Reduction of Product Driven System emulation models based on neural network: impact of discrete data

Philippe Thomas, André Thomas, Marie-Christine Suhner

▶ **To cite this version:**

Philippe Thomas, André Thomas, Marie-Christine Suhner. Reduction of Product Driven System emulation models based on neural network: impact of discrete data. International Conference on Industrial Engineering and Systems Management, IESM' 2009, May 2009, Montréal, Canada. pp.CD. hal-00388829

HAL Id: hal-00388829
https://hal.science/hal-00388829

Submitted on 27 May 2009

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

International Conference on Industrial Engineering and Systems Management, IESM' 2009, May 13 - 15, Montréal, Canada, 2009

Reduction of Product Driven System emulation models based on neural network: impact of discrete data*

Philippe THOMAS, André THOMAS, Marie-Christine SUHNER

Centre de Recherche en Automatique de Nancy (CRAN-UMR 7039),

Nancy-University, CNRS

ENSTIB 27 rue du Merle Blanc, B.P. 1041

88051 Epinal cedex 9 France

Abstract

Product Driven Systems (PDS) architecture needs emulation systems [13]. Discrete events simulation is then often used to build this emulation tool, but emulation model design is not a trivial task. Also, the goal of this paper is the study of the design of a simulation model by reducing its complexity. According to theory of constraints, we want to build reduced models composed exclusively by bottlenecks and a neural network. Particularly a multilayer perceptron, is used. The structure of the network is determined by using a pruning procedure. This work focuses on the impact of discrete data on the results. This approach is applied to sawmill internal supply chain.

Key words: multilayer perceptron, reduced model, simulation, neural network, re-scheduling, supply chain

1 Introduction

In Manufacturing, Planning and Control processes (centralized way to control physical flow in a Supply Chain), evaluation of planning or scheduling scenario by simulation is very useful for the decision makers. Indeed, simulation highlights the evolution of the machines states, the WIP (work in process), and the queues. This information is useful in order to perform a "Predictive scheduling" [8] or a rescheduling. On the other hand, in Product Driven processes (distributed way to control physical flow in a Supply Chain) dedicated architectures are implemented. These architectures consist of a control system and an emulation system. The last one is very useful for PDS design and validation and for control scenario evaluation. So, the PDS architectures require the use of emulation models which are sufficiently precise for representing correctly the system while remaining simple in order to decrease computing times. Discrete event simulation is also often used to build this emulation system, but emulation model design, which is not a trivial task, relies on reusability, modularity and genericity concepts [13]. At these levels of planning and control and to estimate how the whole physical system works, the "management of critical resources" (bottlenecks) is often pertinent [16]. Goldratt and Cox, in "The Goal" [6] put forward the Theory of Constraints (TOC), which proposes to manage the whole supply chain by bottlenecks control. Dynamic discrete events simulation of material flow permits this management [12]. In fact, simulation models of actual industrial cases are often very complex and the modellers encounter problems of scale [10]. Also, many works have highlighted the interest to use simplest (reduced/aggregated) models of simulation [1, 2, 17]. In addition, neural networks have proved their abilities to extract performing models from experimental data

* This paper was not presented at any other revue. Corresponding author P. Thomas. Tel. +33(0) 3 29 29 61 73. Fax +33(0) 3 83 68 44 37

Email addresses: philippe.thomas@cran.uhp-nancy.fr (Philippe Thomas), andre.thomas@cran.uhp-nancy.fr (André Thomas), marie-christine.suhner@cran.uhp-nancy.fr (Marie-Christine Suhner).

[14]. So the use of neural networks appears recently as an interesting approach within the framework of the supply chain [15]. Neural networks are generally used in order to perform a mapping between continuous spaces. However, in the considered cases, continuous variable (as length, speed …) are mixed with discrete ones (as category, colour …). The main goal of this paper is to investigate the impact of these discrete data on the learning process and on the quality of neural model used in order to reduce simulation models. This is studied on one industrial example which is a sawmill flow shop case. In the next part, the used approach of reduction model and the multilayer perceptron are presented. The third part will be devoted to the presentation of the industrial application which is a sawmill flow shop case. The fourth part presents the inputs and output data of the neural network, and the learning. The results are investigated in order to evaluate the comportment of the network in function of the considered data in the last part.

2 The model reduction

2.1 The algorithm

Zeigler [18] has been the first to deal with the problem of model reduction. For Him, complexity of a model is relative to the number of elements, connections and model calculations. He suggests different ways to simplify a discrete simulation model, in replacing part of the model by a random variable, in degrading the range of values taken by a variable and in grouping parts of a model together. Innis et al. [7] first listed 17 simplification techniques for general modelling. Their approach is comprised of four steps: hypotheses (identify the important parts of the system), formulation (specify the model), coding (build the model) and experiments. Brooks and Tobias [1] suggest a "simplification of models" approach for cases where the indicators to be followed are the average throughput rates. Other cases have been studied. The reduction algorithm used [15] is an extension of those presented by Thomas and Charpentier [12]. Its principal steps are recalled and explained below:

1. Identify structural bottleneck (work center (WC) which for several years has been mainly constrained in capacity).

2. Identify conjunctural bottleneck for the bundle of Manufacturing Orders (MO) of the considered MPS.

3. Among the WC not listed in 1 and 2, identify the one (synchronisation WC) satisfying these conditions:

   - present at least in one of the MO using a bottleneck,

   - widely used considering the whole MO.

4. If all MO have been considered go to 5 if not go to 3.

5. Use neural networks for model the intervals between WC which has been found during preceding steps (figure 1).

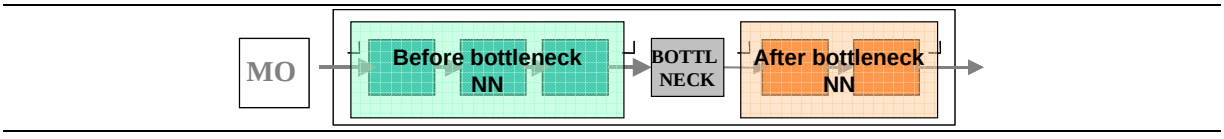

Fig. 1. Reduction model algorithm

So, the Work Centers (WC) remaining in the model are either conjunctural or structural bottlenecks or WC which are vital to the synchronization of the MO. All other WC are incorporated in "aggregated blocks" upstream or downstream of the bottlenecks.

"Conjunctural bottleneck" is a WC which is saturated for the MPS and predictive scheduling in question. This is to say that it uses all available capacity. By "structural bottleneck" we mean a WC which (in the past) has often been in such a condition. Effectively, for one specific portfolio (one specific MPS) there is only one bottleneck – the most loaded WC – but this WC can be another WC than the traditional bottlenecks.

"Synchronization work centers" are resources used jointly with bottlenecks for at least one MO and used for the planning of different MO which do not use bottleneck. To minimize the number of these "synchronization work centers", the WC which have the most in common amongst all this bundle of MO using no bottlenecks and which figure in the routing of at least one MO using bottleneck must be found.

In function of the variation of the MO, the bottleneck may vary. So, different structures should be performed in order to consider the different case occurring.

2.2 The multilayer perceptron

Works of Cybenko [3] and Funahashi [5] have proved that a multilayer neural network with only one hidden layer using a sigmoïdal activation function and an output layer using a linear activation function can approximate all non linear functions with the wanted accuracy. This result explains the great interest of this type of neural network which is called multilayer perceptron (MLP). In this research work, our hypothesis lies in the fact that a part of the modelized production system could be approximate by a non linear function obtained thanks to a MLP. The structure of the multilayer perceptron is recalled here. Its architecture is shown in figure 2. The i-th neuron in the hidden layer (figure 2) receives n_0 inputs $\left\{x_1^0,\cdots,x_{n_0}^0\right\}$ with associated weights $\left\{w_{i1}^0,\cdots,w_{in_0}^0\right\}$. This neuron first computes the weighted sum of the n_0 inputs:

$$z_i^1 = \sum_{h=1}^{n_0} w_{ih}^1 . x_h^0 + b_i^1 \tag{1}$$

where b_i^1 is a bias or threshold term. The output of the neuron is given by an activation function of the sum (1):

$$x_i^1 = g\left(z_i^1\right). \tag{2}$$

where g(.) is chosen as an hyperbolic tangent:

$$g(x) = \frac{2}{1+e^{-2x}} - 1 = \frac{1-e^{-2x}}{1+e^{-2x}}\,. \tag{3}$$

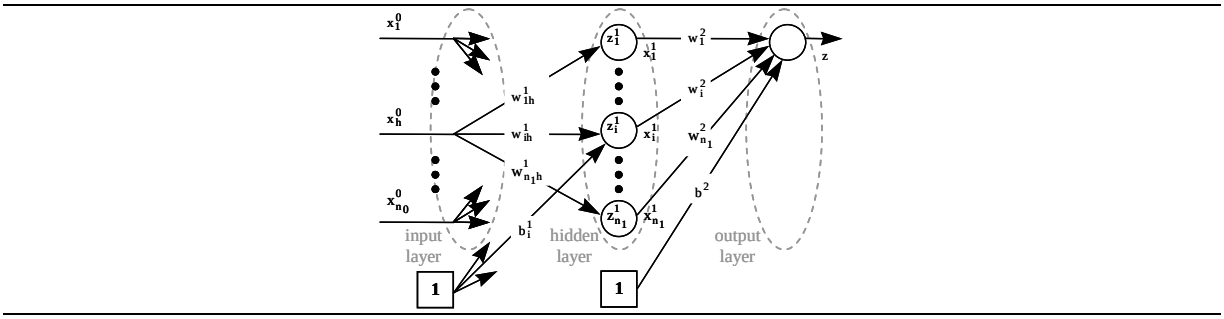

Fig. 2. Architecture of the multilayer perceptron

The neuron in the last layer simply performs the following sum, its activation function being chosen linear:

$$z = \sum_{i=1}^{n_1} w_i^2 . x_i^1 + b^2 \tag{4}$$

where w_i^2 are the weights connecting the output of the hidden neurons with the output neuron and b^2 is the threshold of the output neuron. Now, only the number of hidden neurons is always unknown. In order to determine it, the learning starts from an overparametrized structure. A weight elimination method is used to remove spurious parameters [11]. The learning of the MLP is performed in three steps:

- Initialisation of the weights of an oversized structure by using the Nguyen Widrow algorithm [9].
- Learning of the parameters by using Levenberg-Marquard algorithm with robust criterion [14].
- Weights elimination by using the algorithm proposed by Setiono [11].

3 The overview of the sawmill

At the time of the study, the considered sawmill had a capacity of 270.000 m3 / year, a 52 million euros turnover and 300 employees. The sawmill objective is to transform logs into main and secondary products respecting a cutting plan. The considered cutting plan is presented into figure 3.

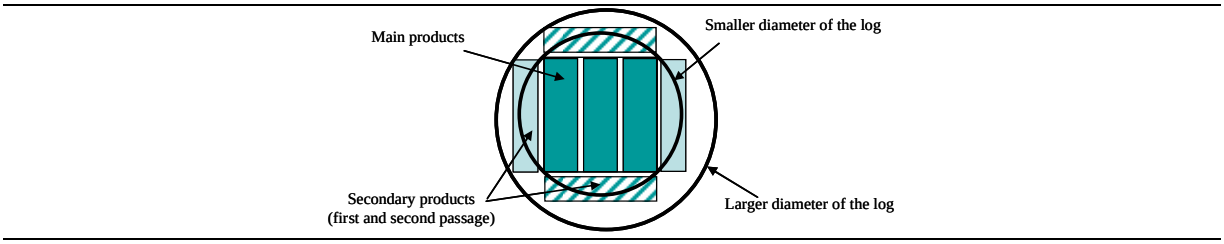

Fig. 3. The cutting plan

The physical industrial production system is composed of sequential work centers (kockums saw, trimmer, sorter …) and queues or conveyors (named respectively RQM4, RQM5, RQM7 …). The log enters the system in RQM1 then it is steered to RQM4 or 5 according to its characteristics. After that, it passes to the cutting machine (Canter). It then enters the edger. After this phase, the log is transformed into main and secondary products. The final operation is the cross cutting which consists in cutting up products to length. Two important steps occur during this process. The first one is the choice of the conveyors RQM4 or RQM5 in order to store the arrival log. In function of this choice, the time spending by the log to wait the Canter saw may be very different. The second one is the type of product considered. When the cutting plan is considered, two types of products appear: main and secondary ones. Only the secondary products have to use the kockums saw when secondary and main products use the trimmer. However, when the physical industrial system is considered, three types of products have to be considered. In fact the Cutting machine Canter works into three steps. First, one saw (CSMK) cuts two faces of the considered log and produces the two secondary products (hatched on figure 3). These two products are driven to kockums saw in order to be finished. Next the log is rotated of 90° and stored into conveyor RQM7. After that, the log is driven once again to the Canter machine. The saw (CSMK) cuts the two other faces of the log, and produces the two other secondary products which are driven to kockums saw. At this time, a parallelepiped is obtained which is divided into three main products by another saw (MKV). The main products are finally driven to the trimmer.

4 The simulation model

4.1 The complete model

During preceding works [12] the complete model of the sawmill process has been constructed. This model is presented figure 4. It is composed of different modules. The first one serves to model the log arrival which follows a homogeneous Poisson process. In this module, the characteristics of the log which are measured by a scanner are associated with it. A second module, the "input sorter", directs the log to RQM4 or RQM5 in function of its characteristics as explain in the preceding part. It may also eject the log of the process if it is machine gunned or if its dimensions are out of range. The logs go to the next module which models the RQM4 and RQM5 queues. Conveyors RQM4, RQM5 and even RQM7 serve of input inventory for the canter line. So a policy is needed in order to determine the priority for choosing between these conveyors the next log treated by the Canter line. Two modules are used for the simulation of the canter line and the passage of the squared in RQM7. Canter line model uses two sub-models for the management of main and secondary products. Canter line has three outputs which lead to kockums line for the secondary products and to trimmer line for the main ones.

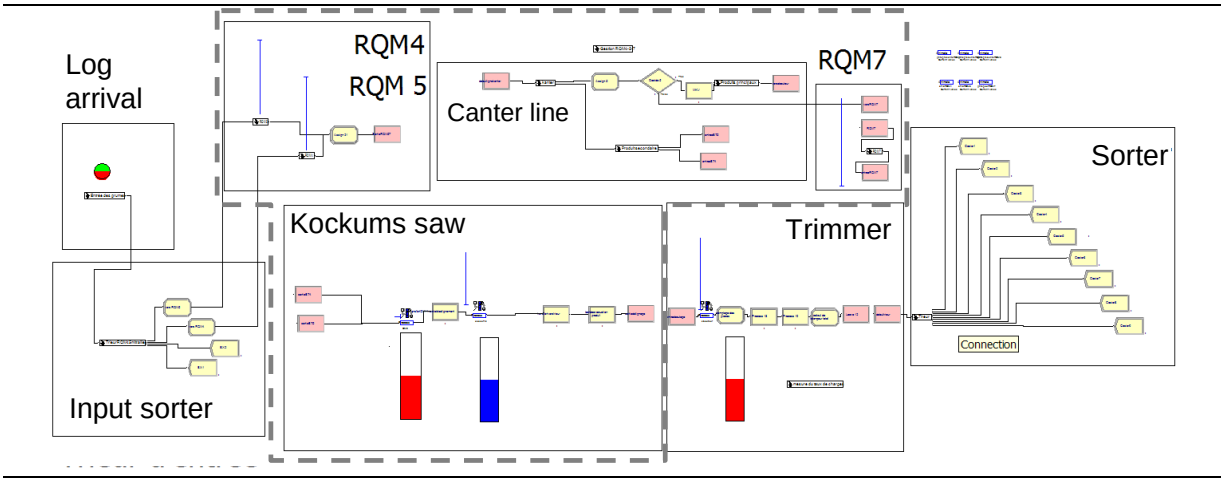

Fig. 4. The complete model

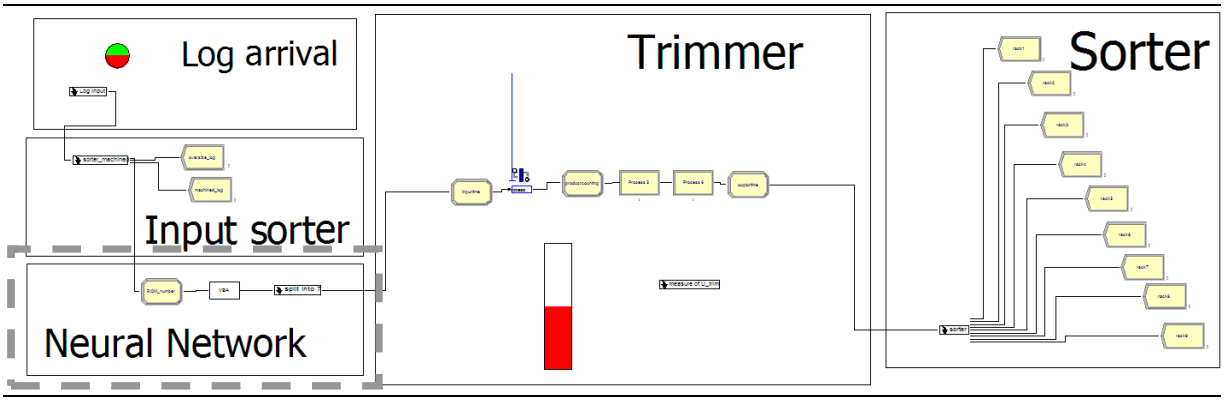

Fig. 5. The reduced model

4.2 The reduced model

As we can see, the design of a complete model for the simulation of a workshop is a difficult task which leads to a complex model. Somewhere else, the bottleneck of this line is the trimmer [12], and to optimise the use of bottleneck is the main industrial objective according to theory of constraints concept [6]. Consequently and

within this framework, modelled the functioning of inventories RQM4 RQM5 and RQM7 and of the canter line is unnecessary. Also, all the part surrounded by grey dashed line on figure 4 gives no direct and useful information for the evaluation of a MPS. In fact, only the arrival times of the products in trimmer queue are useful in order to simulate the load of this bottleneck. Also a multilayer perceptron is used to replace all the part surrounded by grey dashed line on figure 4. Then neural network uses the available shop floor information. The reduced, and so, simplified model is presented on figure 5. The determination of the neural model is the core of the problem.

4.3 The data and the learning

Neural model is a black box obtained with a supervised learning of a non linear relation between input and output data sets. For this, we need to collect the available input data of the process and to determine the desired output [15]. First, each log gives information which is collected by a scanner in input of the canter line. This information is relating to the product dimension, as length (Lg) and three values for timber diameter (diaPB ; diaGB ; diaMOY). These variables serve to control the log to RQM4 or RQM5 queues which is additional information (RQM). In addition of this dimensional information, we have to characterise the process variables at the time of the log arrival. Particularly, the input stock of the trimmer (Q_trim), the utilisation rate of the trimmer (U_trim) and the number of logs present in the different conveyors RQM4, RQM5 and RQM7 (Q_rqm4; Q_rqm5; Q_rqm7) must be taken. Moreover, the sum of these number is also used (Q_rqm = Q_rqm4+Q_rqm5+Q_rqm7). The last type of information is related to the cutting plan of the logs. In fact, each log will be cut into *n* main or secondary products. In our application, the cutting plan (figure 3) divides the log into 7 products:

- 2 secondary products resulting from the first step of cutting process on saw CSMK of the canter line,
- 2 secondary products resulting from the second step cutting process on saw CSMK of the canter line after staying in the RQM7 queue,
- 3 main products resulting from the third step of cutting process on saw MKV of the canter line.

These two saws (CSMK and MKV) belong to the canter line. These 7 products can be classified into three categories according to the location (CSMK or MKV) and the time during the cutting process (first or second cutting). This information is given by the variable (T_piece) which can take as values type1 type2 and type3. The last information is the thickness (in mm) of the product which is also the reference. In our case, we are taking into account only two references: main products 75; secondary products 25 (ref). However, preceding works [16] have shown that this data has no impact on the result and so it will no be taken into account. Consequently, the neural networks input variables are: Lg; diaGB; diaMoy; diaPB; T_piece; Q_trim; U_trim; Q_rqm; Q_rqm4; Q_rqm5; Q_rqm7; RQM. In our application 12775 products are simulated. Among these 12 inputs data, two different categories exist:

- Continuous one (quantitative) [Lg; diaGB; diaMoy; diaPB; Q_trim; U_trim; Q_rqm; Q_rqm4; Q_rqm5; Q_rqm7]. These data are continuous ones and so are well adapted to be used by learning procedure.
- Discrete one (qualitative) [T_piece; RQM]. These data are qualitative. So the study of their impact on the learning process is the core of this paper.

Our objective is to estimate the delay (ΔT) corresponding to the duration of the throughput time for the 12775 products. ΔT is measured between the process input time and the trimmer queue input time. In practice ΔT is the output of the neural network:

$$\Delta T = \sum_{i=1}^{n_i} w_i^2 . g\left(\sum_{h=1}^{12} w_{ih}^1 . x_h^0 + b_i^1 \right) + b^2 \qquad (5)$$

The learning of the network is supervised. So, it is necessary to divide the database into two datasets, learning and validation ones. Only the number of hidden neurons is always unknown and should be determined. In order to determine it, the learning starts from an overparametrized structure and a weight elimination method is used to

remove spurious parameters [11]. The initial overparametrized structure used 10 hidden neurons. So, the learning begins with a structure using n_i=10 hidden neurons (5) which correspond to 141 parameters.

5 The results

The learning approach corresponds to a local search of a minimum. So, in function of the initial weights, the results may be different. In order to evaluate the dispersion of the results, 30 different sets of initials weights are used. In a first step, all the input data are presented to the network in order to produce the model. In a second step, in order to improve the results, a second approach will be presented.

5.1 First approach

Table 1

	Learning residual Mean	Learning residual StD	Validation residual Mean	Validation residual StD
Mean	78.61	586.09	74.33	582.06
StD	43.94	146.50	41.61	145.44
Min	17.11	408.45	12.35	413.93
Max	213.08	1168.80	206.75	1170.93

The 30 learning on the different weights sets have been performed with the 12 inputs and the 10 hidden neurons. In the table 1, the mean and the standard deviation of the residuals obtained on the learning and the validation data sets are presented.

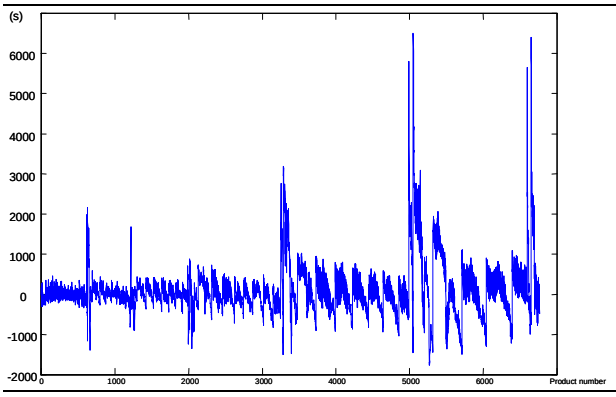

Fig. 6. Residual obtained on the learning data set

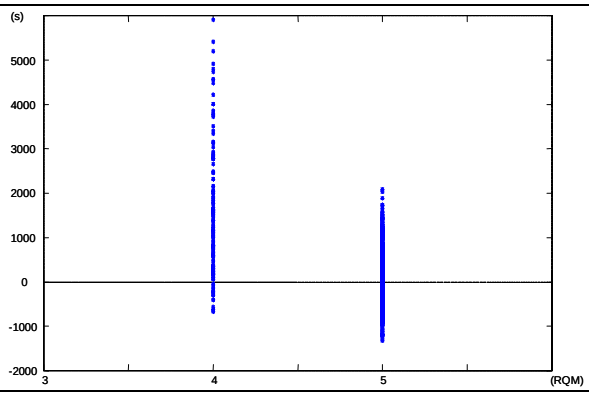

Fig. 7. Residual function of RQM

These results shows that the residuals obtained are always bad. In particular, the mean of the residual obtained may vary, in function of the initial weights from 17.11s to 213.08s on the learning data set. For the validation data set, the results are very similar, with a mean of residual varying from 12.35s to 206.75s. It can be noticed that the mean of the residuals is lower than 30s in only 10% of the cases in learning and 16.67% of the cases in validation. Concerning the standard deviation values, they are large and varying from 408.45 to 1168.8 for the learning data set and from 413.93 to 1170.93 for the validation data set. These two facts show that the learning is not efficient. Figure 6 shows an example of residual characteristic of those obtained on the validation and learning data sets for the 30 different initial weights. The residual presented figure 6 shows clearly that some dynamics of the input data are not taken into account. This fact may be due to different causes: Some explicative variables aren't present in the input data, or the number of hidden neuron is not sufficient, or the neural network doesn't succeed to learn some dynamics.

Other tries with much more hidden neurones (up to 35) have shown that 10 hidden neurons are sufficient. Moreover, the pruning algorithm prunes some of these ten hidden neurons into 56% of the cases. In order to determine if some dynamics presents in the data aren't taken into account by the learning, the correlation between the different inputs and the residuals can be performed on the learning data set (table 2). The table 2 presents the mean, standard deviation, minimal and maximal values of the absolute value of the correlation coefficients obtained between the 30 residuals and the 12 inputs on the learning data set. Similar results can be obtained on the validation data set. These results are very similar with those obtained on the validation data set. It can be noticed that Lg, diaGB, diaMoy, T_piece, U_trim presents a correlation coefficient with residuals which is never significant (always smaller than 0.0959). U_trim, Q_rqm, Q_rqm5, Q_rqm7 present a minimal value of correlation to 0 because the pruning algorithm, in some case have pruned these inputs. Only two inputs have always a significant coefficient correlation with the residual: diaPB and RQM. So, on the two discrete inputs, T_piece and RQM, the correlation coefficients show that the dynamic of the first one is well taken into account by the network when the RQM not. However, these two data are discrete ones. So, the correlation test isn't the most significant. Figure 7 presents an example of the residuals in function of RQM. It can be noticed that two different residuals exist depending of the value of RQM. So, in order to estimate the influence of RQM on the residual the best approach is to compare these two samples.

Table 2

	Lg	diaGB	diaMoy	diaPB	T_piece	Q_trim	U_trim	Q_rqm	Q_rqm4	Q_rqm5	Q_rqm7	RQM
mean	0.0354	0.0118	0.0393	0.1619	0.0350	0.0484	0.0298	0.0707	0.0628	0.0697	0.0525	0.2875
StD	0.0245	0.0096	0.0238	0.0692	0.0261	0.0324	0.0211	0.0467	0.0531	0.0456	0.0355	0.1310
Min	0.0002	0.0013	0.0014	0.064	0.0001	0.0002	0	0	0.0025	0	0	0.1124
Max	0.0882	0.0342	0.0843	0.3411	0.0959	0.1172	0.0813	0.1774	0.2280	0.1831	0.1314	0.6706

For this, two tests can be performed. The first one is the T Student test which tests if the two samples of mean μ_1 and μ_2 have the same mean. The null hypothesis (H0) and its alternative (H1) are:

$$\begin{cases} H0: \ \mu_1 - \mu_2 = 0 \\ H1: \ \mu_1 - \mu_2 \neq 0 \end{cases} \tag{6}$$

The second test is the F Fisher test which is the ratio of the two variances σ^2_{max} and σ^2_{min} of the samples. The null hypothesis (H0) and its alternative (H1) are:

$$\begin{cases} H0: \ \sigma^2_{max} / \sigma^2_{min} = 1 \\ H1: \ \sigma^2_{max} / \sigma^2_{min} > 1 \end{cases} \tag{7}$$

The table 3 presents the results of these two tests with a confidence of 95% and 99% for the two data RQM and T_piece for the 30 tries on the validation data set. The results on the learning data set are very similar. The data T_piece can take 3 values: type1; type2 and type3. So, the F test and the T test have to be performed two by two.

Table 3

	RQM		T_piece 1-2		T_piece 2-3		T_piece 1-3	
	F test	T test	F test	T test	F test	T test	F test	T test
Threshold 95%	1.092	1.961	1.070	1.961	1.077	1.961	1.070	1.961
Reject H0	100%	100%	96.67%	73.33%	43.33%	76.67%	96.67%	66.67%
threshold 99%	1.130	2.583	1.101	2.583	1.127	2.583	1.101	2.583
Reject H0	100%	100%	93.33%	60%	10%	63.33%	90%	36.67%

These results show that RQM has an important influence on residual. Even with a confidence interval of 99% no relation can be found between residuals obtained with RQM=4 and RQM=5. This is not the case with the T_piece data because the hypothesis of equality of mean (T test) is often not rejected and even the hypothesis of

equality of variance (F test) is accepted to 90% between T_piece type2 and type3 with a confidence interval of 99%.

5.2 Second approach

The RQM data has a great influence on the system. The comportment of the system is very different if RQM is 4 or if RQM is 5. So, an approach to deal with this fact is to make two different models in order to model the comportment of the system in these two cases and to switch from one to another with the value of RQM. This approach can be related to the multiple-model approach [4]. Two neural models have to be learned by using respectively the RQM=4 data uniquely and the RQM=5 data uniquely. These two neural networks have 11 inputs: Lg; diaGB; diaMoy; diaPB; T_piece; Q_trim; U_trim; Q_rqm; Q_rqm4; Q_rqm5; Q_rqm7. The learning begins with a structure using n_i=10 hidden neurons (5) which correspond to 131 parameters. 30 different sets of initials weights are used. The table 4 presents the mean and the standard deviation of the residuals obtained on the learning and the validation data sets by using uniquely RQM=4 data and RQM=5 data.

Table 4

	RQM = 4				RQM = 5			
	Learning residual	Learning residual	Validation residual	Validation residual	Learning residual	Learning residual	Validation residual	Validation residual
	Mean	StD	Mean	StD	Mean	StD	Mean	StD
Mean	12.36	478.00	8.40	528.33	7.22	332.80	7.75	335.54
StD	13.15	66.30	13.88	64.08	19.99	42.64	19.35	41.28
Min	-3.68	352.33	-19.55	376.15	-39.92	291.57	-37.28	291.83
(abs)	0.17		0.33		0.02		1.02	
Max	35.09	620.01	34.28	678.21	33.48	485.03	33.95	482.42

Table 5

	Lg	diaGB	diaMoy	diaPB	T_piece	Q_trim	U_trim	Q_rqm	Q_rqm4	Q_rqm5	Q_rqm7
mean	0.0225	0.0366	0.0371	0.0225	0.0257	0.0263	0.0189	0.0135	0.0157	0.0283	0.0168
StD	0.0313	0.0262	0.0262	0.0237	0.0227	0.0174	0.0208	0.0134	0.0132	0.0216	0.0106
Min	0.0005	0.0007	0.0020	0.0011	0.0000	0.0016	0.0014	0.0005	0.0003	0.0002	0.0011
Max	0.1305	0.0854	0.0870	0.1093	0.1123	0.0653	0.0791	0.0397	0.0451	0.0798	0.0398

Table 6

	Lg	diaGB	diaMoy	DiaPB	T_piece	Q_trim	U_trim	Q_rqm	Q_rqm4	Q_rqm5	Q_rqm7
mean	0.0135	0.0195	0.0227	0.0189	0.0405	0.0501	0.0275	0.0770	0.0722	0.0623	0.0530
StD	0.0257	0.0213	0.0272	0.0314	0.0453	0.0566	0.0267	0.0682	0.0695	0.0609	0.0445
Min	0.0009	0.0009	0.0002	0.0000	0.0000	0.0012	0.0016	0.0000	0.0000	0.0020	0.0005
Max	0.1058	0.0760	0.1053	0.1326	0.1330	0.2267	0.0920	0.2211	0.1847	0.2349	0.1577

The line (abs) presents the minimum of the mean in absolute value. It can be noticed that these values are very close to 0 to be compared with the results presented table 1 where the mean value is always greater than 12.35s. These results show that neural models present very similar residuals. In particular, the mean of the residuals is in the worst case, to 35.09s for the RQM=4 data and to 33.95s for the RQM=5 data. These results are to be compared with those presented table 1 where the mean of the residuals moves from 12.35s to 213.08s and where only 10% of the cases in learning and 16.67% of the case in validation give a mean lower than 30s. In order to determine if some dynamics present in the data aren't taken into account by the learning of the two neural models, the correlation between the different inputs and the residuals can be performed on the learning data set for the RQM=4 data(table 5) and for the RQM=5 data (table 6). Similar results can be obtained on the validation

data set. The table 5 and 6 present the mean, standard deviation, minimal and maximal values of the absolute value of the correlation coefficients obtained between the 30 residuals and the 11 inputs on the learning data set for the RQM=4 neural network and the RQM=5 neural network respectively. It can be noticed that, for the two neural models, no input is significantly correlated with the residual. In the worst case, the correlation coefficient obtained between Q_rqm5 input and the residual for the RQM=5 neural network is of 0.2349. However, for this input, in 76.67% of the cases, the correlation coefficient is lower than 0.01.

6 Epilogue

The use of neural network in order to construct a reduced model of emulation is investigated here. Within this framework, this paper focuses on the impact of discrete data on the learning results of the neural model. The results have shown that some discrete data (T_piece) are perfectly taken into account without adaptation. This can be explained by the fact that, even if, these discrete data are useful for the comprehension of the system, they don't produce some very different comportment and a unique neural model can explain all its evolution. However, some discrete data (RQM) implies that some different comportments of the process occur. These data imply that different models should be used in order to model all the system. Our perspectives are to investigate how to use these discrete data in the best way and validate this approach on different application cases. In addition, the system modelised may be changing. In this case, it may be interesting to use an on line learning rule in order to adapt the neural model to the evolution. Another perspective will be to investigate the advantages and disadvantages of this reduction model algorithm comparatively to a complete model. The computing times will be particularly studied.

7 References

[1] R.J. Brooks , and A.M. Tobias. Simplification in the simulation of manufacturing systems. Int. J. Prod. Res., 38(5): 1009-1027, 00.
[2] L. Chwif, R.J. Paul, and M.R. Pereira Barretto. Discret event simulation model reduction: A causal approach. Simulation Modelling Practice and Theory, 14: 930-944, 06.
[3] G. Cybenko. Approximation by superposition of a sigmoïdal function. Math. Control Systems Signals, 2(4): 303-314, 89.
[4] F. Delmotte, L. Dubois, and P. Borne. A general scheme for multi-model controller using trust. Mathematics and Computers in Simulation, 41:173-186, 96.
[5] K. Funahashi. On the approximate realisation of continuous mapping by neural networks. Neural Networks, 2: 183-192, 89.
[6] E. Goldratt., and J. Cox. The Goal : A process of ongoing improvement, North River Press; 2nd Revised edition, Great Barrington, USA, 92.
[7] G.S. Innis , and E. Rexstad. Simulation model simplification techniques. Simulation, 41: 7-15, 83.
[8] P. Lopez, and F. Roubellat. Ordonnancement de la production. Hermès, Paris, 01.
[9] D. Nguyen, and B. Widrow. Improving the learning speed of 2-layer neural networks by choosing initial values of the adaptative weights. Proc. of the Int. Joint Conference on Neural Networks IJCNN'90, 3: 21-26, 90.
[10] E.H. Page, D.M. Nicol, O. Balci, R.M. Fujimoto, P.A. Fishwick, P. L'Ecuyer, and R. Smith. An aggregate production planning framework for the evaluation of volume flexibility. Winter Simulation Conf., 1509-1520, 99.
[11] R. Setiono, and W.K. Leow. Pruned neural networks for regression. 6th Pacific RIM Int. Conf. on Artificial Intelligence PRICAI'00, Melbourne, Australia, 500-509, 00.
[12] A. Thomas, and P. Charpentier. Reducing simulation models for scheduling manufacturing facilities. European Journal of Operational Research, 161(1): 111-125, 05.
[13] A. Thomas, P. Genin, and S. Lamouri. Mathematical programming approaches for stable tactical and operational planning in supply chain and aps context. *Journal of Decision Systems*, 17: 425-455, 08.
[14] P. Thomas, G. Bloch, F. Sirou, and V. Eustache. Neural modeling of an induction furnace using robust learning criteria. J. of Integrated Computer Aided Engineering, 6(1): 5-23, 99.
[15] P. Thomas, and A. Thomas. Sélection de la structure d'un perceptron multicouches pour la réduction d'un modèle de simulation d'une scierie. CIFA'08, Bucarest, Roumanie, 08.
[16] T.E. Vollmann, W.L. Berry, and D.C. Whybark. Manufacturing, Planning and Systems Control. The Business One Irwin, 92.
[17] S.C. Ward. Argument for constructively simple models. J. of the Op. Research Society, 40(2): 141-153, 89.
[18] B.P. Zeigler. Theory of modelling and simulation. Wiley, New York, 76