Proceedings of the
47th IEEE Conference on Decision and Control
Cancun, Mexico, Dec. 9-11, 2008

On- and off-line model identification for power management of Web service systems

Mara Tanelli, Danilo Ardagna and Marco Lovera

Abstract— **In the context of Web services, hosting centers need to comply with the Service Level Agreements (SLAs) stipulated with their customers while minimizing the operating costs, mainly related with the energy expenses due to servers operations. The problem can be effectively formalized using system identification and control theory: the SLAs are translated into set-points for the response time of the servers and tracking performance are traded-off with energy saving objectives on the basis of suitable models for server dynamics. Two approaches are analysed for modeling such systems: recursive and LPV state space system identification methods. The suitability of both identification methods is investigated and their performance assessed on experimental data measured on a custom implementation of a workload generator and a micro-benchmarking Web service application.**

I. Introduction

In recent years, large service centers have been setup to provide computational capacity on demand to many customers who share a pool of IT resources. Furthermore, with the development of the Service Oriented Architecture (SOA), multiple service providers can offer functionally equivalent Web services which may differ for their Quality of Service (QoS) parameters. At run time, service requestors address their invocation to the most suitable provider according to their QoS preferences. QoS requirements are difficult to satisfy because of the high variability of Internet workloads. To handle workload variations, many service centers employ autonomic self-managing techniques (see, e.g., [1], [2], [3]), such that resources are dynamically allocated among running Web services on the base of short-term demand estimates. An emerging problem in this context is related to energy management. Power consumption per rack (i.e., the modules where physical servers are installed) has increased from 1kW in 2000 to 8kW in 2006 and is expected to rise to 20kW by 2010. IT analysts predict that, by 2012, up to 40% of an enterprise's technology budget will be consumed by energy costs, [4]. To reduce energy costs, early autonomic techniques switched servers on and off based on the service center workload, [2]. More recent proposals, see e.g., [5], [6] have started reducing the frequency of operation of servers by exploiting the Dynamic Voltage Scaling (DVS) mechanisms available on new servers. DVS varies both CPU supply voltage and operating frequency. The adoption of DVS is very promising, as power consumption is proportional to the cube of the operating frequency, while servers performance varies linearly with the operating frequency. Furthermore, DVS does not introduce any system overhead.

Several research contributions have proposed autonomic self-managing techniques and can be classified mainly in two categories: (i) utility-based optimization techniques, and (ii) feedback control-theoretic approaches. Utility-based approaches have been introduced to optimize the degree of user satisfaction by expressing their goals in terms of user-level performance metrics. Typically, the system is described by means of a performance model based on queueing theory, embedded within an optimization framework. Utility based approaches can handle multiple decision variables (e.g., admission control, resource allocation, load balancing, etc.) but are based on the assumption that the system is at *steady state*. Hence, these techniques are effective on a medium term control time horizon, e.g., half an hour, [2], [5]. Vice versa, genuine control-theoretic approaches can accurately model system transients and can adjust the system configuration within a very short time frame. The first control-oriented contributions applied to the management of Web services are reported in [7] and use feedback control to limit the utilization of bottleneck resources by means of admission control and resource allocation.

In the practice of control engineering, when a single control system must be designed to guarantee satisfactory closed-loop operation of a given plant in many different operating conditions, two broad classes of methods are available, namely gain scheduling and adaptive control: as is well known, a wide body of design techniques is available for both approaches, which can be reliably exploited provided that the corresponding modelling issues (off- and on-line, respectively) can be effectively dealt with.

As far as gain scheduling is concerned, Linear Parametrically Varying (LPV) models ([8], [9], [10], [11], [12]) have been proposed as a way of dealing with this kind of problems. In this paper, the results obtained in the identification of LPV models for the performance control of Web services are presented. Specifically, the suitability of LPV subspace model identification (SMI) methods for this task is verified. With respect to the approach presented in [5], where input/output (I/O) LPV identification was considered, the method adopted here is more appropriate to provide system models tailored to LPV control design, as they are directly identified in state space form and avoid all the issues - not addressed in [5] - related with equivalence notions between I/O and state space LPV realizations. Furthermore, state space LPV identification allows a straightforward extension

The work of M. Tanelli and M. Lovera has been partially supported by MIUR project "New methods for Identification and Adaptive Control for Industrial Systems". M. Tanelli, D. Ardagna and M. Lovera are with the Dipartimento di Elettronica e Informazione, Politecnico di Milano, Piazza Leonardo da Vinci 32, 20133 Milano, Italy {tanelli, ardagna, lovera}@elet.polimi.it. M. Tanelli is also with the Dipartimento di Ingegneria dell'Informazione e Metodi Matematici, Università degli Studi di Bergamo, Viale Marconi 5, 24044 Dalmine, Italy.

to the MIMO case, which is needed if more that one class of customers needs to be taken into account.

On the adaptive control side, the paper deals with the problem of recursively estimating a model for the dynamic response of the web server to varying workload conditions by resorting to recursive subspace model identification (RSMI) methods (see [13]), which have now reached a sufficient level of maturity both on the algorithmic and the theoretical side.

II. Problem statement

In this paper, a Web server that manages client requests to access a Web service application in a single queue with a FIFO policy is considered. In the queueing theory context, [14], the following quantities are commonly employed to describe the incoming workload over a time interval $[k\Delta t,\ (k+1)\Delta t]$, where Δt is the sampling time:

- λ_k indicates the rate at which the requests arrive to the server;
- s_k represents the *service time*, which is defined as the amount of time needed to process a request if the queue is empty. In common Web services applications, this quantity represents the CPU server time needed to serve the customer. Note, of course, that s_k is inversely proportional to the server CPU frequency. When physical servers are endowed with DVS capabilities, the effect of - say - lowering the CPU frequency when a light workload is present in the system, causes an increase of the effective CPU time needed to serve a request. Thus, the effective service time can be defined as $s_{u,k} = s_k/u_k$. The inverse of such quantity is commonly referred to as the service rate and indicated with μ_k;
- T_k is the server response time, that is the overall time a request stays in the system, which is given by the sum of the service time of such request and the time it spends in the queue before accessing the server;

As already mentioned, classical queueing theory provides a description of the system which relies on steady-state assumptions, and it is therefore reliable only over long time horizons. For control purposes, a dynamical model of the application server capable of capturing transients must be derived. We recall, in fact, that the aim of our work is to obtain a control-oriented dynamical description of the server behavior. The final control system will need to operate at a very fine grained time resolution (e.g., seconds), in order to ensure that SLAs requirements are met while minimizing energy costs. Furthermore, as the behavior of the server response time is highly time varying and the workload conditions change substantially within the same business day, the LPV and the recursive frameworks seem very promising for modeling such systems.

More precisely, in this work we focus on state space LPV models, in the form

$$\begin{aligned} x_{k+1} &= A(\delta_k)x_k + B(\delta_k)u_k \\ y_k &= C(\delta_k)x_k + D(\delta_k)u_k, \end{aligned} \tag{1}$$

where $\delta \in \mathbb{R}^s$ is the parameter vector and $x \in \mathbb{R}^n$, $u \in \mathbb{R}^m$, $y \in \mathbb{R}^l$. It is often necessary to introduce additional assumptions regarding the way in which δ_k enters the system matrices. In the following we will focus on two model classes.

1) Affine parameter dependence (LPV-A):

$$A(\delta_k) = A_0 + A_1\delta_{1,k} + \ldots + A_s\delta_{s,k} \tag{2}$$

and similarly for B, C and D, and where by $\delta_{i,k}, i = 1,\ldots,s$ we denote the i-th component of vector δ_k. This form can be immediately generalised to polynomial parameter dependence.

2) Input-affine parameter dependence (LPV-IA): this is a particular case of the LPV-A parameter dependence in which only the B and D matrices are considered as parametrically-varying, while A and C are assumed to be constant: $A = A_0$, $C = C_0$.

III. Off-line state space LPV identification algorithms

Identifying LPV models in general state space form is a difficult task. It is usually convenient to consider first the simplest form, i.e., the LPV-IA one, as its parameters can be retrieved by using conventional subspace model identification (SMI) algorithms for LTI systems by suitably extending the input vector. In this work the MOESP class of SMI algorithms has been considered. LPV-IA models also provide a useful initial guess for iterative methods which can be used for the identification of fully parameterised models in LPV-A form, along the lines of [8], [11].

Indeed, the classical way to perform linear system identification is by minimizing the error between the real output and the predicted output of the model. A similar approach can be used for LPV state-space systems of the form (1). Letting the system matrices of (1) be completely described by a set of parameters θ, identification can be carried out by minimizing the cost function

$$V_N(\theta) := \sum_{k=1}^{N} ||y_k - \widehat{y}_k(\theta)||_2^2 = E_N^T(\theta)E_N(\theta),$$

with respect to θ, where

$$E_N^T(\theta) = \left[\Big(y_1 - \hat{y}_1(\theta)\Big)^T \quad \cdots \quad \Big(y_N - \hat{y}_N(\theta)\Big)^T \right],$$

y_k denotes the measured output and $\widehat{y}_k(\theta)$ denotes the output of the LPV model to be identified. In general, the minimization of $V_N(\theta)$ is a nonlinear, nonconvex optimization problem. Different algorithms exist to numerically search for a solution to such an optimization problem. One popular choice is a gradient search method known as the Levenberg-Marquardt algorithm [15].

An important question that arises is how to choose the parameters θ to describe the system matrices in (1). Here a full parameterization is used: the interested reader is referred to [8] for a detailed discussion of the issues associated with the nonuniqueness of such a parameterisation.

IV. On-line state space identification algorithms

The problem of recursive subspace model identification (RSMI) has been an active area of research in recent years (see, e.g., [13] and the references therein). In this paper we will focus on the algorithms discussed in the cited references, which recursively estimate the $\{\mathbf{A}, \mathbf{B}, \mathbf{C}, \mathbf{D}\}$ state space matrices of an LTI model at each new data acquisition and can be therefore applied in a straightforward way to the recursive estimation of LPV-IA models. The proposed methods are based on the estimation of a basis for the observability subspace from the input-output (I/O) relation

$$\begin{aligned}\mathbf{y}_f(t) &= \mathbf{\Gamma}_f\mathbf{x}(t) + \mathbf{H}_f\mathbf{u}_f(t) + \mathbf{G}_f\mathbf{e}_f(t) = \\ &= \mathbf{\Gamma}_f\mathbf{x}(t) + \mathbf{H}_f\mathbf{u}_f(t) + \mathbf{b}_f(t) \end{aligned} \tag{3}$$

where the stacked vectors $\mathbf{y}_f$, $\mathbf{u}_f$ and $\mathbf{e}_f$ are defined as

$$\mathbf{y}_f(t) = \begin{bmatrix}\mathbf{y}^T(t) & \cdots & \mathbf{y}^T(t+f-1)\end{bmatrix}^T \in \mathbb{R}^{n_y f\times 1} \tag{4}$$

with $f > n_x$, $\mathbf{\Gamma}_f$ is the observability matrix

$$\mathbf{\Gamma}_f = \begin{bmatrix}\mathbf{C}^T & (\mathbf{CA})^T & \cdots & (\mathbf{CA}^{f-1})^T\end{bmatrix}^T, \tag{5}$$

$\mathbf{H}_f$ is the block Toeplitz matrix of the impulse responses from $\mathbf{u}$ to $\mathbf{y}$ and $\mathbf{b}_f = \mathbf{G}_f\mathbf{e}_f$ with $\mathbf{G}_f$ the block Toeplitz matrix of the impulse responses from $\mathbf{e}$ to $\mathbf{y}$. The class of techniques considered herein is based on the application of the so-called propagator method to the recursive estimation of $\mathbf{\Gamma}_f$. To this purpose, note that letting $\mathbf{Y}_f \in \mathbb{R}^{n_y f\times N}$, $\mathbf{U}_f \in \mathbb{R}^{n_u f\times N}$ and $\mathbf{B}_f \in \mathbb{R}^{n_y f\times N}$ be the Hankel I/O data matrices defined as

$$\mathbf{Y}_f(\bar{t}) = \begin{bmatrix}\mathbf{y}_f(t) & \cdots & \mathbf{y}_f(t+N-1)\end{bmatrix}, \tag{6}$$

with $N >> f > n$ and $\bar{t} = t + N + f - 2$, equation (3) can be written in matrix form as

$$\mathbf{Y}_f(\bar{t}) = \mathbf{\Gamma}_f\mathbf{X}(\bar{t}) + \mathbf{H}_f\mathbf{U}_f(\bar{t}) + \mathbf{B}_f(\bar{t}), \tag{7}$$

where $\mathbf{X}(\bar{t}) = \begin{bmatrix}\mathbf{x}(\bar{t}) & \cdots & \mathbf{x}(\bar{t}+N-1)\end{bmatrix}$. As is well known from the off-line SMI literature, a quantity directly related to the observability subspace can be obtained by computing the projection $\mathbf{Z}_f$ of $\mathbf{Y}_f$ on the kernel of $\mathbf{U}_f$

$$\mathbf{Z}_f(\bar{t}) = \mathbf{Y}_f(\bar{t})\mathbf{\Pi}_{\mathbf{U}_f^{\perp}}(\bar{t}) = \left(\mathbf{\Gamma}_f\mathbf{X}(\bar{t}) + \mathbf{B}_f(\bar{t})\right)\mathbf{\Pi}_{\mathbf{U}_f^{\perp}}(\bar{t}). \tag{8}$$

Considering now the time update of $\mathbf{Z}_f$

$$\mathbf{Z}_f(\bar{t}) = \begin{bmatrix}\mathbf{Z}_f(\bar{t}-1) & \mathbf{z}_f(\bar{t})\end{bmatrix}, \tag{9}$$

it is clear that the *observation vector* $\mathbf{z}_f(\bar{t})$ will carry all the relevant information for the estimation of the observability subspace contained in the data at time $\bar{t}$. Therefore, a two-step procedure for the recursive estimation of the system matrices can be devised:

1) the update of the observation vector $\mathbf{z}_f$ from the I/O measurements by considering a recursive formulation of the orthogonal projection performed in (8);
2) the estimation of a basis of $\mathbf{\Gamma}_f$ from this observation vector via the propagator method.

In this paper, the update of the observation vector is performed using the matrix inversion lemma. The idea (see [13]) is to recursively update the quantity

$$\mathbf{Z}_f(\bar{t}) = \mathbf{Y}_f(\bar{t})\left\{\mathbf{I} - \mathbf{U}_f^T(\bar{t})\left(\mathbf{U}_f(\bar{t})\mathbf{U}_f^T(\bar{t})\right)^{-1}\mathbf{U}_f(\bar{t})\right\} \tag{10}$$

at each new data acquisition, letting

$$\mathbf{U}_f(\bar{t}) = \begin{bmatrix}\sqrt{\beta}\mathbf{U}_f(\bar{t}-1) & \mathbf{u}_f(\bar{t})\end{bmatrix} \tag{11}$$

$$\mathbf{Y}_f(\bar{t}) = \begin{bmatrix}\sqrt{\beta}\mathbf{Y}_f(\bar{t}-1) & \mathbf{y}_f(\bar{t}),\end{bmatrix} \tag{12}$$

where $0 < \beta \leq 1$ is a forgetting factor, by applying the matrix inversion lemma to $\left(\mathbf{U}_f\mathbf{U}_f^T\right)^{-1}$.

Once the observation vector is estimated, the second step of the recursive subspace identification procedure consists in the online update of the observability matrix using the propagator method. Since $\mathbf{\Gamma}_f \in \mathbb{R}^{n_y f\times n_x}$ with $n_y f > n_x$, the extended observability matrix has at least n_x linearly independent rows, which can be gathered in a submatrix $\mathbf{\Gamma}_{f_1}$. Then, the complement $\mathbf{\Gamma}_{f_2}$ of $\mathbf{\Gamma}_{f_1}$ can be expressed as a linear combination of these n_x rows. So, there is a unique linear operator $\mathbf{P}_f \in \mathbb{R}^{n_x\times(n_y f - n_x)}$, named *propagator*, such that $\mathbf{\Gamma}_{f_2} = \mathbf{P}_f^T\mathbf{\Gamma}_{f_1}$. Furthermore, it is easy to verify that

$$\mathbf{\Gamma}_f = \begin{bmatrix}\mathbf{\Gamma}_{f_1}\\ \mathbf{\Gamma}_{f_2}\end{bmatrix} = \begin{bmatrix}\mathbf{\Gamma}_{f_1}\\ \mathbf{P}_f^T\mathbf{\Gamma}_{f_1}\end{bmatrix} = \begin{bmatrix}\mathbf{I}_{n_x}\\ \mathbf{P}_f^T\end{bmatrix}\mathbf{\Gamma}_{f_1}. \tag{13}$$

Thus, since $\operatorname{rank}\{\mathbf{\Gamma}_{f_1}\} = n_x$,

$$\operatorname{span}_{\text{col}}\{\mathbf{\Gamma}_f\} = \operatorname{span}_{\text{col}}\left\{\begin{bmatrix}\mathbf{I}_{n_x}\\ \mathbf{P}_f^T\end{bmatrix}\right\}. \tag{14}$$

Equation (14) implies that it is possible to estimate the observability matrix (in a particular basis) by estimating the propagator. This operator can be estimated from

$$\mathbf{z}_f(\bar{t}) = \mathbf{\Gamma}_f\tilde{\mathbf{x}}(\bar{t}) + \tilde{\mathbf{b}}_f(\bar{t}). \tag{15}$$

Indeed, applying a data reorganization to the observation vector so that the first n_x rows of $\mathbf{\Gamma}_f$ are linearly independent, (15) can be partitioned as

$$\mathbf{z}_f(t) = \begin{bmatrix}\mathbf{z}_{f_1}(t)\\ \mathbf{z}_{f_2}(t)\end{bmatrix} = \begin{bmatrix}\mathbf{I}_{n_x}\\ \mathbf{P}_f^T\end{bmatrix}\mathbf{\Gamma}_{f_1}\tilde{\mathbf{x}}(t) + \begin{bmatrix}\tilde{\mathbf{b}}_{f_1}(t)\\ \tilde{\mathbf{b}}_{f_2}(t)\end{bmatrix} \tag{16}$$

where $\mathbf{z}_{f_1} \in \mathbb{R}^{n_x\times 1}$ and $\mathbf{z}_{f_2} \in \mathbb{R}^{(n_y f - n_x)\times 1}$ are the components of $\mathbf{z}_f$ corresponding to $\mathbf{\Gamma}_{f_1}$ and $\mathbf{\Gamma}_{f_2}$ respectively. In the noise free case, it is easy to show that

$$\mathbf{z}_{f_2} = \mathbf{P}_f^T\mathbf{z}_{f_1}. \tag{17}$$

In the presence of noise, this relation no longer holds. However, by assuming we have collected (or are about to collect) N I/O measurements, an estimate of the propagator $\mathbf{P}_f$ can be obtained by minimising the criterion

$$J_{IV}(\mathbf{P}_f) = \left\|\hat{\mathbf{R}}_{\mathbf{z}_{f_2}\boldsymbol{\xi}}(N) - \mathbf{P}_f^T\hat{\mathbf{R}}_{\mathbf{z}_{f_1}\boldsymbol{\xi}}(N)\right\|_F^2, \tag{18}$$

where $\boldsymbol{\xi} \in \mathbb{R}^{n_\xi\times 1}$ is an instrumental variable, assumed uncorrelated with the noise but sufficiently correlated with the state vector $\mathbf{x}$, in the sense that $\mathbf{R}_{\mathbf{e}_f\boldsymbol{\xi}} = \mathbf{0}$ and $\operatorname{rank}\{\mathbf{R}_{\mathbf{x}\boldsymbol{\xi}}\} = n_x$.

By assuming that the input is sufficiently "rich", J_{IV} in (18) can be optimised recursively using the `EIVPM` algorithm (see [13] for details).

V. Experimental results

A. Testbed setting and experiment design

In the experimental framework, a workload generator and a micro-benchmarking Web service application have been used. The workload generator is based on a custom extension of the Apache *JMeter* 2.3.1 workload injector, [16], which allows to generate workload according to an open model [14] with a Poisson arrival process. Several analyses of actual e-commerce site traces, see for example [17], have shown that the Internet workload follows a Poisson distribution with a good approximation. The micro-benchmarking Web service application is hosted within the Apache *Tomcat* 6.0 application server, [18], where the number of concurrent threads has been limited to one in order to implement a FIFO scheduling. The Web service is a Java servlet designed to consume a fixed CPU time which allows to emulate the DVS of the physical server (a Pentium D machine with no DVS support). The adoption of a micro benchmarking application allows to validate the effectiveness of our approach both for computational intensive applications (where each request requires a significant CPU time) and workload intensive applications (where incoming requests require few CPU milliseconds only and the high computational requirements are due to the high number of incoming requests). The application has been instrumented to accurately determine the service time of each request; note, however, that this is not a limitation as there exist several techniques to assess the number of CPU cycles consumed by requests both at application level (e.g., the Application Resource Measurement API, [19]) or at operating system level (e.g., kernel-based measurements, [1]). For system identification purposes, the incoming request rate λ varies stepwise every 1 minute, with values between 0.15 req/s and 1.5 req/s. Note that, even though the average request rate is only changed each minute, the arrival times of the single requests t_{req} do not uniformly span the time interval, but are generated as samples of an exponential distribution with parameter λ, i.e., $t_{req} \sim \mathcal{E}(\lambda)$.

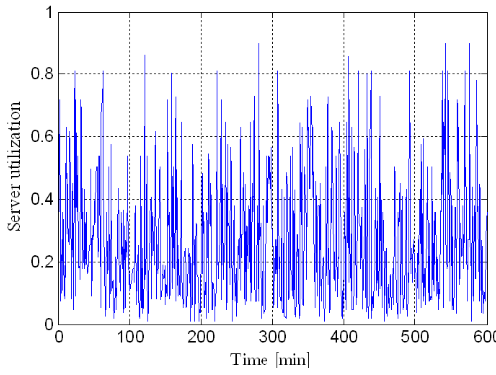

Fig. 1. Time history of the server utilization used in identification experiments.

During each time interval, each request requires a fixed deterministic service time s varied between 0.06 s and 1.1 s. Overall, 1,000 intervals have been considered and the average utilization of the physical server (defined as $\rho = \lambda s$, [7]), has been varied between 1% to 95%, in order to analyze the behavior of the physical server under light load and close to saturation conditions. Figure 1 shows a plot of the server utilization ρ used in the identification tests.

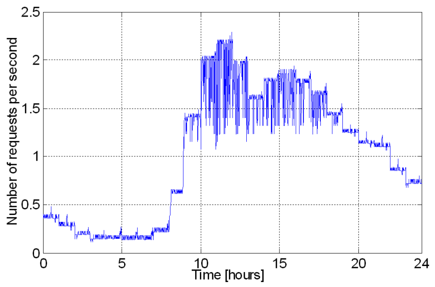

Fig. 2. Time history of the request rate applied during a validation test.

For model validation, a synthetic workload inspired by a real-world usage has been employed. The incoming workload reproduces a 24 hour trace obtained from an Internet Web site. The workload injector is configured to follow a Poisson process with request rate changing every minute according to the trace. Figure 2 reports, as an example, the request rate applied during a validation test. The request rate follows a bimodal distribution with two peaks around 12.00 and 16.00 (see Figure 2). The application service time follows a log-normal distributions (as observed for several real applications, [17]), such that, in each 1 minute time interval, the standard deviation is four times larger than the average service time. To emulate the DVS of a physical server, the average service time has been scaled to obtain $s_{u,k} = s_k/u_k$, where $u_k \in \{0.1, 0.2, \ldots, 1\}$ emulates the DVS of a physical server with 10 CPU frequency levels.

During the test, the average utilization of the physical server has been limited to 90%, to account for the fact that real systems usually implement admission control or other overload protection mechanisms in order to provide QoS guarantees, [7].

B. Off-line identification results

We consider as input the scaled service time $s_{u,k}$ and two possible choices for the scheduling parameters, namely the server utilization (i.e., $p_1 = \lambda s$, see also [5]) and the server utilisation and its square (that is, $p_2 = \left[\lambda s \quad (\lambda s)^2\right]$). The system output is the service response time T_k. The model order - after some comparative analysis - was set to $n = 2$ (which is same used in [5]), and the sampling time was initially set to $\Delta t = 1$ min. To quantitatively evaluate the models, both on identification and validation data, two metrics based on the results obtained in simulation (note that the focus on prediction would not be appropriate as the aim is to evaluate control-oriented models) will be considered: the percentage Variance Accounted For (VAF), defined as

$$VAF = 100\left(1 - \frac{Var[y_k - y_{sim,k}]}{Var[y(k)]}\right), \qquad (19)$$

where y_k is the measured signal and $y_{sim,k}$ is the output obtained simulating the identified model and the percentage average simulation error e_{avg}, computed as

$$e_{avg} = 100 \left(\frac{E\left[|y_k - y_{sim,k}|\right]}{E\left[|y_k|\right]} \right). \qquad (20)$$

In [20], based on the collected identification data, a comparison between the estimation performance of LPV-IA and LPV-A models was performed, which led to the conclusion that the LPV-IA model with scheduling parameters p_2 offers performances which are very close to those of the fully parametrised LPV-A model in terms of VAF and even better in terms of e_{avg}. Due to the decisive advantage offered by the simpler LPV-IA model structure, from here on we will focus on this type of models only. Figure 3 shows the results obtained with LPV-IA models on the identification data.

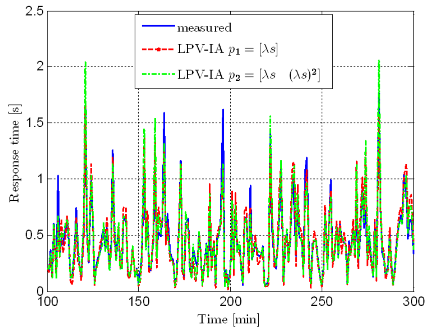

Fig. 3. Detail of the measured (solid line) and simulated response time obtained with an LPV-IA model with $p_1 = \lambda\, s$ (dashed line) and $p_2 = [\lambda\, s\; (\lambda\, s)^2]$ (dash-dotted line) on identification data.

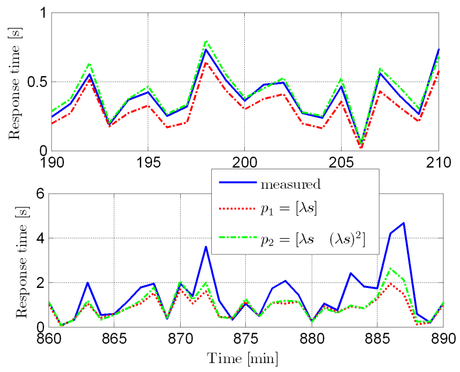

Fig. 4. Detail of the measured (solid line) and simulated response time obtained with $\Delta t = 1$ min and the identified LPV-IA models with $p_1 = \lambda\, s$ (dashed line) and $p_2 = [\lambda\, s\; (\lambda\, s)^2]$ (dash-dotted line) on validation data: light load case (top) and heavy load case (bottom).

Further, the LPV-IA models performance (with both choices of the scheduling parameters) has been tested on the validation data collected based on the bimodal request rate shown in Figure 2. The first results are shown in Figure V-B, where a detail of the measured and simulated response time is shown both in the light load (top) and heavy load (bottom) part of the data. By inspecting these figures it is apparent that the models cannot fully capture the heavy workload intensity, which results in a significant underestimation of the response time in heavy load conditions. This is confirmed by the quantitative analysis shown in Table I, where VAF and e_{avg} metrics obtained on validation data with the LPV-IA models identified with sampling interval $\Delta t = 1$ min are reported. To better analyze the results, note that in Table I the partial results on the light load part of the day (1-8)h and on the heavy load part of the day (9-20)h are also given. These data show that the identified models are extremely effective at light load (notably the performance in validation is better than that in identification), but lose generalization capabilities when tested at heavy load.

Valid. Performance $\Delta t = 1$ min	LPV-IA(p_1)	LPV-IA(p_2)
VAF on 24h	58.14%	65.18%
VAF light load (1-8)h	91.7%	89.8%
VAF heavy load (9-20)h	52.4%	60.1%
e_{avg} on 24h	25.7%	19.37%
e_{avg} light load (1-8)h	20.9%	10.5%
e_{avg} heavy load (9-20)h	29.2%	24.17%

TABLE I
PERFORMANCE OF THE IDENTIFIED MODELS WITH $\Delta t = 1$ MIN ON VALIDATION DATA.

This is due to the fact that, using as sampling interval $\Delta t = 1$ min, the request rates used as inputs in the identification phase are stepwise constant, and cannot account for the non-uniformity of the single arrival times within the interval itself. By analyzing the effective arrival times at which the single requests are issued in the system, the high correlation between such data and the peaks in the response time is apparent. To capture this variability, the sampling interval must be reduced. To this end, we processed again the identification data to extract the average values over a sampling interval $\Delta t = 10$s and two new LPV-IA models, one with $p_1 = \lambda\, s$ and the other with $p_2 = [\lambda\, s \quad (\lambda\, s)^2]$ have been identified (see Figure 5 for a plot of a detail of the results).

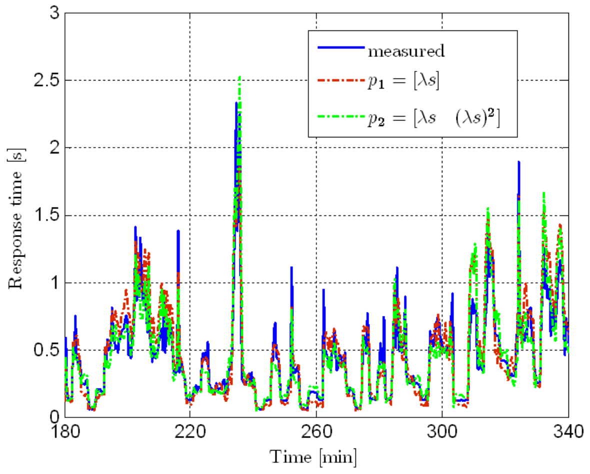

Fig. 5. Detail of the measured (solid line) and simulated response time obtained with $\Delta t = 10$ s an LPV-IA model with $p_1 = \lambda\, s$ (dashed line) and $p_2 = [\lambda\, s\; (\lambda\, s)^2]$ (dash-dotted line) on identification data.

It is interesting to analyse the performance of such models on validation data: on heavy load validation data, the effectiveness of having reduced the sampling interval is apparent:

now the models are capable of providing a response time which correctly follows the peaks of the measured one. This can be appreciated in the quantitative analysis shown in Table II, where VAF and e_{avg} metrics obtained on validation data with the LPV-IA identified with sampling interval $\Delta t = 10\,\mathrm{s}$ are reported. Overall, these final models allow to obtain a validation VAF of 71.5% and an $e_{avg} = 7.4\%$, which can be regarded as very good performances, especially for control purposes. Unfortunately, our results cannot be quantitatively compared to those in [5] - where I/O LPV identification was employed - as no performance metrics for the identification part were provided.

Valid. Performance $\Delta t = 10\,\mathrm{s}$	LPV-IA(p_1)	LPV-IA(p_2)
VAF on 24h	54.01%	71.5%
VAF light load (1-8)h	78.6%	80.2%
VAF heavy load (9-20)h	48.5%	67.1%
e_{avg} on 24h	20.3%	7.4%
e_{avg} light load (1-8)h	20.02%	2.5%
e_{avg} heavy load (9-20)h	22.5%	9.25%

TABLE II

PERFORMANCE OF THE IDENTIFIED MODELS WITH $\Delta t = 10\mathrm{S}$ ON VALIDATION DATA.

C. On-line identification results

The RSMI algorithm presented in Section IV has been applied directly to the validation dataset corresponding to the request rate time history illustrated in Figure 2, considering three different choices for the definition of the input vector u, namely $u_1 = s$, $u_2 = [s \quad \lambda\, s]$ and $u_3 = [s \quad \lambda\, s \quad (\lambda\, s)^2]$. More precisely, the role of the choice of the forgetting factor β in the RSMI algorithm has been analysed.

A picture of the achievable performance as a function of both model structure and forgetting factor is provided by Table III: as can be seen from the results reported therein, it is possible to obtain an almost perfect simulation of the measured server response time by feeding the model with information on the actual server utilisation and by choosing a sufficiently small forgetting factor. Clearly, the choice of small values of β implies that the estimated state space matrices will have a very high variability.

VAF ($\Delta t = 1\,\mathrm{min}$)	RSMI(u_1)	RSMI(u_2)	RSMI(u_3)
$\beta = 0.75$	87.40%	96.06%	99.53%
$\beta = 0.95$	55.34%	68.82%	83.98%
$\beta = 1$	31.13%	48.68%	65.01%
e_{avg} ($\Delta t = 1\,\mathrm{min}$)	RSMI(u_1)	RSMI(u_2)	RSMI(u_3)
$\beta = 0.75$	29.02%	15.16%	6.01%
$\beta = 0.95$	55.88%	45.59%	33.40%
$\beta = 1$	81.12%	64.94%	46.91%

TABLE III

PERFORMANCE OF THE RSMI ALGORITHM ON VALIDATION DATA WITH $\Delta t = 1\,\mathrm{MIN}$.

VI. CONCLUDING REMARKS AND FUTURE WORK

This paper presented the results obtain in the application of on- and off-line model identification methods for the performance control of Web services. Specifically, the suitability of subspace LPV and recursive identification methods has been checked against experimental data measured on a custom implementation of a workload generator and a micro-benchmarking Web service application. Future work will be devoted to further validate our approach on real applications and to extend the models considering a multi-class framework.

REFERENCES

[1] B. Urgaonkar and P. Shenoy, "Sharc: Managing CPU and Network Bandwidth in Shared Clusters," *IEEE Transactions on Parallel and Distributed Systems*, vol. 15, no. 1, pp. 2–17, 2004.

[2] D. Ardagna, M. Trubian, and L. Zhang, "SLA based resource allocation policies in autonomic environments," *Journal of Parallel and Distributed Computing*, vol. 67, no. 3, pp. 259–270, 2007.

[3] B. Urgaonkar, G. Pacifici, P. J. Shenoy, M. Spreitzer, and A. N. Tantawi, "Analytic modeling of multitier Internet applications," *ACM Transaction on Web*, vol. 1, no. 1, January 2007.

[4] V. Metha, "A Holistic Solution to the IT Energy Crisis," 2007. [Online]. Available: http://greenercomputing.com/

[5] W. Qin and Q. Wang, "Modeling and control design for performance management of web servers via an LPV approach," *IEEE Transactions on Control Systems Technology*, vol. 15, no. 2, pp. 259–275, 2007.

[6] D. Kusic and N. Kandasamy, "Risk-Aware Limited Lookahead Control for Dynamic Resource Provisioning in Enterprise Computing Systems," in *ICSOC 2004 Proc.*, 2004.

[7] T. Abdelzaher, K. G. Shin, and N. Bhatti, "Performance Guarantees for Web Server End-Systems: A Control-Theoretical Approach," *IEEE Transactions on Parallel and Distributed Systems*, vol. 15, no. 2, March 2002.

[8] L. Lee and K. Poolla, "Identification of linear parameter-varying systems using nonlinear programming," *ASME Journal of Dynamic Systems, Measurement and Control*, vol. 121, no. 1, pp. 71–78, 1999.

[9] M. Lovera, M. Verhaegen, and C. T. Chou, "State space identification of MIMO linear parameter varying models," in *Proceedings of the International Symposium on the Mathematical Theory of Networks and Systems*, Padua, Italy, 1998, pp. 839–842.

[10] B. Bamieh and L. Giarré, "Identification of linear parameter varying models," in *Proc. of the IEEE Conference on Decision and Control*, Phoenix, AZ, USA, 1999.

[11] V. Verdult, "Nonlinear system identification: A state-space approach," Ph.D. dissertation, University of Twente, Faculty of Applied Physics, Enschede, The Netherlands, 2002. [Online]. Available: http://www.tup.utwente.nl

[12] F. Previdi and M. Lovera, "Identification of a class of nonlinear parametrically varying models," *International Journal on Adaptive Control and Signal Processing*, vol. 17, pp. 33–50, 2003.

[13] G. Mercère and M. Lovera, "Convergence analysis of instrumental variable recursive subspace identification algorithms," *Automatica*, vol. 43, no. 8, pp. 1377–1386, 2007.

[14] L. Kleinrock, *Queueing Systems*. John Wiley and Sons, 1975.

[15] J. J. Moré, "The Levenberg-Marquardt algorithm: Implementation and theory," in *Numerical Analysis*, ser. Lecture Notes in Mathematics, G. A. Watson, Ed. Berlin: Springer Verlag, 1978, vol. 630, pp. 106–116.

[16] Apache, "Apache JMeter." [Online]. Available: http://jakarta.apache.org/jmeter/index.html

[17] L. H. Gomes, C. Cazita, J. M. Almeida, V. Almeida, and W. M. Jr., " Workload Models of SPAM and Legitimate E-mails," *Performance Evaluation*, vol. 64, no. 7-8, pp. 690–714, 2007.

[18] Apache, "Apache Tomcat." [Online]. Available: http://tomcat.apache.org/

[19] The Open Group, "Application Resource Measurement - ARM." [Online]. Available: http://www.opengroup.org/tech/management/arm/

[20] M. Tanelli, D. Ardagna, and M. Lovera, "LPV model identification for power management of web service systems," in *Submitted - 2008 IEEE Multi-conference on Systems and Control, San Antonio, USA.*, 2008.