vn valconn]';
oput = [debitn tempoutn pressn hautern]';
[u,v] = size(iput);
%
iitst = 2:4:v;iival = 4:4:v;iitr = [1:4:v 3:4:v];
%
VV.P = iput(:,iival);VV.T = oput(:,iival);VP = VV.P';VT=VV.T';
TV.P = iput(:,iitst);TV.T = oput(:,iitst);TP = TV.P';TT = TV.T';
P = iput(:,iitr);T = oput(:,iitr);PP =P';
%%
net = newff([-1 1;-1 1;-1 1;-1 1],[16 40 4],{'tansig','tansig','purelin'},'trainscg');
```

```
net.trainParam.show = 5;
net.trainParam.lr = 0.8;
net.trainParam.epochs =10000;
net.trainParam.goal = 1e-5;
%net.trainParam.mu    =1;
%net.trainParam.mu_dec =  0.9;
%net.trainParam.mu_inc =  1.5;
%net.trainParam.max_fail = 10;
Q = 75;
[net,tr]=train(net,P,T,[],[],VV,TV);
%%%%%%
k=sim(net,P);x=sim(net,VV.P);y=sim(net,TV.P);
W1 = net.IW{1,1};
W2 = net.LW{2,1};
W3 = net.LW{3,2};
b1 = net.b{1};
b2= net.b{2};
b3= net.b{3};
%%%%%%%
minval = 0; maxval = 0;
mnmx =[mintempin minval  minvr21v minvalcon mindebit mintempout minpress
minhauter ; maxtempin maxval maxvr21v maxvalcon maxdebit maxtempout maxpress
maxhauter ];
%%%%%%%%%%%%%%%
%a=postmnmx(an,mint,maxt);
aa = postmnmx(PP(:,1),mintempin,maxtempin);
bb = PP(:,2);
cc = postmnmx(PP(:,3),minvr21v,maxvr21v);
dd = postmnmx(PP(:,4),minvalcon,maxvalcon);
%%
vaa = postmnmx(VP(:,1),mintempin,maxtempin);
vbb = VP(:,2);
vcc = postmnmx(VP(:,3),minvr21v,maxvr21v);
vdd = postmnmx(VP(:,4),minvalcon,maxvalcon);
%%%
%%%%%%%%%%Print
[m,n]=size(W1);
[mm,nn] = size(W2);
[mmm,nnn] = size(W3);
fo=fopen('W1.txt','a');
for i = 1 : m
     for j = 1 : n
       if (j < n)
          fprintf(fo,'%f\t',W1(i,j));
```

```
        else
          fprintf(fo,'%f',W1(i,j));
        end;
      end;
  if (i < m )
    fprintf(fo,'\n');
    end ;
end;
fclose(fo);
%%%%%%%%%%
f1=fopen('W2.txt','a');
for i = 1 : mm
      for j = 1 : nn
        if (j < nn)
          fprintf(f1,'%f\t',W2(i,j));
        else
          fprintf(f1,'%f',W2(i,j));
        end;
      end;
  if (i < mm )
    fprintf(f1,'\n');
    end ;
end;
fclose(f1);
%%%%%%%%
f2=fopen('b1.txt','a');
for i = 1 : m
  for j = 1
    fprintf(f2,'%f',b1(i,j));
  end;
  if (i < m )
    fprintf(f2,'\n');
    end ;
end;
fclose(f2);
%%%%%%%%%
f3=fopen('b2.txt','a');
for i = 1 : mm
  for j = 1
    fprintf(f3,'%f',b2(i,j));
  end;
  if (i < mm )
    fprintf(f3,'\n');
    end ;
```

```
end;
fclose(f3);
%%%%%%
f4 = fopen('001.txt','a');
for i = 1:2
  for j = 1 : 8
    if (j < 8)
       fprintf(f4,'%f\t',mnmx(i,j));
      else
       fprintf(f4,'%f',mnmx(i,j));
      end;
    end;
 if (i < 2)
   fprintf(f4,'\n');
   end ;
end;
fclose(f4);
%%%%%%%%%%%%%%
 f7=fopen('W3.txt','a');
for i = 1 : mmm
     for j = 1 : nnn
       if (j < nnn)
         fprintf(f7,'%f\t',W3(i,j));
       else
         fprintf(f7,'%f',W3(i,j));
       end;
     end;
  if (i < mmm )
    fprintf(f7,'\n');
    end ;
end;
fclose(f7);
%%%%%%%%%
%%%%%%%%%%
 f8 =fopen('b3.txt','a');
for i = 1 : mmm
  for j = 1
    fprintf(f8,'%f',b3(i,j));
  end;
  if (i < mmm )
    fprintf(f8,'\n');
    end ;
end;
fclose(f8);
```

```
 %%%%
%%%%%
VVV = [vaa vbb vcc vdd];
[vp,vvp]=size(VVV);
 f10 = fopen('ValidInput.txt','a');
 for i = 1:vp
   for j = 1 : 4
     if (j < 4)
         fprintf(f10,'%f ',VVV(i,j));
       else
         fprintf(f10,'%f',VVV(i,j));
       end;
     end;
  if (i < vp)
    fprintf(f10,'\n');
    end ;
 end;
 fclose(f10);
PPP =[aa bb cc dd];
[mp,np]=size(PPP);
 f6 = fopen('Input.txt','a');
 for i = 1:mp
   for j = 1 : 4
     if (j < 4)
         fprintf(f6,'%f ',PPP(i,j));
       else
         fprintf(f6,'%f',PPP(i,j));
       end;
     end;
  if (i < mp)
    fprintf(f6,'\n');
    end ;
 end;
 fclose(f6);
 %%%%%%%%%%%%%%%%5
kk = k';
ee = postmnmx(kk(:,1),mindebit,maxdebit);
ff = postmnmx(kk(:,2),mintempout,maxtempout);
gg = postmnmx(kk(:,3),minpress,maxpress);
hh = postmnmx(kk(:,4),minhauter,maxhauter);
xx=x';
xee = postmnmx(xx(:,1),mindebit,maxdebit);
xff = postmnmx(xx(:,2),mintempout,maxtempout);
xgg = postmnmx(xx(:,3),minpress,maxpress);
```

```
xhh = postmnmx(xx(:,4),minhauter,maxhauter);
xxx =[xee xff xgg xhh];
 [xk,xxk]=size(xxx);
 f15 = fopen('ValidOutput.txt','a');
 for i = 1:xk
   for j = 1 : xxk
     if (j < xxk)
         fprintf(f15,'%f ',xxx(i,j));
       else
         fprintf(f15,'%f',xxx(i,j));
       end;
     end;
  if (i < xk)
    fprintf(f15,'\n');
    end ;
 end;
 fclose(f15);
yy=y';
yee = postmnmx(yy(:,1),mindebit,maxdebit);
yff = postmnmx(yy(:,2),mintempout,maxtempout);
ygg = postmnmx(yy(:,3),minpress,maxpress);
yhh = postmnmx(yy(:,4),minhauter,maxhauter);
yyy =[yee yff ygg yhh];
 [yk,yyk]=size(yyy);
 f16 = fopen('TestOutput.txt','a');
 for i = 1:yk
   for j = 1 : yyk
     if (j < yyk)
         fprintf(f16,'%f ',yyy(i,j));
       else
         fprintf(f16,'%f',yyy(i,j));
       end;
     end;
  if (i < yk)
    fprintf(f16,'\n');
    end ;
 end;
 fclose(f16);
%%%%
kkk = [ee ff gg hh];
 [mk,nk]=size(kkk);
 f5 = fopen('Output.txt','a');
 for i = 1:mk
   for j = 1 : nk
```

```
      if (j < nk)
          fprintf(f5,'%f ',kkk(i,j));
        else
          fprintf(f5,'%f',kkk(i,j));
        end;
      end;
   if (i < mk)
     fprintf(f5,'\n');
     end ;
 end;
 fclose(f5);
taa = postmnmx(TP(:,1),mintempin,maxtempin);
tbb = TP(:,2);
tcc = postmnmx(TP(:,3),minvr21v,maxvr21v);
tdd = postmnmx(TP(:,4),minvalcon,maxvalcon);
%%%
TTT = [taa tbb tcc tdd];
[tp,ttp]=size(TTT);
 f11 = fopen('TestInput.txt','a');
 for i = 1:tp
    for j = 1 : 4
      if (j < 4)
          fprintf(f11,'%f ',TTT(i,j));
        else
          fprintf(f11,'%f',TTT(i,j));
        end;
      end;
   if (i < tp)
     fprintf(f11,'\n');
     end ;
 end;
 fclose(f11);


%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% The program for choosing steady data %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%
clear;close;clc;
load basedata; %variable d1
% Press Temp2 Hauter Temp1  Val VR2-1V ValveCond Debit
% Reading data from  file 'basedata.math'
t2=d(:,1);
tt=d(:,2);
t3=d(:,3);
```

```
t4=d(:,4);
t5=d(:,5);
t6=d(:,6);
t7=d(:,7);
h =d(:,8);
% Input data
y4=t4';%(temp1);
y5=t5';%(Val);
y6=t6';%(VR2-1V);
y7=t7';%(Valve condensate)
%Output data
y=h';%debit
y1=tt';%Temp2
y2=t2';%Press
y3=t3';%Hauter
x=1:1:length(y);
m=zeros([1,length(y)]);
mtemp=zeros([1,length(y1)]);
mpress=zeros([1,length(y2)]);
mha=zeros([1,length(y3)]);
% Increment to consider for choosing steady data
c = 30;
dd=35;
% Preprocessing
for i = 1: c : length(y)
  b=1;
  for j = i : (i+c-1)
       if j <= length(y)
      t(b)=x(j);
      v(b)=y(j);
      b=b+1;
    end
  end
   P=polyfit(t,v,1);
 alpha = atan(P(1));
        if abs(alpha)  < 0.2268
             for j = i : (i+c-1)
               if j <= length(y)
                 m(j)= y(j);
               end
                        end
        end
end
```

```
for i = 1: dd : length(y)
  bb=1;
  for j = i:(i + dd-1)
    if j <= length(y)
    ttl(bb)=x(j);
    temp(bb)=y1(j);
    press(bb)=y2(j);
    ha(bb)=y3(j);
    bb=bb+1;
     end;
   end
% Least Mean Square
  P1=polyfit(ttl,temp,1);
  P2=polyfit(ttl,press,1);
  P3=polyfit(ttl,ha,1);
% Caculating slope
    altemp = atan(P1(1));
    alpress = atan(P2(1));
    alha = atan(P3(1));
% Choosing steady data
                if abs(altemp) < 0.007
             for j = i : (i+dd-1)
               if j <= length(y1)
                 mtemp(j)= y1(j);
               end
                        end
      end

                        %%%%%
        if abs(alpress) < 0.007
          for j = i : (i+dd-1)
            if j <= length(y2)
              mpress(j)= y2(j);
            end
                        end
      end
     %%%%%
       if abs(alha) < 0.00522
          for j = i : (i+dd-1)
            if j <= length(y3)
              mha(j)= y3(j);
            end
                        end
      end
```

```
end

 % Choosing steady data together for making graphic
 zz=1;
 zz1=1;
 zz2=1;
 zz3=1;
 zz4=1;
 zz5=1;
 zz6=1;
 zz7=1;
 for i = 1: length(y)
   if ((m(i)~= 0) & (mtemp(i)~=0) & (mpress(i) ~= 0) & (mha(i)~= 0 ))
     mdebit(zz)=m(i);
     mtempp(zz1)=mtemp(i);
     mpresss(zz2)=mpress(i);
     mhaa(zz3)=mha(i);
     %input data
     intemp(zz4)=y4(i);
     inval(zz5)=y5(i);
                invr21v(zz6)=y6(i);
     invalve(zz7)=y7(i);
   end
   zz=zz+1;
   zz1=zz1+1;
   zz2=zz2+1;
   zz3=zz3+1;
   zz4=zz4+1;
        zz5=zz5+1;
        zz6=zz6+1;
   zz7=zz7+1;

end

% Choosing steady data together for training with Neural Network
 w=1;
 w1=1;
 w2=1;
 w3=1;
 w4=1;
 w5=1;
 w6=1;
 w7=1;
 for kk = 1: length(mdebit)
```

```
    if ((mdebit(kk)~= 0) & (mtempp(kk)~=0) & (mpresss(kk) ~= 0) & (mhaa(kk)~=0 ))
      odbt(w)=mdebit(kk);
      otemp(w1)=mtempp(kk);
      opress(w2)=mpresss(kk);
      ohauter(w3)=mhaa(kk);
    w=w+1;
    w1=w1+1;
    w2=w2+1;
    w3=w3+1;
      %input data
      iitemp(w4)=intemp(kk);
      iival(w5)=inval(kk);
                  iivr21v(w6)=invr21v(kk);
      iivalve(w7)=invalve(kk);
    w4=w4+1;
         w5=w5+1;
         w6=w6+1;
    w7=w7+1;
end
end
%%%%%%%%%%%%%%%%%%
md= (mdebit/10);
 % Graphics
 f=1;
 figure(f);
   hold on
    plot(x,y','-rs','LineWidth',0.5,...
          'MarkerEdgeColor','k',...
          'MarkerFaceColor','g',...
          'MarkerSize',2);
        m;
        hold on
        plot(x,m,'-b','LineWidth',2);
         xx=1:1 :length(mdebit);
        plot(xx,mdebit,'-m','LineWidth',2);
hold on
        plot(xx,mtempp,'-r','LineWidth',2);
       hold on
plot(x,y1','-g','LineWidth',0.5,...
          'MarkerEdgeColor','k',...
          'MarkerFaceColor','g',...
          'MarkerSize',0.1);
        hold on
        plot(x,y2','-c','LineWidth',0.5,...
```

```
            'MarkerEdgeColor','k',...
            'MarkerFaceColor','g',...
            'MarkerSize',0.1);
          hold on
       plot(x,mpress,'-k','LineWidth',2);
  plot(xx,mpresss,'-b','LineWidth',1);
%
hold on
plot(x,y3','-k','LineWidth',1,...
            'MarkerEdgeColor','k',...
            'MarkerFaceColor','g',...
            'MarkerSize',0.1);
          hold on
           plot(x,mha,'-g','LineWidth',2);
  plot(xx,mhaa,'-m','LineWidth',1);
  grid;
   XLABEL('Time');
  title('Choosing Steady-Data from experiment data')
  % Graphic
    f=f+1;
  figure(f);
  plot(xx,md,'-m','LineWidth',1);
  hold on
  plot(xx,mtempp,'-r','LineWidth',1);
  hold on
  plot(xx,mpresss,'-b','LineWidth',1);
  hold on
  plot(xx,mhaa,'-g','LineWidth',1);
  hold on
  plot(xx,intemp,'-r','LineWidth',2);
  hold on
  plot(xx,inval,'-y','LineWidth',1);
  hold on
  plot(xx,invr21v,'-c','LineWidth',1);
  hold on
  plot(xx,invalve,'-k','LineWidth',1);
  legend(' OUTPUT debit','temp','press','hauter',' INPUT temp','Val','VR2-1V','Valve
cond.',2)
  grid;
  %%%%%%%%%%5
  figure(3);
  xxx=1:length(odbt);
  plot(xxx,odbt/10,'-m','LineWidth',1);
  hold on
```

```
plot(xxx,otemp/1.5,'-r','LineWidth',1);
hold on
plot(xxx,opress,'-b','LineWidth',1);
hold on
plot(xxx,ohauter,'-g','LineWidth',1);
hold on
%%input
plot(xxx,iitemp,'-r','LineWidth',2);
hold on
plot(xxx,iival,'-y','LineWidth',1);
hold on
plot(xxx,iivr21v,'-c','LineWidth',1);
hold on
plot(xxx,iivalve,'-k','LineWidth',1);
 AXIS([0 length(odbt) 0 450]);
XLABEL('Time');
title('Steady-Data')
legend(' OUTPUT debit','temp','press','hauter',' INPUT temp','Val','VR2-1V','Valve
cond.',2);
grid;
%%%%Save input and output data
od = odbt;
ot = otemp;
op = opress;
oh = ohauter;
%%input
it = iitemp;
ival = iival;
ivr2 = iivr21v;
iv = iivalve;
p=[it; ival; ivr2;iv];
T=[od; ot; op; oh];pp=[p;T];
ppp = pp';
%%%%%%%%%%%%%%%%5
f6 = fopen('Steadydatabypass.txt','a');
fprintf(f6,' Temp1    Val    Vr21v    Val_Con    Debit    Temp2    Press    Hauter
\n');
for i = 1:length(it)
  for j = 1 : 8
    if (j < 8)
       fprintf(f6,'%f\t ',ppp(i,j));
      else
       fprintf(f6,'%f',ppp(i,j));
      end;
```

```
    end;
  if (i < length(it))
    fprintf(f6,'\n');
    end ;
 end;
 fclose(f6);

 %#############################################################
%The program for validation the model with neural network
%#############################################################
clear;close;clc;
%---lecture données---------
%load mars_14.txt;
%load mars_19.txt;
%load dec_04.txt;
%load nov_20_a.txt;
%load nov_20_bypass.txt;
load nov_21.txt;
%load data14_03.txt
load data21_11.txt;
d = nov_21(:,:);% data matrix
dd = data21_11(:,:);
n=length(d(:,1));
m = length(dd(:,2));
if n >= m
   for i = 1:m
   ddd(i,:)=d(i,:);
   end
else
   for i = 1: n
    ddd(i,:)=d(i,:);
  end
end
   ddd;
   nn=length(ddd(:,1));
%calcul débit masse
for i=1:(nn-60)
          clear moyt
  dm(i)=(ddd(i+60,5)-ddd(i,5))/(0.08333);
  b(i)=i;
  for j=1:(13*5-5+1)
    moyt(j)=dd(j+i-1,2);
  end
  dvtn(i)=mean(moyt);
```

```
end
[y,z]=size(b);
for k = 1:z
   a(k,:)=ddd(k,:);
   c(k,:)=dd(k,:);
end
a;
c;
%figure
figure(1)
plot(b,dm/5,'b-',b,dvtn/5,'g-',b,a(:,1)*2,'r-',b,c(:,3)*2,'c-',b,a(:,2),'m-',b,c(:,4),'g--
',b,a(:,3),'k-',b,c(:,5),'r--')
axis([0 z 0 500])
legend('debit mass/5','debit vortex/5','Pression exp*2','Pression model*2','Temp
exp','Temp model','Hauter exp','Hauter model',1)
title('compare between experiment data and model 21/11')
grid
figure(2)
plot(b,a(:,1),'r-',b,c(:,3),'b-')
axis([0 2200 0 500])
xlabel('Time');
ylabel('Psia')
legend('Experiment','Model',1)
title('Comparison of presurre (21/11)')
grid
figure(3)
plot(b,a(:,2),'m-',b,c(:,4),'g-')
axis([0 2200 0 500])
xlabel('Time');
ylabel('K')
legend('Experiment','Model',1)
title('Comparison of temperature (21/11)')
grid
figure(4)
plot(b,a(:,3),'k-',b,c(:,5),'b-')
axis([0 2200 0 500])
xlabel('Time');
ylabel('in')
legend('Experiment','Model',1)
title('Comparison of the liquid level (21/11)')
grid
figure(5)
plot(b,dm,'b-',b,dvtn,'g-')
axis([0 2200 0 2500])
```

```
xlabel('Time');
ylabel('lb/h')
legend('Experiment','Model',1)
title('Comparison of flow (21/11)')
grid
%%%%%%%%%%%%

%%%%%%%%%%%%%%%%%
function p = enthalpy_props(name,T)

load enthalpy.txt;
label = char('temp','press','w','s');
i=strmatch(name,label);
y = enthalpy(:,i);
x = enthalpy(:,1);
n = length(x);
if(T < x(1))T = x(1);end
if(T > x(n))T = x(n);end
p = interp1(x,y,T);
switch name
case 'w'
  p = p*1000;
case 's'
   p = p*1000;
end

function a=inter_temp(name,P)
load tempress.txt;
label = char('press','T');
b=strmatch(name,label);
y = tempress(:,b);
x = tempress(:,1);
n = length(x);
if(P < x(1))P = x(1);end
if(P > x(n))P = x(n);end
a = interp1(x,y,P);

%%%%%%%%%%%%%%%%%%%
function p = enthalpyfg_props(name,T)
%%%%%%%%%%%%%%%%%%%

load fg_props.txt;
label = char('temp','hfg');
```

```
i=strmatch(name,label);
y = fg_props(:,i);
x = fg_props(:,1);
n = length(x);
if(T < x(1))T = x(1);end
if(T > x(n))T = x(n);end
p = interp1(x,y,T);

%%%%%%%%%%%%%%%%%%
function p = steam_props(name,T)
%%%%%%%%%%%%%%%%%%

load steam.txt;
label = char('temp','ro','cp','my','k','pr','pess');
i=strmatch(name,label);
y = steam(:,i);
x = steam(:,1);
n = length(x);
if(T < x(1))T = x(1);end
if(T > x(n))T = x(n);end
p = interp1(x,y,T);
switch name
case 'cp'
  p = p*1000;
%  1 bar = 10^5 Pa
case 'pess'
  p = p*100;%  change to kPa
end

function p = water_props(name,T)

load water.txt;
label = char('temp','ro','cp','my','k','pr','pess');
i=strmatch(name,label);
y = water(:,i);
x = water(:,1);
n = length(x);
if(T < x(1))T = x(1);end
if(T > x(n))T = x(n);end
p = interp1(x,y,T);
%%%
switch name
case 'cp'
  p = p*1000;
```

```
%  1 bar = 10^5 Pa
case 'pess'
p = p*100;%  change to kPa
end
```

BIBLIOGRAPHY

[1] Botsch T.W, Stephan K, Alcock, J.-L, Webb D.R, An Experimental Investigation of the Dynamic Behaviour of a Shell-and-Tube Condenser. *Int. J. Heat Mass Transfer* 1997, Vol. 40, No. 17, pp 4129-4135.

[2] J. R. White, *Lecture notes, System Dynamics.* Uni. of Mass-Lowell 1997, website:http://chemical.caeds.eng.uml.edu/onlinec/white/sdyn/s6/s6intro/s6intro.html.

[3] Botsch T.W, Stephan K, Alcock, J.-L, Webb D.R, Modelling and Simulation of the Dynamic Behaviour of a Shell-and-Tube Condenser. *Int. J. Heat Mass Transfer* 1997, Vol. 40, No. 17, pp. 4137-4149.

[4] Holger R. Maier, Graeme C. Dandy, Neural networks for the prediction and forecasting of water resources variables: a review of modelling issues and applications, *Environmental modelling and software 15* , 2000, pp.101-124.

[5] M. Norgaard, O. Ravn, N.K. Poulsen, *Neural network for modelling and control of dynamic systems,* Spinger-Verlag, London 2000.

[6] Neural network toolbox, Version 4 The Math works Inc 2000.

[7] K. H. Johansson, The quadruple-Tank process a multivariable laboratory process with an adjustable zero, *IEEE Transactions on control systems technology* 2000, Vol. 8, No.3, pp 456-465.

[8] Robert Sabourin, *Réseaux de Neurones et systèmes flous, SYS-843, Notes de cours*, Département de génie de la production automatisée, Ecole de Technologie Supérieure de Montréal, 2000.

[9] Curtis D. Johnson, *Process control instrumentation technology,* Fifth edition, Prentice Hall, 1997.

[10] Nicolas Vincent, *Développement d'une méthode de calibration des débimètres de vapeur basée sur la mesure du poids de condensât,* Mémoire de maîtrise, École de Technologie Supérieure de Montréal, 2002

[11] M. F. Moller, A scaled conjugate gradient algorithm for fast supervised learning. *Neural networks* 1993, Vol.6, pp 525-533.

[12] W. L. Luyben, *Process modelling, simulation, and control for chemical engineers.* Second edition, McGraw-Hill, New York, 1990.

[13] Frank P. Incropera, David P. DeWitt, *Introduction to Heat transfer*, Third edition, John Wiley & Sons, INC, 1996.

[14] F.G. Shinskey, *Process Control Systems,* Second edition, Mc Graw - Hill 1979.

[15] Baliga Kabi, *Fluid flow and heat transfer equipment,* Lectures notes, Department of Mechanical Engineering, Mc Gill University, Montreal 2002.

[16] Robert H Bishop, *Learning with Labview 6i,* Prentice Hall, 2002.