PADOVA UNIVERSITY PRESS
Dolomites Research Notes on Approximation
Volume 12 · 2019 · Pages 17–27

Learning with subsampled kernel-based methods: Environmental and financial applications

M. Aminian Shahrokhabadi[a] · A. Neisy[b] · E. Perracchione[c] · M. Polato[c]

Communicated by S. De Marchi

Abstract

Kernel machines are widely used tools for extracting features from given data. In this context, there are many available techniques that are able to predict, within a certain tolerance, the evolution of time series, i.e. the dynamics of the considered quantities. However, the main drawback is that measurements are usually affected by noise/errors and might have gaps. For instance, gaps might be due to several problems of the physical instruments that produce measurements. In these cases the learning and prediction steps for capturing the *trend* of time series become very hard. To alleviate these difficulties, we construct a primary kernel-based approximant, which is indeed a model, with the double aim: to fill the gaps and to filter noisy data. The so-constructed smoothed samples are used as training sets for a kernel-based online model. We claim that the subsampled training phase makes the predicted results more stable. Applications to real data for environmental and financial observations support the validity of our results.

1 Introduction

Kernel-based methods are well-established numerical tools which find their natural applications in a wide variety of fields. In particular, over the last years the topic of numerical approximation of multivariate data via Radial Basis Functions (RBFs) [23], also recently extended to the rational case [7], has gained popularity in various disciplines, such as numerical solution of PDEs, image registration [5], magnetic particle imaging [8], neural networks, Support Vector Machines (SVMs) or Support Vector Regression (SVR) [21, 22], finance [1, 19, 25] and population dynamics [11].

In this paper we focus on kernel machines for SVR. Their theoretical foundation can be traced back to the work of Mercer [14] who in 1909 studied the concept of *positive definite kernels*. Within this context, many interesting applications have been later investigated by RBF-networks and kernel machines. Both methods call for a *regularization approach* to filter data. This is due to the fact that in applications data are always affected by errors that need to be smoothed out to prevent wild oscillatory behavior in the approximation. Furthermore, data might be *scattered*, i.e. present gaps. These issues can lead to inaccurate approximations.

To avoid these drawbacks we adopt a kind of filtering constructed via a greedy approach as in [24]. In this way we are able to pre-process data by keeping only a reduced number of data-dependent bases and consequently construct a model useful to train the kernel machine. In this sense, we refer to the scheme presented in [24] as Subsampled Bases (SB). By acting in this way, the SVR task turns out to be less affected by noise and hence easier to learn, making the prediction more reliable.

The so-constructed algorithm is then tested on 1D real world data, affected by errors and scattered. Comparisons with the classical SVR and polynomial ridge regression are carried out. The results are promising and show that our proposed regularization is needed to truly capture the trend of data, without loosing accuracy when oscillations occur. We also point out that the experiments, carried out with PYTHON, are freely available for the scientific community on the Github repository at

https://github.com/makgyver/vlabtestrepo/.

The guidelines of the paper are as follows. In Section 2, we briefly review the basic aspects of RBF theory, focusing on the so-called *reproducing kernel property*. Section 3 presents our scheme for RBF regression. Such algorithm is tested via extensive numerical experiments in Section 4. The last section deals with conclusions and future work.

2 Reproducing kernel property

In this section we review the main theoretical aspects of kernel-based methods. For further details refer to the books [4, 9, 10, 23].

Let us suppose that we have $X_N = \{\boldsymbol{x}_i,\ i = 1,\dots,N\} \subset \Omega$, a set of distinct data points (or data sites or nodes), arbitrarily distributed on a domain $\Omega \subseteq \mathbb{R}^d$, and an associated set $\mathcal{F}_N = \{f_i = f(\boldsymbol{x}_i),\ i = 1,\dots,N\}$ of data values (or measurements or

[a] Department of Mathematics, Shahid Beheshti University, Tehran, Iran
[b] Department of Mathematics, Faculty of Mathematical Science, Allameh Tabataba'i University, Teheran, Iran
[c] Department of Mathematics “Tullio Levi-Civita”, University of Padova

function values), obtained by sampling some (unknown) function $f : \Omega \longrightarrow \mathbb{R}$ at the nodes $\boldsymbol{x}_i$. A way to model such data consists in finding a function $R : \Omega \longrightarrow \mathbb{R}$ so that:

$$R(\boldsymbol{x}_i) = f_i, \quad i = 1, \dots, N. \tag{1}$$

Since such a model exactly matches the data values at the nodes, R is known as interpolating function. Focusing on conditionally positive definite radial kernels K of order l on $\mathbb{R}^d$, R assumes the form [9]

$$R(\boldsymbol{x}) = \sum_{k=1}^{N} c_k K(\boldsymbol{x}, \boldsymbol{x}_k) + \sum_{m=1}^{L} \gamma_m p_m(\boldsymbol{x}), \quad \boldsymbol{x} \in \Omega, \tag{2}$$

where $p_1, \dots, p_L$, are a basis for the L-dimensional linear space Π_{l-1}^d of polynomials of total degree less than or equal to $l-1$ in d variables, with

$$L = \binom{l-1+d}{l-1}.$$

More specifically, we take RBFs and thus we suppose that there exist a function $\phi : [0, \infty) \to \mathbb{R}$ and (possibly) a shape parameter $\varepsilon > 0$ such that for all $\boldsymbol{x}, \boldsymbol{y} \in \Omega$,

$$K(\boldsymbol{x}, \boldsymbol{y}) = K_\varepsilon(\boldsymbol{x}, \boldsymbol{y}) = \phi_\varepsilon(||\boldsymbol{x} - \boldsymbol{y}||_2) := \phi(r).$$

Here, $r = ||\boldsymbol{x} - \boldsymbol{y}||_2$. For several examples of RBFs and their regularities we refer the reader to Table 1. Note that, some of them are shape parameter-free and this might be an advantage in the sense that for the user there is no need to select *good* values for ε.

K	$\phi(r)$	l
Gaussian C^∞ (GA)	$e^{-\varepsilon^2 r^2}$	0
Inverse Multiquadric C^∞ (IM)	$(1 + r^2/\varepsilon^2)^{-1/2}$	0
Generalized Multiquadric C^∞ (GM)	$(1 + r^2/\varepsilon^2)^{3/2}$	2
Matérn C^2 (M2)	$e^{-\varepsilon r}(1 + \varepsilon r),$	0
Matérn C^6 (M6)	$e^{-\varepsilon r}(15 + 15\varepsilon r + 6(\varepsilon r)^2 + (\varepsilon r)^3)$	0
Linear C^0 (L0)	r	1
Cubic C^2 (C2)	r^3	2

Table 1: Examples of conditionally positive definite RBFs.

To satisfy (1), one needs to solve the following linear system

$$\begin{pmatrix} A & P \\ P^\intercal & O \end{pmatrix} \begin{pmatrix} \boldsymbol{c} \\ \boldsymbol{\gamma} \end{pmatrix} = \begin{pmatrix} \boldsymbol{f} \\ \boldsymbol{0} \end{pmatrix}, \tag{3}$$

where

$$A_{ik} = K(\boldsymbol{x}_i, \boldsymbol{x}_k), \quad i, k = 1, \dots, N,$$

$$P_{im} = p_m(\boldsymbol{x}_i), \quad i = 1, \dots, N, \quad m = 1, \dots, L.$$

Moreover, $\boldsymbol{c} = (c_1, \dots, c_N)^\intercal$, $\boldsymbol{\gamma} = (\gamma_1, \dots, \gamma_L)^\intercal$, $\boldsymbol{f} = (f_1, \dots, f_N)^\intercal$, $\boldsymbol{0}$ is a zero vector of length L and O is a $L \times L$ zero matrix.

For understanding when such problem admits a unique solution, we introduce the definition of *unisolvent set*.

Definition 2.1. A set of points $X_N = \{\boldsymbol{x}_i, i = 1, \dots, N\} \subseteq \Omega$ is called q-unisolvent if the only polynomial of total degree at most q interpolating zero data on X_N is the zero polynomial.

We remark that in case K in (2) is conditionally positive definite of order l on $\mathbb{R}^d$ and the set $X_N = \{\boldsymbol{x}_i, i = 1, \dots, N\} \subseteq \Omega$ of data points forms a $(l-1)$-*unisolvent set*, then the system of linear equations (3) admits a unique solution; refer e.g. to Theorem 7.2 [9] p. 64.

For simplicity, we now restrict to the case $l = 0$. It leads to conditionally positive definite functions of order zero, i.e. strictly positive definite functions. As a consequence, we interpolate without the polynomial term in (2) and the system reduces to $A\boldsymbol{c} = \boldsymbol{f}$.

To each positive definite and symmetric kernel K we associate a real Hilbert space, the so-called *native space* $\mathcal{N}_K(\Omega)$. To this aim, we need some background reviewed in what follows; refer e.g. to Definition 13.1 [9] p. 103.

Definition 2.1. Let $\mathcal{H}$ be a real Hilbert space of functions $f : \Omega \longrightarrow \mathbb{R}$, with inner product $(\cdot, \cdot)_\mathcal{H}$. A function $K : \Omega \times \Omega \longrightarrow \mathbb{R}$ is called a reproducing kernel for $\mathcal{H}$ if:

i. $K(\cdot, \boldsymbol{x}) \in \mathcal{H}$, for all $\boldsymbol{x} \in \Omega$,

ii. $f(\boldsymbol{x}) = (f, K(\cdot, \boldsymbol{x}))_\mathcal{H}$, for all $f \in \mathcal{H}$ and for all $\boldsymbol{x} \in \Omega$ (reproducing property).

Then, we define the following space

$$\mathcal{H}_K(\Omega) = \text{span}\{K(\cdot,\boldsymbol{x}), \boldsymbol{x} \in \Omega\},$$

equipped with the bilinear form $\langle \cdot,\cdot \rangle_{\mathcal{H}_K(\Omega)}$ defined as

$$\left\langle \sum_{i=1}^{m} c_i K(\cdot,\boldsymbol{x}_i), \sum_{k=1}^{n} d_k K(\cdot,\boldsymbol{x}_k) \right\rangle_{\mathcal{H}_K(\Omega)} = \sum_{i=1}^{m}\sum_{k=1}^{n} c_i d_k K(\boldsymbol{x}_i,\boldsymbol{x}_k).$$

The space $\mathcal{H}_K(\Omega)$ is a pre-Hilbert space with reproducing kernel K, i.e. $\langle K(\boldsymbol{x},\cdot), f\rangle_{\mathcal{H}_K(\Omega)} = f(\boldsymbol{x})$ for all $f \in \mathcal{H}$ and for all $\boldsymbol{x} \in \Omega$. Then, we define the native space $\mathcal{N}_K(\Omega)$ of K as the completion of $\mathcal{H}_K(\Omega)$ with respect to the norm $||\cdot||_{\mathcal{H}_K(\Omega)}$, that is $||f||_{\mathcal{H}_K(\Omega)} = ||f||_{\mathcal{N}_K(\Omega)}$, for all $f \in \mathcal{H}_K(\Omega)$; cf. [9, 23].

We also remark that the choice of the RBF affects the fitting process, indeed the error decreases according to the fill-distance h_X, where

$$h_X = \sup_{\boldsymbol{x}\in\Omega}\left(\min_{\boldsymbol{x}_k \in X_N} \|\boldsymbol{x} - \boldsymbol{x}_k\|_2\right),$$

and the interpolation of functions $f \in \mathcal{N}_K(\Omega)$ with a C^{2k} smooth kernel has error bound of order k; refer e.g. to Theorem 14.5 [9] p. 121. Furthermore, we remark that the selection of the shape parameter is a tedious computational issue. For a general overview about techniques that allow to choose *safe* values for ε we refer the reader to [10].

We now need some technical considerations and we introduce the operator $T : L_2(\Omega) \longrightarrow L_2(\Omega)$,

$$T[f](\boldsymbol{x}) = \int_\Omega K(\boldsymbol{x},\boldsymbol{y}) f(\boldsymbol{y}) d\boldsymbol{y}.$$

By virtue of the Mercer's Theorem (see e.g. Theorem 2.2. [10] p. 24 or [14]), we know that the operator T has a countable set of eigenfunctions $\{\varphi_k\}_{k\geq 0}$ and eigenvalues $\{\lambda_k\}_{k\geq 0}$. The eigenfunctions are orthonormal in $L_2(\Omega)$ and the kernel can be expressed in terms of the *eigencouples* as

$$K(\boldsymbol{x},\boldsymbol{y}) = \sum_{k\geq 0} \lambda_k \varphi_k(\boldsymbol{x})\varphi_k(\boldsymbol{y}), \quad \boldsymbol{x},\boldsymbol{y} \in \Omega,$$

where the series converges uniformly and absolutely. It is worth to note that we can interpret the Mercer series representation in terms of an inner product (in the sequence space l_2). Indeed,

$$K(\boldsymbol{x},\boldsymbol{y}) = \langle \Phi(\boldsymbol{x}), \Phi(\boldsymbol{y})\rangle_{l_2}, \quad \boldsymbol{x},\boldsymbol{y} \in \Omega,$$

where the series converges uniformly and absolutely and

$$\Phi(\cdot) = \left(\sqrt{\lambda_1}\varphi_1(\cdot), \sqrt{\lambda_2}\varphi_2(\cdot), \ldots\right).$$

Note that this separates K into a *feature* that depends only on $\boldsymbol{x}$ and another one that only depends on $\boldsymbol{y}$. Furthermore, in machine learning literature a mapping Φ from Ω to $\mathcal{H}$, where $\mathcal{H}$ is called the so-called feature space, i.e. a Hilbert space, is often referred to as the feature map. The feature space can be a reproducing kernel Hilbert space, so that we obtain the *canonical features* $\Phi(\boldsymbol{x}) = K(\cdot,\boldsymbol{x})$. An example of a mapping Φ from Ω to the feature space $\mathcal{H}$ is shown in Figure 1.

As a final remark, we point out that, in case $\mathcal{H}$ is a reproducing kernel Hilbert space, the Mercer expansion gives a feature map. However, depending on the feature space, we can associate to any kernel many different feature maps; refer e.g. to [2].

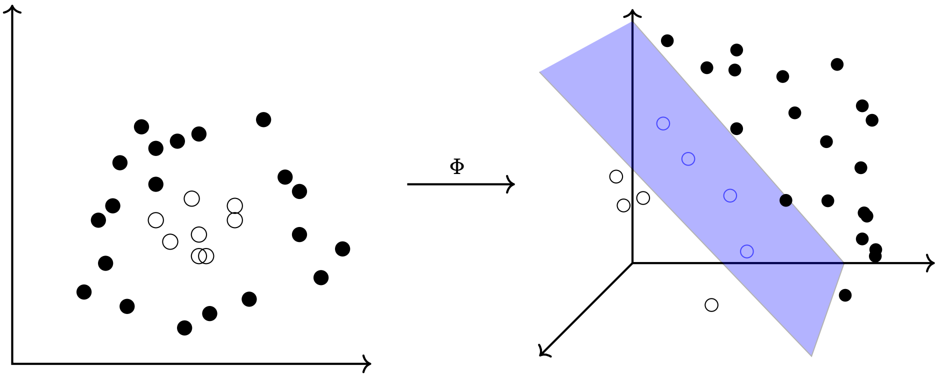


Figure 1: Illustrative figure for binary classification. Examples in the input space, i.e., $\mathbb{R}^2$, are not linearly separable (left), but after the projection through Φ (right), onto an higher dimensional space, i.e., the feature space $\mathbb{R}^3$, examples become separable by, for instance, the blue hyperplane. This step in machine learning literature is often referred to as the *kernel trick*.

In the next section we better point out how kernel machines work; for further details the reader can refer e.g. to [21]. Moreover, since the trend of data is usually hard to predict when noise is introduced, we develop a reduced RBF model which will be the input for the kernel machine, i.e. the training set.

3 Learning with subsampled kernel methods

The main scope of machine learning, briefly reviewed in Subsection 3.1 [21], is to discover patterns from data and which relation links inputs to the outputs. Thinking of time series, this means to give predictions about the future dynamics of the considered quantities. However, learning techniques might suffer when data are affected by noise and present gaps, for instance due to some technical problem of the measuring devices. Noisy data might lead to serious computational issues in context of SVM (and SVR). For example in the classification context, noisy data points can transform a linearly separable problem into a non-linearly separable one which can lead to a poor selection of the hyper-parameters, and this can have impacts on the generalization capability of the kernel machine.

3.1 Kernel machines

Kernel-based methods are one of the most used machine learning approaches. The basic idea behind this kind of schemes is related to the so-called kernel trick which allows to implicitly compute vector similarities (defined in terms of dot-product) in potentially infinite dimensional spaces. In the following we give a brief overview about kernel algorithms.

In particular, SVM is the most famous and successful kernel method. It is usually used for classification tasks, but it can be easily adapted to regression [6, 22]. For the former, the problem consists in predicting data values which are labels. For simplicity we take $y_i \in \{-1, +1\}$. The predictor that allows us to assign an appropriate label has the form $\mathrm{sign}(h(\boldsymbol{x}))$, where h denotes a hyperplane separating the given measurements. A typical loss function for SVM classification is given by the *h*inge loss:

$$L(y, h(\boldsymbol{x})) = \max(1 - y h(\boldsymbol{x}), 0).$$

We now focus on linear and non-linear SVR. Let us assume

$$R(\boldsymbol{x}) = \boldsymbol{x}^\intercal \boldsymbol{w} + b,$$

where $\boldsymbol{w}$ is the unit normal vector to the hyperplane R and b is called bias. Then, let us consider the ϵ-insensitive loss function

$$L(f, R(\boldsymbol{x})) = \max\left(|f - R(\boldsymbol{x})| - \epsilon, 0\right).$$

To determine $\boldsymbol{w}$ and b, we introduce the following constrained optimization problem with regularization parameter C and slack variables ξ_i, $i = 1, \dots, N$, that allow us to deal with the case where the given measurements are hard to fit with R:

$$\min_{\boldsymbol{w}, b, \xi, \xi^*} \left[\frac{1}{2} \boldsymbol{w}^\intercal \boldsymbol{w} + C \sum_{i=1}^{N} (\xi_i + \xi_i^*) \right],$$

subject to:

$$\begin{aligned} & R(\boldsymbol{x}_i) - f_i \leq \epsilon + \xi_i, \quad i = 1, \dots, N, \\ & f_i - R(\boldsymbol{x}_i) \leq \epsilon + \xi_i^* \quad i = 1, \dots, N, \\ & \xi_i, \xi_i^* \geq 0, \end{aligned}$$

where the objective function aims to minimize the norm of the hypothesis in order to get a smooth function and maintaining the number of mis-classifications as low as possible. $C \geq 0$ represents the so-called *trade-off parameter* and is indeed a smoothing parameter. The *hyper-parameter* $\epsilon \geq 0$ indicates the width of the *tube* in which the samples can fall into without being counted as errors. We refer the reader to Figure 2.

This problem is usually solved in its dual form defined as:

$$\min_{\boldsymbol{\alpha}, \boldsymbol{\alpha}^*} \left[\frac{1}{2} \sum_{i=1}^{N} \sum_{j=1}^{N} (\alpha_i - \alpha_i^*)(\alpha_j^* - \alpha_j) \boldsymbol{x}_i^\intercal \boldsymbol{x}_j + \epsilon \sum_{i=1}^{N} (\alpha_i^* + \alpha_i) - \sum_{i=1}^{N} f_i (\alpha_i^* - \alpha_i) \right],$$

subject to:

$$\begin{aligned} & 0 \leq \alpha_i, \alpha_i^* \leq C, \quad i = 1, \dots, N, \\ & \sum_{i=1}^{N} (\alpha_i^* - \alpha_i) = 0. \end{aligned}$$

Consequently,

$$\boldsymbol{w} = \sum_{i=1}^{N} (\alpha_i^* - \alpha_i) \boldsymbol{x}_i.$$

Finally, from the Karush Kuhn Tucker conditions (see e.g. [15]), we have [21]:

$$\begin{aligned} b &= f_i - \boldsymbol{x}_i^\intercal \boldsymbol{w} - \epsilon, \quad \text{for } \alpha_i \in (0, C), \\ b &= f_i - \boldsymbol{x}_i^\intercal \boldsymbol{w} + \epsilon, \quad \text{for } \alpha_i^* \in (0, C). \end{aligned}$$

Observe that in the computation of b any given index i can be used. However, to make b uniquely defined and for stability purposes it is computed via an average over all candidates.

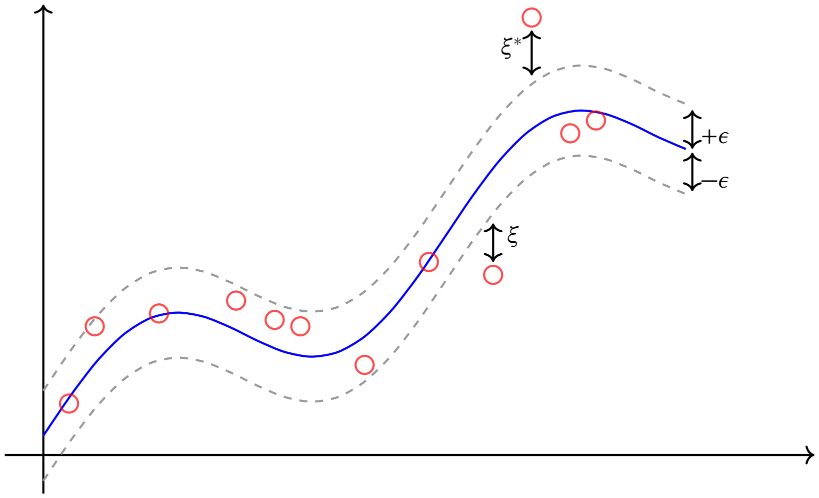


Figure 2: Graphical illustration of the SVR ϵ-tube. Data points inside the tube do not contribute to the loss, while points outside the tube have a cost proportional to the distance from the tube.

Then, for the non-linear regression we use the kernel trick and thus, in view of the considerations made above, we only need to replace the dot product with the kernel evaluation and the measurements $\boldsymbol{x}_i$ with the feature map $\Phi(\boldsymbol{x}_i)$, $i = 1, \dots, N$. In other words the non-linear SVR function is given by

$$\boldsymbol{w} = \sum_{i=1}^{N} (\alpha_i^* - \alpha_i)\Phi(\boldsymbol{x}_i), \quad R(\boldsymbol{x}) = \Phi(\boldsymbol{x})^\intercal \boldsymbol{w} + b = \sum_{i=1}^{N} (\alpha_i^* - \alpha_i) K(\boldsymbol{x}, \boldsymbol{x}_i) + b,$$

and

$$b = f_i - \sum_{j=1}^{N} (\alpha_j^* - \alpha_j) K(\boldsymbol{x}_i, \boldsymbol{x}_j) - \epsilon, \quad \text{for } \alpha_i \in (0, C),$$
$$b = f_i - \sum_{j=1}^{N} (\alpha_j^* - \alpha_j) K(\boldsymbol{x}_i, \boldsymbol{x}_j) + \epsilon, \quad \text{for } \alpha_i^* \in (0, C).$$

Concerning the computation of b, again it is defined via an average over all candidates.

As already pointed out, this approach properly works if data have only little noise. Therefore, instead of using the original and real data in the kernel machine, we propose to filter such points by constructing via greedy methods surrogate models, namely in what follows Subsampled Bases (SB) models. This enables us to extract better the trend of the measurements.

3.2 Subsampled kernel-based training

If the data are affected by errors, i.e. $f_i = f(\boldsymbol{x}_i) + \delta_i$, one no longer wants to interpolate data. Here we assume that one has Gaussian white noise, i.e. $\boldsymbol{\delta} = (\delta_1, \dots, \delta_N)^\intercal \sim \mathcal{N}(0, \sigma^2 I)$, where I is the $N \times N$ identity matrix and σ is the standard deviation. Then, we compute the coefficients as

$$\boldsymbol{c} = (A + \lambda I)^{-1} \boldsymbol{f}, \tag{4}$$

where $\lambda \geq 0 \in \mathbb{R}$ is the so-called Tikhonov parameter: see e.g. [20] for details. Note that, trivially, (4) is the solution of the following unconstrained optimization problem:

$$\min_{\boldsymbol{c} \in \mathbb{R}^d} \left[(\boldsymbol{f} - A\boldsymbol{c})^\intercal (\boldsymbol{f} - A\boldsymbol{c}) + \lambda \boldsymbol{c}^\intercal A \boldsymbol{c} \right]. \tag{5}$$

Necessary and sufficient condition for the characterization of the minimum is to enforce that the gradient of the above functional is equal to zero. Indeed the problem is quadratic and therefore

$$\nabla_{\boldsymbol{c}} \left[(\boldsymbol{f} - A\boldsymbol{c})^\intercal (\boldsymbol{f} - A\boldsymbol{c}) + \lambda \boldsymbol{c}^\intercal A \boldsymbol{c} \right] = 0,$$

which implies

$$-\boldsymbol{f} + A\boldsymbol{c} + \lambda \boldsymbol{c} = 0.$$

The latter equation proves the equivalence between (4) and (5).

Since the data we consider might be affected by errors or noise, our approximant $\tilde{R}$ will be constructed via Tikhonov regularization as

$$\tilde{R}(\boldsymbol{x}) = \sum_{i=1}^{N} c_i K(\boldsymbol{x}, \boldsymbol{x}_i),$$

where the coefficients are computed as in (4).

Furthermore, together with this regularization we also consider greedy approaches to construct surrogate models; refer e.g. to [24].

We drive our attention to the problem of constructing a surrogate model which involves only a small subset of bases, i.e. a smaller number of nodes M. Following [24], given the initial set consisting of N data the aim is the one of finding a suitable subspace (reduced), spanned by M centres. Here, the selection will be carried out by means of a greedy approach. In this sense, we will have training and validation sets. Of course the points that will form the subspace will be selected so that the error estimated via the validation set is below a certain tolerance.

Given $X_0 \neq \emptyset$, the idea can be summarized in the following steps:

1. Compute $\boldsymbol{x}_k = \text{argmax}_{\boldsymbol{x}\in X_N/X_{k-1}} |f(\boldsymbol{x}) - \tilde{R}(\boldsymbol{x})|$, where $\tilde{R}$ is obtained via (4);
2. Define the new set of nodes as $X_k = X_{k-1} \cup \{\boldsymbol{x}_k\}$ if and only if $|f(\boldsymbol{x}_k) - \tilde{R}(\boldsymbol{x}_k)| > \tau$, where τ is a fixed tolerance;
3. A suitable subspace of M bases, with (usually) $M \ll N$ is found.

Note that the algorithm converges [12] to a number of centers M that depends on the tolerance. Thus, such method enables us to construct smoother times series useful to predict data via machine learning tools. Indeed we can evaluate the RBF interpolant at a point $\boldsymbol{x}$ as

$$\tilde{R}(\boldsymbol{x}) = \boldsymbol{\psi}^{\intercal}(\boldsymbol{x})\boldsymbol{c},$$

where

$$\boldsymbol{\psi}^{\intercal}(\boldsymbol{x}) = (K(\boldsymbol{x},\boldsymbol{x}_1),\ldots,K(\boldsymbol{x},\boldsymbol{x}_M)).$$

Once this subsampled and smoothed model for the training set is constructed, the kernel machine enables us to extract trends and give more robust predictions of the considered quantities.

As a final discussion on this section, we need to understand which kind of prediction we are interested in. Indeed, we point out that another drawback of SVR is that it works with fixed length input vectors. Thinking of time series, the model is trained using the very next data point. And thus at a certain time step t_k, $k \in \mathbb{N}$, $k > 0$, we are able to predict only t_{k+1}. And if we want to go further, we need to assimilate the true input at the $(k+1)$-th time or use the approximated one. To address this problem we introduce a sliding window strategy to create the training instances. In what follows we refer to this method as Multi SVR (MSVR) scheme.

3.3 Training phase of MSVR scheme

Given a sequence of real numbers $t_1,\ldots,t_n$ and a window size $k > 0 \in \mathbb{N}$, the sliding window approach creates $n-k+1$ training vectors such that $\boldsymbol{t}_i = (t_i,\ldots,t_{j+k-1})^{\intercal}$ for $j = 1,\ldots,n-k+1$. Finally, we need to associate to each instance a target value (the one to predict). Assuming we want to predict the very next data point in the sequence for an instance $\boldsymbol{t}_i$, its associated target value f_i is t_{j+k}. So, in general, if we want to build a model for predicting the data point at Δt steps in the future, the associated target value to $\boldsymbol{t}_i$ will be $f_i = t_{j+k+\Delta t-1}$. Figure 3 gives a graphical intuition of the sliding window strategy.

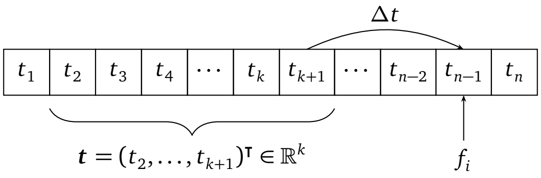


Figure 3: Illustration of the sliding window mechanism.

Note that if one or more missing values fall into a window such training instance will be discarded, and this once more stresses the importance of the preliminary reconstruction of the subsampled model via kernel based methods. Numerical experiments in this direction are provided in the next section.

4 Numerical experiments

Experiments have been carried out in Python using the scientific module scikit-learn [16] on a MacBook Pro late 2012 with 16GB of RAM and CPU Intel® Core i7 @ 2.70GHz.

For testing the accuracy of the prediction on m forecast values, we evaluate the Root Mean Square Error (RMSE), defined as

$$\text{RMSE}(\hat{\boldsymbol{f}}_m, \boldsymbol{f}_m) = \sqrt{\frac{1}{m}\sum_{i=1}^{m}(\hat{f}_i - f_i)^2},$$

and the RMSE percentage (RMSEP),

$$\text{RMSEP}(\hat{\boldsymbol{f}}_m, \boldsymbol{f}_m) = \sqrt{\frac{1}{m}\sum_{i=1}^{m}\left(\frac{\hat{f}_i - f_i}{f_i}\right)^2},$$

where $\boldsymbol{f}_m \in \mathbb{R}^m$ are the ground truth values and $\hat{\boldsymbol{f}}_m \in \mathbb{R}^m$ are the predicted ones.

In this section, for constructing the model used in the training phase via SB method, we consider the Matérn C^2 kernel with $\varepsilon = 7$. The non-optimal selection of this parameter is due to the fact that it depends on the data set, i.e. at each step of the SB method it should be recomputed. By fixing it instead we gain in terms of efficiency. The remaining parameters are fixed as follows: $\tau = 1\text{E}-03$ and $\lambda = 1.00\text{E}-05$. Finally, for both SVR and MSVR prediction, we perform a standard 5-fold cross validation using a grid search strategy [3] for model selection. In particular, we validate $\varepsilon^2 \in [10^{-5}, 10^{-1}]$, $\epsilon \in [10^{-3}, 10^{-1}]$ and $C \in [10^{-1}, 10^{5}]$. We compare our approaches against a polynomial ridge regression [13] scheme (in the remainder we will refer to it with RIDGE). The polynomial regressor is defined as:

$$\min_{\boldsymbol{z}} ||\boldsymbol{f} - \pi_d(\mathbf{X})\boldsymbol{z}||_2^2 + \alpha||\boldsymbol{z}||_2^2,$$

where $\alpha \geq 0$ is the regularization hyper-parameter and $\pi_d(\mathbf{X})$ is a polynomial (of degree d) feature expansion of the training instances $\mathbf{X}$, where $\mathbf{X}$ is the matrix containing all the training instances arranged in the rows. In our case, each instance is composed by a single feature, and thus the corresponding training matrix $\mathbf{X}$ is actually a column vector. The degree of the polynomial has been validated in the range [2,15], while the regularization hyper-parameter α has been validated in the set of values $\{0, 0.01, \ldots, 0.09\} \cup \{0.1, \ldots, 1\}$.

In the following we briefly describe the two real world data sets we consider. For both data sets we perform a time series prediction using SVR and MSVR. These experiments aim to show whether our subsampled data model is able to perform better than using the raw data for training. We will refer to these schemes as “subsampled and classical training”.

As it will be evident, the first data set contains moderately smooth data, while the second one is truly affected by noise. In the latter case, we thus expect a sensible improvement in terms of the accuracy when the training is carried out with SB approximants rather than when using the original data. On the opposite, in the first case, results should be comparable.

A final remark for all this section is that when we use the term SVR, we mean that the prediction is made only on the very next step and then, the approximated value is used to go further in the prediction. MSVR instead, is able to predict on different time steps without the need of assimilating data at each step. Of course the prediction via MSVR is more challenging, but at the same time more realistic. Indeed in applications, assimilating data at each time step is not always possible.

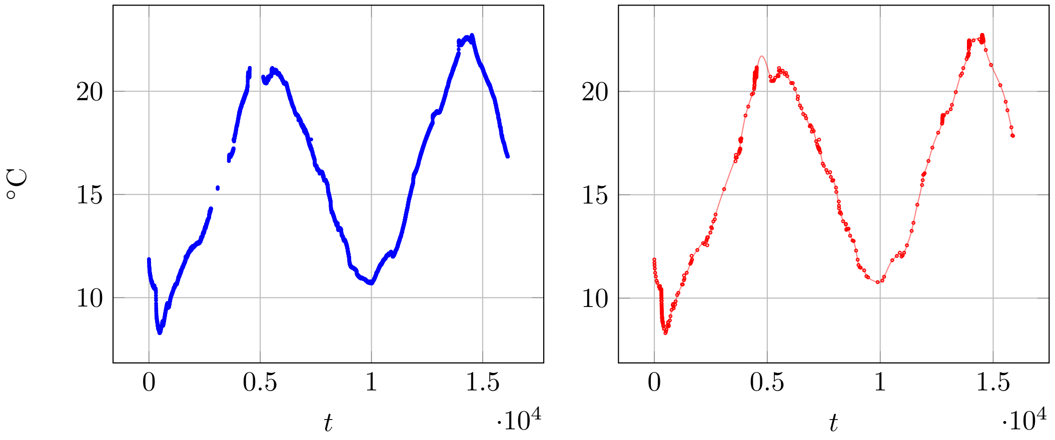
4.1 Environmental data

We consider the data collected in the South-Eastern part of the Veneto Region, available at

http://voss.dmsa.unipd.it/.

The above mentioned data set has been created for an experimental study of the organic soil compaction and prediction of the land subsidence related to climate changes in the South-Eastern area of the Venice Lagoon catchment (VOSS - Venice Organic Soil Subsidence). Such data were collected with the contribution of the University of Padova (UNIPD) from 2001 to 2006. Different physical quantities, measured each hour, are available. For instance, here we consider the temperature in Celsius (°C) sampled one meter below the soil.

In our experiments we use the temperature collected from 14/11/2001 to 14/11/2003. Even if data are not too noisy, they have gaps due to temporary fails of the instruments. For this reason it is suggested to pre-process data with the SB method. The total number of considered data points is 16114 in which 12 are not considered in the training phase since are the points we want to forecast. From the total of 16102 training points only 14637 are valued, the remaining ones are missing. Using the SB method we extract 393 basis. Both data points and the extracted basis together with the so-construed model are depicted in Figure 4.



(a) Temperature raw data points

(b) Extracted points (red dots) and subsampled data model (red curve)

Figure 4: Left: environmental (temperature) data. Right: the extracted points and the SB model.

The prediction via SVR is carried out both via subsampled and classical training phases, to compare our approach with the standard one. We also use the MSVR and for that we take 48 hour (2 days) as time window which should be reasonably large in order to capture the trend of the temperature. The accuracy indicators are reported in Table 2, while Figures 5 and 6 show the graphical results. The reader should note that both methods provide suitable results and robust approximations. In this case, the results with raw and smooth training phase are comparable for SVR. Due to the moderate smoothness of data, SVR performs slightly better than MSVR. For the same reason, RIDGE is able to reach good performances in both settings. Even though the overall best remains SVR, we argue that the difference is not statistically significant.

Remark 1. It is worth to note that, in this particular dataset ridge regression is the method that had the highest benefit from the smooth training data. For MSVR, we have to note that the smoothing scheme provides appreciable benefits. Indeed, gaps for the raw data require to discard several sliding windows. Such drawback is solved by pre-processing data via SB methods.

	Subsampled training		**Classical training**	
	RMSE	**RMSEP**	**RMSE**	**RMSEP**
RIDGE	0.0142	0.0838	0.0163	0.0964
SVR	0.0178	0.1053	0.0136	0.0804
MSVR	0.0204	0.1209	0.0208	0.1233

Table 2: Results for environmental data.

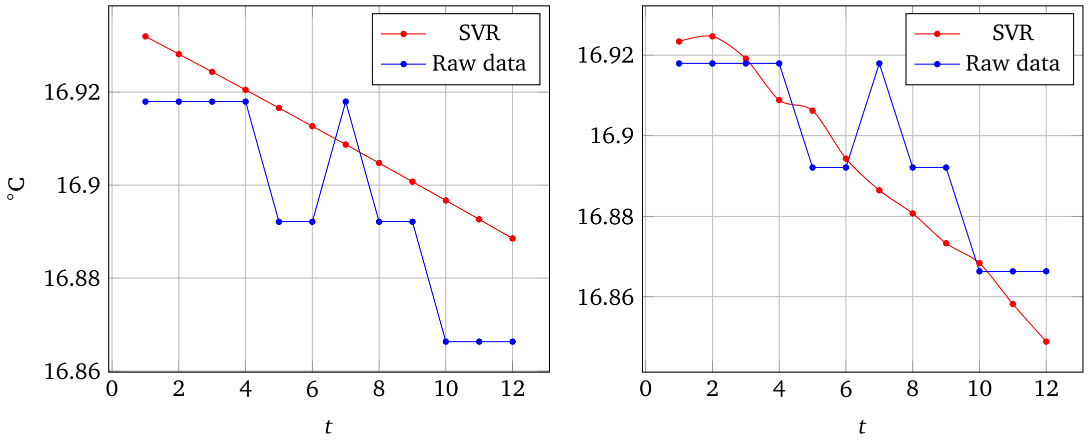


Figure 5: SVR with subsampled (left) and classical (right) training for environmental data.

4.2 Financial data

The data we consider can be found at

http://tsetmc.ir.

This is the website of Tehran Securities Exchange Technology Management Co. This website reports the volume of daily trades of all kind of bonds in Tehran Stock Exchange Market. The data that we use here concerns the daily closing price of a stock named Behran Oil with short name Shabharn. Behran Oil is the stock of Behran Company, see

http://www.behranoil.com/.

that produces lubricants, oil and non-oil products and cooling fluids. This company also trades these products and provides services related to them. This stock belongs to the main board stock exchange market. Like all other stock markets around the world, this has its own laws which one of the most important is the so-called *5% rule*. This rule implies that the prices of current day can vary within the difference of 5% of the closing price of the previous working day. This can be easily modelled. Indeed, a point is added to the data set according to our algorithm if and only if both the tolerance on the error is not satisfied and the 5% rule does not hold on the approximation.

Of course the stock market closes during holidays and weekends. Such missing data cannot be considered as gaps, differently from other closing dates decided by the Securities and Exchange High Council. The reasons of these closing dates are, for instance, changing the category of stock like from a secondary board to the main one, holding the general meeting of shareholders or leaking of important information about profits of stock. After that period, based on the council decision the difference of opening price and the previous closing price might be more than 5%. This procedure of course creates gaps and noise. Therefore, we need to pre-process data with the SB method.

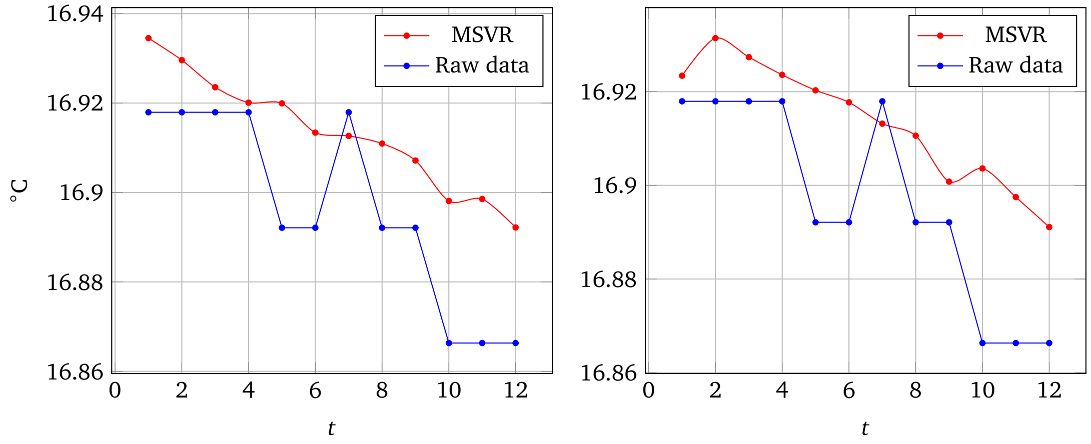
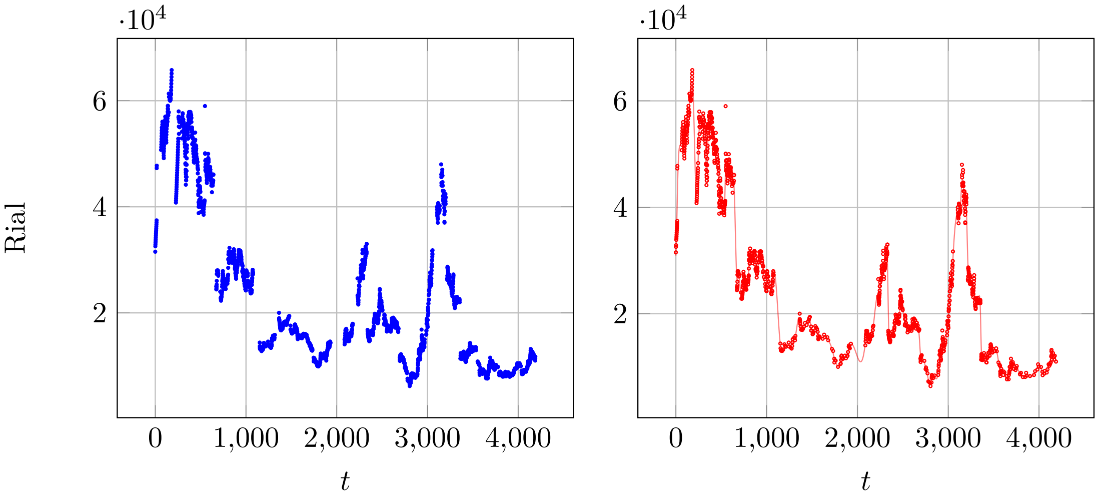


Figure 6: MSVR with subsampled (left) and classical (right) training for environmental data.



(a) Financial market raw data points

(b) Extracted points (red dots) and subsampled data model (red curve)

Figure 7: Left: financial data. Right: the extracted basis and the SB model.

The considered data have been collected from 16/04/2001 to 01/04/2018, for a total of 4172 data points, of which only 3369 are valued. Values are reported in *Rial*, the official currency used in Iran. In this case we consider as prediction time window the last 10 data points. For the training data, at first, we extract 1471 basis using the SB algorithm. Both data points and the extracted basis together with the so-construed model are depicted in Figure 7.

For MSVR we employ a 30 weekdays window. In this case, since financial data tend to vary quite quickly and without any real trend, a larger sliding window would not be useful.

The results of the prediction carried out both via raw data and smoother nodes are summarized in Table 3, while Figures 8 and 9 show the graphical approximation.

Remark 2. In this example with financial data, the use of a smoother learning approach turns out to be particularly meaningful and gives reliable predictions. This is evident especially for the MSVR that fails without any pre-processing on the data. In this framework our approach shows its robustness and effectiveness, leading to accurate approximations. It is also interesting to notice that the ridge regression shows reasonable performances when trained with the raw data.

5 Conclusions and work in progress

In this paper we have presented a robust tool which enables us to extract trends and features from given data. The key step consists in learning pre-processed and smoother data. Even if for the numerical experiments we only focused on time series, such scheme in principle can be applied to any dimension. Unfortunately, in this case the implementation is not trivial.

Therefore, work in progress consists in extending the procedure to higher dimensions. This would allow us to deal with a

	Subsampled training		Classical training	
	RMSE	**RMSEP**	**RMSE**	**RMSEP**
RIDGE	1458.7	12.658	570.99	4.8223
SVR	502.40	4.3751	771.27	6.7010
MSVR	315.03	2.7175	642.01	5.5867

Table 3: Results for financial data.

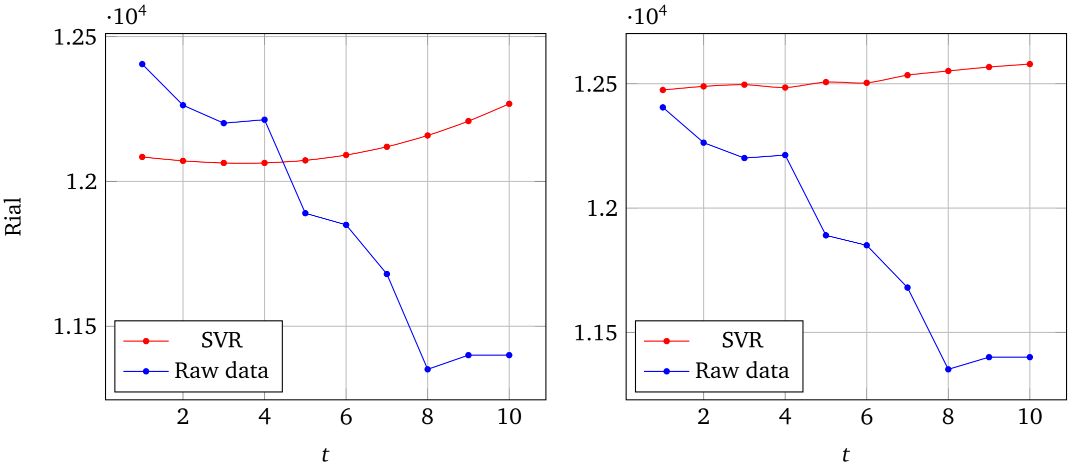


Figure 8: SVR with subsampled (left) and classical (right) training for financial data.

large variety of applications. Among them, we mention here the one of modelling data from satellite, i.e. images [17]. In that case, a reduced model for the image could be obtained by considering the state-of the art presented in [18].

Acknowledgments

We sincerely thank the reviewers for helping us to significantly improve the manuscript. This research has been accomplished within Rete ITaliana di Approssimazione (RITA), partially funded by GNCS-INδAM and through the European Union's Horizon 2020 research and innovation programme ERA-PLANET, grant agreement no. 689443, via the GEOEssential project.

References

[1] J. Amani Rad, J. Höök, E. Larsson, L. von Sydow, *Forward deterministic pricing of options using Gaussian radial basis functions*, J. Comput. Science **24** (2018) pp. 1555–1580.

[2] A. Berlinet, C. Thomas-Agnan, *Reproducing Kernel Hilbert Spaces in Probability and Statistics*, Dordrecht, Kluwer, 2004.

[3] J. Bergstra, R. Bardenet, Y. Bengio, B. Kégl, *Algorithms for Hyper-parameter Optimization*, Proceedings of the 24th International Conference on Neural Information Processing Systems (2011) pp. 2546–2554.

[4] M.D. Buhmann, *Radial Basis Functions: Theory and Implementation*, Cambridge Monogr. Appl. Comput. Math., vol. 12, Cambridge Univ. Press, Cambridge, 2003.

[5] R. Cavoretto, A. De Rossi, H. Qiao, *Topology analysis of global and local RBF transformations for image registration*, Math. Comput. Simul. **147** (2018) pp. 52–72.

[6] C. Cortes, V.N. Vladimir, *Support-vector networks*, Machine Learning **20** (1995) 273–297.

[7] S. De Marchi, A. Martínez, E. Perracchione, *Fast and stable rational RBF-based partition of unity interpolation*, J. Comput. Appl. Math. **349** (2019) 331–343.

[8] S. De Marchi, W. Erb, F. Marchetti, *Spectral filtering for the reduction of the Gibbs phenomenon for polynomial approximation methods on Lissajous curves with applications in MPI*, Dolomites Res. Notes Approx, **10** (2017), pp. 128–137.

[9] G.E. Fasshauer, *Meshfree Approximations Methods with* Matlab, World Scientific, Singapore, 2007.

[10] G.E. Fasshauer, M.J. McCourt, *Kernel-based Approximation Methods Using* Matlab, World Scientific, Singapore, 2015.

[11] E. Francomano, F.M. Hilker, M. Paliaga, E. Venturino, *An efficient method to reconstruct invariant manifolds of saddle points*, Dolom. Res. Notes Approx. **10** (2017) pp. 25–30.

[12] B. Haasdonk, G. Santin, *Greedy Kernel Approximation for Sparse Surrogate Modeling*, in W. Keiper et al. (Eds.), Reduced-Order Modeling (ROM) for Simulation and Optimization: Powerful Algorithms as Key Enablers for Scientific Computing, pp. 21–45, 2018.

[13] T. Hastie, R. Tibshirani, J. Friedman, *The Elements of Statistical Learning*, Springer, New York, NY, USA, 2001.

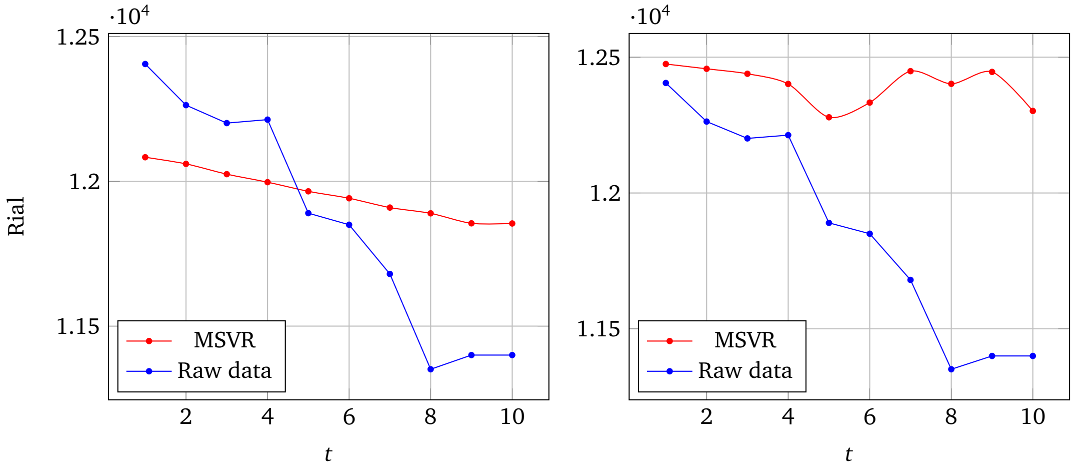


Figure 9: MSVR with subsampled (left) and classical (right) training for financial data.

[14] J. Mercer, *Functions of positive and negative type and their connection with the theory of integral equations*, Phil. Trans. Royal Society **209** (1909) pp. 415–446.

[15] J. Nocedal, S.J. Wright, *Numerical Optimization*, Springer-Verlag, New York, 1999.

[16] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, E. Duchesnay, *Scikit-learn: Machine Learning in Python*, Journal of Machine Learning Research, **12** (2011) pp. 2825–2830.

[17] E. Perracchione, M. Polato, D. Tran, F. Piazzon, F. Aiolli, S. De Marchi, S. Kollet, C. Montzka, A. Sperduti, M. Vianello, M. Putti, *Modelling and processing services and tools*, 2018, GEO Essential Deliverable 1.3; http://www.geoessential.eu/wp-content/uploads/2019/01/GEOEssential-D_1.3_final.pdf.

[18] F. Piazzon, A. Sommariva, M. Vianello, *Caratheodory-Tchakaloff Subsampling*, Dolom. Res. Notes Approx. **10** (2017) pp. 5–14.

[19] A. Safdari-Vaighani, E. Larsson, A. Heryudono, *Radial basis function methods for the Rosenau equation and other higher order PDEs*, J. Sci. Comput. **75** (2018) pp. 1555–1580.

[20] S.A. Sarra, *The* Matlab *radial basis function toolbox*, J. Open Research Software, **5** (2017) pp. 1–10.

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

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

[23] H. Wendland, *Scattered Data Approximation*, Cambridge Monogr. Appl. Comput. Math., vol. 17, Cambridge Univ. Press, Cambridge, 2005.

[24] D. Wirtz, N. Karajan, B. Haasdonk, *Surrogate modelling of multiscale models using kernel methods*, Int. J. Numer. Met. Eng. **101** (2015) pp. 1–28.

[25] D. Wu, K. Warwick, Z. Ma, M.N. Gasson, J.G. Burgess, S. Pan, T.Z. Aziz, *Prediction of parkinson's disease tremor onset using a radial basis function neural network based on particle swarm optimization*, Int. J. Neur. Syst. **20** (2010) pp. 109–116.