# An Improved Hammerstein-Wiener System Identification with Application to Virtualized Software System

*by* Dharma Aryani

# An Improved Hammerstein-Wiener System Identification with Application to Virtualized Software System

Dharma Aryani, Liuping Wang and Tharindu Patikirikorala

*Abstract*— This paper proposes a system identification procedure to approximate virtualized software system dynamics within the framework of a Hammerstein-Wiener model. The approach is an extension of the existing works where the linear dynamics are estimated in Frequency Sampling Filter (FSF) structure and the inverse static output nonlinearity are synthesized in B-Spline curve functions. Furthermore, the issue on parameter selection for B-spline model approximation is addressed by using a data clustering method. An experimental test-bed of virtualized software system is established to generate experimental data which are used to confirm the performance of the proposed approach. The identification results have shown that the model efficacy is increased with the proposed approach because the dimension of the nonlinear model is reduced significantly while maintaining the desired accuracy.

## I. INTRODUCTION

The increasing range of applications and services maintained by software systems motivates the growing popularity of virtualization technology to intensify the computing performance. In a virtualization framework, the pool of shared resources needs to be attained dynamically by adjusting the resources according to a user requirement while acknowledging the requirement of other users [1]. These issues can be addressed efficiently on the basis of dynamic management [2]. The effectiveness of dynamic management for resource provisioning and performance properties between multiple users in a shared resources application is highly determined by the accuracy of model prediction. The relationship between resources provisioning and performance properties in virtualized software systems consists of linear and nonlinear dynamics [12]. However, most of existing literature pays little attention to the nonlinear dynamics by only using linear models in their control system designs, e.g. [3], [4].

One approach to nonlinear system identification is characterized by synthesizing linear and static nonlinear element [15]. This idea is evolved into Hammerstein-Wiener model estimation which is a favorable and generic class for nonlinear dynamic model estimation [11]. A wide range of applications have implemented Hammerstein-Wiener model structure because the method profoundly presents good performance in system identification for processes with significant nonlinearity issues [10], [8]. In the Hammerstein-Wiener structure, the nonlinear dynamics could be modeled

in its inversion form [5]. The purpose of capturing nonlinear dynamics in term of its inversion is to readily compensate the nonlinearity dynamics [7], [16] in the later applications. The nonlinear dynamics can be estimated accurately using a B-Spline curve function. B-spline curve estimation has been very well established in computer aided geometry design [19]. Several studies have demonstrated satisfactory results by using B-spline for nonlinear model estimation [17], [9]. However, the accuracy of the model prediction is very sensitive to the locations of knots especially when severe local nonlinearities exist.

This paper contributes to the improvement of the previous work in system identification of virtualized software system. The linear model is represented by non-parametric Frequency Sampling Filter (FSF) model and the inverse static nonlinear model in a polynomial function or a B-Spline function. Furthermore, the issue on parameter selection for B-spline model approximation is addressed by using a data clustering method which benefits for more precise knot points position. This approach reduces the model complexity because the parameters of a nonlinear model can be reduced significantly leading to less control points for the curve. The estimation of linear and nonlinear model parameters in Wiener block is performed in a straightforward manner. An experimental test-bed of virtualized software system is established to generate real observational data which are used to confirm the proposed approach performance.

The structure of the paper is outlined as follows. Section 2 covers the process description of the test-bed followed by dynamic nonlinearities formulation in Section 3. The algorithms for system identification and the identification results are addressed in Section 4. Section 5 gives the summary of the study as a conclusion.

## II. PROCESS DESCRIPTION

For experimental purposes, an architecture of virtualized software system is built on the scenario of multi-tier applications by implementing RUBiS. It is an online auction site benchmark in a three tiers application of e-commerce website which models the behavior of ebay.com. RUBiS has been used in several existing studies in software system management (eg. [4], [14]). A common pool of server resources is shared and each tier is hosted in a virtual machine. This type of shared resources environment systems has engaged both the embedded system and business domains due to its efficiency of infrastructure utilization and maintenance cost in data center. Figure 1 represents the structure of the test-bed. It consists of a server, a database and a client simulator
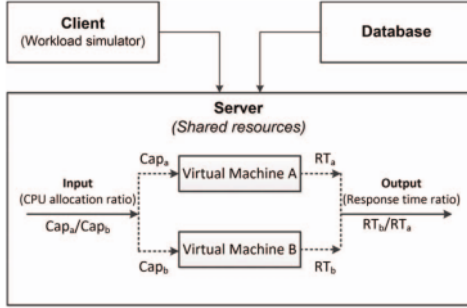
Dharma Aryani and Liuping Wang are with School of Electrical and Computer Engineering, RMIT University, Australia dharma.aryani@rmit.edu.au
Dharma Aryani is also with The State Polytechnic of Ujung Pandang, Indonesia
Tharindu Patikirikorala is with Faculty of Computer Engineering, Swinburne University of Technology, Australia

Fig. 1.   Virtualized software system



Fig. 2.   Hammerstein-Wiener structure

machine which are connected in an isolated network. Server is the shared infrastructure for two virtual machines (VMs) which will utilize the resources proportionally. Therefore, the server is functioning as the host machine while the two VMs are the guest machines. For virtualization, hypervisor is $Xen2.6$ is implemented to create VM by virtually partitioning the host machine hardware and to manage guests operating systems. This hypervisor comes with a credit-based scheduler that gives advantages for resource allocation. The scheduler dynamically allocates a certain share of CPU capacity to each VM. In order to adjust this function, an actuator was installed to feed the desired ratio of CPU to the system. Since the system sends intensive client requests, one CPU is dedicated only for the management of virtual machine ($dom0$) while all virtual machines are pinned to one CPU. This setting provides an efficiency for CPU capacity provisioning among VMs. In addition, a sensor program was attached to each VM to calculate the response time to incoming requests. Also, Apache Httpd 2.2 server is installed in all machines for customer application settings. The host and guest machines are running on $CentOS$ (Community Enterprise operating system). RUBiS workload generator is set up to simulate the workloads for each VM and the database of RUBiS benchmark was deployed in the other machine.

### III. Dynamic Nonlinearities

In a shared resources environment with unpredictable workload variations, the main challenge to manage performance guarantees is to provide a relative management scheme. To represent this scheme, input and output variables are defined as the ratio values of the related variables from two virtual machines. $Cap_a(k)$ and $Cap_b(k)$ are denoted as the CPU allocation for $VM_a$ and $VM_b$ respectively, leading to the total CPU capacity $Cap_{total} = Cap_a + Cap_b$. It yields the input equation to

$$u(k) = \frac{Cap_a(k)}{Cap_b(k)} \tag{1}$$

The portion of resource sharing is in the percentage of total CPU capacity where full CPU capacity equals to 100%. Therefore, to prevent a shortage of resources when workload requests suddenly increase in an unpredictable condition,
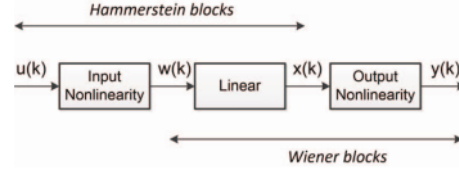
CPU share is constrained to a minimum capacity. In this experiment, $Cap_{a,min}(k), Cap_{b,min}(k) = 20$. So, it will ensure that a certain share of CPU capacity is assigned for each VM during runtime. On the output side, measured response time is the output of this virtualized system which is expressed in $y(k)$. $RT_a(k)$, $RT_b(k)$ represent response time to the workloads of $VM_a$ and $VM_b$ respectively. Thus,

$$y(k) = \frac{RT_b(k)}{RT_a(k)} \tag{2}$$

The setting of this input-output variables has been confirmed by a survey study by [13] that it is widely used for software system modeling because of its combination supporting the performance monitoring and management on the basis of relative guarantee management. Nonlinearities in the input variable is caused by the relationship between variable $Cap_a((k))$ and $Cap_b((k))$ in form of their ratio as in (1) and the minimum bound for the input signal $Cap_{min}$ which leads to static restriction of the operating points in the input variable. For instance, if $Cap_{total} = 100$ and $Cap_{min} = 20$, then the possible operating points configuration will be $n = 61$ points $(u = u_1, u_2, u_3, \cdots, u_{n-1}, u_n)$

$$u = \frac{20}{80}, \frac{21}{79}, \ldots, \frac{50}{50}, \ldots, \frac{79}{21}, \frac{80}{20}.$$

The order of these points are nonsymmetrical because the deviation between the points are unequal. Similar dynamics also appear in the output variable. If workloads variations for $VM_a$ and $VM_b$ have a large difference, the $RT_a(k)$ and $RT_b(k)$ will also have a large range of time differences. When the response time $VM_a$ is larger than the response time $VM_b$ $(RT_a(k) > RT_b(k))$, the output increases to a higher value. But when $(RT_a(k) < RT_b(k))$, the $y(k)$ value decreases in a higher rate as well.

### IV. System Identification Approach

The estimation of system dynamics in Hammerstein-Wiener model is comprised of a block-oriented model that consists of a linear model and static nonlinearities on both input and output elements (see Fig. 2). With this model, the nonlinear block precedes the linear block in the Hammerstein model and succeeds the linear block in the Wiener model.

#### A. Hammerstein block

Nonlinear model in the Hammerstein block is assigned to get the relationship between input signal $u$ and intermediate input $w$ in the form of inverse static nonlinearity function. To estimate the inverse static input nonlinearity, the operating

points $u(k)$ are transformed to equally spaced operating points. This approach has been used in the authors' previous studies [12], [18]. The technique was formulated by defining an intermediate variable $w$ where $w_{min} \leq w \leq w_{max}$. To have an equally spaced operating points, a fixed deviation is determined by $\delta w = \frac{w_{max} - w_{min}}{n-1}$.

$$w = w_1, w_2, w_3, \cdots, w_n$$

where $w_1 = w_{min}, w_2 = w_1 + \delta w, w_3 = w_2 + \delta w$ and $w_n = w_{max}$. Then, the respective operating points of input $u_i$ are mapped to the $w_i$ points. Furthermore, the relationship between data pairs from the mapping is approximated in inverse static nonlinear function $u(i) = f^{-1}(w(i))$. This inverse function is expressed in the polynomial form:

$$u(i) = \beta_0 + \beta_1 w(i) + \beta_2 w(i)^2 + \cdots + \beta_m w(i)^m \quad (3)$$

Then the coefficient vector of inverse static input nonlinearity is estimated by the least squares method, leading to

$$\hat{\Theta} = (\Phi^T \Phi)^{-1} (\Phi^T U) \quad (4)$$

where the parameter vector $\theta = \begin{bmatrix} \beta_0 & \beta_1 & \cdots & \beta_m \end{bmatrix}^T$ and data vector $\phi(i) = \begin{bmatrix} 1 & w(i) & \cdots & w(i)^m \end{bmatrix}^T$

### B. Wiener block

The linear model is represented in the Frequency Sampling Filters (FSF) model. Moreover, the identification of nonlinear model will be delivered in terms of inverse static nonlinearities by assigning B-spline curve function for inverse static nonlinear model estimation.

*1) Frequency Sampling Filter:* FSF model represents the model in fewer parameters than an FIR model and the estimation is unbiased and reliable to reduce noise effect [20]. Additionally, FSF model estimation only requires prior knowledge about settling time of the process with the assumption that the system is stable, linear, and time-invariant process. The transfer function of FSF model is

$$G(z) = \sum_{l=-\frac{M-1}{2}}^{\frac{M-1}{2}} G(e^{j\omega_l}) H_l(z) \quad (5)$$

where

$$H_l(z) = \frac{1}{M} \frac{1 - z^{-M}}{1 - e^{j\omega_l} z^{-1}} \quad (6)$$

where $M$ is filter order chosen as an odd integer number. Following the finding of [20], the FSF function can be formulated in a reduced $m$th-order FSF model which is written as follows

$$G(z) = \sum_{l=-\frac{m-1}{2}}^{\frac{m-1}{2}} G(e^{j\omega_l}) H_l(z) \quad (7)$$

Equation (7) is associated as the $j$-th FSF with $\omega_l = \frac{2\pi l}{M}$ as the center frequency in $l = 0, \pm 1, \pm 2, \cdots, \pm \frac{m-1}{2}$. $m$ is the effective order which indicates the significant parameters of FSF model and $M$ represents the order of individual FSF filters corresponding to the process settling time $M = T_s / \delta t$

and $\delta t$ is a sampling interval. $m$ is odd number and much smaller than $M$. The input and output variable for linear model in this study are $w(k)$ and $x(k)$ respectively. Therefore, the FSF model can be approximated as

$$x(k) = \sum_{l=-\frac{m-1}{2}}^{\frac{m-1}{2}} G(e^{j\omega_l}) f_l(k) \quad (8)$$

where $f_l(k) = H_l(z) w(k)$.

*2) B-Spline:* The estimation of B-spline function is motivated by initial finding of [6] which is widely known as "de-Boor" algorithm. The initial stage in a B-spline approximation is to select a set of parameters along data points range. These points will be used to calculate the knot vector intervals (knot spans). The intervals will direct the control points to form the curve. Therefore, this step determines the shape of the curve because inappropriate selection of parameters could lead to an unpredictable curve estimation [9]. Nonuniform parameters are implemented to provide flexibility for mapping parameters into curve by changing the knot spans length. In the B-spline form, a curve is represented by combining control points and the basis functions. A parametric B-spline curve $B(q)$ is defined by

$$B(q) = \sum_{i=0}^{n} N_{i,p}(q) P_i \quad (9)$$

where $N_{i,p}(q)$ are the normalized $p$th-degree basis function of B-Spline defined on a knot vector $Q = [Q_0, Q_1, \cdots, Q_{s+p}, Q_{s+p+1}]$ and $P_i$ $(i = 0, \cdots, s)$ are control points. In this study, nonperiodic knot vector is used which is characterized by the multiplicity at the first and the last knots. Suppose $s + 1$ parameters are selected from observational data $T = \{t_0, t_1, \ldots, t_s\}$ and the B-spline degree is $p$, there will be $r + 1$ knots to be used in the curve modeling using (10), where $r = s + p + 1$.

$$Q_0 = Q_1 = \cdots = Q_p = t_{min} \quad (10a)$$

$$Q_{j+p} = \frac{1}{p} \sum_{i=j}^{j+p-1} t_i \quad for \, j = 1, 2, \ldots, s - p \quad (10b)$$

$$Q_{r-p} = Q_{r-p+1} = \cdots = Q_r = t_{max} \quad (10c)$$

Basis function values for B-spline curve with degree $p$ are calculated using (11) and (12).

$$N_{i,0}(q) = \begin{cases} 1 & \text{if } Q_i \leq q < Q_{i+1} \\ 0 & \text{otherwise} \end{cases} \quad (11)$$

$$N_{i,p}(q) = \frac{q - Q_i}{Q_{i+p} - Q_i} N_{i,p-1}(q) + \frac{Q_{i+p+1} - q}{Q_{i+p+1} - Q_{i+1}} N_{i+1,p-1}(q) \quad (12)$$

$N_{i,p}$ is a composite curve of degree $p$ polynomials with joining points in knot span $[Q_i, Q_{i+p+1}]$. The representative formulation of B-spline curve $B(y(k))$ using $y(k)$ as the input and $x(k)$ as the output (see Figure 2) is

$$x(k) = \sum_{i=0}^{n} N_{i,p}(y(k)) P_i \quad (13)$$

*3) Data Clustering:* Data clustering is commenced from a given representation of $n$ objects and based on quantitative comparison in similarities or relationship, the objects are partitioned into $c$ smaller groups (clusters). The most considerable algorithm in partitioning clustering is K-means approach. In this study, K-means clustering is functioning as a tool to automatically select the dominant points of the data which is the center point of clustered data. The basic idea of K-means is to find the clusters that minimize the distance from data points in a cluster to the related cluster center. K-means is commonly used in Euclidean distance as the distance metric for the partitioning criterion. Suppose that $V = \{v_1, v_2, \cdots, v_n\}$ as the set of data points to be analyzed. This data will be partitioned into $c$ sets $R = \{R_1, R_2, \cdots, R_c\}$ in regards to the squared Euclidean distance which is defined as $d(v_i, \mu_j) = ||v_i - \mu_j||^2$. Therefore, the goal is to find $c$ cluster centers $\mu = \{\mu_1, \mu_2, \cdots, \mu_c\}$ which minimize the objective function as expressed as

$$J_{k-means} = \underset{R}{argmin} \sum_{j=1}^{c} \sum_{V \in [R_j]} ||V - \mu_j||^2 \qquad (14)$$

where $\mu_j$ is calculated as the mean of points in $S_j$ set. The algorithm requires two user-specified parameters: the number of clusters $c$ and initial setting for center point of each cluster (centroid). The basic algorithm for k-means clustering are as follows:

1) Select $c$ points within the range of data set as initial centroids of the clusters
2) For each data point, calculate the Euclidean distance to all centroids and assign each data points to the closest centroid to form $c$ clusters
3) Recompute the centroid of the new clusters by calculating the mean value of all points within each clusters.
4) Repeat (2) and (3) until the result convergence

To identify inverse static output nonlinearity, the procedure begins with a crude estimation where the models are estimated using random parameters. This step is important to have rough description of model characteristics. Then, K-means performs data clustering and select the center point based on the preference number of clusters. Since inverse static output nonlinearity represents the relationship of $y(k)$ as input and $x(k)$ as the output ($x(k) = f(y(k))$), data clustering and centroids calculation are based on data pairs of these variables. In the estimation, parameter $c$ which is the number of required clusters is set equal to the number of required parameters $T = \{t_0, t_1, \ldots, t_s\}$ minus 2 points ($s - 2$) since $t_0$ is equal to the minimum value of data range and $t_s$ is equal to the maximum point of data range. The results are $c$ centroid points in the form of data pairs of $\{y_i, x_i\}$. Therefore, the $y$ value is taken to represent the centroid value because it will be used as a parameter to estimate the B-spline with respect to the output variable $y(k)$. The centroid points resulted from this clustering process are used as a set of parameters for knot vector generation. Then, the knot vector of B-spline curve estimation is calculated using parameter $T = \{t_0, t_1, \ldots, t_s\}$ which is replaced by the centroid points $\mu = \{\mu_1, \mu_2, \cdots, \mu_c\}$.

This approach eliminates the pre-analysis effort that has to be performed in order to define the appropriate knots.

*4) Parameter Estimation:* Parameter estimation for linear model and inverse static nonlinearity in Wiener block is performed in a straightforward manner. Process output can be formulated by equating the linear model (8) and inverse static nonlinear model (13). Using the assumption that the inverse static function is a single-valued smooth function, it is also assumed that $P_0 = 1$ without lose of generality. The output function is composed as

$$N_{0,p}(\hat{y}(k)) = \sum_{l=-\frac{m-1}{2}}^{\frac{m-1}{2}} G(e^{j\omega_l}) f_l(k) - N_{1,p}(y(k))P_1 \\ - N_{2,p}(y(k))P_2 - \cdots - N_{i,p}(y(k))P_i \qquad (15)$$

To solve the prediction of model parameters, Equation (15) is transformed to matrix notations where the model is composed by parameter and regressor vector. Model parameter vector is

$$\Theta = \begin{bmatrix} G(e^{j0}) & G(e^{j\omega_1}) & \cdots & G(e^{-j\omega_{\frac{m-1}{2}}}) & P_1 & \cdots & P_s \end{bmatrix}^T \qquad (16)$$

and for data samples $k = 0, 1, \ldots, n$, the corresponding regression vector as

$$\phi(k)^T = \begin{bmatrix} A(t) & B(t) \end{bmatrix} \qquad (17)$$

where

$$A(t) = \begin{bmatrix} f_0(k) & f_1(k) & f_{-1}(k) & \cdots & f_{-\frac{m-1}{2}}(k) \end{bmatrix}$$

$$B(t) = \begin{bmatrix} -N_{1,p}(y(k)) & -N_{2,p}(y(k)) & \cdots & -N_{i,p}(y(k)) \end{bmatrix}$$

The output vector is

$$\hat{Y}^T = \begin{bmatrix} N_{0,p}(y(0)) & N_{0,p}(y(1)) & \cdots & N_{0,p}(y(n)) \end{bmatrix} \qquad (18)$$

Parameters of $\Theta$ can be solved by least squares estimation.

## V. IDENTIFICATION RESULTS

Data set for identification is from observed input and output data using the experimental test-bed. 400 samples of data pairs are generated using multi-level sinusoidal input with minimum 100 requests/s workloads for each client class, see Figure 3. Input variable settings are total available CPU capacity $Cap_{total} = 100$ and the minimum capacity allocation for each VM is 20 CPU caps. The validation of estimated
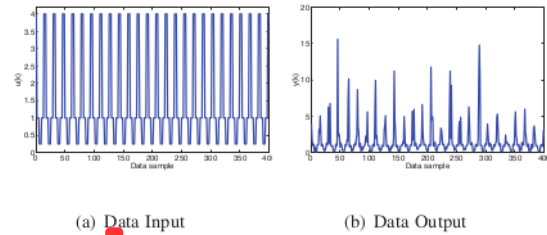


(a) Data Input          (b) Data Output

Fig. 3.   Data set for system identification

models is based on Mean Squared Error (*MSE*) value which

provides the difference between predicted result over the true value,

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(\hat{Y}_i - Y_i)^2 \qquad (19)$$

where $\hat{Y}$ is the response of predicted model and $Y$ is the observed response.

### A. Hammerstein Model

The inverse static input nonlinearity model $u(k) = f^{-1}(w(k))$ are approximated in polynomial function. $w(k)$ values are set arbitrarily by $w_{min} = -15$, $w_{max} = 15$ where $\delta w = 0.5$. Using least squares method (4), the polynomial functions is approximated in (20) with fitness value $R^2 = 0.99$ and MSE value 0.000068. After identifying input nonlinearity, the rest of the system will be estimated as a Wiener system where

$$u(k) = 4.17e^{-7}w(k)^5 + 9.34e^{-6}w(k)^4 + 1.018e^{-4}w(k)^3$$
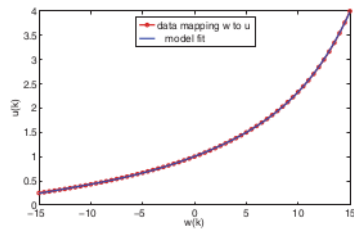$$+ 0.0028w(k)^2 + 0.08w(k) + 1.0045$$
$$\qquad (20)$$



Fig. 4.   Inverse static input nonlinearity $(u(k) = f^{-1}w(k))$

### B. Wiener Model

In the Wiener model, the FSF model represents the linear model and B-spline functions capture the inverse output nonlinearity. For nonlinear model evaluation, the nonuniform random parameters selection is compared to parameters selection using clustering method.

*1) Linear model using FSF structure:* For model analysis purpose, parameters $M$ and $m$ are presented to have a clear understanding about their influence on the modeling accuracy. Curve parameters for knots distribution in nonlinear model is set to 12 points. The prediction results are presented in Table 1.

Table 1. The MSE values of model estimation in different FSF parameter

| M/m | 30/9 | 30/13 | 30/17 | 70/9 | 70/13 | 70/17 |
|-----|------|-------|-------|------|-------|-------|
| MSE | 0.0203 | 0.0183 | 0.0170 | 0.0234 | 0.0216 | 0.0208 |

It can be seen from the table that increasing parameter $M$ leads to less accuracy of prediction. On the other hand, the variation of model order selection $m$ shows that the higher

the model order, the better prediction will be gained. Based on the MSE values, the best parameters are $M = 30$ and $m = 13$. These parameters will be used on the next system identification analysis in this study.

*2) Inverse static output nonlinearity in B-spline function:* The degree of spline is set to 3 for cubic spline model and knot vector is non-uniform. For parameter $T$ selection, two different approaches are examined, randomly selected parameters and the parameters selected from the centroids of clustered data.

*Parameters from random selection:* The sequence of parameters is selected randomly along data points of system output $(y)$. The number of parameters is varied $(6, 7, 8, 9)$ to express the effect of the length of parameter set when it is chosen randomly. It is inevitable that the predicted model is indefinite since any changes in random values leads to different estimated curve. For each estimation, the identification procedures are repeated 5 times and selected the best model with the lowest MSE value. It can be seen from the model fitting in Figure 5 and the MSE value in Table 2 that more parameters give better prediction for the B-spline curve. However, this fact raises the issue about redundancy in parameters of the predicted model.



(a) 6 curve parameters   (b) 7 curve parameters

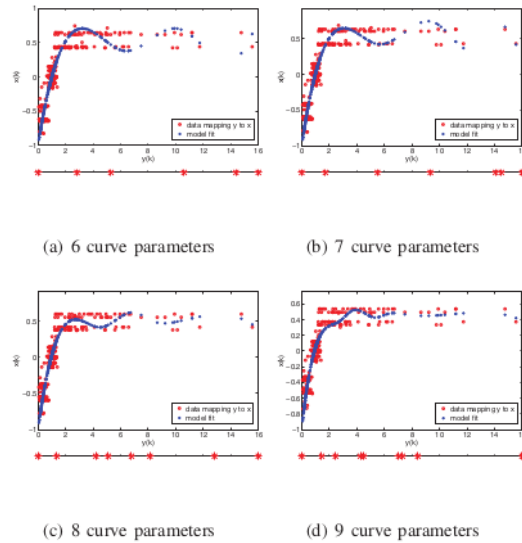(c) 8 curve parameters   (d) 9 curve parameters

Fig. 5.   Estimated B-spline model of inverse static output nonlinearity and the position of nonuniform parameters from random selection

Table 2. The MSE values of model estimation with non-uniform parameters from random selection

| Model Order | 6 | 7 | 8 | 9 |
|-------------|---|---|---|---|
| MSE | 0.0301 | 0.0281 | 0.0257 | 0.0239 |

*Parameters from the centroids of clustered data:* Results in Figure 6 show that more points occupied the complex and

dense area and fewer points on the flat area. This is the superiority of the proposed method over basic parameter selection method. More accurate models with clustering approach can be achieved with less parameter (See Table 3). With only 6 parameters from clustering method, the error prediction is less than the estimation with 9 parameters in random selection method and no multiplicity issue appears. In spite of the fact that increasing the number of cluster gives smaller MSE value of the prediction, the estimated model will be overfitting the real data which is unfavorable for control system design. Therefore, the curve parameters should be located properly to have an estimated model. Eventually, the proposed approach to apply clustering method for proper selection curve parameters is very constructive to address these related objectives.



(a) 6 curve parameters     (b) 7 curve parameters

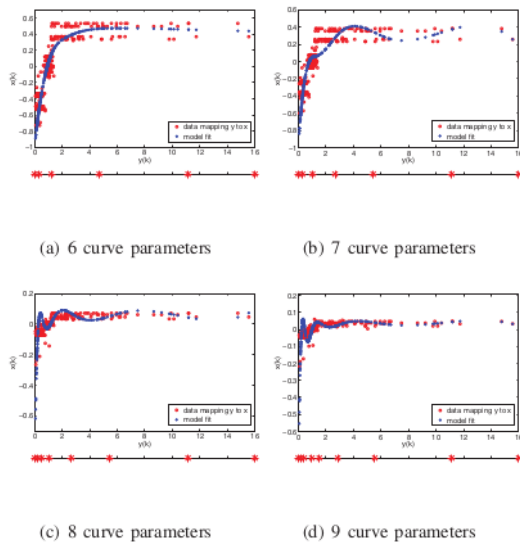(c) 8 curve parameters     (d) 9 curve parameters

Fig. 6. Estimated B-spline model of inverse static output nonlinearity and the position of nonuniform parameters from centroids of clustered data

Table 3. The MSE values of model estimation with non-uniform parameters from the centroids of clustered data

| Model Order | 6 | 7 | 8 | 9 |
|---|---|---|---|---|
| MSE | 0.0236 | 0.0198 | 0.0061 | 0.0039 |

## VI. Conclusions

In this paper, Hammerstein-Wiener structure has been used for model estimation of virtualized software system in a straightforward approach with significant improvement for Wiener models, where the linear dynamic is represented in FSF model and the nonlinear dynamic in B-Spline function. The considerably improved results are exhibited from the implementation of data clustering method in parameter selection to define dominant points as parameter for knot vector calculation, where more parameters are located for higher

density region and complex curve. The proposed approach contributes to the model parameter reduction and model efficacy in nonlinear system identification for virtualized software system.

## References

[1] P. Barham, B. Dragovic, K. Fraser, S. Hand, T. Harris, A. Ho, R. Neugebauer, I. Pratt and A. Warfield, Xen and the art of virtualization, Proceedings of the Nineteenth ACM Symposium on Operating Systems Principles, SOSP 2003, Bolton Landing, NY, USA, pp. 164–177.

[2] L. Chenyang, T. F. Abdelzaber, J. A. Stankovi, and S.H. Son, A feedback control approach for guaranteeing relative delays in web servers, Proceedings of Real-Time Technology and Applications Symposium, 2001, Seventh IEEE, 51–62.

[3] Y. Lu, T. F. Abdelzaher, C. Lu, L. Sha, and X. Liu, Feedback control with queueing-theoretic prediction for relative delay guarantees in web servers, IEEE Real-Time and Embedded Technology and Applications Symposium, pp 208–217, 2003.

[4] P. Padala, K.G. Shin, X. Zhu, M. Uysal, S. Wang, and S. Singhal, Adaptive control of virtualized resources in utility computing environments, SIGOPS/EuroSys European Conference on Computer Systems, EuroSys 07, pp 289–302, 2007.

[5] A.D. Kalafatis, L. Wang, and W.R. Cluett, Identification of Wiener-type nonlinear systems in a noisy environment, International Journal of Control, vol. 66, pp 923–941, 1997.

[6] C. DeBoor, A Practical Guide to Splines, Applied Mathematical Sciences, Springer-Verlag, New York, 1978.

[7] K. P. Fruzzetti, A. Palazoglu and K. A. McDonald, Nonlinear model predictive control using Hammerstein models, Journal of Process Control, vol.7, pp 31–41, 1997.

[8] X. Hong, Y. Gong, and S. Chen, A Wiener model for high power amplifiers using B-spline function approximation, International Conference of Digital Signal Processing, 6-8 July 2011, pp 1–5.

[9] X. Hong, R.J. Mitchell and S. Chen, System identification of Wiener systems with B-spline functions using De Boor recursion, International Journal of Systems Science, vol. 44(9), pp 1666–1674, 2013.

[10] F. Jurado, Hammerstein-model-based predictive control of microturbines, International Journal of Energy Research, vol. 7, pp 511–521, 2006.

[11] L. Ljung, System Identification: Theory for the users, Prentice Hall, Upper Saddle River, 1987.

[12] T. Patikirikorala, L. Wang, A. Colman and J. Han, Hammerstein-wiener nonlinear model based predictive control for relative QoS performance and resource management of software systems, Control Engineering Practice, vol. 20, pp. 49–61, 2011.

[13] T. Patikirikorala, J. Han, A. Colman, and L. Wang, A systematic survey on the design of self-adaptive software systems using control engineering approaches, Proceedings of Software Engineering for Adaptive and Self-Managing Systems (SEAMS), pp 33-42, 2012.

[14] T. Patikirikorala, L. Wang, A. Colman and J. Han, Differentiated performance management in virtualized environments using nonlinear control. http://www.ict.swin.edu.au/personal/tpatikirikorala/docs/tharindu_pds_2013.pdf, 2013.

[15] S. A. Billings and S.Y. Fakhouri, Nonlinear System Identification using the Hammerstein Model, International Journal of Systems Science, vol. 10, pp 567–578, 1979.

[16] H.H. Bloemen, T.J. Van den Boom, and H.B. Verbruggen, Model Based Predictive Control for Hammerstein Systems, Proceedings of the 39th IEEE Conference on Decision and Control, pp 4963–4968, 2000.

[17] Y. Zhu, Distillation column identification for control using Wiener model, Proceedings of the American Control Conference, San Diego, CA, USA, pp 3462–3466, 1999.

[18] D. Aryani, L. Wang, and T. Patikirikorala, Control oriented system identification for virtualized software systems, The 19th World Congress of the International Federation of Automatic Control, Cape Town, South Africa, 24–29 August 2014, pp. 4122–4127.

[19] L. Piegl and W. Tiller, The NURBS book, Springer-Verlag, New York, 1995.

[20] L. Wang and W. R. Cluett, Frequency sampling filters: an improved model structure for step response identification, Automatica, vol. 33, pp 939–944, 1997.

# An Improved Hammerstein-Wiener System Identification with Application to Virtualized Software System