DOI: http://dx.doi.org/10.1109/TNNLS.2012.2205018

A Hierarchical Approach for Multi-scale Support Vector Regression

Francesco Bellocchio, *Member, IEEE*, Stefano Ferrari, *Member, IEEE*, Vincenzo Piuri, *Fellow, IEEE*, and N. Alberto Borghese, *Member, IEEE*

***Abstract*—Support Vector Regression is based on a linear combination of displaced replicas of the same function, called kernel. When the function to be approximated is non-stationary, the single kernel approach may be not effective, as it is not able to follow the variations in the frequency content in the different regions of the input space. The Hierarchical Support Vector Regression (HSVR) model presented here aims to provide a good solution also in these cases. HSVR is constituted of a set of hierarchical layers, each containing a standard SVR with Gaussian kernel at a given scale. Decreasing the scale layer by layer, details are incorporated inside the regression function. HSVR has been widely applied to noisy synthetic and real datasets and it has shown the ability in denoising the original data, obtaining an effective multi-scale reconstruction of better quality than that obtained by standard SVR. Results compare favorably also with multi-kernel approaches. Furthermore, tuning the SVR configuration parameters is strongly simplified in the HSVR model.**

***Index Terms*—Support Vector Machine, Support Vector Regression, Multiple kernels, Multi-scale regression.**

I. Introduction

Support Vector Machines (SVM) have been introduced as a powerful method for classification [1][2]. They are based on setting the classification boundary, which divides the points having different labels, such that the distance of the boundary from the closest data point is maximized. The boundary between classes is defined by a hyperplane computed as a linear combination of a subset of the data points, called Support Vectors (SV). To identify the SVs and their associated coefficients, the problem is reformulated as a quadratic optimization problem that, being convex, guarantees the uniqueness and the optimality of the solution.

It was soon recognized that the method was able only to classify the data that exhibit linear separability, which is not sufficient for many applications. For this reason, a mapping machinery that transforms the classification problem from its natural space into a higher dimensional space, called feature space, is used. Goal of this mapping is to obtain linear separability in this higher dimensional space. This is not an obstacle for the computation of the solution since the determination of the coefficients does not require the computation of the mapped value of any single data points, but only the value of the inner product of mapped data pairs, that is computed applying a kernel function to the pairs. Therefore mapping is only implicitly computed (a scheme known as "kernel trick"); the kernel output gives a similarity measure of a data pair.

The SVM approach was more recently extended to regression problems [3], domain in which it was Support Vector Regression (SVR). The output of a SVR is computed as

$$Y_{\text{svr}}(x) = \sum_{i=1}^{n} \beta_i \, k(x;\, x_i) + b \tag{1}$$

where β_i and x_i are respectively the weight and the position of each SV, n is the number of SVs, b is the bias and $k(\cdot;\, x_i)$ is the kernel function corresponding to x_i. In the standard approach, a single kernel function is used, which shape is characterized by a set of parameters. Like other methods based on kernels, the quality of the regression depends on the choice of the kernel function and of its parameters, which must be suitable to the current data. In general, this choice, also known as kernel selection [4], is a difficult task: the kernel is often chosen by trial and error, genetic optimization or user expertize.

Besides, the choice of a single kernel function can be questioned. In fact, when the data are characterized by space varying frequency content, the use of a single kernel is not able to produce accurate solutions and approaches based on multiple kernels have been recently investigated [5][6][7][8]. In [5][6] the kernel is defined as a mixture of predefined basic kernels. In this case, the form of the output becomes:

$$Y_{\text{svr}}(x) = \sum_{i=1}^{n} (\beta_i \sum_{j=1}^{m} \mu_j k_j(x,\, x_i)) + b \tag{2}$$

where the type and the number, m, of kernels, $k_j(\cdot;\, \cdot)$, used in the linear combination have to be chosen a priori and the mixing coefficients, μ_j, are determined in the optimization phase. However, even in this case, the solution is a linear combination of copies of the same (multiple) kernel function.

The problem of using a single kernel is highlighted in the example reported in Fig. 1, similar to that discussed in [9]. The data points have been sampled on the curve $h(\cdot)$

$$h(x) = \sin(2\pi x^4) + x \tag{3}$$

N.A. Borghese is with the Department of Computer Science, Università degli Studi di Milano, Italy, e-mail: borghese@dsi.unimi.it.

F. Bellocchio, S. Ferrari and V. Piuri are with the Department of Information Technology, Università degli Studi di Milano, Italy, e-mail {francesco.bellocchio,stefano.ferrari,vincenzo.piuri}@unimi.it.

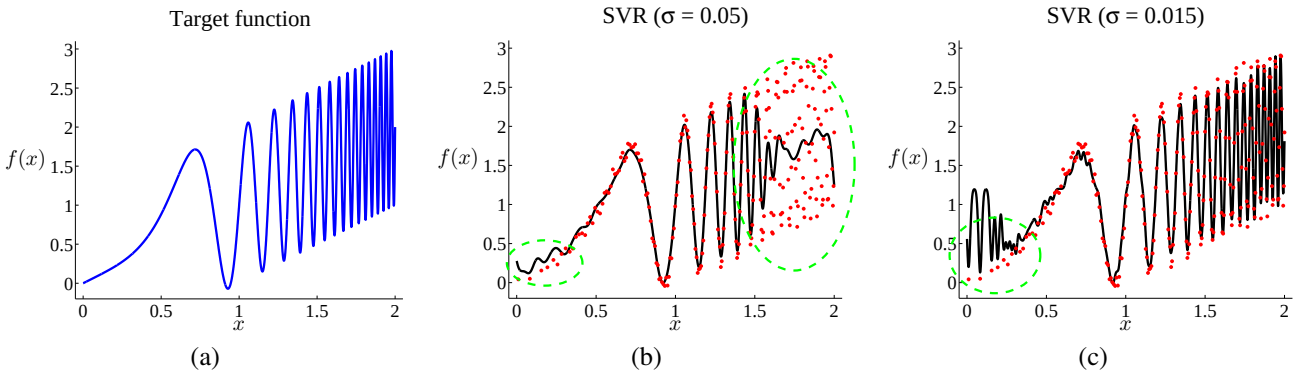


Fig. 1. (a) A function with non-stationary frequency content, and (b)–(c) two SVR using a single Gaussian kernel with two different scale parameters, σ. (b) A large scale kernel provides a smooth regression on the samples taken from the function in (a), but is unable to reconstruct the details, while (c) a small scale kernel suffers of overfitting providing spurious oscillations and poor generalization.

whose local frequency content increases with x. The sampling step is decreased with the local frequency according to $\frac{1}{120x}$. The regression computed with a large kernel fails in reconstructing the details as shown in Fig. 1b. On the other side, using a kernel with a small scale, such as the one used in Fig. 1c, the regression is prone to overfitting and lacks generalization in scarcely sampled regions.

A possible solution is to adapt the multiple kernel to the local frequency content as suggested in [7][8] where a localized multiple kernel model is presented. In this case, the parameters μ_i become a function of both the SVs position, x_i, and the data position, x, and therefore $\mu_j = \mu_j(x_i, x)$ in (2). Optimization becomes more complex, but an elegant two passes solution is shown in [8].

A different solution is based on using the same kernel with different scales. In [9] the solution is computed using a set of groups of kernels, where the kernels in each group have the same scale. The optimization problem is written to take into account also the multiple scales of the kernel inside the optimization function. This increases significantly the computational complexity and the method is just suitable for small size problems. Furthermore it requires the a priori selection of the set of the different kernel scales.

In [10] the multiple scales are originated from a wavelet decomposition. The approach requires that the coefficients of all the levels are estimated together, which leads again to a very large number of parameters for the optimization engine.

A more manageable solution can be obtained when the regression is built through hierarchical iterative approximation. Such approach has been explored both in the neural networks [11][12][13] and in the machine learning domains [14]. In the neural networks domain, a self-organizing multi-scale model, called Hierarchical Radial Basis Function (HRBF), was proposed in [12]. Thanks to the allocation of units with smaller scales only where the data contain higher frequencies, a uniform residual error is guaranteed with the use of a limited number of units. This principle has been explored also in machine learning domain. In [15] a first approximation at a very coarse level is first produced with a SVR featuring a kernel with a large scale and then refined using kernels with smaller scales. However, in such approach the number of SVs becomes quickly extremely large with prohibitive computational time. To reduce the number of training points and hence the number of SVs, iterative partitioning of the input space was introduced and the training set formed of the centroids of the samples inside each partition. This procedure might reduce the quality of the input data, especially for data spaces of high dimensionality.

Another approach is based on boosting, originally introduced in the classification domain [16], and then extended to regression [17][18]. Boosting is a technique that iteratively uses suboptimal solvers (called weak learners) for increasing the accuracy of the regression. Recently [14], boosting has been extended to multi-scale regression: one basis function at a time is added, to decrease the global error. Each time, the parameters are updated. The scale is decreased when there is no error reduction with the actual scale. This produces a very long configuration time.

Another major problem with SVR, even more important when using multi-scale kernels, is the number of SVs. This problem has been studied in several works [19][20][21][22][23] targeted to classification. These approaches can be classified into two main families. The first redefine the optimization problem in order to control the number of SVs inserted [20][21]. The second family realizes the reduction of SVs in two steps: the standard training of the SVM is performed first and then a pruning procedure is applied; in general, pruning methods involve a tradeoff between SVs reduction and accuracy loss.

In [19] the set of row vectors of the kernel matrix (K in (6)) is first partitioned using K-means algorithm. Pruning operates cluster-wise, deleting a row vector if it is similar enough to its orthogonal projection in the space spanned by the others vectors of its cluster. After pruning the coefficients of the SVs left have to be recomputed. Although this step requires only 0.006 of the optimization time (for each SV pruned), when a medium/large number of SVs is present ($>$ 100 SVs) pruning time exceeds that of training.

In [24] the hyperplane found by the SVM is seen from a mechanical point of view. Each SV exerts a force on the hyperplane and the minimum of the optimization problem is determined when the system is in an equilibrium state. The pruning is operated by substituting two SVs with a SV that exerts an equivalent force with the rationale that, after

substitution, the system is, again, in an equilibrium state. However, the new SVs, generally, do not exert exactly an equivalent force of those deleted and an approximation is accepted. The selection of the pairs of SVs for substitution is performed by means of a greedy heuristic procedure: the pair such that their substitution determines the minimum deviation from the previous solution is selected. The stop criterion is based on monitoring the accuracy loss for each replacement: when it goes over a given threshold, the pruning procedure is stopped. This approach has the advantage of controlling the loss of accuracy in each pruning step, but, as the previous one, it is very computational expensive.

We present here a novel approach that adapts the local scale to that of the data keeping the number of SVs and configuration time comparable with classical SVR. The approach is inspired to the HRBF model [12] and is based on interleaving a regression estimate step with a pruning step, in a novel way.

The paper is structured as follows. After reviewing the SVR model in Section II, we introduce our HSVR model in Section III and show some results in Section IV. We discuss them in Section V and draw some conclusions in Section VI.

II. Support Vector Regression

Let $S = \{(x_1, z_1), ..., (x_n, z_n)\}$ be the set of n data points that constitute the training set, where $x_i \, (1 \leq i \leq n)$ belongs to $X \subseteq \mathbb{R}^D$ and $z_i \in Z \subseteq \mathbb{R}$. Aim of SVR [3] is to find a regression function, $f : R^D \rightarrow R$ of this kind:

$$z = f(x) = \omega^T \phi(x) + b, \tag{4}$$

where $\phi(x)$ maps a data point x into a higher dimensional space, called feature space. The mapped point is then projected over Z through the weights ω and the threshold constant b, $b \in \mathbb{R}$, which can be found solving the optimization problem:

$$\begin{aligned} \min_{\omega, b, \xi_i^+, \xi_i^-} \quad & \tfrac{1}{2}\omega^T\omega + C\sum_{i=1}^n \xi_i^+ + C\sum_{i=1}^n \xi_i^- \\ \text{s.t.} \quad & z_i - \omega^T\phi(x_i) - b \leq \varepsilon + \xi_i^+ \\ & \omega^T\phi(x_i) + b - z_i \leq \varepsilon + \xi_i^- \\ & \xi_i^+, \xi_i^- \geq 0, \; i = 1, \ldots, n \end{aligned} \tag{5}$$

where $\varepsilon \geq 0$ determines a "tube" around the regression curve inside which the points are considered sufficiently close to the curve itself and therefore do not contribute to the cost function (ε-insensitive loss function, see Fig. 2). Stated in a different way, ε controls the admissible uncertainty on the data points.

The parameter C adjusts the tradeoff between the closeness of the solution to the data points and its smoothness. $\xi^+ = \{\xi_1^+, \ldots, \xi_n^+\} \in \mathbb{R}^n$ and $\xi^- = \{\xi_1^-, \ldots, \xi_n^-\} \in \mathbb{R}^n$ are slack variables that measure the distance of each data point from the ε-tube: they take into account that some points may be distant more than ε from the regression curve.

Introducing the Lagrange multipliers α_i^+ on the constraints corresponding to ξ_i^+ and α_i^- on those corresponding to ξ_i^-,

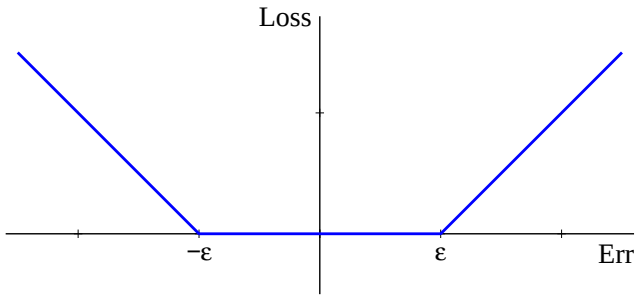


Fig. 2. Loss function commonly used for SVR: data points that are close to the regression curve for less than ε do not contribute to the penalization of the current solution.

the following dual problem is obtained:

$$\begin{aligned} \max_{\alpha^+, \alpha^-} \quad & -\tfrac{1}{2}(\alpha^+ - \alpha^-)^T K(\alpha^+ - \alpha^-) \\ & -\varepsilon\textstyle\sum_{i=1}^n(\alpha_i^+ + \alpha_i^-) + \sum_{i=1}^n z_i(\alpha_i^+ - \alpha_i^-) \\ \text{s.t.} \quad & \textstyle\sum_{i=1}^n(\alpha_i^+ - \alpha_i^-) = 0 \\ & \alpha_i^+, \alpha_i^- \in [0, C], \; i = 1, \ldots, n \end{aligned} \tag{6}$$

where $\alpha^+ = \{\alpha_1^+, \ldots, \alpha_n^+\} \in \mathbb{R}^n$ and $\alpha^- = \{\alpha_1^-, \ldots, \alpha_n^-\} \in \mathbb{R}^n$ are the dual variables, and $K \in \mathbb{R}^{n \times n}$, $K_{i,j} = k(x_i, x_j)$ [3], is the kernel matrix evaluated from a kernel function $k : X \times X \rightarrow \mathbb{R}$.

To determine α^+, α^-, and b non-linear optimization engines are used, like [25] and [26]. Using the stationary Karush-Kuhn-Tucker (KKT) conditions derived from (6) and introducing the new variables $\beta_i = \alpha_i^+ - \alpha_i^-$, the regression function (4) can be rewritten as:

$$f(x) = \sum_{i=1}^{n} \beta_i \, k(x, \, x_i) + b \tag{7}$$

where $f(\cdot)$ is expressed as a linear combination of replicas of the kernel function, $k(\cdot, \cdot)$, relative to each point x_i. This substitutes the mapping function, $\phi(\cdot)$ used in (4).

For the KKT conditions and (6) the β_i coefficients have to satisfy the following relationships:

$$\begin{cases} |\beta_i| = 0, & |z_i - f(x_i)| < \varepsilon \\ |\beta_i| \in [0, \, C], & |z_i - f(x_i)| = \varepsilon \\ |\beta_i| = C, & |z_i - f(x_i)| > \varepsilon \end{cases} \tag{8}$$

The points that have non-zero β coefficients are called support vectors (SV). In practice, a tolerance threshold δ, is introduced to allow finding a reasonable number of points on the tube boundary, and relationships (8) are substituted by:

$$\begin{cases} |\beta_i| = 0, & |z_i - f(x_i)| < \varepsilon - \delta \\ |\beta_i| \in [0, \, C], & \varepsilon - \delta \leq |z_i - f(x_i)| \leq \varepsilon + \delta \\ |\beta_i| = C, & |z_i - f(x_i)| > \varepsilon + \delta \end{cases} \tag{9}$$

The SVs that lie outside the ε-tube are called "bounded" and the absolute value of their associated β is set equal to C. As a result, the sum in (7) can be limited only to the SVs.

Among others, the most commonly used kernel for regression is the Gaussian kernel:

$$k(x,\, x_i) = G(||x - x_i||; \sigma) = \exp\left(-\frac{||x - x_i||^2}{\sigma^2}\right) \quad (10)$$

The parameter σ affects the extension of the influence of each support vector, x_i, in its neighborhood. Using Gaussians of very small scale would allow reconstructing the finest details, while a large scale kernel would provide a rough approximation. However, the use of a single scale kernel may not be the best choice when the dataset is sampled from a non stationary source that generates data over a smooth manifold in some regions, and data with rapid variations in others (Fig. 1a). In fact, when operating with a kernel with a small scale, the reconstruction of smooth regions may induce, in the best case, a waste of computational resources. Moreover, when the training samples are too spaced with respect to the scale parameter, the resulting regression does provide poor generalization. On the other hand, when a kernel with a large scale is employed, rapidly varying regions could be reconstructed only at the price of using a very large number of SVs and a very large value of C is required (which increase the computational cost and can increase the overfit of the data).

An approximation scheme that would adapt, for each SV, the scale of the kernel to the frequency content of the region in which the SV is situated, represents a better solution. This is the approach proposed in the following.

III. The HSVR Model

The Hierarchical Support Vector Regression (HSVR) model is composed of a pool of L layers, each constituted of a single-kernel SVR, $\{a_l(\cdot)\}$ characterized by a proper scale. The different layers are organized as a hierarchy where the scale, determined by a parameter, σ_l, decreases when the layer number increases, that is $\sigma_l \geq \sigma_{l+1}$ holds. The output of the HSVR model is obtained as the sum of the output of the layers:

$$f(x) = \sum_{l=1}^{L} a_l(x;\, \sigma_l) \quad (11)$$

When the kernel is the Gaussian function, the output of each l-th layer can be written as:

$$a_l(x; \sigma_l) = \sum_{k=1}^{M_l} \beta_{l,k}\, G(||x - x_{l,k}||; \sigma_l) + b_l \quad (12)$$

where M_l is the number of SVs, $\beta_{l,k}$ is the coefficient of the k-th SV, and b_l is the bias of the l-th layer. Therefore, the l-th SVR layer realizes a reconstruction up to a certain scale, determined by σ_l. HSVR configuration proceeds adding and configuring one layer at a time, proceeding from the layer featuring the largest scale to that featuring the smallest one.

The first layer is trained such that the distance between the regression curve produced by the first layer itself and the data is minimized (5). All the other layers are trained to approximate the residual that is the difference between the original data and the output of the HSVR model produced by the layers configured up to that stage. The measure of the residual for each layer, r_l, is given as:

$$r_l(x_i) = r_{l-1}(x_i) - a_l(x_i) \quad (13)$$

The l-th layer will be configured with the training set, S_l, defined as: $S_l = \{(x_1, r_{l-1}(x_1)), ..., (x_n, r_{l-1}(x_n))\}$ and $r_0(x_i) = z_i$ is assumed.

The value of the scale parameter of the first layer, σ_1, is somehow arbitrary; for instance it can be chosen proportional to the size of the input domain (e.g., the length of the diagonal of the data bounding box). Also the decrease rule of σ can be arbitrary. Generally, halving the value of σ for each layer ($\sigma_{l+1} = \frac{\sigma_l}{2}$), as done in wavelet decomposition, produces satisfactory results. If the σ decreases more slowly the accuracy of the solution could improve, but the number of layers (and the number of SVs) increases as well. New layers are added during training until a given stopping criterion is satisfied (e.g., when the validation error does not decrease anymore).

Two other parameters are defined for each layer: C_l, the tradeoff between the regression error and the smoothness of the solution, and ε, which controls the amplitude of the ε-insensitivity tube around the solution itself.

Although few attempts to determine a proper value of C have been proposed in regularization theory [27][28], C is usually experimentally set by trial and error. In this work, C_l is chosen, for each layer, as J times the standard deviation of the residuals used to configure the $l_t h$ layer:

$$C_l = J\, \text{std}(r_{l-1}(x_i)) \quad (14)$$

with the following motivation. First, we notice that as C_l is the value assumed by the Lagrange multipliers associated to the SVs of the l-th layer ((6) and (8)), its value represents the maximum weight that can be associated to each kernel in (7). For the input space regions where the Gaussians associated to SVs have no significant overlap (this depends both on the Gaussian scale parameter and on the data density), the value of C_l is approximately the maximum value that can be assumed by the regression function in those regions as the Gaussian kernel maximum is equal to one (10)). For this reason, C_l should be large enough to allow the regression curve reaching the maximum or minimum value of the data points inside the whole input domain. On the other hand, a too large value of C_l could favor overfitting. Experimental results on different datasets have suggested to choose the value of J in the interval $(0,\, 5]$, that represents a tradeoff between these two requirements. Inside the above range, results are largely independent on the value of J.

Similar to SVR, the ε parameter cannot be determined from the dataset. ε could be set proportional to the accuracy required for the regression as its value is related to noise amplitude [29].

A. Training set reduction

The drawback of the previous scheme is the total number of SVs, which is significantly higher than in standard SVR.

Moreover, experiments shows that in HSVR the layers with a larger value of σ have a number of SVs similar to the layers

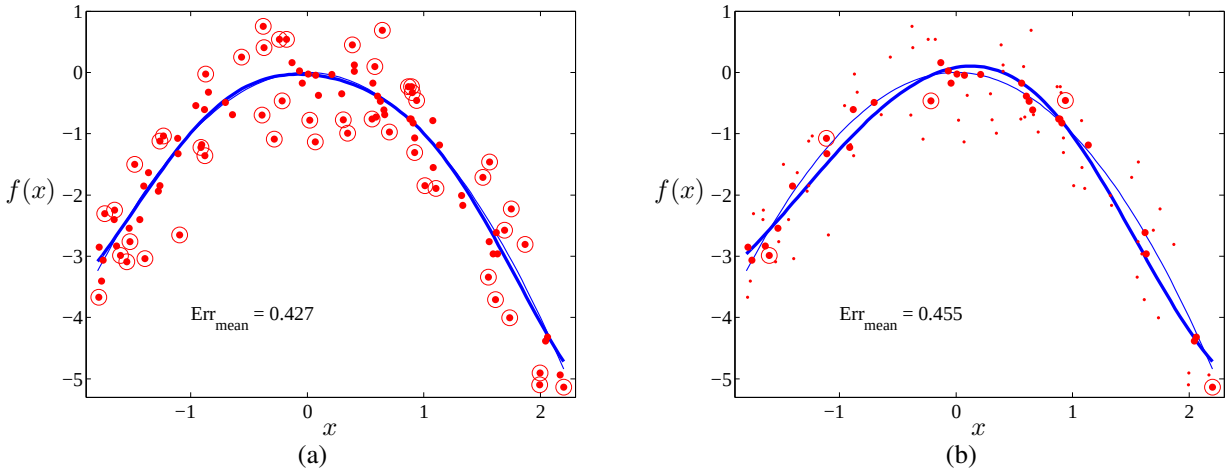


Fig. 3. Data points reduction. A smooth function is shown in thin line in both panels. A set of 100 points have been randomly sampled over it and a Gaussian random quantity has been added to them. These points are displayed as dots. Thick dots represent all the points used by the optimization engine to determine the regression represented as a thick line. Circled dots represent the SVs. In panel (a) the SVR curve obtained through standard SVR ($\varepsilon = 0.416$, $C = 9.67$, $\sigma = 1.66$, $\text{Err}_{\text{mean}} = 0.427$) and in panel (b) the regression curve obtained considering only the points in S_l' (15), where Err_{mean} is the mean of the absolute reconstruction error. Notice that in the latter case only 32 data points are used in the optimization procedure (the unused points are shown as small dots). The number of SVs drops from 49 to 5. Both regression curves are contained inside a ε-tube around the real function.

with a smaller σ. This appears in contrast with common sense, as fewer units should be required to realize a reconstruction at a larger scale, but it is due to the fact that all the data points distant from the regression curve by more than ε are selected as SVs (cfr. Fig. 3). Hence, in the first layers, where the HSVR output has a low frequency content, many data points lie far from the curve and are still selected as SVs, thus leading to an unnecessary high number of SVs.

To avoid this, after each layer has been configured, a pruning step is carried out to reduce the number of the SVs. The cost function (6) is then minimized a second time, considering only the reduced training set to obtain the final approximation for each current layer.

To reduce the number of the SVs, we first notice that the distance of a training point from the regression curve measures the suitability of the current curve to describe the information conveyed by that data point. In this sense, points too distant from the regression curve cannot be "explained" by the curve and their utility can be questioned: they can be regarded as outliers. For these reasons, an acceptable approximation of the regression curve should be obtained using only those points that lie close to the curve.

This intuition has been confirmed experimentally. We have observed that the quality of the regression at a given scale does not degrade significantly if we compute the regression considering only the points close to the ε-tube (cf. Fig. 3). The closeness of a point to the ε-tube can be assessed only after the computation of the regression itself, that is carried out considering all the training points. Afterwards, in a second pass, the regression is computed again considering only the points close to the ε-tube. More formally, let us consider the l-th layer and the regression computed for that layer, $a_l(x)$, using the complete training set, S_l. Let us define S_l' the set constituted only of those SVs which lie on the border of the ε-tube and those whose distance from $a_l(x)$ is less than $\varepsilon/2$ (i.e., those that belong to the most internal part of that tube):

$$S_l' = \left\{ (x_i,\, r_{l-1}(x_i)) \;\middle|\; ||r_l(x_i)| - \varepsilon| < \delta \vee |r_l(x_i)| < \frac{\varepsilon}{2} \right\} \quad (15)$$

where δ is the tolerance parameter that determines the thickness of the ε-tube margin (9).

Therefore, we have structured the configuration phase of each layer in two sequential steps: the first one provides the regression curve, a_l, considering all the training points, while the second one, a_l', realizes an efficient representation of the regression curve by considering only a selected subset of these points. It should be noticed that when pruning takes place, a_l' substitutes a_l in (11) and (13).

In order to cope with the diminished points density in S_l', the value of the parameter C_l is increased proportionally in the second optimization step:

$$C_l' = C_l \frac{|S_l|}{|S_l'|} = J \,\text{std}(r_{l-1}(x_i)) \frac{|S_l|}{|S_l'|} \quad (16)$$

It should be remarked that, similarly to [12][13], any reconstruction error induced by the information loss due to the reduction of the training set will flow into the residual, r_l, which is used to configure the next layer of the model. Therefore such error is not critical, as it will be taken care by the next layers. The reduction procedure could be applied also to standard SVR. However, as in standard SVR there is no chance to recover the error introduced by pruning, more care should be taken in selecting the points in the reduced training set. Results on the reduction procedure is discussed in Section V.

IV. Results

We report here the results obtained using the HSVR model on both simulated and real data and compare them with those obtained with standard SVR [3], in terms of number of SVs,

TABLE I
ACCURACY ON SYNTHETIC DATASET

	Err_{mean}	Err_{std}	RMSE	#SVs	Time [s]
HSVR	0.0282	0.0262	0.0385	1545	0.308
HSVR (red.)	0.0313	0.0338	0.0460	243	0.382
SVR	0.0816	0.167	0.186	149	0.451

computational time and accuracy. The accuracy is assessed through the root mean square error (RMSE), and the mean absolute error (Err_{mean}) and its standard deviation (Err_{std}). These are computed over a test set, different from the training one. A third set, different from both, called validation set, is used for choosing the optimal value of the parameters, which are ε, σ, and C for the standard SVR model, and of ε and C only for the HSVR model. C is set according to (14) and (16) and the value of σ for the first layer, σ_0 , is set equal to the size of the input domain for the HSVR model in order to cover the entire domain.

The validation set is used also to decide when the growth of the HSVR model has to be stopped: no new layer is added when the validation error does not decrease anymore and the last layer is discarded, as it can easily contribute to overfitting.

The optimization problem in (6), which arises for both the hierarchical and the standard SVR approach, was solved through the LibCVM Toolkit Version 2.2 [25]. This software has shown the same accuracy of SVM^{light} [26] (that is one of the most used software packages for SVM) with a substantial saving in computational time. This was measured, on a PC equipped with an Intel Pentium 4, at 2.40 GHz, 512 KB of cache, and 512 MB of memory.

A. Regression on synthetic data

The space-varying function $h : \mathbb{R} \to \mathbb{R}$, defined in (3) and plotted in Fig. 1a, is considered here as it allows stressing the limits of the single scale SVR approach. The training dataset has been obtained sampling (3) in 252 points such that the sampled data density is proportional to the local frequency content, and adding a random uniform quantity in $[-0.1, 0.1]$ to simulate sampling error. The regression has been evaluated using a test set and a validation set, each composed of 500 points sampled from $h(\cdot)$ with a uniform distribution.

The accuracy of standard SVR was evaluated on the validation set for all the possible combinations of the following values of the parameters ε, σ, and C:

$$\varepsilon \in \{0.0, 0.005, 0.01, 0.025, 0.05, 0.075, 0.1, 0.2\} \quad (17)$$

$$\sigma \in \{0.015, 0.022, 0.0313, 0.0625, 0.125, 0.25\} \quad (18)$$

$$C \in \{0.5, 1, 1.5, 2, 5, 10, 20\} \quad (19)$$

and the best SVR is considered in the comparison. This is shown in Fig. 4a and it was obtained with $\varepsilon = 0.05$, $\sigma = 0.022$, and $C = 20$. Notice the poor approximation on the right side of the curve and the spurious oscillations on the left side. These are not present in the curve provided by the HSVR model shown in Figs. 4b–c.

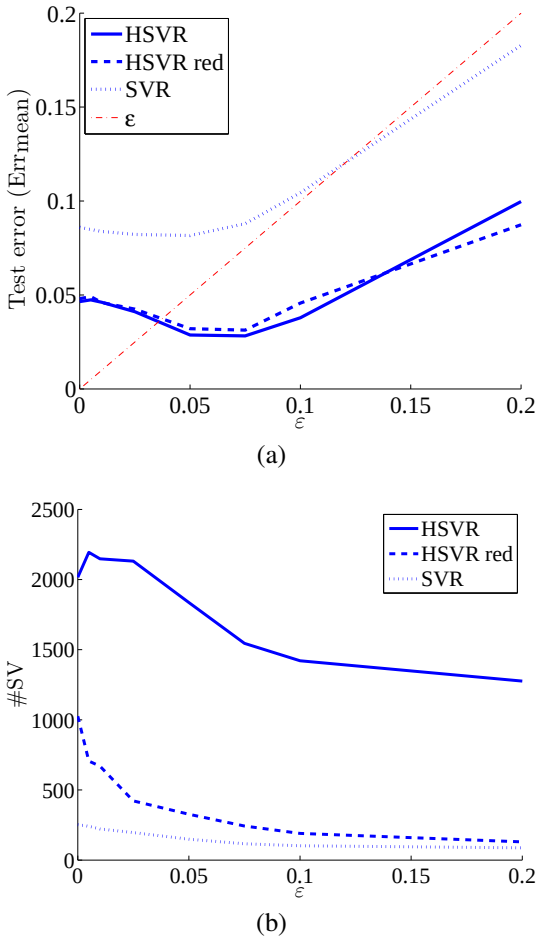


Fig. 5. (a) Mean test error and (b) number of SVs used, as a function of ε. For reference, in panel (a) the value of ε has been reported as a dot-dashed line in panels (a)-(c).

This observation is supported by the quantitative data reported in Table I on the accuracy of the different models.

The computational time for the HSVR model is referred to the entire process of configuring the 9 layers required before the growth stops, while for the SVR model it does not consider the process for searching for optimum value of ε, C and σ, but is referred only to the computation of the solution with the best parameters, $\varepsilon = 0.05$, $\sigma = 0.022$, and $C = 20$.

If we considered also the search for the optimal value of the parameters, the total configuration time would increase significantly to 43.8 s. For sake of comparison, the configuration time for HSVR increases to 3.27 s and 4.49 s for HSVR without and with reduction, to test the eight values of ε.

Enlarging the search space of C up to 100,000, the accuracy of SVR improves. In fact, the test error decreases from 0.0816 down to 0.0517, although it remains higher than that of HSVR. However, the time required to compute this solution increases enormously to 3,129 s.

We have also investigated the role of ε. As it can be seen in Fig. 5, the test error produced by the HSVR model is below ε for $\varepsilon > 0.05$. This is much smaller of the optimal value of 0.075. This means that the data points, on average, lie inside the ε-tube around the curve. Instead, for SVR the optimal value

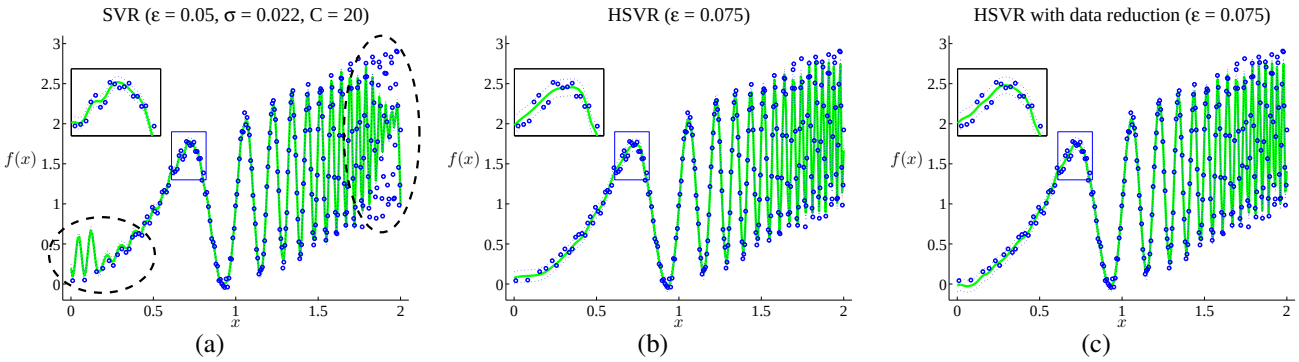


Fig. 4. Reconstruction provided by standard SVR (a) with the best parameters ($\varepsilon = 0.05$, $\sigma = 0.0313$, $C = 20$). Notice the poor approximation on the right side of the curve and spurious oscillations on the left. Reconstruction provided by HSVR (b) and HSVR with data points reduction (c) with $\varepsilon = 0.075$. The dashed lines limit the ε-insensitive region (i.e., the data points that lie inside this region do not increase the cost function value (5)).

TABLE II
PERFORMANCE OF THE HSVR MODEL ($\varepsilon = 0.075$) ON THE SYNTHETIC DATASET

	HSVR					HSVR red.				
# Layer	Err_{mean}	Err_{std}	RMSE	#SVs (tot.)	time [s]	Err_{mean}	Err_{std}	RMSE	#SVs (tot.)	time [s]
1	0.479	0.333	0.583	237 (237)	0.017 (0.017)	0.520	0.302	0.602	3 (3)	0.017 (0.017)
2	0.458	0.337	0.569	240 (477)	0.02 (0.037)	0.453	0.341	0.567	4 (7)	0.018 (0.035)
3	0.412	0.368	0.552	227 (704)	0.019 (0.056)	0.436	0.361	0.566	6 (13)	0.02 (0.055)
4	0.359	0.366	0.511	220 (924)	0.02 (0.076)	0.373	0.380	0.532	9 (22)	0.021 (0.076)
5	0.294	0.367	0.470	201 (1125)	0.047 (0.123)	0.321	0.385	0.501	15 (37)	0.052 (0.128)
6	0.212	0.329	0.391	171 (1296)	0.078 (0.201)	0.237	0.344	0.418	35 (72)	0.102 (0.23)
7	0.104	0.221	0.244	131 (1427)	0.065 (0.266)	0.141	0.273	0.307	51 (123)	0.073 (0.303)
8	0.0328	0.0390	0.051	82 (1509)	0.036 (0.302)	0.0367	0.0446	0.0577	79 (202)	0.067 (0.37)
9	0.0282	0.0262	0.0385	36 (1545)	0.006 (0.308)	0.0313	0.0338	0.046	41 (243)	0.012 (0.382)

of ε is 0.05, which is smaller than the test error achieved (0.0816). This means that a relatively large number of data points are not contained inside the ε-tube as can be seen in Fig. 4a. Moreover, as expected, the number of SVs decreases with the increase of ε.

The increase in the detail of the HSVR model is shown in Fig. 6, where the output of each layer is shown. Notice that data reduction makes the approximation smoother in the first layers, but smoothing tends to disappear in the last layer. This is highlighted in Fig. 7 where the test error obtained with data reduction is superimposed to that obtained without data reduction. Quantitative figures are summarized in Table II.

B. Regression on real data

Figure 8 shows a typical point cloud sampled from a real artifact (a panda mask) through a 3D scanner [13]. As the points have been sampled from a single point of view, they belong to a 2.5D surface that can be described as a $\mathbb{R}^2 \rightarrow \mathbb{R}$ function. These points are therefore suitable to SVM regression. The dataset is composed of $22,000$ points, $18,000$ of which, randomly chosen, are used for training, $2,000$ for validation, and $2,000$ for testing. Since the input features can be measured in different domains, the data are often normalized before optimization. Here, each coordinate of the points was normalized to fit inside $[-1, 1]$.

Besides, in order to limit border effects, validation and test error has been computed in the inner region of the dataset, considering only points distant from the closest boundary by more than 0.1.

SVR was computed with all the combinations of the following values of the parameters (ε, σ, and J):

$$\varepsilon \in \{0,\ 0.0025,\ 0.005,\ 0.01,\ 0.02\} \tag{20}$$

$$\sigma \in \{0.188,\ 0.0938,\ 0.0469,\ 0.0234\} \tag{21}$$

$$J \in \{0.5,\ 1,\ 2,\ 5\} \tag{22}$$

The parameter C, similarly to (14), was set proportionally to the standard deviation of the height, z, of the points through the proportionality factor J:

$$C = J\,\text{std}(z) \tag{23}$$

The HSVR was computed using all the combinations of the values of J in (22) and ε in (20). The surface associated to the lowest test error is shown in Fig. 9a–c for SVR and HSVR with and without reduction. However, although sampled points lie inside the ε-tube region, the results are not nice due to high frequency variability. A better visual appearance can be obtained only from HSVR models, discarding a few of the last layers. This is clearly evident in Fig. 9e–f.

The test error and number of SVs for HSVR and SVR, averaged over five different randomizations of the dataset have been reported in Fig. 10. For each ε the best, the average and the worst case are plotted with respect to the other configuration parameters. The best results are reported

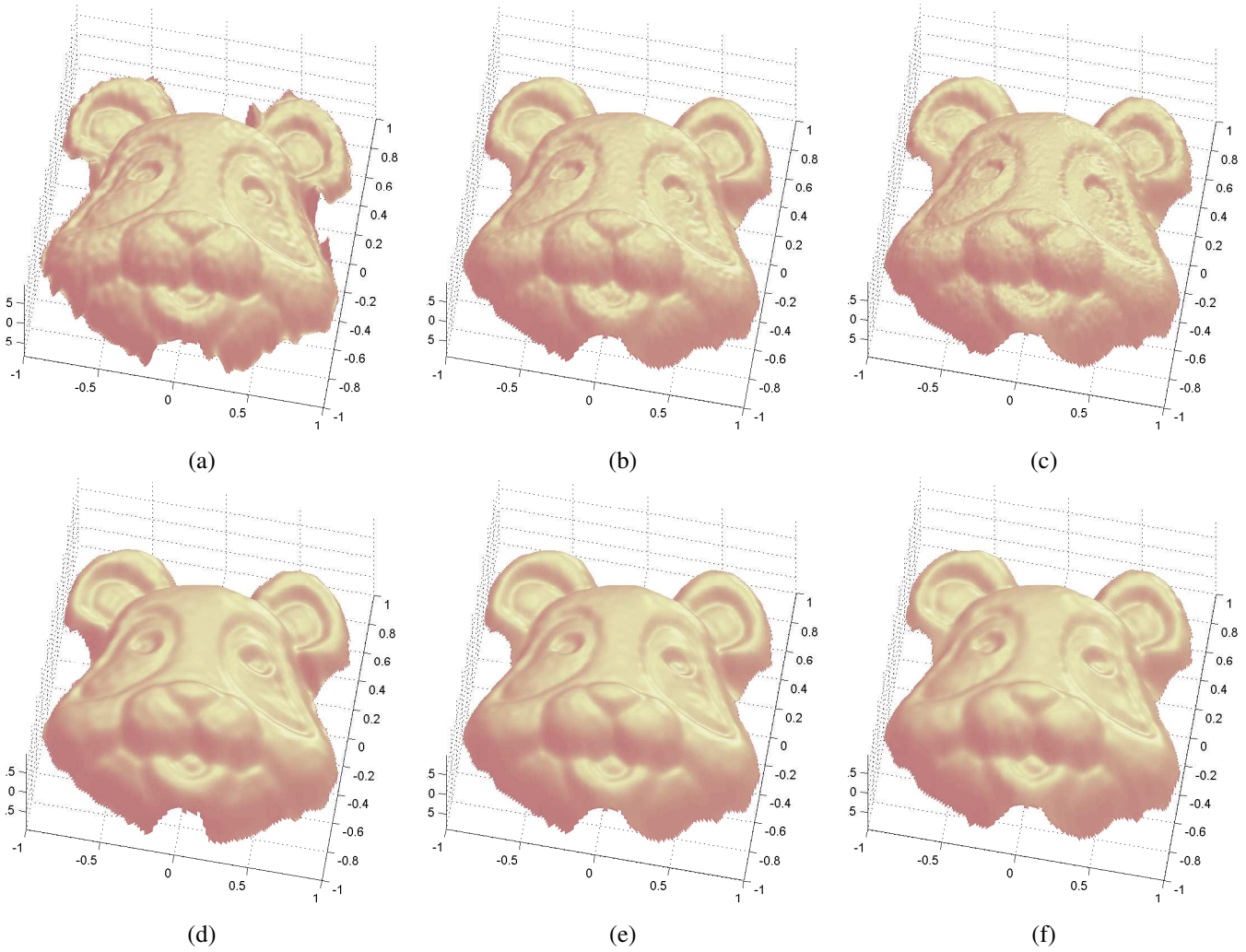


Fig. 9. Panels (a), (b), and (c) show the surfaces that determine the lowest test error for SVR, HSVR and HSVR with reduction. The parameter used were $J = 0.5$, $\varepsilon = 0.005$ and $\sigma = 0.0469$ for SVR, $J = 0.5$ and $\varepsilon = 0.01$ for HSVR, and $J = 1$ and $\varepsilon = 0.01$ for HSVR with reduction. Although these surfaces are optimal in terms of test error, their visual appearance is not of good quality. A better result is shown in panels (d), (e), and (f). In (d) the surface obtained through SVR with a suboptimal set of parameters ($J = 5$, $\varepsilon = 0.005$, and $\sigma = 0.0938$) is shown. In panels (e) and (f) the surface from the same HSVR models for (b) and (c) are used, but discarding some of the last layers (one of seven for (e) and three of ten for (f)).

TABLE III
RESULTS FOR "PANDA MASK" DATASET

	Err_{mean}	Err_{std}	RMSE	#SVs	time [s]
HSVR	0.0110	0.0115	0.0160	100 448	682
HSVR (red.)	0.0112	0.0119	0.0163	11 351	1 104
SVR	0.0111	0.0117	0.0161	12 442	382

in Table III that show that the best test error is similar in all the three models (all in the range $[0.0110, 0.0112]$). This consideration can be extended to all the best models obtained for a given value of ε (Fig. 10a).

On the contrary, the average and worst case test error (computed over all the other considered parameters combinations) are much higher for SVR than for HSVR. In fact, the average (worst) test error is 0.0113 (0.012), 0.0116 (0.015), and 0.0140 (0.019) for respectively HSVR, HSVR with reduction, and SVR.

Although the test error for all the three optimal models is very similar, the configuration time is very different: 682 s for HSVR, 1,104 s for HSVR with reduction, and 382 s for SVR. This large difference becomes smaller when suboptimal configuration parameters are considered. In fact, average configuration time is 1,024 s, 1,241 s, and 1,093 s for HSVR, HSVR with reduction and SVR.

However, the time required to explore the parameters space has to be added to this time. The dimensionality of this space is smaller for HSVR, as σ has not to be considered. In the present case, where 80 parameters combinations have been used for SVR and 20 combinations for HSVR, a time saving of 25.4% has been observed. In fact, for SVR the total configuration time, including parameters space search, is 87,410 s, while for HSVR and HSVR with pruning the total configuration time is 16,845 s and 22,168 s respectively. If we set a predefined value of $J = 1$, that was almost optimal for all the analyzed data sets, the configuration time of HSVR drops further to 4,472 s, less than 1/20th of standard SVR.

We have challenged HSVR also on datasets reported in the literature. Table IV summarizes the results on the "motorcycle"

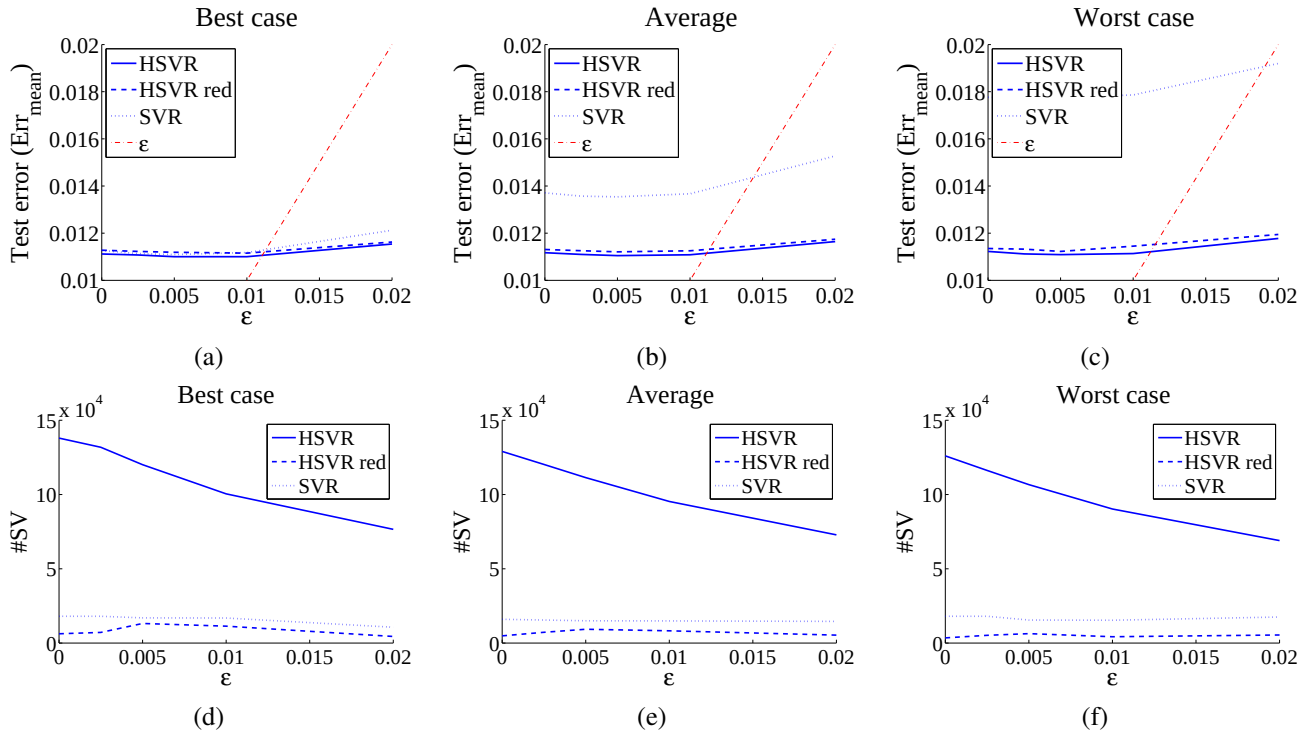


Fig. 10. Test error (a–c) and number of SVs (d–f) used by the SVR and HSVR model for the Panda dataset as a function of ε are reported for the best, average and worst cases, with respect to the different parameter combinations (20)–(22). For reference, the value of ε has been reported as a dot-dashed line.

dataset [30]. Results of HSVR are averaged over ten different randomizations of the data set to create different training and test sets, and they are compared with the multi-scale approach of [9]. We were not able to replicate the same error reported in [9] using SVR: the error obtained by our implementation of SVR was in fact of 0.2272 against 0.2334 reported in [9]. This may depend on different randomizations of the data set and/or on the different optimization engine used. In any case, we remark that both HSVR and MS-SVR do produce a regression error smaller than standard SVR. HSVR uses less SVs than standard SVR demonstrating its ability to catch local frequency content, although spending more time for the configuration (0.003 s for SVR versus 0.011 s for HSVR); it should be noticed that no search for the hyperparameters has been carried out, since, for sake of comparison, we used the same values reported in [9]. As expected, the approach in [9] does use even less SVs but it requires solving a global optimization problem in one pass for all the scales; this requires a much larger configuration time and makes this approach feasible only for small datasets. Moreover, the number of scales has to be defined a-priori; this can lead to over or under-fitting the data.

We have challenged HSVR also on the "housing" dataset [31], in which data lie in a 13 dimensions manifold and compare the results with the multi-kernel method described in [32] in Table V. Both HSVR and MKSVR, outperform standard SVR in terms of accuracy. We also remark that HSVR requires a larger number of SVs with respect to both SVR and MKSVR. However the advantage of HSVR of searching a smaller parameters space is reflected in a shorter configuration time (a total of 6.97 s for 125 hyperparameter combinations of SVR and 5.43 s for 20 hyperparameter combinations of HSVR).

TABLE IV
RESULTS FOR "MOTORCYCLE" DATASET

	σ	ε	C	#SVs	RMSE
SVR	1	0.1436	10	51.8	0.2272
HSVR red.	10	0.1436	1	26.1	0.2216
SVR [9]	1	0.1436	10	49.3	0.2334
MS-SVR(E) [9]	0.7–2.8	0.1436	5	8	0.2322
MS-SVR(Q) [9]	0.7–2.8	—	5	7.5	0.2329
MS-SVR(H) [9]	0.7–2.8	0.1436	1	9	0.2329

TABLE V
RESULTS FOR "HOUSING" DATASET

	σ	ε	C	#SVs	MSE
SVR	5	0.1	30	268	0.0829 (0.0262)
HSVR red.	10	0.025	5	737	0.0816 (0.0208)
SVR [32]	—	—	—	205	0.137 (0.122)
MKSVR [32]	—	—	—	159	0.099 (0.022)

V. DISCUSSION

The approach presented here is based on two key elements: a multi-scale incremental approximation and a reduction of the number of the data points passed to the optimization engine. The latter allows reducing also the number of SVs.

As clearly shown in Fig. 4a and Table I the use of a single kernel function comes short in providing a regression of good quality when the structure of the data changes in the input

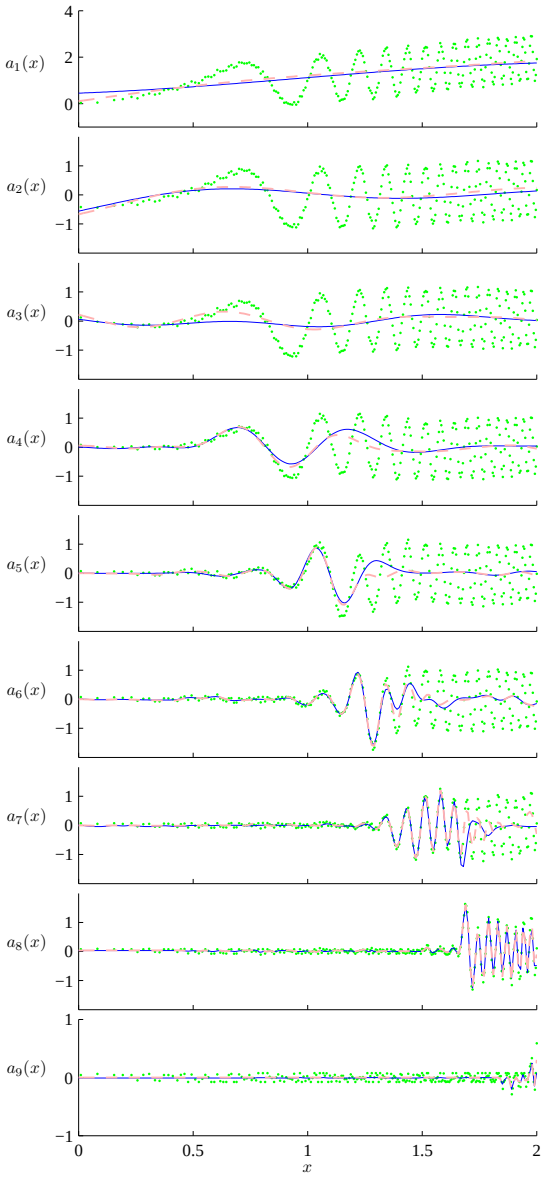


Fig. 6. Reconstruction operated by the different layers of the HSVR model when all the data points (dashed line) and when only the points in S_l' (continuous line) are considered . The residual of each layer (i.e., the training points for that layer) are reported as dots. The data reduction has the effect of smoothing the regression in the first layers, but smoothing tends to disappear in the last layers, where the two curves are almost coincident.

domain. In particular, spurious oscillations are produced in the region of low frequency content when a high frequency kernel is adopted (Fig. 1c). In the multi-scale approach presented here, instead, this low frequency region is reconstructed by the first layers, while the higher layers improve the reconstruction in the high frequency region (Fig. 6). This is made clear in Fig. 11 where the relative contribution to the final reconstruction of each layer, η_l, along the input domain is shown:

$$\eta_l(x) = \frac{Y_l(x)}{\sum_{j=1}^{L} Y_j(x)} \tag{24}$$

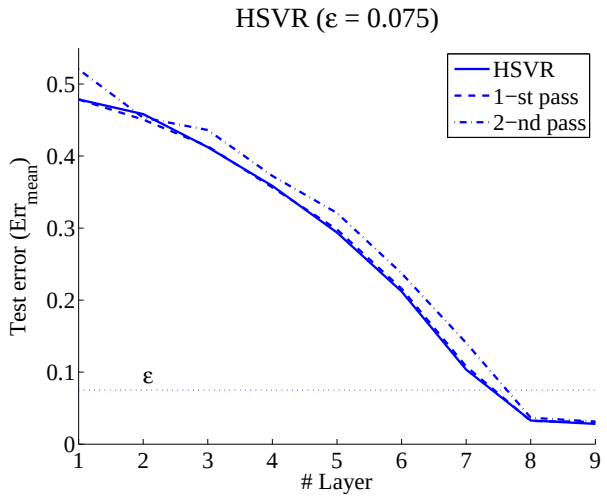
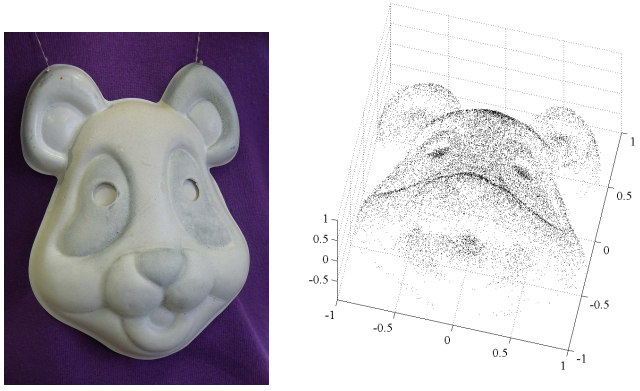


Fig. 7. Evolution of the test error (Err_{mean}) of the HSVR models as new layers are inserted. The continuous line represents the error of the HSVR model with no data reduction. The dashed line represents the error of the model when data reduction was applied in all the previous layers, but not in the current one (1-st optimization pass). The dot dashed line represents the error of the HSVR model when data reduction is applied to all the layers (2-nd optimization pass).



(a) (b)

Fig. 8. Panel (a) shows the artifact (a panda mask) that has been digitized obtaining the data for the experiment (b).

where

$$Y_l(x) = \sum_{k=1}^{M_k} |\beta_{l,k}| \exp\left(-\frac{||x - x_{l,k}||^2}{\sigma_l^2}\right) \tag{25}$$

The other key element of HSVR is pruning that aims here at identifying the data points that can be meaningful for the reconstruction at each level. To achieve this a careful analysis of the points is carried out subdividing them into three sets (15): those inside, outside and on the border of the ε tube. The inner points constitute a "reference": after optimization they have to lie inside the ε tube and, therefore, they constitute a sort of constraint for the function. The outer points and those on the border of the ε tube (SVs) play the role of basis for the computation of the surface. Data reduction is based to the closeness (or relevance) of each point to the surface evaluated as point to surface distance (15). In this respect this pruning procedure can be regarded as an instance of active learning [33] as the points most relevant to the

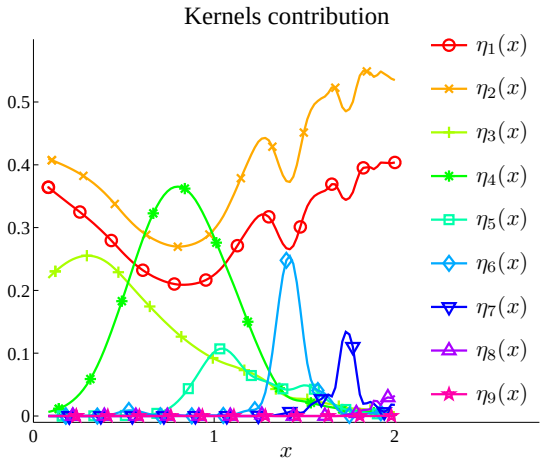


Fig. 11. Relative contribution of the kernels in the different layers (computed as in (24)) to the reconstruction of $h(\cdot)$, described in (3) (cfr. Figs 4c and 6). Notice that the SVs have the same position over the different layers. The first two layers have kernels with a large weight over all the input domain and contribute to the bias, while in the higher layers, the weight of the Gaussian basis increases with the x axis inversely proportional to the scale. Therefore an automatic scale selection is obtained.

reconstruction are identified. Other criteria to determine the relevance of each point could be based on a ranking measure as for instance in [34]. The evaluation of points with respect to a reference function is different from classical clustering paradigms in which data points are reduced according to a similarity measure between the points themselves [35]. While, in clustering, sub-sets of points are substituted by a single point that usually lie in a position different from that of any point in the sub-set, here the selected points lie in the same position of the data points. This allows avoiding smoothing out valleys and ridges of the function. Pruning allows reducing the number of SV by one order of magnitude as shown in Tables II and III, making regression inside the framework of SVR feasible.

The reduction of the SVs that effectively contribute to the output of each layer does not degrade the reconstruction (cf. Fig. 7) and it allows producing a very similar output with much less SVs (243 vs. 1,544 for the synthetic dataset and 11,351 vs. 100,448 for the panda mask dataset). The same is true for the output of the intermediate layers as shown in Fig. 7 and Table II. From this we can draw the conclusion that most of the SVs employed especially in the first layers of a multi-scale reconstruction (cf. Table II) are wasted in the vain attempt of approximating details with a kernel that operates at a too large scale. These SVs can be pruned without degrading the output quality of the model.

Fig. 7 shows also the part of the residual error that can be attributed to data points reduction (this is the height difference between the error "2-nd pass" and the error "HSVR"). This error is recovered in each next layer as can be seen by comparing the error produced when no data reduction is applied to the current layer but it was applied to all the previous layers ("1-st pass"), with that produced when no data reduction is applied ("HSVR"): the two errors are almost coincident. Data reduction is particularly efficient when data sampling is relatively dense; if a few data points are present and these are sparse, pruning might turn out not as efficient.

As can be seen in Table III, the accuracy of HSVR and the number of SVs used are similar to those of standard SVR. The analysis of the configuration time is more critical. When the configuration parameters C, σ, and ε are set properly, the configuration time for SVR is about one third of HSVR: 382 s versus 1,182 s. However, in practice, most of the time required to determine an SVR regression is devoted to the search of good values for C, σ, and ε as the quality of the regression depends critically on these parameters [3]. This makes standard SVR regression extremely time consuming as a three dimensional parameters space has to be searched by the optimization procedure. This search space is reduced to a bidimensional space in HSVR as σ has not to be optimized. This allows a reduction in configuration time that largely overcompensates the need of computing a SVR two times for each level. Moreover, as shown in Fig. 10a–c, the HSVR approach is much less sensitive to the initial value of these two parameters. This is due to the robustness of the hierarchical structure in which the data not explained in one layer can be recovered by the next layers.

Moreover, the parameter C can be set proportional to the standard deviation of the data points value (14), through the factor J. This has been experimentally computed analyzing different datasets: values in the range $[0.1,\ 20]$ were considered and values of J around 1 have always been sufficient to achieve very good results as shown in Fig. 10a–c where best, average and worst case error with different combinations of J and ε are almost similar. If we accept this value for J, we come up with a configuration time of HSVR that is about 1/20th than SVR.

As kernels with a different σ are chosen in the different layers, the criticality in choosing a single value of σ, adequate for the data, disappears: starting from a large σ and halving it in each layer, guarantees that a value of σ adequate to the data is found as shown in Fig. 6. It should be remarked that, in case of equally spaced SVs, (7) represents an approximation in terms of Riesz basis. Moreover, the angle between the spaces spanned in two consecutive layers is quite small, which allows slowly incorporating the details on one side, but also stopping adding new layers when regression starts becoming noisy [12].

Therefore, the HSVR regression cannot be considered the direct result of the minimization of the cost function (5) as in [9][32]. However, we notice that (5) constitutes a regularization function that produces a smooth regression penalizing both the norm of the error and the norm of the kernel coefficients. We do consider the two norms at each layer with the aim of producing a smooth regression incrementally. We favor smoothness, also decreasing the maximum amplitude of the coefficients in each layer (cf. (14)). As shown in the experimental results, this allows finding a good regression with a limited computational time.

The number of layers may be critical to avoid overfitting, that is a problem shared by all incremental approaches. Different strategies can be used to stop the configuration procedure. If no a-priori information is available, the use of validation error guarantees that a good generalization is obtained: the

introduction of a new layer can be stopped when the validation error does not decrease anymore with the new layer. Alternatively, the growth can be stopped when the residual drops below a given threshold, which can be associated to measurement noise [13]. Other criteria, depending on the applicative context, can be adopted. We explicitly remark that, at the end of the configuration phase, the regression at all the considered scales is available (cf. Fig. 6). This can be of great value, for instance, for the construction of models from points clouds in the graphical domain: a smoother surface, although with a slightly larger test error, Fig. 9d–f, can be preferred to a surface with a smaller test error, featuring less smoothness Fig. 9a–c. This would be not possible with classical SVR approach that produces only one regression.

Although, in principle a linear combination of kernels featuring a single large scale can be used to realize an accurate estimate of a function with high frequency content [36], the computational effort required in this case would be too high. In fact, the use of large scale kernels involves large coefficients (high value of C), which may cause numerical instability of the solution. Besides this, the configuration time tends to increase with the value of C and makes using high values of C almost unfeasible for real cases.

In principle, the HSVR configuration scheme can work with kernels other than Gaussians. However, the Gaussian kernel has two main properties: the scale parameter, σ, allows shaping the kernel such that the SVs are sensitive to different frequency ranges, and the non orthogonality, which allows recovering in the next layers the reconstruction error possibly left by the previous layers (13). Although other kernels that enjoy these properties could be used, adequate optimization engines, different from LibCVM Toolkit, should be developed for these kernels, which goes beyond the aim of this work.

VI. Conclusion

A novel approach to multi-scale kernel regression, called HSVR, has been here presented. It is based on an incremental model, adding layers which employ kernels at decreasing scales until an adequate output is obtained. The number of SVs used is similar to that employed by standard SVR, while the configuration time is much lower as a full search in the parameters space is not required. The approach is based also on a data reduction step: only the data that can be considered meaningful for the output of each layer are passed to the optimization procedure with a substantial reduction in the number of the SVs selected that makes this approach feasible also for very large data sets.

References

[1] V. N. Vapnik, *Statistical Learning Theory*. Wiley, 1998.

[2] N. Cristianini and J. Shawe-Taylor, *An Introduction to Support Vector Machines and other kernel-based learning methods*. Cambridge University Press, 2000.

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

[4] G. Lanckriet, N. Cristianini, P. Bartlett, L. E. Ghaoui, and M. I. Jordan, "Learning the kernel matrix with semi-definite programming," *J. of Machine Learning Research*, vol. 5, pp. 27–72, 2004.

[5] O. Chapelle, V. Vapnik, O. Bousquet, and S. Mukherjee, "Choosing multiple parameters for support vector machines," *Machine Learning*, vol. 46, no. 1–3, pp. 131–159, 2002.

[6] Z. Wang, S. Chen, and T. Sun, "MultiK-MHKS: A novel multiple kernel learning algorithm," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 30, no. 2, pp. 348–353, 2008.

[7] M. Gönen and E. Alpaydin, "Localized multiple kernel learning," in *Proc. of the 25th Int. Conf. on Machine learning*, ser. ICML '08. New York, NY, USA: ACM, 2008, pp. 352–359.

[8] M. Gönen and E. Alpaydın, "Localized multiple kernel regression," in *Proc. of the 20th IAPR Int. Conf. on Pattern Recognition*, 2010.

[9] D. Zheng, J. Wang, and Y. Zhao, "Non-flat function estimation with a multi-scale support vector regression," *Neurocomputing*, vol. 70, no. 1-3, pp. 420 – 429, 2006.

[10] H. Peng and J. Wang, "Nonlinear system identification based on multi-resolution support vector regression," in *Int. Conf. on Neural Networks and Brain, ICNN&B*, vol. 1, 2005, pp. 240–243.

[11] J. E. Moody, "Fast learning in multi-resolution hierarchies," in *Neural Information Processing Systems*, 1988, pp. 29–39.

[12] S. Ferrari, M. Maggioni, and N. A. Borghese, "Multi-scale approximation with hierarchical radial basis functions networks," *IEEE Trans. Neural Netw.*, vol. 15, no. 1, pp. 178–188, Jan. 2004.

[13] S. Ferrari, F. Bellocchio, V. Piuri, and N. A. Borghese, "A hierarchical RBF online learning algorithm for real-time 3-D scanner," *IEEE Trans. Neural Netw.*, vol. 21, no. 2, pp. 275–285, Feb. 2010.

[14] C. K. Reddy and J. H. Park, "Multi-resolution boosting for classification and regression problems." in *PAKDD'09*, 2009, pp. 196–207.

[15] F. Steinke, B. Schölkopf, and V. Blanz, "Support vector machines for 3D shape processing," *Comp. Graph. Forum*, vol. 24, pp. 285–294, 2005.

[16] R. E. Schapire, "A brief introduction to boosting," in *Int. Joint Conf. on Artificial Intelligence*, 1999, pp. 1401–1406.

[17] Y. Freund and R. E. Schapire, "A decision-theoretic generalization of on-line learning and an application to boosting," *J. Comput. Syst. Sci.*, vol. 55, pp. 119–139, August 1997.

[18] N. Duffy and D. Helmbold, "Boosting methods for regression," *Mach. Learn.*, vol. 47, pp. 153–200, May 2002.

[19] X. Liang, "An effective method of pruning support vector machine classifiers," *IEEE Trans. Neural Netw.*, vol. 21, no. 1, pp. 26–38, 2010.

[20] G. M. Fung, O. L. Mangasarian, and A. J. Smola, "Minimal kernel classifiers," *J. Mach. Learn. Res.*, vol. 3, pp. 2303–321, 2002.

[21] S. S. Keerthi, O. Chapelle, and D. D. Coste, "Building support vector machines with reduced classifier complexity," *J. Mach. Learn. Res.*, vol. 7, pp. 1493–1515, 2006.

[22] J. Guo, N. Takahashi, and T. Nishi, "An efficient method for simplifying decision functions of support vector machines," *IEICE Trans. Fund.*, no. 10, pp. 2795–2802, 2006.

[23] X. Zeng and X. Chen, "Smo-based pruning methods for sparse least squares support vector machines," *IEEE Trans. Neural Netw.*, vol. 16, no. 6, pp. 1541–1546, 2005.

[24] D. Nguyen and T. Ho, "An efficient method for simplifying support vector machines," in *Proc. of the 22nd Int. Conf. on Machine learning*, 2005, pp. 617–624.

[25] I. W. Tsang, J. T. Kwok, and P.-M. Cheung, "Core vector machines: Fast SVM training on very large data sets," *J. of Machine Learning Research*, vol. 6, pp. 363–392, 2005.

[26] T. Joachims, "Making large-scale SVM learning practical," in *Advances in Kernel Methods - Support Vector Learning*, B. Schölkopf, C. Burges, and A. Smola, Eds. MIT Press, 1999, ch. 11, pp. 169–184.

[27] D. Wang, X.-B. Wu, and D.-M. Lin, "Two heuristic strategies for searching optimal hyper parameters of C-SVM," in *Proc. of the Eighth Int. Conf. on Machine Learning and Cybernetics*, 2009, pp. 3690–3695.

[28] Y. Tang, W. Guo, and J. Gao, "Efficient model selection for support vector machine with gaussian kernel function," in *IEEE Symp. on Comp. Intell. and Data Mining, 2009. CIDM '09.*, 2009, pp. 40–45.

[29] A. J. Smola, N. Murata, B. Schölkopf, and K.-R. Müller, "Asymptotically optimal choice of ϵ-loss for support vector machines," in *Proc. of the 8th Int. Conf. on Artificial Neural Networks, Perspectives in Neural Computing*. Springer Verlag, 1998, pp. 105–110.

[30] [Online]. Available: http://theoval.cmp.uea.ac.uk/\~gcc/matlab/

[31] [Online]. Available: http://archive.ics.uci.edu/ml/datasets/Housing

[32] S. Qiu and T. Lane, "Multiple kernel learning for support vector regression," Computer Science Department, The University of New Mexico, Albuquerque, NM, USA, Tech. Rep., 2005.

[33] M. Hasenjäger and H. Ritter, "New learning paradigms in soft computing," in *Active learning in neural networks*. Physica-Verlag GmbH, 2002, pp. 137–169.

[34] T. M. Martinetz, S. G. Berkovich, and K. J. Schulten, ""Neural-gas" network for vector quantization and its application to time-series prediction," *IEEE Trans. Neural Netw.*, vol. 4, no. 4, pp. 558–568, 1993.
[35] D. Wunsch and R. Xu, *Clustering*. IEEE Computer Soc. Press, 2008.
[36] F. Girosi, M. Jones, and T. Poggio, "Regularization theory and neural networks architectures," *Neural Computation*, vol. 7, pp. 219–269, 1995.