2009 American Control Conference
Hyatt Regency Riverfront, St. Louis, MO, USA
June 10-12, 2009

Dynamic Engine Modeling Through Linear Programming Support Vector Regression

Zhao Lu, Jing Sun, Dongkyoung Lee and Ken Butts

Abstract— **In this paper, we develop a dynamic model for an internal combustion engine using Support Vector Regression (SVR). In particular, a linear programming SVR (LP-SVR) approach is investigated. The computational advantages and generalization capability of the LP-SVR dynamic engine model are illustrated through a case study, where a model is developed for an L4 gasoline engine. Simulation results are reported to demonstrate the effectiveness of proposed approach and to illustrate the trade-offs among different modeling attributes.**

I. INTRODUCTION

To meet the ever demanding customer expectations and increasingly stringent government emission regulations, and in response to the stiff global market competition, automotive companies are striving to develop advanced powertrain and vehicle technologies under substantial time and cost constraints. Along with the introduction of new engine hardware innovations, there also come with new engine control development processes, where more emphasis has been placed on model-based design and calibration [1] as a way to deal with complexity and to assure robust performance. In this process, a dynamic engine model is a critical tool that facilitates the control strategy design, performance analysis and overall system integration. Much effort has been devoted to engine modeling by the industrial and academic engine control community, and many different types of models with different levels of complexity and fidelity can be found in literature, for example, in [2], [3] and references therein.

Existing engine models that serve to support engine control and calibration are often phenomenological in nature. They capture the cause-effect relationship between the inputs (i.e., the adjustable engine control variables such as throttle, spark, valve timing, etc.) and outputs (namely, the response variables such as the torque, temperature, emissions). The models generally can be categorized into two classes: the grey-box models where certain first principles are combined with system identification techniques in the modeling process, and the black-box models which rely primarily on data and regression techniques for model development. While representative grey-box models can be found in literature (see [2], [3] and references therein), several black-box models have been developed using neural networks [4], [5].

Zhao Lu is with Electrical Engineering Department, Tuskegee University, Tuskegee, Alabama, 36088 zlu@ieee.org

Jing Sun and Dongkyoung Lee are with Department of Naval Architecture and Marine Engineering and the Department of Mechanical Engineering respectively at the University of Michigan, Ann Arbor, MI 48109 jingsun@umich.edu, keyenlee@umich.edu

Ken Butts is with North America Technical Center, Toyota Motor Corporation, Ann Arbor, Michigan, 48105 ken.butts@tema.toyota.com

Even though the thermodynamic processes associated with engine operation have many statistical characteristics due to the empirical engine mapping practice, engine modeling has been rarely attacked from the statistical learning perspective. The aim of our work is to explore the state-of-the-art statistical learning methodologies to develop a framework for engine modeling and control strategy development. In particular, we investigate a Linear Programming Support Vector Regression (LV-SVR) approach which exploits the functional representation capability of the supporting vector regression and the computational advantages of linear programming optimization. Using an L4 engine as a case study, we show that a dynamic engine model developed using the proposed LP-SVR approach has the desired model accuracy as well as computational efficiency. It should be pointed out that several papers can be found in literature that use support vector machine as a modeling tool for automotive systems, such as [6], [7], [8]. However, we believe that this is the first application that exploits the LP-SVR for engine modeling.

The rest of the paper will be organized as follows: In the next section, we provide the general background of statistical learning, with particular emphasis on support vector regression. Then, the soft-constrained LP-SVR, including details of the algorithm and its implementation, is introduced. As a case study, we apply the LP-SVR to the engine modeling of an L4 gasoline engine, and demonstrate the computational and generalization characteristics of the LP-SVR. Future works are outlined together with the conclusions.

The following generic notations will be used throughout this paper: lower case symbols such as $x, y, \alpha, \cdots$ refer to scalar valued objects, lower case boldface symbols such as $\mathbf{x}, \mathbf{y}, \cdots$ and underlined Greek letters such as $\underline{\beta}, \underline{\gamma}, \cdots$ refer to vector valued objects, and finally capital boldface symbols will be used for matrices.

II. SVR FOR DYNAMIC SYSTEM IDENTIFICATION

During the past decade, as a powerful statistical learning technique for predictive data analysis, the support vector machine (SVM) has been gaining popularity in the field of machine learning[9], [10], [11]. Essentially, the SVM is a universal approach for solving problems of multidimensional function estimation. It is based on the Vapnik-Chervonenkis (VC) theory [9]. Initially, it was designed to solve pattern recognition problems, where in order to find a decision rule with good generalization capability, a small subset of the training data, called the support vectors (SVs), are selected. Experiments showed that it is easy to recognize high-dimensional identities using a small basis constructed

978-1-4244-4524-0/09/$25.00 ©2009 AACC

from the selected support vectors. Since the inception of this concept, the idea of support vector learning has also been successfully applied to various fields such as regression, density estimation and solving linear operator equations. When SVM is employed to tackle the problems of function approximation and regression estimation, it is often referred to as the support vector regression (SVR) [9], [12]. Research on this topic has shown that the SVR type of function approximation is very effective [10], [11], [12], especially for the cases involving high-dimensional input space. Another important advantage for using SVR in function approximation is that the number of free parameters in the function approximation scheme is equal to the number of support vectors. Such a number can be obtained by defining the width of a tolerance band through the ε-insensitive loss function. Thus, the selection of the number of free parameters can be directly related to the approximation accuracy and does not have to depend on the dimensionality of the input space or other factors as that in the case of multilayer feedforward neural networks.

The ε-insensitive loss function is attractive because, unlike the quadratic and Huber cost functions where all the data points will be support vectors, the SV solution derived can be sparse. In the realm of data modeling, the sparsity plays a crucial role in improving the generalization performance and computational efficiency. It has been shown that sparse data representations reduce the generalization error as long as the representation is not too sparse, which is consistent with the principle of parsimony [13], [14].

For the purpose of modeling complex nonlinear dynamical systems using sparse representation, SVR has been exploited in the context of nonlinear black-box system identification very recently [15], [16], [17], [18]. Applications to automotive systems have also been reported [6], [7], [8]. Although it is believed that the formulation of SVR embodies the structural risk minimization principle to combine the excellent generalization properties with a sparse model representation, some data modeling practicians have begun to realize that the capability of the standard quadratic programming SVR (QP-SVR) method to produce sparse models has perhaps been overstated. For example, it has been shown that the standard SVR technique does not always lead to parsimonious models in system identification [17]. A recent study has compared the standard SVR and uniformly regularized orthogonal least squares (UROLS) algorithms using time series prediction problems, and has found that both methods have similar excellent generalization performance but the resulting model from SVR is not sparse enough [19]. It is explained that the number of support vectors found by a quadratic programming algorithm in an SVR is only an upper bound on the number of necessary and sufficient support vectors, due to the linear dependencies between support vectors in the feature space.

On the other hand, due to the distinct mechanism used for selecting the support vectors, the linear programming support vector regression (LP-SVR) is advantageous over QP-SVR in model sparsity, ability to use more general kernel functions and fast learning based on linear programming [20], [21]. The idea of linear programming support vector machines is to use the kernel expansion as an ansatz for the solution, but to use a different regularizer, namely the norm of the coefficient vector. In other words, for LP-SVR, the nonlinear regression problem is treated as a linear one in the kernel space, rather than in the feature space as in the case of QP-SVR. Obviously, the choice of kernel plays a critical role in the performance of LP-SVR.

III. SOFT-CONSTRAINED LINEAR PROGRAMMING SVR

Conceptually there are some similarities between the LP-SVR and QP-SVR. Both algorithms adopt the ε-insensitive loss function, and use kernel functions in the feature space. Consider the regression in the following set of functions

$$f(\mathbf{x}) = \mathbf{w}^T \underline{\phi}(\mathbf{x}) + b \tag{1}$$

with given training data, $\{(\mathbf{x}_1, y_1), \cdots, (\mathbf{x}_l, y_l)\}$ where l denotes the total number of exemplars, $\mathbf{x}_i \in \mathscr{R}^n$ are the inputs and $y_i \in \mathscr{R}$ are the target output data. The nonlinear mapping $\underline{\phi} : \mathscr{R}^n \mapsto \mathscr{R}^m (m > n)$ maps the input data into a so-called high dimensional feature space (which can be infinite dimensional) and $\mathbf{w} \in \mathscr{R}^m, b \in \mathscr{R}$. In ε-SV regression, the goal is to find a function $f(\mathbf{x})$ that has at most ε deviation from the actually obtained targets y_i for all the training data, and at the same time, is as flat as possible. In the conventional support vector method, one aims at minimizing the empirical risk subject to elements of the structure

$$\begin{array}{ll} \textit{minimize} & \frac{1}{2}\|\mathbf{w}\|^2 + C\sum_{i=1}^{l}(\xi_i + \xi_i^*) \\ \textit{subject to} & \left\{ \begin{array}{c} y_i - \langle \mathbf{w}, \underline{\phi}(\mathbf{x}) \rangle - b \le \varepsilon + \xi_i \\ \langle \mathbf{w}, \underline{\phi}(\mathbf{x}) \rangle + b - y_i \le \varepsilon + \xi_i^* \\ \xi_i, \xi_i^* \ge 0 \end{array} \right. \end{array} \tag{2}$$

where ξ_i and ξ_i^* are the slack variables corresponding to the size of the excess deviation for positive and negative direction respectively. This is a classic quadratic optimization problem with inequality constraints, and the optimization criterion penalizes data points whose $y-$values differ from $f(\mathbf{x})$ by more than ε. The constant $C > 0$ determines the trade-off between the flatness of f and the amount up to which deviations larger than ε can be tolerated. By defining the ε-insensitivity loss function,

$$L(y_i - f(\mathbf{x}_i)) = \left\{ \begin{array}{ll} 0, & if\ |y_i - f(\mathbf{x}_i)| \le \varepsilon \\ |y_i - f(\mathbf{x}_i)| - \varepsilon, & otherwise \end{array} \right. \tag{3}$$

the optimization problem (2) is equivalent to the following regularization problem,

$$\textit{minimize}\ R_{reg}[f] = \sum_{i=1}^{l} L(y_i - f(\mathbf{x}_i)) + \lambda \|\mathbf{w}\|^2 \tag{4}$$

where $f(\mathbf{x})$ is in the form of (1) and $\lambda\|\mathbf{w}\|^2$ is the regularization term. The ε-insensitivity loss function (3) defines an ε-tube. According to the well-known Representer Theorem [11], the solution to the regularization problem (4) can be

written as the SV kernel expansion

$$f(\mathbf{x}) = \sum_{i=1}^{l} \beta_i k(\mathbf{x}_i, \mathbf{x}) \tag{5}$$

provided $k(\mathbf{x}_i, \mathbf{x}_i) = 1$, where $k(\mathbf{x}_i, \mathbf{x})$ is the kernel function. Obviously, the kernel functions $k(\mathbf{x}_i, \mathbf{x})$ play a crucial role in determining the characteristics of the model (5). In support vector learning algorithms, the kernel function provides an elegant way of working in the feature space, thereby avoiding all the troubles and difficulties inherent in high dimensions. Several commonly-used kernel functions in the literatures are:

- Linear kernel:
$$k(\mathbf{x}, \mathbf{x}') = \langle \mathbf{x}, \mathbf{x}' \rangle$$
where $\langle \cdot, \cdot \rangle$ denotes the inner product of two vectors;
- Gaussian radial basis function (GRBF) kernel:
$$k(\mathbf{x}, \mathbf{x}') = exp\left(\frac{-\|\mathbf{x} - \mathbf{x}'\|^2}{2\sigma^2} \right);$$
- Polynomial kernel:
$$k(\mathbf{x}, \mathbf{x}') = (1 + \langle \mathbf{x}, \mathbf{x}' \rangle)^q;$$
- Sigmoid kernel:
$$k(\mathbf{x}, \mathbf{x}') = tanh(\alpha \langle \mathbf{x}, \mathbf{x}' \rangle + \gamma);$$
- Thin plate spline kernel:
$$k(\mathbf{x}, \mathbf{x}') = \|\mathbf{x} - \mathbf{x}'\|^2 ln \|\mathbf{x} - \mathbf{x}'\|.$$

$\sigma, q, \alpha, \gamma$ are adjustable parameters of the kernel functions.

Defining

$$\underline{\beta} = [\beta_1, \beta_2, \cdots, \beta_l]^T,$$

LP-SVR replaces (4) by

$$minimize\ R_{reg}[f] = \sum_{i=1}^{l} L(y_i - f(\mathbf{x}_i)) + \lambda \|\underline{\beta}\|_1 \tag{6}$$

where $f(\mathbf{x})$ is in the form of (5) and $\|\underline{\beta}\|_1$ denotes the l_1-norm of $\underline{\beta}$ in the coefficient space. This regularization problem is equivalent to the following constrained optimization problem

$$\begin{array}{ll} minimize & \|\underline{\beta}\|_1 + C\sum_{i=1}^{l}(\xi_i + \xi_i^*); \\ subject\ to & \left\{ \begin{array}{c} y_i - \sum_{j=1}^{l} \beta_i k(\mathbf{x}_j, \mathbf{x}_i) \le \varepsilon + \xi_i, \\ \sum_{j=1}^{l} \beta_i k(\mathbf{x}_j, \mathbf{x}_i) - y_i \le \varepsilon + \xi_i^*, \\ \xi_i, \xi_i^* \ge 0. \end{array} \right. \end{array} \tag{7}$$

From the geometric perspective, it can be followed that $\xi_i \xi_i^* = 0$ in SV regression. Therefore, it is sufficient to just introduce slack variables $\xi_i \ge 0$ in the constrained optimization problem (7). Thus, we arrive at the following formulation of SV regression with fewer slack variables

$$\begin{array}{ll} minimize & \|\underline{\beta}\|_1 + 2C\sum_{i=1}^{l} \xi_i; \\ subject\ to & \left\{ \begin{array}{c} y_i - \sum_{j=1}^{l} \beta_i k(\mathbf{x}_j, \mathbf{x}_i) \le \varepsilon + \xi_i, \\ \sum_{j=1}^{l} \beta_i k(\mathbf{x}_j, \mathbf{x}_i) - y_i \le \varepsilon + \xi_i, \\ \xi_i \ge 0. \end{array} \right. \end{array} \tag{8}$$

In an attempt to convert the optimization problem above into a linear programming problem, we decompose β_i and $|\beta_i|$ as follows

$$\beta_i = \alpha_i^+ - \alpha_i^-, |\beta_i| = \alpha_i^+ + \alpha_i^- \tag{9}$$

where $\alpha_i^+, \alpha_i^- \ge 0$. It is worth noting that the decompositions in (9) are unique, i.e., for a given β_i there is only one pair (α_i^+, α_i^-) which satisfies both equations. Furthermore, both variables can not be larger than zero at the same time, i.e., $\alpha_i^+ \cdot \alpha_i^- = 0$. In this way, the l_1-norm of $\underline{\beta}$ can be written as

$$\|\beta\|_1 = \left(\underbrace{1, 1, \cdots, 1}_{l}, \underbrace{1, 1, \cdots, 1}_{l} \right) \begin{pmatrix} \underline{\alpha}^+ \\ \underline{\alpha}^- \end{pmatrix} \tag{10}$$

where $\underline{\alpha}^+ = (\alpha_1^+, \alpha_2^+, \cdots, \alpha_l^+)^T$ and $\underline{\alpha}^- = (\alpha_1^-, \alpha_2^-, \cdots, \alpha_l^-)^T$. Furthermore, the constraints in the formulation (8) can be written in the following vector form

$$\begin{pmatrix} \mathbf{K} & -\mathbf{K} & -\mathbf{I} \\ -\mathbf{K} & \mathbf{K} & -\mathbf{I} \end{pmatrix} \begin{pmatrix} \underline{\alpha}^+ \\ \underline{\alpha}^- \\ \underline{\xi} \end{pmatrix} \le \begin{pmatrix} \mathbf{y} + \varepsilon \\ \varepsilon - \mathbf{y} \end{pmatrix} \tag{11}$$

where $\mathbf{K}_{ij} = k(\mathbf{x}_i, \mathbf{x}_j), \underline{\xi} = (\xi_1, \xi_2, \cdots, \xi_l)^T$ and $\mathbf{I}$ is an $l \times l$ identity matrix. Thus, the constrained optimization problem (8) can be implemented by the following linear programming problem with the variables $\underline{\alpha}^+, \underline{\alpha}^-$, and $\underline{\xi}$:

$$\begin{array}{ll} minimize & \mathbf{c}^T \begin{pmatrix} \underline{\alpha}^+ \\ \underline{\alpha}^- \\ \underline{\xi} \end{pmatrix} \\ subject\ to & \begin{pmatrix} \mathbf{K} & -\mathbf{K} & -\mathbf{I} \\ -\mathbf{K} & \mathbf{K} & -\mathbf{I} \end{pmatrix} \begin{pmatrix} \underline{\alpha}^+ \\ \underline{\alpha}^- \\ \underline{\xi} \end{pmatrix} \le \begin{pmatrix} \mathbf{y} + \varepsilon \\ \varepsilon - \mathbf{y} \end{pmatrix} \end{array} \tag{12}$$

where $\mathbf{c} = \left(\underbrace{1, 1, \cdots, 1}_{l}, \underbrace{1, 1, \cdots, 1}_{l}, \underbrace{2C, 2C, \cdots, 2C}_{l} \right)^T$.

In the QP-SVR case, the set of points not inside the tube coincides with the set of SVs. While, in the LP context, this is no longer true (although the solution is still sparse), any point could be an SV, even if it is inside the tube. Actually, the sparse solution still can be obtained in LP-SVR even though the size of the $\varepsilon-$insensitive tube was set to zero, due to the soft constraints used. However, sparser solution can usually be obtained by setting non-zero ε.

Many standard software packages can be used to solve the LP problem given by (12). In our work, MATLAB© functions are used to perform the LP problem formulated for dynamic engine modeling.

IV. APPLICATION OF LP-SVR FOR ENGINE MODELING

As a case study to evaluate the applicability of the LP-SVR for engine modeling, we consider an L4 gasoline engine as the platform. In addition to the spark, fuel and air, this engine is equipped with the intake VVT (variable valve timing) mechanism. A high fidelity proprietary engine model, which has been extensively validated and is being used for engine

calibration and strategy development activities, is used as a "virtual" engine in this study to generate data and to serve as a test-bed for model validation.

A data set was generated by exciting the high fidelity model with randomly selected (but within feasible range) input combinations (manifold air flow (MAF, g/s), air-to-fuel ratio (λ), spark (deg), and VVT (deg) commands) and engine operating conditions (namely, for different engine speed and load combinations). This assures the richness of the data set. The output variables, for which models will be developed, include engine torque, speed, emissions, exhaust temperature and the covariance of the cylinder pressure which measures the combustion stability. The total data set consists of 1000 data points, of which 500 are used as the training data set for developing the LP-SVR model, and the rest are used for validating the resulting model.

Once the data is generated and the kernel is defined, one can set up the LP problem in the form of (12) for this engine modeling application. The linear programming function of MATLAB© can be used to solve the optimization problem for different engine response variables, namely the engine torque (Tq), feedgas emissions including hydrocarbon (HC) and nitrogen oxide (NOx), engine exhaust and catalyst temperatures (Texh and Tcat respectively), and the covariance of the cycle-to-cycle cylinder pressure (Cov). To determine the model structure, namely the input and output regression windows, we have experimented with different model structures and leveraged our previous extensive experience in engine modeling. The choice of the 2-step output regression and one-step input regression reflect the dominant manifold filling and engine thermal inertia dynamic effects. The final regression engine models have the following form:

$$\begin{aligned}
y_{Tq}(k)(Nm) &= f_{tq}(y_{Tq}(k-1), y_{Tq}(k-2), \mathbf{u}(k-1)) \\
y_{HC}(k)(g/s) &= f_{HC}(y_{HC}(k-1), y_{HC}(k-2), \mathbf{u}(k-1)) \\
y_{NOx}(k)(g/s) &= f_{NOx}(y_{NOx}(k-1), y_{NOx}(k-2), \mathbf{u}(k-1)) \\
y_{Texh}(k)(K) &= f_{Texh}(y_{Texh}(k-1), y_{Texh}(k-2), \mathbf{u}(k-1)) \\
y_{Tcat}(k)(K) &= f_{Tcat}(y_{Tcat}(k-1), y_{Tcat}(k-2), \mathbf{u}(k-1)) \\
y_{Cov}(k) &= f_{Cov}(y_{Cov}(k-1), y_{Cov}(k-2), u_1(k-1))
\end{aligned} \tag{13}$$

where $\mathbf{u} = [MAF, VVT, spark, \lambda, N]^T$, $u_1 = MAF$ and N is the engine speed, and the sampling time is chosen as $0.1sec$.

Figure 1 shows the structure of the LP-SVR engine model.

Remark: Once the engine model structure is fixed, other model parameters, such as the kernel function, the ε and C used in the SVR, are chosen through trial-and-error to achieve a good trade-off between model accuracy and model sparsity. More details on the choice of those parameters will be discussed in the next section in connection with the numerical results.

V. SIMULATION AND VALIDATION RESULTS

The numerical results of the LP-SVR model are analyzed in terms of several key pertinent attributes:

- Sparsity: Sparsity is quantified by the ratio (in percentage) of the number of resulting supporting vectors to the number of total data points in the training data set. Lower sparsity is desirable for several reasons, as it leads to less memory cells required for model implementation, lower computation time in simulation, and, in general, better generalization capabilities.
- Optimization time: This is the time used to determine the support vectors and to derive the approximating function. Even though the current modeling is done off-line, for which the computational time is usually not an issue, understanding the computational implications of the algorithm will help us to evaluate their potential for future on-line applications. In this paper, all the times given are based on using MATLAB in solving the optimization on Intel Pentium (R)4 3.4 GHz processors with 2047MB RAMS.
- Absolute and relative modeling accuracy: The absolute accuracy for each output variable is measured by the root mean squares (rms) value of the error between the SVR model prediction (i.e., the output of our model) and the output of the high fidelity engine model (namely, the output of the "virtual" engine). The relative accuracy is defined as the ratio (in percentage) of the absolute rms error over the range of the corresponding variable. For most engine modeling work, 5% of relative error is often considered as an excellent regression result.
- Simulation time: The CPU times used to simulate the resulting model in MATLAB© for a given input profile on Intel Pentium (R)4 3.4 GHz processors with 2047MB RAMS are given in Tables I and II. Whenever the numbers are compared for different models, the same input profile is assumed for all different models.

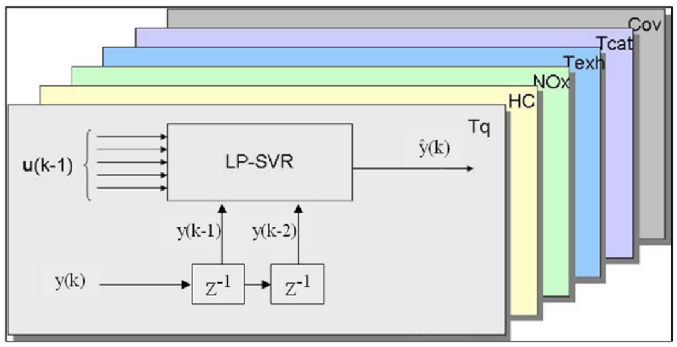

Fig. 1. DIAGRAM of LP-SVR ENGINE MODEL

The results of the LP-SVR modeling are summarized in Table I for each function. The excellent generalization property of the model can be observed by noting the relative accuracies in the training and validation: they are very close. Other metrics for measuring generalization properties, such as the n-fold cross validation accuracy [24], can be used to perform more systematic evaluation, and will be considered in our future work. The last row in the table gives the kernel functions used for each output. We experimented 4 different kernel functions: the linear kernel (lin.) and GRBF kernel with different σ values (rbf1,rbf2,rbf3,rbf4),

TABLE I
SUMMARY OF LP-SVR ENGINE MODEL

	Tq	HC	NOx	Texh	Tcat	Cov
sparsity (%)	3.12	2.39	0.94	0.63	0.62	0.21
Rel. rms for training(%)	7.56	5.44	5.93	3.07	0.08	3.28
Rel. rms for validation(%)	6.97	5.72	5.54	4.40	0.17	4.97
Opt. time (sec)	242.1	331.7	96.70	64.10	146.6	95.38
Sim. Time for validation (sec)	0.73	0.52	0.20	0.14	0.13	0.03
kernel	rbf1	rbf2	rbf3	rbf4	lin.	lin.

TABLE II
COMPARISON OF LP-SVR (LP) AND QP-SVR (QP) MODELS

	Tq		Tcat		NOx	
	LP	QP	LP	QP	LP	QP
Sparsity (%)	3.12	20.92	0.62	3.85	0.94	36.94
Rel. rms for training	7.59	6.51	0.08	0.07	5.93	5.73
Rel. rms for validation	6.97	6.37	0.17	0.09	5.54	5.34
Opt time for training	242.2	14915	146.6	14421	96.70	15130
Sim time for validation	0.73	6.69	0.13	0.70	0.20	9.50
kernel	rbf1	rbf1	lin.	lin.	rbf3	rbf3

and chose the one that gives the best training result for each output variables in terms of modeling accuracy. One can see the computational times required for optimization and for simulation are directly correlated to the sparsity of the model and to the type of the kernel used. The linear kernel requires less computational time, as one would expect. For those functions that use the same type of kernel, the computational time of the resulting model is almost linearly dependent on the sparsity (see (Tq, HC, NOx,Texh) for the GBRF kernel and (Tcat, Cov) for the linear kernel).

Figure 2 shows the time traces of three outputs of the LP-SVR model for the validation run, comparing three outputs of the LP-SVR with the high fidelity model. One can see that the prediction of the LP-SVR model is very close to that given by the "virtual" engine, indicating an excellent generalization. The plots for other outputs are omitted, as similar trends are exhibited.

To understand the advantages/disadvantages of the LP-SVR modeling approach, especially in relation to other known methodologies, a regression model is developed using standard least squares algorithm to fit the data using a standard polynomial function. Moreover, an SVR model using quadratic programming (QP-SVR) is also developed to clarify the trade-offs between the computational effort and generalization capability. The LP-SVR and QP-SVR models use the same inputs and the same model structure (i.e., the same regression window for inputs and output), and their main attributes are summarized in Table II for three representative output variables: engine torque, Tcat and NOx that compares LP-SVR with QP-SVR. Note, however, that the parameters of the SVR model (namely the values

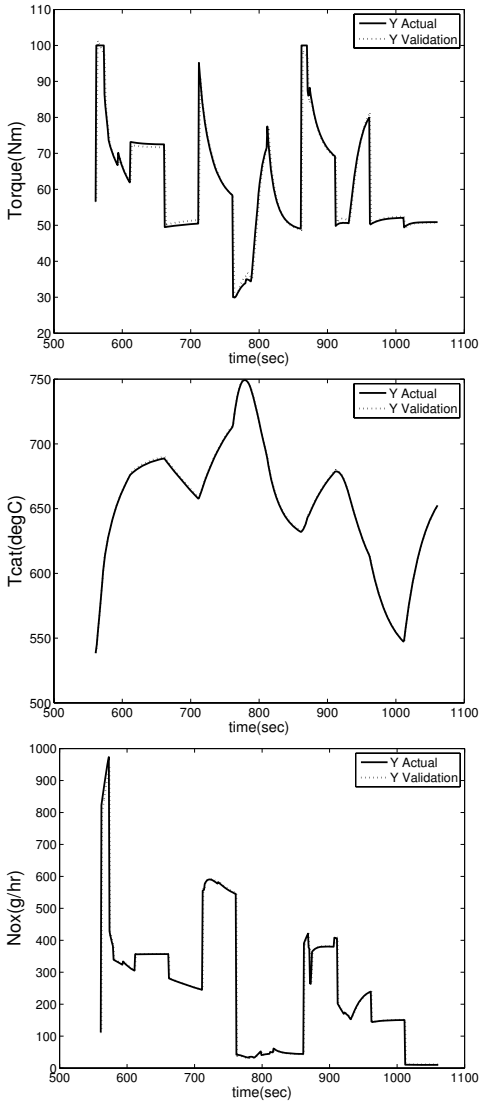

Fig. 2. SIMULATION RESULTS OF THE LP-SVR MODEL FOR VALIDATION.

for ε and C) are different for LP-SVR and QP-SVR, and they are optimized for each case to achieve the best model accuracy. Similar characteristics are revealed by models for the other output variables which are not listed here due to space limitation. One can see from Table 2 that, while the LP-SVR model has slightly higher rms, its substantial computational advantage (in terms of optimization time and simulation time) over the QP-SVR is very obvious.

VI. CONCLUSIONS

In this paper, we presented a novel approach for engine modeling using linear programming support vector regression. Through a case study, we demonstrated that the dynamic engine model developed using the proposed approach has several special characteristics that make it attractive. First, compared to the quadratic programming SVR, the linear programming SVR has better sparsity which directly translates to model simplicity. Second, the LP-SVR

is computationally much more tractable than QP-SVR, as measured by the time it takes to perform the optimization in determining the support vectors. This computational efficiency is achieved without substantial compromise for the model accuracy. It also makes the LP-SVR more attractive for on-line applications. Third, the striking similarity in the prediction accuracy between the training phase and validation phase confirms the excellent generalization property. The LP-SVR also provides several mechanisms, such as the choice of the kernel and the ε-insensitive loss function, that will allow the user to tune the process to achieve the desired modeling results.

At this stage, the use of statistical learning tools such as the support vector regression has been primarily limited to off-line applications, such as the one illustrated in this paper. The resulting models are also validated for series-parallel implementation where the measured output is used for future predictions. While the series-parallel model is useful for prediction purpose, a parallel model where the measured output is not used for prediction is needed for stand-alone simulation applications. The excellent sparsity and computational simplicity demonstrated by the LP-SVR, however, suggest the potential of this approach for on-line applications such as adaptation. Additional mechanisms, such as iterative training and weights updating, have been explored to assure the validity of model in the parallel implementation. Our future work will focus on exploring the statistical learning tools for on-line control and adaptation, aimed at improving the engine control robust performance over its life-cycle.

REFERENCES

[1] T. Ueda, A. Ohata, "Trends of Future Powertrain Development and the Evolution of Powertrain Control Systems," in *Proc. 2004 SAE - Convergence Conference*, October, 2004.

[2] L. Guzzella and C.H. Onder, *Introduction to Modeling and Control of Internal Combustion Engine Systems*, Springer, London, 2004.

[3] Jeffrey A. Cook, Jing Sun, Julia H. Buckland, Ilya V. Kolmanovsky, Huei Peng, and Jessy W. Grizzle, "A Survey: Automotive Powertrain Control," *Asian Journal of Control*, vol. 8, no. 3, pp. 237-260, 2006.

[4] Mohamed Ayeb, Dirk Lichtenthaler, Heinz J. Theuerkauf,H. Winsel, "SI Engine Modeling Using Neural Networks," SAE paper No. 980790,1998.

[5] X. Dovifaaz, M. Oulandsine, A. Rachid, and G. Bloch, "Neural Modeling for Diesel Engine Control," *Proc. 15th Triennial World Cong.,* Barcelona, Spain, 2002.

[6] I. V. Kolmanovsky and E. Gilbert, "Suppport Vector Machine-based Determination of Gasoline Direct Injection Engine Admissible Operating Envelope," *SAE Paper* 2002-01-1301, 2002.

[7] J. Wang, K. H. Guo and Y. L. Lei, "Support Vector Machine Theory Based Shift Quality Assessment for Automotive Mechanical Transmission (AMT)," *SAE Paper* 2007-01-1588, 2007.

[8] E. Gani and C. Manzie, "Indicated Torque Reconstruction from Instananeous Engine Speed in a Six-Cylinder SI Engine Using Support Vector Machines," *SAE Paper,* 2005-01-0030, 2005.

[9] V. N. Vapnik, *The Nature of Statistical Learning Theory*, 2nd Edition, Springer-Verlag, 2000.

[10] N. Cristianini, J. Shawe-Taylor, *An Introduction to Support Vector Machines and Other Kernel-based Learning Methods*, Cambridge University Press, Cambridge, MA, 2000.

[11] B. Schlkopf, A.J. Smola, *Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond*, MIT Press, Cambridge, MA, 2002.

[12] A.J. Smola, B. Schlkopf, "A tutorial on support vector regression", *Statistics and Computing*, vol.14, pp.199-222, 2004.

[13] N. Ancona, R. Maglietta, E. Stella, "Data representation in kernel based learning machines", *Proceedings of the International Conference on Artificial Intelligence and Soft Computing*, 2004.

[14] S. Chen, "Local regularization assisted orthogonal least squares regression", *Neurocomputing*, vol. 69, pp.559-585, 2006.

[15] A. Gretton, A. Doucet, R. Herbrich, P.J.W. Rayner, B. Schlkopf, "Support vector regression for black-box system identification", *Proceedings of the 11th IEEE Workshop on Statistical Signal Processing*, 2001.

[16] J.L. Rojo-Alvarez, M. Martinez-Ramon, M. Prado-Cumplido, A. Artes-Rodriguez, A.R. Figueiras-Vidal, "Support vector machines for nonlinear kernel ARMA system identification", *IEEE Trans. Neural Networks*, vol. 17, pp.1617-1622, 2006.

[17] P.M.L. Drezet, R.F. Harrison, "Support vector machines for system identification", *Proceedings of the UKACC International Conference on Control*, 1998.

[18] W.C. Chan, C.W. Chan, K.C. Cheung, C.J. Harris, "On the modeling of nonlinear dynamic systems using support vector neural networks", *Engineering Applications of Artificial Intelligence*, vol.14, pp.105-113, 2001.

[19] K.L. Lee, S.A. Billings, "Time series prediction using support vector machines, the orthogonal and the regularized orthogonal least-squares algorithms", *Int. J. Systems Science*, vol.33, pp.811-821, 2002.

[20] V. Kecman, *Learning and Soft Computing: Support Vector Machines, Neural Networks, and Fuzzy Logic Models*, MIT Press, Cambridge, MA, 2001.

[21] I. Hadzic, V. Kecman, "Support vector machines trained by linear programming: theory and application in image compression and data classification", *Proceedings of the IEEE 5th Seminar on Neural Network Applications in Electrical Engineering*, 2000.

[22] S.H. Lee, R.J. Howlett, S.D. Walters, C. Crua, "Modeling and control of internal combustion engines using intelligent techniques", *Cybernetics and Systems*, vol. 38, pp.09-533, 2007.

[23] T. Holliday, A.J. Lawrance, T.P. Davis, "Engine-mapping experiments: a two-stage regression approach", *Technometrics*, vol.40, pp.120-126, 1998.

[24] Andrew Webb, *Statistical Pattern Recognition* (2nd Edition), John Wiley & Sons Ltd., England, 2002.