Support vector regression from simulation data and few experimental samples

Gérard Bloch, Fabien Lauer, Guillaume Colin, Yann Chamaillard

▶ **To cite this version:**

Gérard Bloch, Fabien Lauer, Guillaume Colin, Yann Chamaillard. Support vector regression from simulation data and few experimental samples. Information Sciences, 2008, 178 (20), pp.3813-3827. 10.1016/j.ins.2008.05.016 . hal-00286610

HAL Id: hal-00286610
https://hal.science/hal-00286610

Submitted on 10 Jun 2008

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

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

Support vector regression from simulation data and few experimental samples

Gérard Bloch [a] , Fabien Lauer [a,*] ,
Guillaume Colin [b] and Yann Chamaillard [b]

[a] *Centre de Recherche en Automatique de Nancy (CRAN), Nancy–University, CNRS, Parc Robert Bentz F-54500 Vandœuvre-les-Nancy.*

[b] *Institut Pluridisciplinaire de Recherche en Ingénierie des Systèmes, Mécanique, Energétique (PRISME), University of Orléans, 8, rue Léonard de Vinci, 45072 Orléans Cedex 2, France.*

Abstract

This paper considers nonlinear modeling based on a limited amount of experimental data and a simulator built from prior knowledge. The problem of how to best incorporate the data provided by the simulator, possibly biased, into the learning of the model is addressed. This problem, although particular, is very representative of numerous situations met in engine control, and more generally in engineering, where complex models, more or less accurate, exist and where the experimental data which can be used for calibration are difficult or expensive to obtain. The first proposed method constrains the function to fit to the values given by the simulator with a certain accuracy, allowing to take the bias of the simulator into account. The second method constrains the derivatives of the model to fit to the derivatives of a prior model previously estimated on the simulation data. The combination of these two forms of prior knowledge is also possible and considered. These approaches are implemented in the linear programming support vector regression (LP-SVR) framework by the addition, to the optimization problem, of constraints, which are linear with respect to the parameters. Tests are then performed on an engine control application, namely, the estimation of the in-cylinder residual gas fraction in Spark Ignition (SI) engine with Variable Camshaft Timing (VCT). Promising results are obtained on this application. The experiments have also shown the importance of adding potential support vectors in the model when using Gaussian RBF kernels with very few training samples.

Key words: Support Vector Machine (SVM), nonlinear modeling, prior knowledge, SI engine, engine control, residual gas fraction

1 Introduction

The general problem of how to efficiently incorporate knowledge given by a prior simulation model into the learning of a nonlinear model from experimental data can be presented from an application point of view. Consider the modeling of the in-cylinder residual gas fraction in Spark Ignition (SI) engine with Variable Camshaft Timing (VCT) for engine control. In this context, experimental measurements are complex and costly to obtain. On the other hand, a simulator built from physical knowledge can be available but cannot be embedded in a real time controller.

In engine control design (modeling, simulation, control synthesis, implementation and test), two types of models are commonly used:

- Low frequency models or Mean Value Engine Models (MVEM) with average values for the variables over the engine cycle. These models are often used in real time engine control [1,5]. However, they must be calibrated on experiments in sufficiently large number in order to be representative.
- High frequency simulation models that can simulate the evolution of the variables during the engine cycle [6]. These models, of various complexity from zero-dimensional to three-dimensional models, are mostly based on fewer parameters with physical meaning. However, they cannot be embedded in real time controllers.

The idea is thus to build an embeddable black box model by taking into account a prior simulation model, which is representative but possibly biased, in order to limit the number of required measurements. The prior model is used to generate simulation data for arbitrarily chosen inputs in order to compensate for the lack of experimental samples in some regions of the input space. This problem, although particular, is representative of numerous situations met in engine control, and more generally in engineering, where complex models, more or less accurate, exist, providing prior knowledge in the form of simulation data, and where the experimental data which can be used for calibration are difficult or expensive to obtain. The following of the paper studies various methods for the incorporation of these simulation data into the training of the model.

In nonlinear function approximation, kernel methods, and more particularly Support Vector Regression (SVR) [24], have proved to be able to give excel-

* Corresponding author.
Email addresses: gerard.bloch@esstin.uhp-nancy.fr (Gérard Bloch),
fabien.lauer@esstin.uhp-nancy.fr (Fabien Lauer),
guillaume.colin@univ-orleans.fr (Guillaume Colin),
yann.chamaillard@univ-orleans.fr (Yann Chamaillard).

lent performances in various applications [18,17,14]. SVR aims at learning an unknown function based on a training set of N input-output pairs $(\boldsymbol{x}_i, y_i)$ in a black box modeling approach. It originally consists in finding the function that has at most a deviation ε from the training samples with the smallest complexity [22]. Thus, SVR amounts to solve a constrained optimization problem, in which the complexity, measured by the norm of the parameters, is minimized. Allowing for the cases where the constraints can not all be satisfied (some points have larger deviation than ε) leads to minimize an ε-insensitive loss function, which yields a zero loss for a point with error less than ε and corresponds to an absolute loss for the others. The SVR algorithm can thus be written as a quadratic programming (QP) problem, where both the ℓ_1-norm of the errors larger than ε and the ℓ_2-norm of the parameters are minimized. To deal with nonlinear tasks, SVR uses kernel functions, such as the Radial Basis Function (RBF) kernel, which allow to extend linear methods to nonlinear problems via an implicit mapping in a higher dimensional feature space. Compared to neural networks, SVR has the following advantages: automatic selection and sparsity of RBF centers, intrinsic regularization, no local minima (convex problem with a unique solution), and good generalization ability from a limited amount of samples. In addition, the ε-insensitive loss improves the robustness to outliers compared to quadratic criteria.

Other formulations of the SVR problem minimizing the ℓ_1-norm of the parameters can be derived to yield linear programs (LP) [25,23,16]. Some advantages of this latter approach can be noticed compared to the QP formulation such as an increased sparsity of support vectors [25,23] or the ability to use more general kernels [15]. The remaining of the paper will thus focus on the LP formulation of SVR (LP-SVR).

After a presentation of the LP-SVR problem (section 2), the paper uses the framework of [10] to extend the problem with additional constraints, that are linear with respect to the parameters, in order to include prior knowledge in the learning (section 3). The methods are exposed respectively for the inclusion of knowledge on the output values (section 3.1), on the derivatives of the model (section 3.2) and the addition of potential support vectors (section 3.3). Finally, the various ways of incorporating prior knowledge in the form of simulation data with these techniques are tested on the in-cylinder residual gas fraction data in section 4.

Notations: all vectors are column vectors written in boldface and lowercase letters whereas matrices are boldface and uppercase, except for the ith column of a matrix $\boldsymbol{A}$ that is denoted $\boldsymbol{A}_i$. The vectors $\mathbf{0}$ and $\mathbf{1}$ are vectors of appropriate dimensions with all their components respectively equal to 0 and 1. For $\boldsymbol{A} \in \mathbb{R}^{d\times m}$ and $\boldsymbol{B} \in \mathbb{R}^{d\times n}$ containing d-dimensional sample vectors, the "kernel" $\boldsymbol{K}(\boldsymbol{A}, \boldsymbol{B})$ maps $\mathbb{R}^{d\times m} \times \mathbb{R}^{d\times n}$ in $\mathbb{R}^{m\times n}$ with $\boldsymbol{K}(\boldsymbol{A}, \boldsymbol{B})_{i,j} = k(\boldsymbol{A}_i, \boldsymbol{B}_j)$, where $k : \mathbb{R}^{d\times d} \to \mathbb{R}$ is the kernel function. In particular, if $\boldsymbol{x} \in \mathbb{R}^d$ is a column

vector then $\boldsymbol{K}(\boldsymbol{x}, \boldsymbol{B})$ is a row vector in $\mathbb{R}^{1\times n}$. The matrix $\boldsymbol{X} \in \mathbb{R}^{N\times d}$ contains all the training samples $\boldsymbol{x}_i$, $i = 1, \ldots, N$, as rows. The vector $\boldsymbol{y} \in \mathbb{R}^N$ gathers all the target values y_i for these samples. Uppercase Z is a set containing $|Z|$ vectors that constitute the rows of the matrix $\boldsymbol{Z}$.

2 Support Vector Regression (SVR)

In nonlinear regression by kernel methods, the function of input $\boldsymbol{x} \in \mathbb{R}^d$ is approximated by a kernel expansion on the N training samples

$$f(\boldsymbol{x}) = \sum_{i=1}^{N} \alpha_i k(\boldsymbol{x}, \boldsymbol{x}_i) + b = \boldsymbol{K}(\boldsymbol{x}, \boldsymbol{X}^T)\boldsymbol{\alpha} + b \ , \tag{1}$$

where $\boldsymbol{\alpha} = [\alpha_1 \ \ldots \ \alpha_i \ \ldots \ \alpha_N]^T$ and b are the parameters of the model and $k(.,.)$ is the kernel function. Typical kernel functions are the linear, Gaussian RBF, polynomial and sigmoidal kernels. In this paper, a local, the Gaussian RBF kernel: $k(\boldsymbol{x}, \boldsymbol{x}_i) = \exp\left(-\|\boldsymbol{x} - \boldsymbol{x}_i\|^2/2\sigma^2\right)$, and a non-local, the polynomial kernel: $k(\boldsymbol{x}, \boldsymbol{x}_i) = (\boldsymbol{x}^T\boldsymbol{x}_i + 1)^\gamma$, are considered.

In kernel regression via linear programming (LP), the ℓ_1-norm of the parameters $\boldsymbol{\alpha}$ of the kernel expansion is minimized together with the ℓ_1-norm of the errors $\xi_i = y_i - f(\boldsymbol{x}_i)$ by

$$\min_{(\boldsymbol{\alpha},b)} \|\boldsymbol{\alpha}\|_1 + C\sum_{i=1}^{N} |\xi_i| \ , \tag{2}$$

where a hyperparameter C is introduced to tune the trade-off between the error minimization and the maximization of the function flatness. Instead of the absolute value $|\xi|$, the ε-insensitive loss function defined as

$$|\xi|_\varepsilon = \begin{cases} 0 & \text{if } |\xi| \leq \varepsilon \ , \\ |\xi| - \varepsilon & \text{otherwise,} \end{cases} \tag{3}$$

can be used to yield Linear Programming Support Vector Regression (LP-SVR). A possible formulation of the corresponding problem involves $4N + 1$ variables [23]. In this paper, we will follow the approach of [16] that involves only $3N + 1$ variables. Introducing two sets of optimization variables, in two positive slack vectors $\boldsymbol{a}$ and $\boldsymbol{\xi}$, this problem can be implemented as a linear program solvable by standard optimization routines such as MATLAB *linprog*.

In this scheme, the LP-SVR problem becomes

$$\begin{aligned} \min_{(\boldsymbol{\alpha},b,\boldsymbol{\xi},\boldsymbol{a})} & \frac{1}{N}\mathbf{1}^T\boldsymbol{a} + \frac{C}{N}\mathbf{1}^T\boldsymbol{\xi} \\ \text{s.t.} \quad -\boldsymbol{\xi} \leq & \boldsymbol{K}(\boldsymbol{X}^T,\boldsymbol{X}^T)\boldsymbol{\alpha} + b\mathbf{1} - \boldsymbol{y} \leq \boldsymbol{\xi} \\ \mathbf{0} \leq & \quad \mathbf{1}\varepsilon \quad \leq \boldsymbol{\xi} \\ -\boldsymbol{a} \leq & \quad \boldsymbol{\alpha} \quad \leq \boldsymbol{a} \ . \end{aligned} \tag{4}$$

The last set of constraints ensures that $\mathbf{1}^T\boldsymbol{a}$, which is minimized, bounds $\|\boldsymbol{\alpha}\|_1$. In practice, sparsity is obtained as a certain number of parameters α_i will tend to zero. The input vectors $\boldsymbol{x}_i$ for which the corresponding α_i are non-zero are called *support vectors* (SVs).

The parameter ε can also be introduced as a variable in the objective function to be tuned automatically by the algorithm [23,16].

In the LP formulation, only symmetry of the kernel is required [16]. It is not necessary for the kernel to satisfy Mercer's conditions (or positive semidefiniteness) as in the original QP form of the SVR problem [22].

Remark: *In this paper,* balanced *formulations of the learning problems are used. For instance, in problem (4), the sums* $\mathbf{1}^T\boldsymbol{a}$ *and* $\mathbf{1}^T\boldsymbol{\xi}$ *involve* N *terms. They are thus normalized by adding a factor* $1/N$ *in the objective function. When the sums involve different numbers of terms, as in the following, this normalization simplifies the tuning of the hyperparameters.*

3 Incorporating prior knowledge

In this section, two different forms of prior knowledge on the function to be approximated are included in the learning. The first one considers knowledge, possibly approximate, on the output of the function. The second one considers knowledge on the derivatives of the function, possibly given by a prior model. The combination of both types of prior knowledge is also considered and the addition of potential support vectors is discussed at the end of the section.

3.1 Knowledge on output values

Prior knowledge on the function to approximate is assumed to take the form of a set Z of points $\boldsymbol{z}_p$, $p = 1, \ldots, |Z|$, regrouped as rows in the matrix $\boldsymbol{Z}$, for which the output values y_p, $p = 1, \ldots, |Z|$, regrouped in $\boldsymbol{y}_p$, are provided by a

prior model or a simulator. In this setting, the aim is to enforce the equality constraints

$$f(\boldsymbol{z}_p) = y_p,\ p = 1, \dots |Z| \ , \tag{5}$$

on the model f. However, applying constraints such as (5) to the problem (4) would lead to an exact fit to these points, which may not be advised if these are given by a simulator possibly biased. Moreover, all these constraints may lead to an unfeasible problem if they cannot all be satisfied simultaneously. To deal with this case, the equalities (5) can rather be included as soft constraints by introducing a vector $\boldsymbol{u} = [u_1 \ \dots \ u_p \ \dots \ u_{|Z|}]^T$ of positive slack variables bounding the error on (5) as

$$|y_p - f(\boldsymbol{z}_p)| \leq u_p,\ p = 1, \dots |Z| \ . \tag{6}$$

The ℓ_1-norm of the slack vector $\boldsymbol{u}$ is then added to the criterion of (4), with a trade-off parameter λ, in order to be minimized. The trade-off parameter λ allows to tune the influence of the prior knowledge on the model and thus incorporate approximate knowledge.

It is also possible to include almost exact or biased knowledge by authorizing violations of the equality constraints (5) that are less than a threshold. Using the ε-insensitive loss function (3) on the prior knowledge errors u_p with a threshold ε_p, different than the one, ε, used for the training set, leads to the following linear program

$$\begin{aligned}
&\min_{(\boldsymbol{\alpha},b,\boldsymbol{\xi},\boldsymbol{a},\boldsymbol{u})} \ \frac{1}{N}\mathbf{1}^T\boldsymbol{a} + \frac{C}{N}\mathbf{1}^T\boldsymbol{\xi} + \frac{\lambda}{|Z|}\mathbf{1}^T\boldsymbol{u} \\
\text{s.t.} \quad & -\boldsymbol{\xi} \leq \ \boldsymbol{K}(\boldsymbol{X}^T, \boldsymbol{X}^T)\boldsymbol{\alpha} + b\mathbf{1} - \boldsymbol{y} \ \leq \boldsymbol{\xi} \\
& \mathbf{0} \leq \ \mathbf{1}\varepsilon \ \leq \boldsymbol{\xi} \\
& -\boldsymbol{a} \leq \ \boldsymbol{\alpha} \ \leq \boldsymbol{a} \\
& -\boldsymbol{u} \leq \ \boldsymbol{K}(\boldsymbol{Z}^T, \boldsymbol{X}^T)\boldsymbol{\alpha} + b\mathbf{1} - \boldsymbol{y}_p \ \leq \boldsymbol{u} \\
& \mathbf{0} \leq \ \mathbf{1}\varepsilon_p \ \leq \boldsymbol{u} \ ,
\end{aligned} \tag{7}$$

where the two last sets of $|Z|$ constraints stand for the inclusion of prior knowledge.

3.2 Knowledge on the derivatives

Knowledge on the derivatives allows to include information on local maxima or minima, saddle-points and so on. In some applications, one may also wish to retain certain properties of a prior model such as the shape or the roughness at some specific points $\boldsymbol{z}_p$ in Z. This problem corresponds to learn a new model

while constraining certain of its derivatives to equal those of the prior model at these particular points.

Consider that prior knowledge on the derivative of the function f with respect to the jth component x^j of $\boldsymbol{x} \in \mathbb{R}^d$ is available as

$$\left.\frac{\partial f(\boldsymbol{x})}{\partial x^j}\right|_{\boldsymbol{z}_p} = y'_p, \ \forall \boldsymbol{z}_p \in Z \ . \tag{8}$$

This prior knowledge can be enforced in the training by noticing that the kernel expansion (1) is linear with respect to the parameters $\boldsymbol{\alpha}$, which allows to write the derivative of the model output with respect to x^j as

$$\frac{\partial f(\boldsymbol{x})}{\partial x^j} = \sum_{i=1}^{N} \alpha_i \frac{\partial k(\boldsymbol{x}, \boldsymbol{x}_i)}{\partial x^j} = \boldsymbol{r}_j(\boldsymbol{x})^T \boldsymbol{\alpha} \ , \tag{9}$$

where $\boldsymbol{r}_j(\boldsymbol{x}) = [\partial k(\boldsymbol{x}, \boldsymbol{x}_1)/\partial x^j \ \ldots \ \partial k(\boldsymbol{x}, \boldsymbol{x}_i)/\partial x^j \ \ldots \ \partial k(\boldsymbol{x}, \boldsymbol{x}_N)/\partial x^j]^T$ is of dimension N. The derivative (9) is linear with respect to $\boldsymbol{\alpha}$. In fact, the form of the kernel expansion implies that the derivatives of any order with respect to any component are linear with respect to $\boldsymbol{\alpha}$. See for instance [10] for Gaussian RBF kernels. The prior knowledge (8) can thus be included by defining $\boldsymbol{R}(\boldsymbol{Z}^T, \boldsymbol{X}^T) = \left[\boldsymbol{r}_j(\boldsymbol{z}_1) \ \ldots \ \boldsymbol{r}_j(\boldsymbol{z}_p) \ \ldots \ \boldsymbol{r}_j(\boldsymbol{z}_{|Z|})\right]^T$ of dimension $|Z| \times N$, $\boldsymbol{y}'_p = [y'_1 \ \ldots \ y'_p \ \ldots \ y'_{|Z|}]^T$ and solving the problem

$$\begin{aligned} \min_{(\boldsymbol{\alpha}, b, \boldsymbol{\xi}, \boldsymbol{a}, \boldsymbol{v})} & \frac{1}{N}\mathbf{1}^T\boldsymbol{a} + \frac{C}{N}\mathbf{1}^T\boldsymbol{\xi} + \frac{\lambda}{|Z|}\mathbf{1}^T\boldsymbol{v} \\ \text{s.t.} \quad -\boldsymbol{\xi} \leq & \ \boldsymbol{K}(\boldsymbol{X}^T, \boldsymbol{X}^T)\boldsymbol{\alpha} + b\mathbf{1} - \boldsymbol{y} \leq \boldsymbol{\xi} \\ \mathbf{0} \leq & \ \mathbf{1}\varepsilon \leq \boldsymbol{\xi} \\ -\boldsymbol{a} \leq & \ \boldsymbol{\alpha} \leq \boldsymbol{a} \\ -\boldsymbol{v} \leq & \ \boldsymbol{R}(\boldsymbol{Z}^T, \boldsymbol{X}^T)\boldsymbol{\alpha} - \boldsymbol{y}'_p \leq \boldsymbol{v} \ , \end{aligned} \tag{10}$$

where a vector $\boldsymbol{v} = [v_1 \ \ldots \ v_p \ \ldots \ v_{|Z|}]^T$ of positive slack variables bounding the error on (8) has been introduced.

The extension to any order of derivative is straightforward. Thus, this method allows to incorporate prior knowledge on any derivative for any set of points possibly different for each derivative. In comparison, the methods described in [13] and [12] require the prior derivative values on all the training points and these points only.

Combining prior knowledge on the output values as well as on the derivative values is also possible and amounts to a concatenation of constraints. Thus a method that considers the case where information is available on both y_p and

y'_p, leads to the complete problem

$$\begin{array}{rrcl} \min\limits_{(\boldsymbol{\alpha},b,\boldsymbol{\xi},\boldsymbol{a},\boldsymbol{u},\boldsymbol{v})} & \frac{1}{N}\mathbf{1}^T\boldsymbol{a} + \frac{C}{N}\mathbf{1}^T\boldsymbol{\xi} + \frac{\lambda_1}{|Z|}\mathbf{1}^T\boldsymbol{u} + \frac{\lambda_2}{|Z|}\mathbf{1}^T\boldsymbol{v} & & \\ \text{s.t.} \quad -\boldsymbol{\xi} \leq & \boldsymbol{K}(\boldsymbol{X}^T,\boldsymbol{X}^T)\boldsymbol{\alpha} + b\mathbf{1} - \boldsymbol{y} & \leq \boldsymbol{\xi} & \\ \mathbf{0} \leq & \mathbf{1}\varepsilon & \leq \boldsymbol{\xi} & \\ -\boldsymbol{a} \leq & \boldsymbol{\alpha} & \leq \boldsymbol{a} & \\ -\boldsymbol{u} \leq & \boldsymbol{K}(\boldsymbol{Z}^T,\boldsymbol{X}^T)\boldsymbol{\alpha} + b\mathbf{1} - \boldsymbol{y}_p & \leq \boldsymbol{u} & \\ \mathbf{0} \leq & \mathbf{1}\varepsilon_p & \leq \boldsymbol{u} & \\ -\boldsymbol{v} \leq & \boldsymbol{R}(\boldsymbol{Z}^T,\boldsymbol{X}^T)\boldsymbol{\alpha} - \boldsymbol{y}'_p & \leq \boldsymbol{v}\ , \end{array} \tag{11}$$

where λ_1 and λ_2 are the trade-off parameters for the knowledge on the output values and on the derivatives, respectively. In this problem, the first three sets of constraints correspond to the standard learning (4) on the training set $(\boldsymbol{X}, \boldsymbol{y})$. The next two sets of constraints include prior knowledge on the output values as in section 3.1. The last set of constraints involving the slack vector $\boldsymbol{v}$ incorporates the information on the derivatives as in (10).

3.3 Adding potential support vectors

In the case where the training data do not cover the whole input space, extrapolation occurs, which can become a problem when using local kernels such as the RBF kernel. To avoid this problem, the points $\boldsymbol{z}_p$ can be included as potential support vectors (or RBF centers). However, these samples are not introduced in the training set, so that the model is not required to fit to the corresponding output values y_p.

For instance, in this setting, the problem (7) may be written as

$$\begin{array}{rrcl} \min\limits_{(\boldsymbol{\alpha},b,\boldsymbol{\xi},\boldsymbol{a},\boldsymbol{u})} & \frac{1}{N+|Z|}\mathbf{1}^T\boldsymbol{a} + \frac{C}{N}\mathbf{1}^T\boldsymbol{\xi} + \frac{\lambda}{|Z|}\mathbf{1}^T\boldsymbol{u} & & \\ \text{s.t.} \quad -\boldsymbol{\xi} \leq & \boldsymbol{K}(\boldsymbol{X}^T,[\boldsymbol{X}^T\ \boldsymbol{Z}^T])\boldsymbol{\alpha} + b\mathbf{1} - \boldsymbol{y} & \leq \boldsymbol{\xi} & \\ \mathbf{0} \leq & \mathbf{1}\varepsilon & \leq \boldsymbol{\xi} & \\ -\boldsymbol{a} \leq & \boldsymbol{\alpha} & \leq \boldsymbol{a} & \\ -\boldsymbol{u} \leq & \boldsymbol{K}(\boldsymbol{Z}^T,[\boldsymbol{X}^T\ \boldsymbol{Z}^T])\boldsymbol{\alpha} + b\mathbf{1} - \boldsymbol{y}_p & \leq \boldsymbol{u} & \\ \mathbf{0} \leq & \mathbf{1}\varepsilon_p & \leq \boldsymbol{u}\ . \end{array} \tag{12}$$

The difference with (7) is that the samples in $\boldsymbol{Z}$ are added as potential support vectors. It must be noted that the number of parameters in the vector $\boldsymbol{\alpha}$ is

now $N + |Z|$ and that the dimension of the kernel matrix $\boldsymbol{K}(\boldsymbol{X}^T, [\boldsymbol{X}^T\ \boldsymbol{Z}^T])$ has increased to $N \times (N + |Z|)$.

Potential SVs can also be added in algorithms (10) and (11) by replacing the matrices $\boldsymbol{K}(\boldsymbol{X}^T, \boldsymbol{X}^T)$, $\boldsymbol{K}(\boldsymbol{Z}^T, \boldsymbol{X}^T)$ and $\boldsymbol{R}(\boldsymbol{Z}^T, \boldsymbol{X}^T)$ by $\boldsymbol{K}(\boldsymbol{X}^T, [\boldsymbol{X}^T\ \boldsymbol{Z}^T])$, $\boldsymbol{K}(\boldsymbol{Z}^T, [\boldsymbol{X}^T\ \boldsymbol{Z}^T])$ and $\boldsymbol{R}(\boldsymbol{Z}^T, [\boldsymbol{X}^T\ \boldsymbol{Z}^T])$.

4 Estimation of the in-cylinder residual gas fraction

The application deals with the estimation of residual gases in the cylinders of Spark Ignition (SI) engines with Variable Camshaft Timing (VCT). VCT allows the timing of the intake and exhaust valves to be changed while the engine is in operation. VCT is used to improve performance in terms of emissions, fuel economy, peak torque, and peak power [8].

The air path control of SI engines is a crucial task because the torque provided by the engine is directly linked to the air mass trapped in the cylinders [2]. When considering new air actuators such as VCT, the estimation of in-cylinder air mass is more involved than for basic SI engines. Indeed, VCT authorizes phenomena such as air scavenging (from intake to exhaust manifolds, with turbocharging) or backflow (from exhaust manifold to cylinders).

In this context, it is important to estimate the residual gas mass fraction

$$\chi_{res} = \frac{m_{res}}{m_{tot}}, \tag{13}$$

where m_{tot} is the total gas mass trapped in the cylinder and m_{res} is the mass of residual gases, which are burned gases present in the cylinder when the valves are closed before the new combustion and which are due to the dead volumes or the backflow. Knowing this fraction allows to control torque as well as pollutant emissions.

There is no standard sensor to measure this fraction online. There exists a corresponding mean value model, proposed by Fox et al. [3], that includes some constants to be identified from experiments on a particular engine. As mentioned in the introduction, only average values of the variables over the cycle are considered in mean value models. In this context, the residual gas mass fraction χ_{res} can be expressed as a function of the engine speed N_e, the ratio p_{man}/p_{exh}, where p_{man} and p_{exh} are respectively the intake manifold pressure and the exhaust pressure, and an overlapping factor OF, which is an image of the time during which the valves are opened together. This residual

gas fraction also depends, but only slightly, on the opening and closing instants of the valves, which are not taken into account here, as in [3].

The available data are provided, on one hand, by the modeling and simulation environment Amesim [7], which uses a high frequency zero-dimensional thermodynamic model [4] and, on the other hand, by off line measurements, which are accurate, but complex and costly to obtain, by direct in-cylinder sampling [4]. The problem is thus as follows. How to obtain a simple, embeddable, black box model with a good accuracy and a large validity range for the real engine, from precise real measurements as less numerous as possible and a representative, but possibly biased, prior simulation model?

4.1 Setup of the experiments

Three datasets are built from the available data composed of 26 experimental samples plus 26 simulation samples:

- the training set composed of a limited amount of real data (N samples),
- the test set composed of independent real data ($N_{test} = 26 - N$ samples),
- the simulation set composed of data provided by the simulator ($N_{pr} = 26$ samples).

It must be noted that the inputs of the simulation data do not exactly coincide with the inputs of the experimental data. Various sizes N of the training set will be considered in order to study the effect of the number of training samples on the model.

The residual gas mass fraction χ_{res}, given in percentages, takes values in the range $[5, 30]$. The ranges of values for the three inputs are: N_e (rpm) $\in \{1000, 2000\}$, $p_{man}/p_{exh} \in [0.397, 0.910]$ and OF ($^\circ CA/m$) $\in [0, 2.8255]$. Table 1 presents the chosen values for OF with the corresponding opening and closing instants of the valves. The datasets are shown in Figure 1.

Table 1
Intake valve opening (IVO) and exhaust valve closing (EVC) timings (at 0 mm lift) for the different values of the overlapping factor OF.

OF ($^\circ/m$)	IVO ($^\circ CA$)	EVC ($^\circ CA$)	overlap duration ($^\circ CA$)
0	0	0	0
0.41	0	42	42
0.58	-24	24	48
1.16	-33	30	63
2.83	-42	42	84

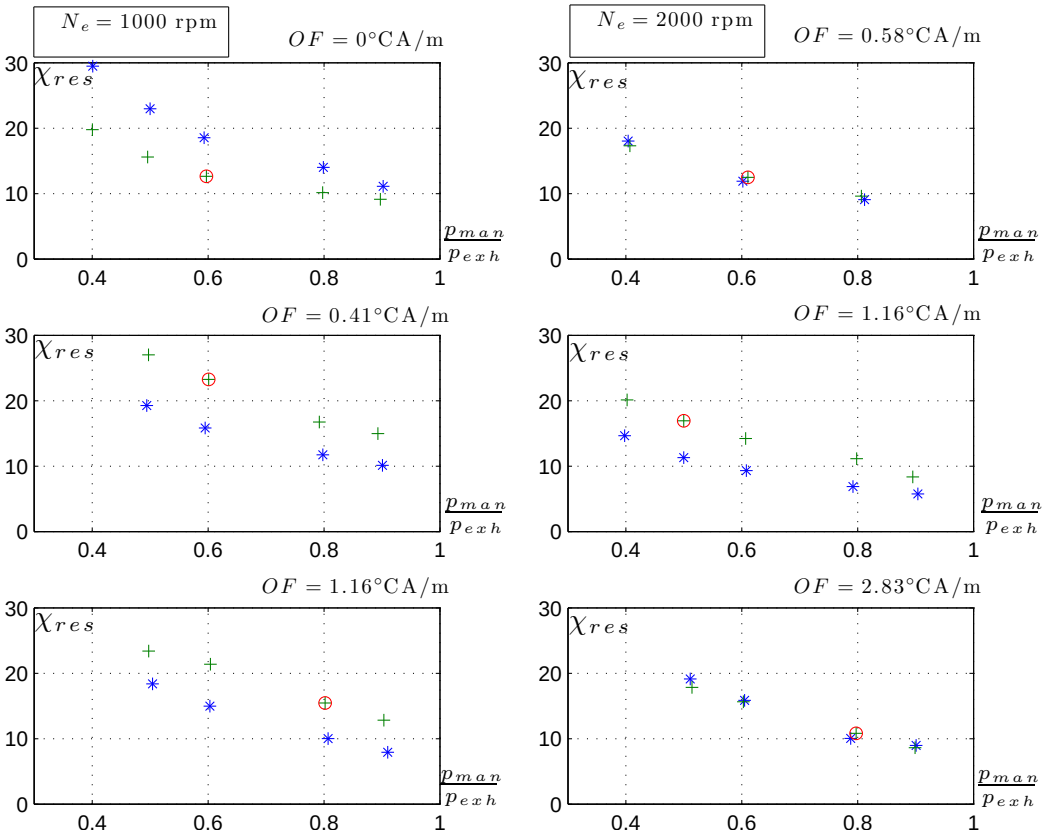

Fig. 1. Residual gas mass fraction χ_{res} in percentages as a function of the ratio p_{man}/p_{exh} for two engine speeds N_e and different overlapping factors OF. The 26 experimental data are represented by plus signs (+), with a superposed circle ($\oplus$) for the training samples when $N = 6$. The 26 simulation data appear as asterisks ($*$).

Different situations and various ways of incorporating the simulation data are considered in the following models.

(1) *Experimental model*. The simulation data are not available. Standard LP-SVR training (4) on the training set only is used to determine the model.
(2) *Prior model*. The experimental data are not available. LP-SVR training (4) on the simulation set only is used to determine the model.
(3) *Mixed model*. All the data are available and mixed together. LP-SVR training (4) on the training set extended with the simulation data is used to determine the model. This may be the first and most simple approach, similar to the virtual sample approach, which has been extensively studied in pattern recognition for the incorporation of transformation-invariance [19,20,9] and has been very successful in this field [21,11]. However, in the case considered here, the simulation data can be biased as the physical model is not fully accurate.
(4) *O-model*. The simulation data are considered as prior knowledge on *Output* values. Algorithm (7) is used to train the model. This approach allows

to take into account a possible bias of ε_p between the simulation and real data.

(5) *OP-model*. Same as the *O-model* but with the simulation data as *Potential* SVs. Algorithm (12) is used to train the model. This approach allows to compensate for the local behavior of RBF kernels which may become a serious drawback when reducing the number of training samples.

(6) *D-model*. The simulation data are used to build the *prior model*, which is then used to provide prior knowledge on *Derivative* values with respect to the input p_{man}/p_{exh}, taken at the simulation input points. Algorithm (10) with derivative constraints is used to train the final model. This approach allows to retain the overall shape of the *prior model* while fitting it to the available real data.

(7) *DP-model*. Same as the *D-model* but with the simulation data as *Potential* SVs.

(8) *OD-model*. The simulation data are both considered as prior knowledge on *Output* values and used to build a *prior model* in order to give prior knowledge on *Derivative* values as for model 5. Algorithm (11) is used to train the OD-model.

(9) *ODP-model*. Same as the *OD-model* but with the simulation data as *Potential* SVs.

These models are evaluated on the basis of three indicators defined below.

- *RMSE test*: root mean square error on the test set (N_{test} samples)
- *RMSE total*: root mean square error on the whole real dataset ($N + N_{test}$ samples)
- *MAE total*: maximum absolute error on the whole real dataset ($N + N_{test}$ samples)

Before training, the variables are normalized with respect to their mean and standard deviation. When both experimental and simulation data are available, the simulation data are preferred since they are supposed to cover a wider region of the input space. Thus, the mean and standard deviation are determined on the simulation data for all the models except for the *experimental model*, in which case the training set must be used to determine the normalization parameters.

The hyperparameters of the method can be classified in two categories: internal parameters, such as the kernel parameters σ or d; and the user-level parameters, such as C, λ (or λ_1 and λ_2), ε and ε_p, allowing to tune the algorithm in accordance with some prior knowledge. As kernel parameter tuning is out of the scope of the paper, the kernel parameters are set according to the following heuristics. Since all standard deviations of the inputs equal 1 after normalization, the RBF kernel width σ is set to 1. The degree γ of the polynomial kernels is set to the lowest value yielding a reasonable training error on

$N = 15$ samples, i.e. $\gamma = 4$. For the threshold parameters of the ε-insensitive loss functions, ε is set to 0.001 in order to approximate the real data well. In addition, when using prior knowledge on output values (*O-*, *OP-*, *OD-* and *ODP-models*) and setting ε_p, one has both the real and simulation data at hand. Thus, it is possible to compute an estimate of the maximum absolute error (MAE) obtained by the simulator by looking at the MAE obtained by the *prior model* on the training set (available real data). ε_p is then set accordingly by taking into account the normalization step ($\varepsilon_p = 1.6$). Regarding the remaining hyperparameters, the following sections respectively discuss the use of fixed values, the sensitivity of the algorithm to their values, and their tuning by cross-validation.

4.2 Fixed hyperparameters

The hyperparameters C, λ (or λ_1 and λ_2) are first set as follows. One goal of the problem is to obtain a model that is accurate on both the training and test samples (the training points are part of the performance index *RMSE total*). Thus C is set to a large value ($C = 1000$) in order to ensure a good approximation of the training points. The additional hyperparameter of the method λ is set to $\lambda = C = 1000$ to give as much weight to the prior knowledge than to the data. For the *OD-* and *ODP-models*, the two types of prior knowledge are considered with the same weight, i.e. $\lambda_1 = \lambda_2 = C = 1000$.

The first experiments are performed with RBF kernels, which exhibit a local behavior. The number N of points retained as training samples is first set to 15, which is about half of the available experimental data. The results in this setting appear at the top of Table 2. These show that the model of the simulator, the *prior model*, is actually biased and leads to a large error when tested on the real data. As a consequence, the *mixed model* that simply considers simulation data as training samples cannot yield good results from inhomogeneous and contradictory data. Regarding the various forms of prior knowledge considered, it seems that the information on the derivative is the most relevant. However, in order to improve the model, potential support vectors must be added, as in the *DP-* and *ODP-models*.

Now, the effect of reducing the number of training samples to $N = 6$ is studied. The results in Table 2 show that a good *experimental model* cannot be determined with so few samples. On the other hand, adding prior knowledge allows to obtain good results as shown on Figure 2 for the *ODP-model*. Incorporating prior information on the derivative, as in the *DP-* and *ODP-models*, is more efficient than simply incorporating simulation samples with a large threshold ε_p on the corresponding error as implemented in the *O-* and *OP-models*. It must be noted that the test error of the *DP-* and *ODP-models* is less than half

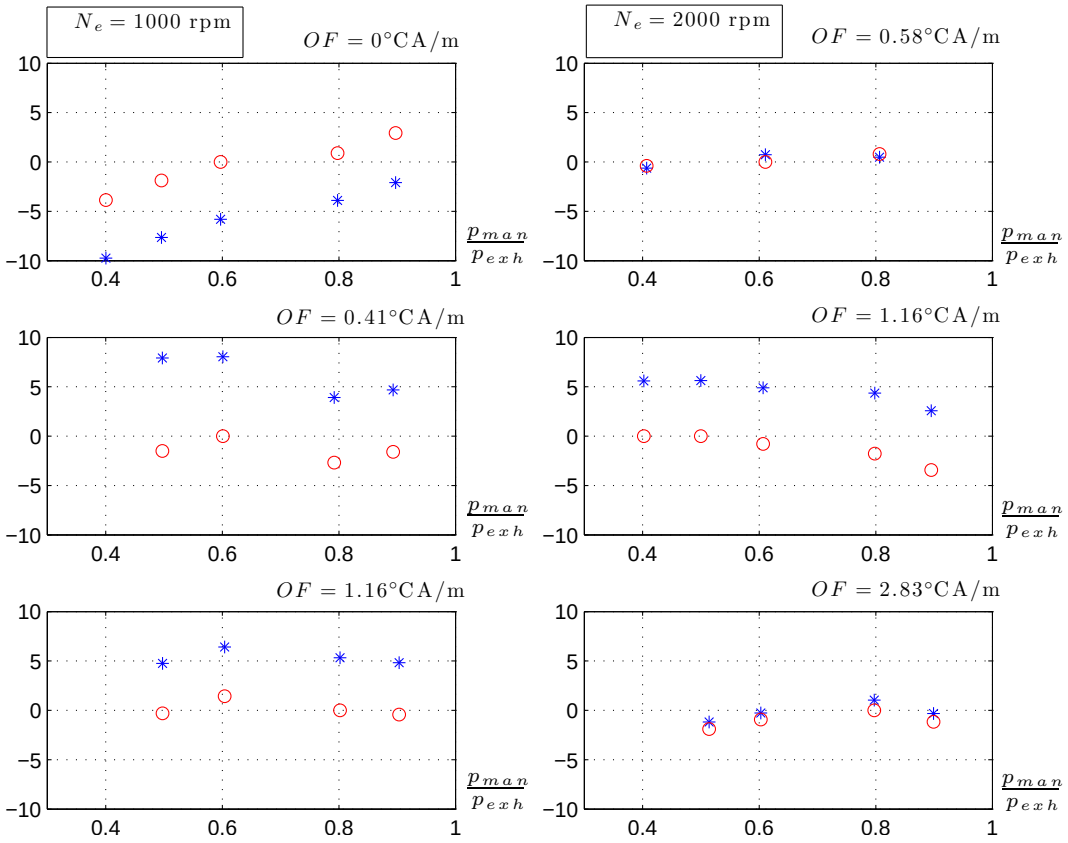

Fig. 2. Errors on the experimental data (training and test samples) for the *prior model* ($*$) and the *ODP-model* ($\circ$) trained with a RBF kernel on only six samples ($N = 6$).

the test errors obtained by the *prior* and *mixed models*, which correspond to the standard methods to include simulation data.

When the number of training points becomes too small ($N = 3$), the *O-*, *D-* and *OD-models*, that do not incorporate the simulation data as potential SVs, become almost constant and thus inefficient. This is due to the fact that these models have not enough free parameters (only 3 plus a bias term) and corresponding RBFs; thus they cannot accurately model the data. On the contrary, the *OP-*, *DP-* and *ODP-models* do not suffer from this problem. However, as discussed below, the method becomes more sensitive to the tuning of λ which leads to a very large error for the *DP-model*. On the other hand, the *ODP-model* achieves a reasonable performance considering that only $N = 3$ experimental samples were used.

Table 3 shows that, for a non-local kernel, i.e. a polynomial kernel of degree 4, the method also allows to improve the model. Due to the non-local behavior of this kernel, the effect of considering the simulation data as potential support vectors is less obvious and the *OD-model* without additional SVs yields the best performance for $N = 6$. A common feature of the experiments with the

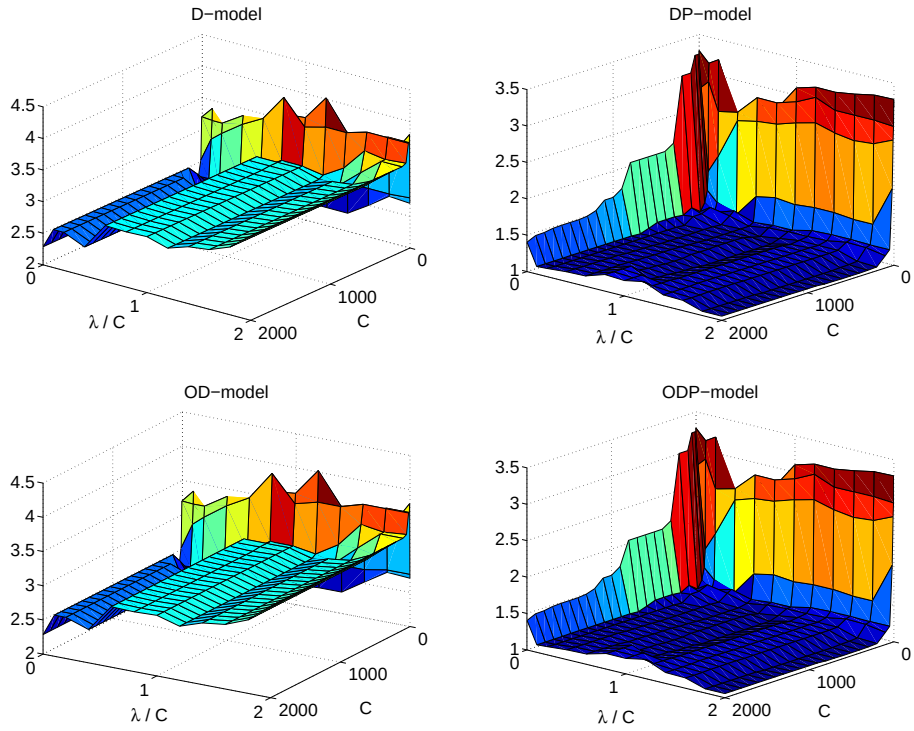

Fig. 3. *RMSE test* versus C and the ratio C/λ for the *D-*, *DP-*, *OP-* and *ODP-models* trained on $N = 15$ data points with a RBF kernel.

two kernels is the superiority of the prior knowledge applied to the derivatives compared to the output values.

4.3 Influence of the hyperparameters C and λ

As shown in the previous section, the models that do not use information on the derivative (*O-* and *OP-models*) always lead to poor performance. Thus only the results for the *D-*, *DP-*, *OD-models* and *ODP-models* are reported here.

The effect of the hyperparameter λ (or λ_1 and λ_2) depends on the value of C as the ratio λ/C defines the weight of the prior knowledge with respect to the weight of the training data. The first set of experiments considers $\lambda_1 = \lambda_2 = \lambda$. The balance between λ_1 and λ_2 is studied at the end of this section.

Figures 3 and 4 show the variations of the *RMSE test* for the models using respectively RBF and polynomial kernels, trained on $N = 15$ data points with $1 \leq \lambda \leq 2C$ and $1 \leq C \leq 2000$. In this setting, the *RMSE test* is not very sensitive to λ, except for a slight increase observed for large values $\lambda > C$. The data are in sufficient number to cover the whole input space and the approximative nature of the knowledge may decrease the performance if more

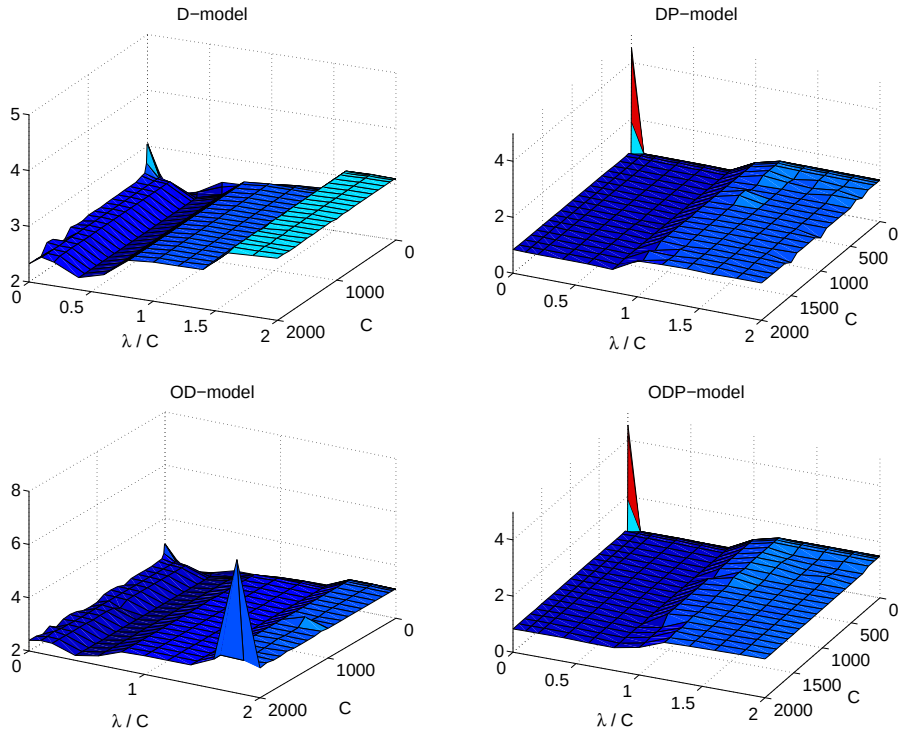

Fig. 4. *RMSE test* versus C and the ratio C/λ for the *D-*, *DP-*, *OP-* and *ODP-models* trained on $N = 15$ data points with a polynomial kernel ($\gamma = 4$).

weight is given to the prior knowledge, i.e. if $\lambda > C$.

For $N = 6$ experimental data points, the test error is also not very sensitive to the tuning of λ, as shown in Figure 5 for RBF kernels and Figure 6 for polynomial kernels. However, the *D-* and *OD-models* with RBF kernels are improved for large values of $\lambda > C$. These models have not enough support vectors, are thus more dependent on the prior knowledge and require a ratio $\lambda/C > 1$. However, even for large values of λ, they still lead to larger test errors than the models with additional support vectors. The models with polynomial kernels are less sensitive to this issue.

When training the models with only $N = 3$ experimental data points, the method becomes more sensitive to λ and the tuning of this hyperparameter may become critical. As shown in Figure 7, a better test error than the one reported in Table 2 for $\lambda/C = 1$ could be obtained by the *ODP-model* with RBF kernels for values of λ around $0.1C$. With polynomial kernels, the *DP-model* actually leads to divergent outputs for many values of the couple C, λ. The other models using polynomial kernels are also more sensitive to λ (Fig. 8).

These experiments also show that the tuning of C is not critical as the models are mostly insensitive to its value, except for $N = 3$. In general, the polynomial models appear less sensitive to the tuning of C than the RBF models.

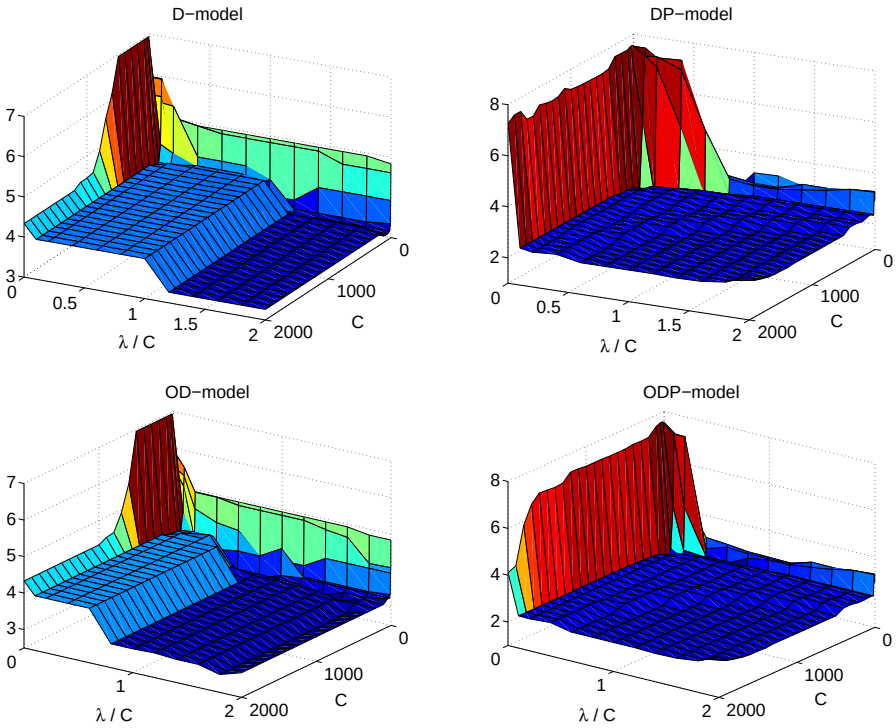

Fig. 5. *RMSE test* versus C and the ratio C/λ for the *D-*, *DP-*, *OP-* and *ODP-models* trained on $N = 6$ data points with a RBF kernel.

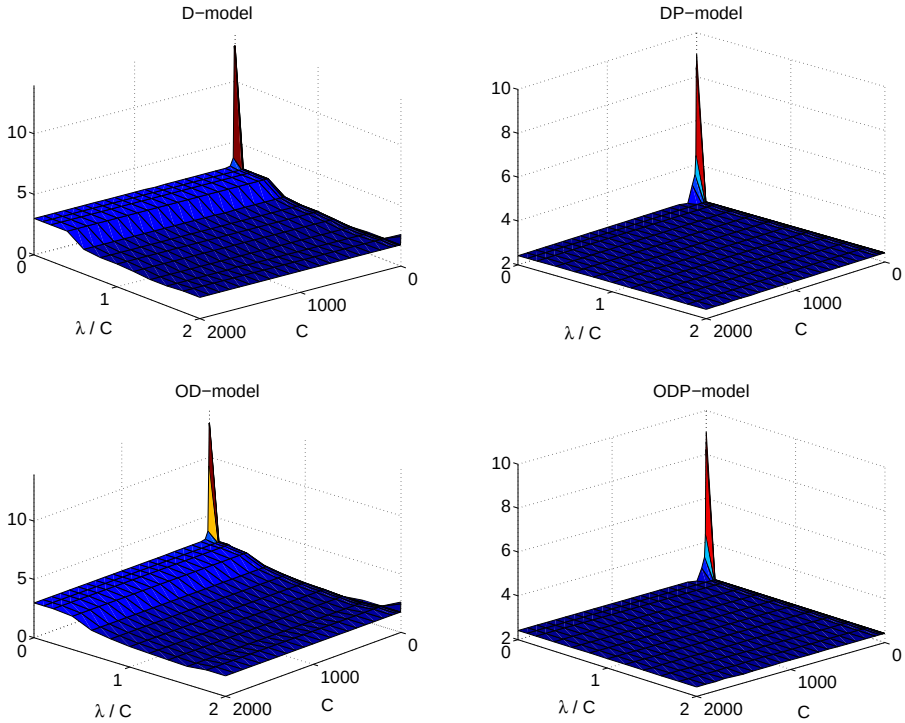

Fig. 6. *RMSE test* versus C and the ratio C/λ for the *D-*, *DP-*, *OP-* and *ODP-models* trained on $N = 6$ data points with a polynomial kernel ($\gamma = 4$).

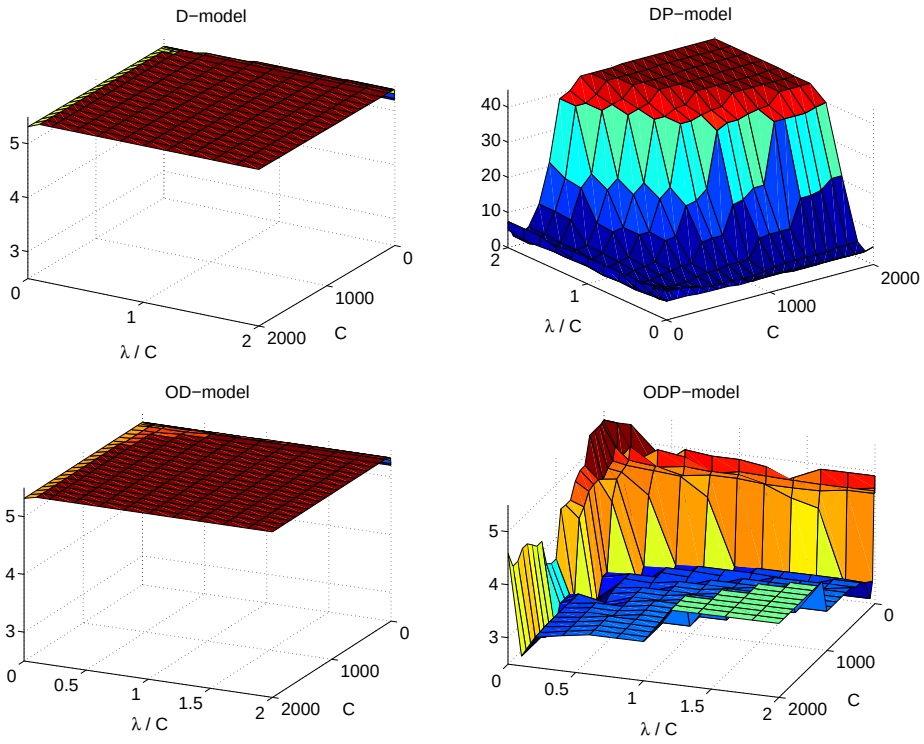

Fig. 7. *RMSE test* versus C and the ratio C/λ for the *D-*, *DP-*, *OP-* and *ODP-models* trained on $N = 3$ data points with a RBF kernel.

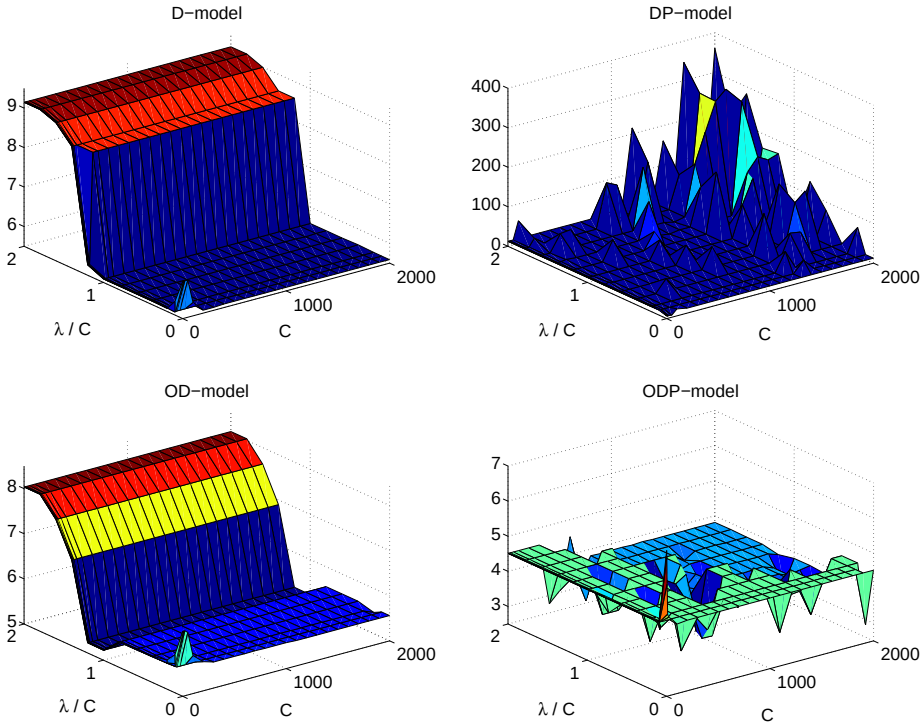

Fig. 8. *RMSE test* versus C and the ratio C/λ for the *D-*, *DP-*, *OP-* and *ODP-models* trained on $N = 3$ data points with a polynomial kernel ($\gamma = 4$).

However, the models with RBF kernels are more stable with respect to the tuning of λ. Moreover, for different training set sizes N, the *ODP-model* always leads to the best performance, whereas this is not the case with the polynomial models. Thus, for practical reasons, it is preferable to use the RBF kernel, though using a polynomial kernel may lead to a better test error for a particular value of the hyperparameter.

Balance between λ_1 and λ_2. Studying the effect of the balance between the prior knowledge on the ouptut (weighted by λ_1) and the prior knowledge on the derivative (weighted by λ_2) shows that the *ODP-model* is mostly influenced by the information on the derivative for both the RBF and the polynomial kernels. Indeed, for a fixed value of λ_2, the model is rather insensitive to λ_1 as long as λ_1 is not too close to zero. Besides, tests with $N = 15$, 6 and 3 have shown that an almost minimal test error can always be obtained with $\lambda_2 = \lambda_1$. Thus in practice, when mixing two types of prior knowledge without further information on the optimal balance between the two, it is advised to choose $\lambda_2 = \lambda_1$, as this reduces the number of hyperparameters to tune.

4.4 Tuning λ by cross validation

The trade-off parameter λ weighting the prior knowledge with respect to the experimental data can be tuned by Leave-One-Out (LOO) cross validation. The LOO procedure allows to compute an estimate of the generalization error, here the generalization root mean square error (RMSE), as follows. The model is trained on $N - 1$ samples from the training set, leaving one sample aside for validation, on which the performance of the model is evaluated. This procedure is repeated N times providing an average performance of the training algorithm for a particular value of the hyperparameter.

This procedure is relevant only for a sufficiently large number of training samples. Thus, only one experiment of Sect. 4.2, for $N = 15$, is reproduced here with, for the training of the *O-*, *OP-*, *D-* and *DP-models*, the optimal value of λ determined by the LOO procedure in the set of 10 values $\{1,\ 10,\ 50,\ 100,\ 200,\ 300,\ 400,\ 500,\ 600,\ 800,\ 1000,\ 1500,\ 2000\}$. These values are to be related to the value of $C = 1000$, which represents the weight of the experimental data in the training. For the *OD-* and *ODP-models*, the relevance of both types of prior knowledge is equivalently balanced and $\lambda_1 = \lambda_2 = \lambda$, where λ is also tuned by the LOO procedure.

Tables 4 and 5 show, respectively for the RBF and polynomial kernels, the optimal value of λ and the corresponding results. The test errors obtained by this tuning procedure are very close to the ones obtained with $\lambda = 1000$ and reported in Tables 2 and 3. Moreover, an improvement in terms of the *RMSE*

total and the *MAE total* can be observed for the two best models using RBF kernels (*DP-* and *ODP-models*). Thus, when no information is available on the optimal balance between the prior knowledge and the data, and when the size of the training set permits it, a cross validation procedure can be used to tune the hyperparameters.

5 Conclusion

This paper uses simple and effective techniques for the incorporation of prior knowledge into LP-SVR learning. This prior information may be given in terms of output values as well as derivative values on a set of points. Various methods based on these techniques have been studied for the inclusion of knowledge in the form of simulation data. The proposed methods have been tested on the estimation of in-cylinder residual gas fraction application. In this context, real data are available only in a limited number due to the cost of experimental measurements, but additional data can be obtained thanks to a complex physical simulator. The output of the simulator being biased but providing rather good information on the overall shape of the model, prior information on the derivatives, provided by a prior model trained on the simulation data, is the most relevant. Models enhanced by this knowledge thus allow to obtain the best performance.

The additional hyperparameters of the method weight the prior knowledge with respect to the data and can thus be chosen in accordance with the confidence in the prior information. Moreover, the sensitivity of the method with respect to the tuning of these hyperparameters has been experimentally shown, but only on a particular example, to be very low as long as the data are not too few. Besides, the experiments have also shown the importance of adding potential support vectors in the model when using a local kernel, such as the Gaussian RBF kernel, with few training samples.

Acknowledgments

The authors thank P. Giansetti for providing the data used in this work.

References

[1] A. Chevalier, M. Müller, and E. Hendricks. On the validity of mean value engine models during transient operation. *SAE Technical Papers*, (2000-01-1261), 2000.

[2] G. Colin, Y. Chamaillard, G. Bloch, and G. Corde. Neural control of fast nonlinear systems – Application to a turbocharged SI engine with VCT. *IEEE Trans. on Neural Networks*, 18(4):1101–1114, 2007.

[3] J. W. Fox, W. K. Cheng, and J. B. Heywood. A model for predicting residual gas fraction in spark-ignition engines. *SAE Technical Papers*, (931025), 1993.

[4] P. Giansetti, G. Colin, P. Higelin, and Y. Chamaillard. Residual gas fraction measurement and computation. *International Journal of Engine Research*, 8(4):347–364, 2007.

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

[6] J. B. Heywood. *Internal Combustion Engines Fundamentals*. McGraw-Hill, New York, NY, USA, 1988.

[7] Imagine. Amesim web site. www.amesim.com, 2006.

[8] M. Jankovic, S. Magner, S. Hsieh, and J. Konesol. Transient effects and torque control of engines with variable cam timing. In *Proc. of the American Control Conference, San Francisco, CA*, volume 1, pages 50–54, Sept. 2000.

[9] F. Lauer and G. Bloch. Incorporating prior knowledge in support vector machines for classification: a review. *Neurocomputing*, 71(7-9):1578–1594, 2008.

[10] F. Lauer and G. Bloch. Incorporating prior knowledge in support vector regression. *Machine Learning*, 70(1):89–118, 2008.

[11] F. Lauer, C. Y. Suen, and G. Bloch. A trainable feature extractor for handwritten digit recognition. *Pattern Recognition*, 40(6):1816–1824, 2007.

[12] M. Lázaro, F. Pérez-Cruz, and A. Artés-Rodriguez. Learning a function and its derivative forcing the support vector expansion. *IEEE Signal Processing Letters*, 12:194–197, 2005.

[13] M. Lázaro, I. Santamaria, F. Pérez-Cruz, and A. Artés-Rodriguez. Support vector regression for the simultaneous learning of a multivariate function and its derivatives. *Neurocomputing*, 69:42–61, 2005.

[14] C. A. M. Lima, A. L. V. Coelho, and F. J. Von Zuben. Hybridizing mixtures of experts with support vector machines: Investigation into nonlinear dynamic systems identification. *Information Sciences*, 177(10):2049–2074, 2007.

[15] O. Mangasarian. Generalized support vector machines. In A. Smola, P. Bartlett, B. Schölkopf, and D. Schuurmans, editors, *Advances in Large Margin Classifiers*, pages 135–146. MIT Press, Cambridge, MA, USA, 2000.

[16] O. L. Mangasarian and D. R. Musicant. Large scale kernel regression via linear programming. *Machine Learning*, 46(1-3):255–269, 2002.

[17] D. Mattera and S. Haykin. Support vector machines for dynamic reconstruction of a chaotic system. In B. Schölkopf, C. J.C. Burges, and A. J. Smola, editors, *Advances in kernel methods: support vector learning*, pages 211–241. MIT Press, Cambridge, MA, USA, 1999.

[18] K.R. Müller, A. Smola, G. Rätsch, B. Schölkopf, J. Kohlmorgen, and V. Vapnik. Predicting time series with support vector machines. In *Proc. of the Int. Conf. on Artificial Neural Networks (ICANN), Lausanne, Switzerland*, volume 1327 of *Lecture Notes in Computer Science*, pages 999–1004, 1997.

[19] T. Poggio and T. Vetter. Recognition and structure from one 2D model view: Observations on prototypes, object classes and symmetries. Technical Report AIM-1347, Massachusetts Institute of Technology, Cambridge, MA, USA, 1992.

[20] B. Schölkopf, C. Burges, and V. Vapnik. Incorporating invariances in support vector learning machines. In *Proc. of the Int. Conf. on Artificial Neural Networks (ICANN), Bochum, Germany*, volume 1112 of *Lecture Notes in Computer Science*, pages 47–52, 1996.

[21] P. Y. Simard, D. Steinkraus, and J. C. Platt. Best practices for convolutional neural networks applied to visual document analysis. In *Proc. of the 7th Int. Conf. on Document Analysis and Recognition, Edinburgh, Scotland, UK*, volume 2, pages 958–962, 2003.

[22] A. J. Smola and B. Schölkopf. A tutorial on support vector regression. *Statistics and Computing*, 14(3):199–222, 2004.

[23] A. J. Smola, B. Schölkopf, and G. Rätsch. Linear programs for automatic accuracy control in regression. In *Proc. of the 9th Int. Conf. on Artificial Neural Networks (ICANN), Edinburgh, UK*, volume 2, pages 575–580, 1999.

[24] V. N. Vapnik. *The nature of statistical learning theory.* Springer-Verlag, New York, NY, USA, 1995.

[25] J. Weston, A. Gammerman, M. O. Stitson, V. Vapnik, V. Vovk, and C. Watkins. Support vector density estimation. In B. Schölkopf, C. J.C. Burges, and A. J. Smola, editors, *Advances in kernel methods: support vector learning*, pages 293–305. MIT Press, Cambridge, MA, USA, 1999.

Table 2
Errors on the residual gas mass fraction for various training set sizes N with $\lambda = C = 1000$ and a RBF kernel. '–' appears when the result is irrelevant (model mostly constant).

N	Model	*RMSE test*	*RMSE total*	*MAE total*
	1 (experimental model)	1.69	1.54	**3.77**
	2 (prior model)	5.02	4.93	9.74
	3 (mixed model)	5.07	4.22	7.80
	4 (O-model)	2.26	1.47	4.11
15	5 (OP-model)	3.23	2.10	4.95
	6 (D-model)	2.89	2.67	8.92
	7 (DP-model)	**1.07**	**1.24**	4.84
	8 (OD-model)	2.89	2.67	8.91
	9 (ODP-model)	**1.07**	**1.24**	4.84
	1 (experimental model)	4.89	4.42	10.16
	2 (prior model)	4.86	4.93	9.74
	3 (mixed model)	4.84	4.88	9.75
	4 (O-model)	3.86	3.38	9.78
6	5 (OP-model)	4.43	3.88	9.08
	6 (D-model)	3.99	3.50	10.42
	7 (DP-model)	**2.24**	**1.96**	5.99
	8 (OD-model)	3.15	2.83	**5.79**
	9 (ODP-model)	**2.24**	**1.96**	5.99
	1 (experimental model)	–	–	–
	2 (prior model)	4.92	4.93	9.74
	3 (mixed model)	4.89	4.86	9.75
	4 (O-model)	–	–	–
3	5 (OP-model)	5.80	5.46	11.53
	6 (D-model)	–	–	–
	7 (DP-model)	38.5	36.2	70.0
	8 (OD-model)	–	–	–
	9 (ODP-model)	**3.24**	**3.05**	**6.04**

Table 3
Errors on the residual gas mass fraction for various training set sizes N with $\lambda = C = 1000$ and a polynomial kernel.

N	Model	*RMSE test*	*RMSE total*	*MAE total*
	1 (experimental model)	2.06	2.17	**4.89**
	2 (prior model)	4.92	4.89	9.77
	3 (mixed model)	4.97	4.55	9.73
	4 (O-model)	3.58	2.33	5.58
15	5 (OP-model)	8.97	5.83	15.99
	6 (D-model)	2.70	2.66	7.25
	7 (DP-model)	**1.29**	**1.66**	5.10
	8 (OD-model)	2.70	2.66	7.25
	9 (ODP-model)	**1.29**	**1.66**	5.10
	1 (experimental model)	13.79	12.31	29.59
	2 (prior model)	4.84	4.89	9.77
	3 (mixed model)	4.84	4.87	9.77
	4 (O-model)	2.93	2.57	6.17
6	5 (OP-model)	4.79	4.20	14.87
	6 (D-model)	1.88	1.70	3.54
	7 (DP-model)	2.42	2.12	6.34
	8 (OD-model)	**1.75**	**1.60**	**3.53**
	9 (ODP-model)	2.42	2.13	6.35
	1 (experimental model)	6.39	6.06	14.25
	2 (prior model)	4.87	4.89	9.77
	3 (mixed model)	4.92	4.89	9.78
	4 (O-model)	5.65	5.31	11.25
3	5 (OP-model)	6.01	5.65	11.38
	6 (D-model)	5.59	5.26	10.11
	7 (DP-model)	12.52	11.77	27.65
	8 (OD-model)	5.22	4.93	11.21
	9 (ODP-model)	**3.59**	**3.38**	**5.68**

Table 4
Errors on the residual gas mass fraction obtained by the models using RBF kernels for a training set size of $N = 15$ and λ tuned by cross validation.

N	Model	λ	*RMSE test*	*RMSE total*	*MAE total*
	1 (experimental model)		1.69	1.54	3.77
	2 (prior model)		5.02	4.93	9.74
	3 (mixed model)		5.07	4.22	7.80
	4 (O-model)	1	2.29	1.49	4.05
15	5 (OP-model)	2000	3.70	2.41	5.78
	6 (D-model)	1500	2.93	2.98	8.88
	7 (DP-model)	500	**1.12**	**0.73**	**1.88**
	8 (OD-model)	1500	2.93	2.98	8.88
	9 (ODP-model)	500	**1.12**	**0.73**	**1.88**

Table 5
Errors on the residual gas mass fraction obtained by the models using polynomial kernels ($\gamma = 4$) for a training set size of $N = 15$ and λ tuned by cross validation.

N	Model	λ	*RMSE test*	*RMSE total*	*MAE total*
	1 (experimental model)		2.06	2.17	4.89
	2 (prior model)		4.92	4.89	9.77
	3 (mixed model)		4.97	4.55	9.73
	4 (O-model)	1	3.04	1.98	5.38
15	5 (OP-model)	1	3.26	2.12	5.91
	6 (D-model)	400	2.22	1.98	6.22
	7 (DP-model)	10	**0.83**	**0.54**	1.54
	8 (OD-model)	300	2.20	1.83	5.81
	9 (ODP-model)	200	0.84	**0.54**	**1.50**