Manifold Learning Regression with Non-stationary Kernels

Alexander Kuleshov, Alexander Bernstein, and Evgeny Burnaev(✉)

Skolkovo Institute of Science and Technology, Skolkovo Innovation Center,
3 Nobel Street, Moscow 121205, Russia
{A.Kuleshov,A.Bernstein,E.Burnaev}@skoltech.ru,
http://adase.group

Abstract. Nonlinear multi-output regression problem is to construct a predictive function which estimates an unknown smooth mapping from q-dimensional inputs to m-dimensional outputs based on a training data set consisting of given "input-output" pairs. In order to solve this problem, regression models based on stationary kernels are often used. However, such approaches are not efficient for functions with strongly varying gradients. There exist some attempts to introduce non-stationary kernels to account for possible non-regularities, although even the most efficient one called Manifold Learning Regression (MLR), which estimates the unknown function as well its Jacobian matrix, is too computationally expensive. The main problem is that the MLR is based on a computationally intensive manifold learning technique. In this paper we propose a modified version of the MLR with significantly less computational complexity while preserving its accuracy.

Keywords: Nonlinear multi-output regression
Manifold learning regression · Non-stationary kernel

1 Introduction

1.1 Nonlinear Multi-output Regression

We formulate a nonlinear multi-output regression task [1–3]: let $\mathbf{f}$ be an unknown smooth mapping from an input space $\mathbf{X} \subset \mathbb{R}^q$ to m-dimensional output space $\mathbb{R}^m$. Given a training data set

$$\mathbf{Z}_{(n)} = \{\mathbf{Z}_i = (\mathbf{x}_i, \mathbf{y}_i = \mathbf{f}(\mathbf{x}_i)), i = 1, 2, \ldots, n\}, \tag{1}$$

The research, presented in Sect. 1 of this paper, was partially supported by the Russian Foundation for Basic Research grants 16-01-00576 A and 16-29-09649 ofi m. The research, presented in other sections, was supported solely by the Ministry of Education and Science of Russian Federation, grant No. 14.606.21.0004, grant code: RFMEFI60617X0004.

L. Pancioni et al. (Eds.): ANNPR 2018, LNAI 11081, pp. 152–164, 2018.
https://doi.org/10.1007/978-3-319-99978-4_12

consisting of input-output pairs, the task is to construct the function $\mathbf{y}^* = \mathbf{f}^*(\mathbf{x}) = \mathbf{f}^*(\mathbf{x}|\mathbf{Z}_{(n)})$ to predict the true output $\mathbf{y} = \mathbf{f}(\mathbf{x})$ for an arbitrary Out-of-Sample (OoS) input $\mathbf{x} \in \mathbf{X}$ with small predictive error $|\mathbf{y}^* - \mathbf{y}|$. In engineering applications $\mathbf{f}^*(\mathbf{x})$ is usually used as a surrogate of some target function [4]. Most of optimization algorithms use gradient of the optimized function; in this case, the regression method also should allow estimating $m \times q$ Jacobian matrix $\mathbf{J}_f(\mathbf{x}) = \nabla_x \mathbf{f}(\mathbf{x})$ of the mapping $\mathbf{f}(\mathbf{x})$ at an arbitrary input point $\mathbf{x} \in \mathbf{X}$.

There exist various regression methods such as least squares (LS) techniques (linear and nonlinear), artificial neural networks, kernel nonparametric regression, Gaussian process regression, kriging regression, etc. [1–3,5–16]. A classical approach is based on Kernel Nonparametric Regression (KNR) [7]: we select the kernel function $K(\mathbf{x}, \mathbf{x}')$ (see [17]) and construct the KNR-estimator

$$\mathbf{f}_{KNR}(\mathbf{x}) = \frac{1}{K(\mathbf{x})} \sum_{j=1}^{n} K(\mathbf{x}, \mathbf{x}_j) \cdot y_j, \;\; K(\mathbf{x}) = \sum_{j=1}^{n} K(\mathbf{x}, \mathbf{x}_j), \tag{2}$$

which minimizes (over $\hat{\mathbf{y}}$) the residual $\sum_{j=1}^{n} K(\mathbf{x}, \mathbf{x}_j) \, |\hat{\mathbf{y}} - \mathbf{y}_j|^2$.

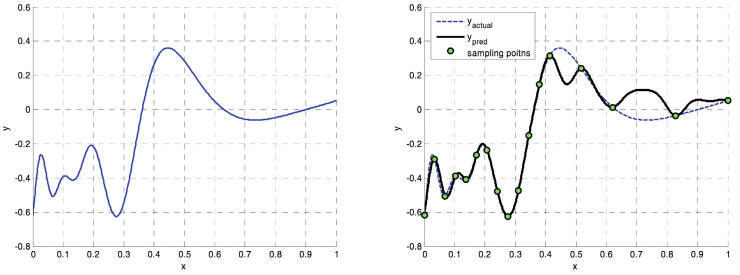
The symmetric non-negative definite function $K(\mathbf{x}, \mathbf{x}')$ can be interpreted as a covariance function of some random field $\mathbf{y}(\mathbf{x})$; thus, the unknown function $\mathbf{f}(\mathbf{x})$ can be interpreted as a realization of the random field $\mathbf{y}(\mathbf{x})$ and $K(\mathbf{x}, \mathbf{x}') = \mathrm{cov}(\mathbf{f}(\mathbf{x}), \mathbf{f}(\mathbf{x}'))$. If we consider only the first and second moments of this random field, then without loss of generality we can assume that this field is Gaussian and as a result obtain so-called Gaussian Process Regression [5,6,18,19].

One of the most popular kernel estimators is kriging, first developed by Krige [20] and popularized by Sacks [21]. Kriging provides both global predictions and their uncertainty. Kriging-based surrogate models are widely used in engineering modeling and optimization [4,22–24].

Kriging regression combines both linear LS and KNR approaches: the deviation of the unknown function $\mathbf{f}(\mathbf{x})$ from its LS estimator, constructed on basis of some functional dictionary, is modeled by a zero mean Gaussian random field with the covariance function $K(\mathbf{x}, \mathbf{x}')$. Thus we can estimate the deviation at the point $\mathbf{x}$ using some filtration procedure and known deviations at the sample points $\{\mathbf{x}_i\}$. Usually stationary covariance functions $K(\mathbf{x}, \mathbf{x}')$ are used that depend on their arguments $\mathbf{x}$ and $\mathbf{x}'$ only through the difference $(\mathbf{x} - \mathbf{x}')$.

1.2 Learning with Non-stationary Kernels

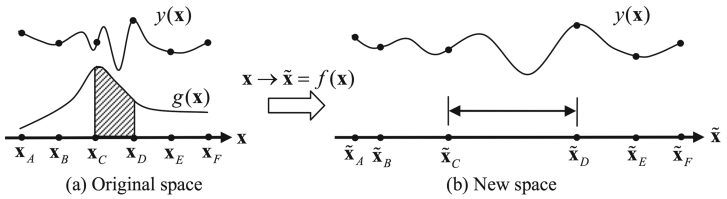
Many methods use kernels that are stationary. However, as indicated e.g. in [2,3,5,6], such methods have serious drawbacks in case of functions with strongly varying gradients. Traditional kriging "is stationary in nature" and has low accuracy in case of functions with "non-stationary responses" (significant changes in "smoothness") [25,26]. Figure 1 illustrates this phenomenon by the Xiong function $\mathbf{f}(\mathbf{x}) = \sin(30(\mathbf{x} - 0.9)^4) \cdot \cos(2(\mathbf{x} - 0.9)) + (\mathbf{x} - 0.9)/2$, $\mathbf{x} \in [0, 1]$, and its kriging estimator with a stationary kernel [25]. Therefore, non-stationary kernels with adaptive kernel width are used to estimate non-regular functions. There are strategies for constructing the non-stationary kernels [26].



(a) True function (b) Kriging prediction (stationary covariance)

Fig. 1. Example of Kriging prediction with a stationary covariance [25].

The interpretable nonlinear map approach from [27] uses the one-to-one reparameterization function $\mathbf{u} = \varphi(\mathbf{x})$ with the inverse $\mathbf{x} = \psi(\mathbf{u})$ to map the Input space $\mathbf{X}$ to $\mathbf{U} = \varphi(\mathbf{X})$, such that the covariance function $k(\mathbf{u}, \mathbf{u}') = K(\psi(\mathbf{u}), \psi(\mathbf{u}')) = \mathrm{cov}(\mathbf{f}(\psi(\mathbf{u})), \mathbf{f}(\psi(\mathbf{u}')))$ becomes approximately stationary. This approach was studied for years in geostatistics in case of relatively low dimensions ($q = 2, 3$), and the general case has been considered in [25] with the reparameterization function $\varphi(\mathbf{x}) = \mathbf{x}_0 + \int_{x_0^{(1)}}^{x^{(1)}} \int_{x_0^{(2)}}^{x^{(2)}} \cdots \int_{x_0^{(q)}}^{x^{(q)}} s(\mathbf{x}) d\mathbf{x}$, where $\mathbf{x} = (x^{(1)}, x^{(2)}, \ldots, x^{(q)})$ and $s(\mathbf{x})$ is a density function, modelled by a linear combination of some "dictionary" functions with optimized coefficients. A simple one-dimensional illustration of such map is provided in Fig. 2.



(a) Original space (b) New space

Fig. 2. A conceptual illustration of the nonlinear reparameterization function [25].

After such reparameterization, KNR-estimator (2) $\mathbf{g}_{KNR}(\mathbf{u})$ for the function $\mathbf{g}(\mathbf{u}) = \mathbf{f}(\psi(\mathbf{u}))$ with the stationary kernel $k(\mathbf{u}, \mathbf{u}')$ is constructed, and the function $\mathbf{f}^*(\mathbf{x}) = \mathbf{g}_{KNR}(\varphi(\mathbf{x}))$ is used as an estimator for $\mathbf{f}(\mathbf{x})$.

1.3 Manifold Learning Regression

A fundamentally different geometrical approach to KNR called Manifold Learning Regression (MLR) was proposed in [10,11]; MLR also constructs the reparameterization function $\mathbf{u} = \varphi(\mathbf{x})$ and estimates the Jacobian matrix $\mathbf{J}_f(\mathbf{x})$.

MLR compares favourably with many conventional regression methods. In Fig. 3 (see [10]) we depict the KNR-estimator $\mathbf{f}_{KNR}$ (2) with a stationary kernel and the MLR-estimator $\mathbf{f}_{MLR}$ for the Xiong function $\mathbf{f}(\mathbf{x})$. The input values in the set $\mathbf{Z}_{(n)}$, $n = 100$ were uniformly randomly distributed on the interval $[0, 1]$.

We see that the MLR method provides the essentially smoother estimate. The mean squared errors $\mathrm{MSE}_{KNR} = 0.0024$ and $\mathrm{MSE}_{MLR} = 0.0014$ were calculated using the test sample with $n = 1001$ uniform grid points in the interval $[0, 1]$.

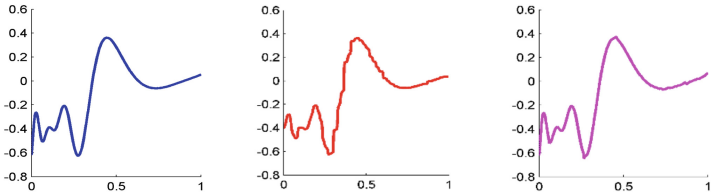


Fig. 3. Reconstruction of the Xiong function (a) by KNR with stationary kernel (b) and MLR (c).

MLR is based on a Manifold Learning approach. Let us represent in the input-output space $\mathbb{R}^p$, $p = q + m$, the graph of the function $\mathbf{f}$ by the smooth q-dimensional manifold (Regression Manifold, RM)

$$\mathbf{M}(\mathbf{f}) = \{\mathbf{Z} = \mathbf{F}(\mathbf{x}) \in \mathbb{R}^p : \mathbf{x} \in \mathbf{X} \subset \mathbb{R}^q\} \subset \mathbb{R}^p, \tag{3}$$

embedded in the ambient space $\mathbb{R}^p$ and parameterized by the single chart

$$\mathbf{F} : \mathbf{x} \in \mathbf{X} \subset \mathbb{R}^q \to \mathbf{Z} = \mathbf{F}(\mathbf{x}) = (\mathbf{x}, \mathbf{f}(\mathbf{x})) \in \mathbb{R}^p. \tag{4}$$

Arbitrary function $\mathbf{f}^* : \mathbf{X} \to \mathbb{R}^m$ also determines the manifold $\mathbf{M}(\mathbf{f}^*)$ (substitute $\mathbf{f}^*(\mathbf{x})$ and $\mathbf{F}^*(\mathbf{x})$ instead of $\mathbf{f}(\mathbf{x})$ and $\mathbf{F}(\mathbf{x})$ in (3) and (4)).

In order to apply MLR, we estimate RM $\mathbf{M}(\mathbf{f})$ using the training data $\mathbf{Z}_{(n)}$ (1) by the Grassmann & Stiefel Eigenmaps (GSE) algorithm [28]. The constructed estimator $\mathbf{M}_{GSE} = \mathbf{M}_{GSE}(\mathbf{Z}_{(n)})$, being also a q-dimensional manifold embedded in $\mathbb{R}^p$, provides small Hausdorff distance $d_H(\mathbf{M}_{GSE}, \mathbf{M}(\mathbf{f}))$ between these manifolds. In addition, the tangent spaces $\mathrm{L}(\mathbf{Z})$ to RM $\mathbf{M}(\mathbf{f})$ at the manifold points $\mathbf{Z} \in \mathbf{M}(\mathbf{f})$ are estimated by the linear spaces $\mathrm{L}_{GSE}(\mathbf{Z})$ with "aligned" bases smoothly depending on $\mathbf{Z}$. GSE also constructs the low-dimensional parameterization $h(\mathbf{Z})$ of the manifold points $\mathbf{Z}$ and the recovery mapping $\mathbf{g}(h)$, which accurately reconstructs $\mathbf{Z}$ from $h(\mathbf{Z})$.

To get the estimator $\mathbf{f}_{MLR}(\mathbf{x})$ of the unknown function $\mathbf{f}$, we solve the equation $\mathbf{M}(\mathbf{f}_{MLR}) = \mathbf{M}_{GSE}$. Using the estimator $\mathrm{L}_{GSE}(\mathbf{F}(\mathbf{x}))$, we also construct $m \times q$ matrix $\mathbf{G}_{MLR}(\mathbf{x})$, which estimates the $m \times q$ Jacobian matrix $\mathbf{J}_f(\mathbf{x}) = \nabla_x \mathbf{f}(\mathbf{x})$ of $\mathbf{f}(\mathbf{x})$ at the arbitrary point $\mathbf{x} \in \mathbf{X}$. Here as the reparameterization function $\mathbf{u} = \varphi(\mathbf{x})$ we use approximation of the unknown function $h(\mathbf{F}(\mathbf{x}))$ (it depends on $\mathbf{f}(\mathbf{x})$, which is unknown at the OoS points $\mathbf{x} \in \mathbf{X}$).

1.4 Paper Contribution

The GSE algorithm contains several very computationally expensive steps such as construction of the aligned bases in the estimated tangent spaces, the embedding mapping and the recovery mappings, the reparameterization mapping, etc. Although the incremental version of the GSE algorithm [29] reduces its complexity, still it remains computationally expensive.

The paper proposes a new modified version of the MLR algorithm (mMLR) with significantly less computational complexity. We developed a simplified version of the MLR algorithm, which does not require computationally expensive steps, listed above, so that we can construct the estimators $(\mathbf{f}_{MLR}(\mathbf{x}), \mathbf{G}_{MLR}(\mathbf{x}))$ while preserving the same accuracy. Then instead of using the KNR procedure with a stationary kernel we developed its version with a non-stationary kernel, which is defined on basis of the constructed MLR estimators.

Note that in this paper we consider the case when the input domain $\mathbf{X} \subset \mathbb{R}^q$ is a "full-dimensional" subset of $\mathbb{R}^q$ (i.e., the intrinsic dimension of $\mathbf{X}$ is equal to q) in contrast to [6,16], where $\mathbf{X}$ is a low-dimensional manifold in $\mathbb{R}^q$. In [30] they reviewed approaches to the regression with manifold valued inputs.

The paper is organized as follows. Section 2 describes some details of the GSE/MLR algorithms; the proposed mMLR algorithm is described in Sect. 3.

2 Manifold Learning Regression

2.1 Tangent Bundle Manifold Estimation Problem

The MLR algorithm is based on the solution of the Tangent bundle manifold estimation problem [31,32]: estimate RM $\mathbf{M}(\mathbf{f})$ (3) from the dataset $\mathbf{Z}_{(n)}$ (1), sampled from $\mathbf{M}(\mathbf{f})$. The manifold estimation problem is to construct:

- the embedding mapping h from RM $\mathbf{M}(\mathbf{f})$ to the q-dimensional Feature Space (FS) $\mathbf{T}_h = h(\mathbf{M}(\mathbf{f}))$, which provides low-dimensional parameterization (coordinates) $h(\mathbf{Z})$ of the manifold points $\mathbf{Z} \in \mathbf{M}(\mathbf{f})$,
- the recovery mapping $\mathbf{g}(t)$ from FS $\mathbf{T}_h$ to $\mathbb{R}^p$, which recovers the manifold points $\mathbf{Z} = \mathbf{g}(t)$ from their low-dimensional coordinates $t = h(\mathbf{Z})$,

such that the recovered value $r_{h,g}(\mathbf{Z}) = \mathbf{g}(h(\mathbf{Z}))$ is close to the initial vector $\mathbf{Z}$:

$$\mathbf{g}(h(\mathbf{Z})) \approx \mathbf{Z}, \tag{5}$$

i.e. the recovery error $\delta_{h,g}(\mathbf{Z}) = |r_{h,g}(\mathbf{Z}) - \mathbf{Z}|$ is small. These mappings determine the q-dimensional Recovered Regression manifold (RRM)

$$\begin{aligned} \mathbf{M}_{h,g} &= r_{h,g}(\mathbf{M}(\mathbf{f})) = \{r_{h,g}(\mathbf{Z}) \in \mathbb{R}^p : \mathbf{Z} \in \mathbf{M}(\mathbf{f})\} \\ &= \{\mathbf{Z} = \mathbf{g}(t) \in \mathbb{R}^p : t \in \mathbf{T}_h = h(\mathbf{M}(\mathbf{f})) \subset \mathbb{R}^q\}, \end{aligned} \tag{6}$$

which is embedded in the ambient space $\mathbb{R}^p$, covered by the single chart $\mathbf{g}$, and consists of all recovered values $r_{h,g}(\mathbf{Z})$ of the manifold points $\mathbf{Z}$. Thanks

to (5) we get proximity of the manifolds $\mathbf{M}_{h,g} \approx \mathbf{M}(\mathbf{f})$, i.e. the Hausdorff distance $d_H(\mathbf{M}_{h,g}, \mathbf{M}(\mathbf{f}))$ between RM $\mathbf{M}(\mathbf{f})$ and RRM $\mathbf{M}_{h,g}$ (6) is small due the inequality $d_H(\mathbf{M}_{h,g}, \mathbf{M}(\mathbf{f})) \leq \sup_{\mathbf{Z}\in\mathbf{M}(\mathbf{f})} \delta_{h,g}(\mathbf{Z})$.

The manifold proximity (5) at the OoS point $\mathbf{Z} \in \mathbf{M}(\mathbf{f}) \setminus \mathbf{Z}_{(n)}$ characterizes the generalization ability of the solution $(h, \mathbf{g})$ at the specific point $\mathbf{Z}$. Good generalization ability requires [32] that the pair $(h, \mathbf{g})$ should provide the tangent proximities $\mathrm{L}_{h,g}(\mathbf{Z}) \approx \mathrm{L}(\mathbf{Z})$ between the tangent spaces $\mathrm{L}(\mathbf{Z})$ to RM $\mathbf{M}(\mathbf{f})$ at points $\mathbf{Z} \in \mathbf{M}(\mathbf{f})$ and the tangent spaces $\mathrm{L}_{h,g}(\mathbf{Z}) = \mathrm{Span}(\mathbf{J}_g(h(\mathbf{Z})))$ (spanned by columns of the Jacobian matrix $\mathbf{J}_g(t)$ of the mapping $\mathbf{g}$ at the point $t = h(\mathbf{Z})$) to RRM $\mathbf{M}_{h,g}$ at the recovered points $r_{h,g}(\mathbf{Z}) \in \mathbf{M}_{h,g}$. Note that the tangent proximity is defined in terms of a chosen distance between these tangent spaces considered as elements of the Grassmann manifold $\mathrm{Grass}(p, q)$, consisting of all q-dimensional linear subspaces in $\mathbb{R}^p$.

The set of manifold points equipped with the tangent spaces at these points is called the Tangent bundle of the manifold [33], and therefore we refer to the manifold estimation problem with the tangent proximity requirement as the Tangent bundle manifold learning problem [31]. The GSE algorithm, briefly described in the next section, provides the solution to this problem.

2.2 Grassmann and Stiefel Eigenmaps Algorithm

The GSE algorithm consists of the three successively performed steps: tangent manifold learning, manifold embedding, and manifold recovery.

Tangent Manifold Learning. We construct the sample-based $p \times q$ matrices $\mathbf{H}(\mathbf{Z})$ with columns $\{\mathbf{H}^{(k)}(\mathbf{Z}) \in \mathbb{R}^p,\ 1 \leq k \leq q\}$, smoothly depending on $\mathbf{Z}$, to meet the relations $\mathrm{Span}(\mathbf{H}(\mathbf{Z})) \approx \mathrm{L}(\mathbf{Z})$ and $\nabla_{\mathbf{H}^{(i)}(\mathbf{Z})}\mathbf{H}^{(j)}(\mathbf{Z}) = \nabla_{\mathbf{H}^{(j)}(\mathbf{Z})}\mathbf{H}^{(i)}(\mathbf{Z})$ (covariant differentiation is used here), $1 \leq i < j \leq q$, for all points $\mathbf{Z} \in \mathbf{M}(\mathbf{f})$.

The latter condition provides that these columns are coordinate tangent fields on RM $\mathbf{M}(\mathbf{f})$ and, thus, $\mathbf{H}(\mathbf{Z})$ is the Jacobian matrix of some mapping [33]. Thus the mappings h and $\mathbf{g}$ are constructed in such a way that

$$\mathbf{J}_g(h(\mathbf{Z})) = \mathbf{H}(\mathbf{Z}). \tag{7}$$

Using Principal Component Analysis (PCA), we estimate the tangent space $\mathrm{L}(\mathbf{Z})$ at the sample point $\mathbf{Z} \in \mathbf{Z}_{(n)}$ [34] by the q-dimensional linear space $\mathrm{L}_{PCA}(\mathbf{Z})$, spanned by the eigenvectors of the local sample covariance matrix

$$\Sigma(\mathbf{Z}|K_p) = \frac{1}{K_p(\mathbf{Z})} \sum_{j=1}^{n} K_p(\mathbf{Z}, \mathbf{Z}_j) \cdot [(\mathbf{Z}_j - \mathbf{Z}) \cdot (\mathbf{Z}_j - \mathbf{Z})^{\mathrm{T}}], \tag{8}$$

corresponding to the q largest eigenvalues; here $K_p(\mathbf{Z}) = \sum_{j=1}^{n} K_p(\mathbf{Z}, \mathbf{Z}_j)$ and $K_p(\mathbf{Z}, \mathbf{Z}')$ is a stationary kernel in $\mathbb{R}^p$ (e.g., the indicator kernel $\mathrm{I}\{|\mathbf{Z} - \mathbf{Z}'| < \varepsilon\}$ or the heat kernel [35] $K_{p,\varepsilon,\rho}(\mathbf{Z}, \mathbf{Z}') = \mathrm{I}\{|\mathbf{Z} - \mathbf{Z}'| \leq \varepsilon\} \cdot \exp\{-\rho \cdot |\mathbf{Z} - \mathbf{Z}'|^2\}$ with the parameters ε and ρ).

We construct the matrices $\mathbf{H}(\mathbf{Z})$ to meet the relations

$$\mathrm{Span}(\mathbf{H}(\mathbf{Z})) = \mathrm{L}_{PCA}(\mathbf{Z}), \tag{9}$$

therefore, the required proximity $\mathrm{Span}(\mathbf{H}(\mathbf{Z})) \approx \mathrm{L}(\mathbf{Z})$ follows automatically from the approximate equalities $\mathrm{L}_{PCA}(\mathbf{Z}) \approx \mathrm{L}(\mathbf{Z})$, which are satisfied when RM $\mathbf{M}(\mathbf{f})$ is "well sampled" and the parameter ε is small enough [36].

The principal components form the orthogonal basis in the linear space $\mathrm{L}_{PCA}(\mathbf{Z})$. Let us denote the $p \times q$ matrix with the principal components as columns by $Q_{PCA}(\mathbf{Z})$. However, for different $\mathbf{Z}$ these bases are not agreed with each other and can be very different even in neighboring points. While preserving the requirements (9), the GSE algorithm constructs other bases in these linear spaces, determined by the $p \times q$ matrices

$$\mathbf{H}_{GSE}(\mathbf{Z}) = Q_{PCA}(\mathbf{Z}) \cdot v(\mathbf{Z}). \tag{10}$$

Here $q \times q$ nonsingular matrices $v(\mathbf{Z})$ should provide smooth dependency of $\mathbf{H}(\mathbf{Z})$ on $\mathbf{Z}$ and coordinateness of the tangent fields $\{\mathbf{H}^{(k)}(\mathbf{Z}) \in \mathbb{R}^p,\ 1 \le k \le q\}$.

At the sample points the matrices $\mathbf{H}_i = \mathbf{H}_{GSE}(\mathbf{Z}_i)$ (10) are constructed to minimize the quadratic form $\sum_{i,j=1}^{n} K_p(\mathbf{Z}_i, \mathbf{Z}_j) \cdot \|\mathbf{H}_i - \mathbf{H}_j\|_F^2$ under the coordinateness constraint and certain normalizing condition, required to avoid a degenerate solution; here $\|\cdot\|_F$ is the Frobenius matrix norm. The exact solution of this problem is obtained in the explicit form; at the OoS points $\mathbf{Z}$, the matrices $\mathbf{H}_{GSE}(\mathbf{Z})$ are constructed using certain interpolation procedure.

Manifold Embedding. After we construct the matrices $\mathbf{H}_{GSE}(\mathbf{Z})$ and assuming that the conditions (5) and (9) are satisfied, we use the Taylor series expansion of the mapping $\mathbf{g}(t)$, $t = h(\mathbf{Z})$ to get the relation $\mathbf{Z}' - \mathbf{Z} \approx \mathbf{H}_{GSE}(\mathbf{Z}) \cdot (h(\mathbf{Z}') - h(\mathbf{Z}))$ for the neighboring points $\mathbf{Z}, \mathbf{Z}' \in \mathbf{M}(\mathbf{f})$. These relations, considered further as regression equations, allow constructing the embedding mapping $h_{GSE}(\mathbf{Z})$ and FS $\mathbf{T}_h = h(\mathbf{M}(\mathbf{f}))$.

Manifold Recovery. After we construct the matrices $\mathbf{H}_{GSE}(\mathbf{Z})$ and the mapping h_{GSE}, using known values $\{\mathbf{g}(t_i) \approx \mathbf{Z}_i\}$ (5) and $\{\mathbf{J}_g(t_i) = \mathbf{H}_i\}$ (9), $t_i = h_{GSE}(\mathbf{Z}_i)$, we construct the mapping $\mathbf{g}_{GSE}(t)$ and the estimator $\mathbf{G}_{GSE}(t)$ for its covariance matrix $\mathbf{J}_g(t)$.

2.3 Manifold Learning Regression Algorithm

We split the p-dimensional vector $\mathbf{Z} = \begin{pmatrix} \mathbf{Z}_{in} \\ \mathbf{Z}_{out} \end{pmatrix}$, $p = q+m$, into the q-dimensional vector $\mathbf{Z}_{in}$ and the m-dimensional vector $\mathbf{Z}_{out}$ and obtain the corresponding partitions

$$\mathbf{H}_{GSE}(\mathbf{Z}) = \begin{pmatrix} \mathbf{H}_{GSE,in}(\mathbf{Z}) \\ \mathbf{H}_{GSE,out}(\mathbf{Z}) \end{pmatrix},\ Q_{PCA}(\mathbf{Z}) = \begin{pmatrix} Q_{PCA,in}(\mathbf{Z}) \\ Q_{PCA,out}(\mathbf{Z}) \end{pmatrix},$$
$$\mathbf{g}_{GSE}(t) = \begin{pmatrix} \mathbf{g}_{GSE,in}(t) \\ \mathbf{g}_{GSE,out}(t) \end{pmatrix},\ \mathbf{G}_{GSE}(t) = \begin{pmatrix} \mathbf{G}_{GSE,in}(t) \\ \mathbf{G}_{GSE,out}(t) \end{pmatrix} \tag{11}$$

of the $p \times q$ matrices $\mathbf{H}_{GSE}(\mathbf{Z})$ and $Q_{PCA}(\mathbf{Z})$, the p-dimensional vector $\mathbf{g}_{GSE}(t)$, and the $p \times q$ matrix $\mathbf{G}_{GSE}(t)$; note that the $q \times q$ matrix $\mathbf{G}_{GSE,in}(t)$ and the

$m \times q$ matrix $\mathbf{G}_{GSE,out}(t)$ are the Jacobian matrices of the mappings $\mathbf{g}_{GSE,in}(t)$ and $\mathbf{g}_{GSE,out}(t)$, respectively.

It follows from the proximities (5), (9) and the partition (11) with $\mathbf{Z} = \mathbf{F}(\mathbf{x})$ (4) that

$$\mathbf{g}_{GSE,in}(h_{GSE}(\mathbf{F}(\mathbf{x}))) \approx \mathbf{x},\ \mathbf{g}_{GSE,out}(h_{GSE}(\mathbf{F}(\mathbf{x}))) \approx \mathbf{f}(\mathbf{x}), \tag{12}$$

but the left part of the latter equation cannot be used for estimating the unknown function $\mathbf{f}(\mathbf{x})$ since it depends on the function $h_{GSE}(\mathbf{F}(\mathbf{x}))$, which in its turn depends on the function $\mathbf{f}(\mathbf{x})$.

According to the MLR approach we construct the estimator $\varphi(\mathbf{x})$ for the function $h_{GSE}(\mathbf{F}(\mathbf{x}))$ as follows. We have two parameterizations of the manifold points $\mathbf{Z} = \mathbf{F}(\mathbf{x}) \in \mathbf{M}(\mathbf{f})$: the "natural" parameterization by the input $\mathbf{x} \in \mathbf{X}$ and the GSE-parameterization $t = h_{GSE}(\mathbf{Z})$, which are linked by the unknown one-to-one mapping $t = \varphi(\mathbf{x})$, whose values $\{\varphi(\mathbf{x}_i) = t_i = h_{GSE}(\mathbf{Z}_i)\}$ are known at the sample inputs $\{\mathbf{x}_i\}$. The relations (5) and (12) imply that $\mathbf{g}_{GSE,in}(\varphi(\mathbf{x})) \approx \mathbf{x}$ and $\mathbf{G}_{GSE,in}(\varphi(\mathbf{x})) \cdot \mathbf{J}_{\varphi}(\mathbf{x}) \approx \mathrm{I}_q$. Thus we get that $\mathbf{J}_{\varphi}(\mathbf{x}) \approx \mathbf{G}^{-1}_{GSE,in}(\varphi(\mathbf{x}))$; here $\mathbf{J}_{\varphi}(\mathbf{x})$ is the Jacobian matrix of the mapping $\varphi(\mathbf{x})$. Therefore, the known matrices $\{\mathbf{G}^{-1}_{GSE,in}(\varphi(\mathbf{x}_i)) = \mathbf{G}^{-1}_{GSE,in}(t_i)\}$ estimate the Jacobian matrices $\{\mathbf{J}_{\varphi}(\mathbf{x}_i)\}$ at the sample inputs $\{\mathbf{x}_i\}$.

Based on the known values $\{(\varphi(\mathbf{x}_i), \mathbf{J}_{\varphi}(\mathbf{x}_i))\}$, $\varphi(\mathbf{x})$ is estimated at the arbitrary point $\mathbf{x}$ by $\varphi_{MLR}(\mathbf{x}) = \frac{1}{K_q(\mathbf{x})} \sum_{j=1}^{n} K_q(\mathbf{x}, \mathbf{x}_j) \cdot \{t_j + \mathbf{G}^{-1}_{GSE,in} \cdot (\mathbf{x} - \mathbf{x}_j)\}$; here $K_q(\mathbf{x}, \mathbf{x}')$ is a stationary kernel in $\mathbb{R}^q$ (like $K_{p,\varepsilon,\rho}$, but defined in $\mathbb{R}^q$).

The relations (12) imply that $\mathbf{G}_{GSE,out}(\varphi(\mathbf{x})) \cdot \mathbf{J}_{\varphi}(\mathbf{x}) \approx \mathbf{J}_f(\mathbf{x})$ and we get

$$\mathbf{f}_{MLR}(\mathbf{x}) = \mathbf{g}_{GSE,out}(\varphi_{MLR}(\mathbf{x})), \tag{13}$$

$$\mathbf{G}_{MLR}(\mathbf{x}) = \mathbf{G}_{GSE,out}(\varphi_{MLR}(\mathbf{x})) \cdot \mathbf{G}^{-1}_{GSE,in}(\varphi_{MLR}(\mathbf{x})) \tag{14}$$

as the estimators for the unknown function $\mathbf{f}(\mathbf{x})$ and its Jacobian matrix $\mathbf{J}_f(\mathbf{x})$.

Note that the estimators (13), (14) require constructing the aligned bases (matrices $\mathbf{H}_{GSE}(\mathbf{Z})$), the embedding mapping $h_{GSE}(\mathbf{Z})$, the recovery mapping $\mathbf{g}_{GSE}(t)$ and the estimator $\mathbf{G}_{GSE}(t)$ for its Jacobian matrix, and the reparameterization mapping $\varphi_{MLR}(\mathbf{x})$. These GSE steps are computationally expensive, even if the incremental version of GSE is used [29].

3 Modified Manifold Learning Regression

The proposed modified version of the MLR method consists of the following parts: constructing both the PCA-approximations for the tangent spaces at the sample points (as in case of the GSE algorithm) and the preliminary estimation of $\mathbf{f}(\mathbf{x})$ for arbitrary inputs (Sect. 3.1), constructing both the PCA-approximations $\mathrm{L}_{PCA}(\mathbf{Z})$ at the OoS points $\mathbf{Z} = \mathbf{F}(\mathbf{x})$ and the estimators $\mathbf{G}_{MLR}(\mathbf{x})$ of the Jacobian matrix $\mathbf{J}_f(\mathbf{x})$ for arbitrary inputs (Sect. 3.2), constructing the non-stationary kernels based on the preliminary MLR estimators and their usage for construction of both the new adaptive PCA-approximations and the final estimators $(\mathbf{f}_{mMLR}(\mathbf{x}), \mathbf{G}_{mMLR}(\mathbf{x}))$.

3.1 Preliminary Estimation of Unknown Functions

We start from the standard PCA-approximations for the tangent spaces $\mathrm{L}(\mathbf{Z})$ at the sample points.

Step 1. Given the training dataset $\mathbf{Z}_{(n)}$ (1), $p \times q$ matrices $Q_{PCA}(\mathbf{Z}_i)$ and linear spaces $\mathrm{L}_{PCA}(\mathbf{Z}_i) = \mathrm{Span}(Q_{PCA}(\mathbf{Z}_i))$, $i = 1, 2, \ldots, n$, are constructed as in Sect. 2.2.

Let $\{\mathbf{H}_{GSE}(\mathbf{Z}_i) = Q_{PCA}(\mathbf{Z}_i) \cdot v(\mathbf{Z}_i)\}$ (10) be the GSE-matrices, computed after the estimation of the aligning matrices $\{v(\mathbf{Z}_i)\}$. It follows from (7) and (9)–(11) that

$$\mathbf{G}_{GSE,in}(h_{GSE}(\mathbf{Z})) = \mathbf{H}_{GSE,in}(\mathbf{Z}) = Q_{PCA,in}(\mathbf{Z}) \cdot v(\mathbf{Z}),$$
$$\mathbf{G}_{GSE,out}(h_{GSE}(\mathbf{Z})) = \mathbf{H}_{GSE,out}(\mathbf{Z}) = Q_{PCA,out}(\mathbf{Z}) \cdot v(\mathbf{Z}).$$

Thus the estimator $\mathbf{G}_{MLR}(\mathbf{x})$ (14) at the sample inputs $\{\mathbf{x}_i\}$ is equal to

$$\mathbf{G}_{MLR}(\mathbf{x}_i) = \mathbf{H}_{GSE,out}(\mathbf{Z}_i) \cdot \mathbf{H}^{-1}_{GSE,in}(\mathbf{Z}_i)$$
$$= Q_{PCA,out}(\mathbf{Z}_i) v(\mathbf{Z}_i) v^{-1}(\mathbf{Z}_i) Q_{PCA,in}(\mathbf{Z}_i) = Q_{PCA,out}(\mathbf{Z}_i) Q^{-1}_{PCA,in}(\mathbf{Z}_i) \quad (15)$$

and depends only on the PCA-matrices $\{Q_{PCA}(\mathbf{Z}_i)\}$, not on the matrices $v(\mathbf{Z}_i)$.

Step 2. Compute the estimators $\{\mathbf{G}_{MLR}(\mathbf{x}_i)\}$ (15) for $i = 1, 2, \ldots, n$.

After the Step 2 we obtain values $\mathbf{G}_{MLR}(\mathbf{x}_i)$ of the Jacobian matrix of $\mathbf{f}(\mathbf{x})$ at the sample inputs. Using the Taylor series expansion we get that $\mathbf{f}(\mathbf{x}) \approx \mathbf{f}(\mathbf{x}') + \mathbf{J}_f(\mathbf{x}') \cdot (\mathbf{x} - \mathbf{x}')$ for the neighboring input points $\mathbf{x}, \mathbf{x}' \in \mathbf{X}$. We construct the estimator $\mathbf{f}^*(\mathbf{x})$ for $\mathbf{f}(\mathbf{x})$ at the arbitrary point $\mathbf{x}$ as a solution to the regression problem with known Jacobian values at sample points [30] by minimizing the residual $\sum_{j=1}^{n} K_q(\mathbf{x}, \mathbf{x}_j) \cdot |\mathbf{y} - \mathbf{y}_j - \mathbf{G}_{MLR}(\mathbf{x}_j) \cdot (\mathbf{x} - \mathbf{x}_j)|^2$ over $\mathbf{y}$.

Step 3. Compute the estimator $\mathbf{f}^*(\mathbf{x})$ at the arbitrary input $\mathbf{x} \in \mathbf{X}$

$$\mathbf{f}^*(\mathbf{x}) = \frac{1}{K_q(\mathbf{x})} \sum_{j=1}^{n} K_q(\mathbf{x}, \mathbf{x}_j) \cdot \{\mathbf{y}_j + \mathbf{G}_{MLR}(\mathbf{x}_j) \cdot (\mathbf{x} - \mathbf{x}_j)\}$$
$$= \mathbf{f}_{sKNR}(\mathbf{x}) + \frac{1}{K_q(\mathbf{x})} \sum_{j=1}^{n} K_q(\mathbf{x}, \mathbf{x}_j) \cdot \mathbf{G}_{MLR}(\mathbf{x}_j) \cdot (\mathbf{x} - \mathbf{x}_j). \quad (16)$$

Here $\mathbf{f}_{sKNR}(\mathbf{x}) = \frac{1}{K_q(\mathbf{x})} \sum_{j=1}^{n} K_q(\mathbf{x}, \mathbf{x}_j) \cdot \mathbf{y}_j$ is the KNR-estimator (2) with a stationary kernel.

Note that the estimators $\mathbf{f}^*(\mathbf{x})$ (16) and $\{\mathbf{G}_{MLR}(\mathbf{x}_i)\}$ (15) coincide with the MLR-estimators (13) and (14) but they have significantly lower computational complexity.

3.2 Estimation of Jacobian Matrix at Arbitrary Point

The $p \times q$ matrix $Q_{PCA}(\mathbf{Z})$ and the tangent space $\mathrm{L}_{PCA}(\mathbf{Z})$ at the OoS point $\mathbf{Z} = \mathbf{F}(\mathbf{x})$ are computed using the estimator $\mathbf{f}^*(\mathbf{x})$ (16). Thus we can define $\mathbf{F}_{MLR}(\mathbf{x}) = (\mathbf{x}, \mathbf{f}^*(\mathbf{x}))$ (4).

Step 4. Compute the $p \times q$ matrix $Q_{PCA}(\mathbf{Z}^*)$ at the point $\mathbf{Z}^* = \mathbf{F}_{MLR}(\mathbf{x})$, such that its columns are the eigenvectors of the matrix $\Sigma(\mathbf{Z}^*|K_p)$ (8) corresponding to the q largest eigenvalues.

The matrix $Q_{PCA}(\mathbf{F}_{MLR}(\mathbf{x}))$ estimates the matrix $Q_{PCA}(\mathbf{F}(\mathbf{x}))$ at the arbitrary input $\mathbf{x} \in \mathbf{X}$. Thus, the relation (14) results in the next step.

Step 5. Compute the preliminary estimator $\mathbf{G}_{MLR}(\mathbf{x})$ for $\mathbf{J}_f(\mathbf{x})$ at the arbitrary input $\mathbf{x} \in \mathbf{X}$

$$\mathbf{G}_{MLR}(\mathbf{x}) = Q_{PCA,out}(\mathbf{F}_{MLR}(\mathbf{x})) \cdot Q^{-1}_{PCA,in}(\mathbf{F}_{MLR}(\mathbf{x})). \tag{17}$$

Then based on (17) we compute the preliminary estimators

$$\begin{aligned}\mathbf{f}_{MLR}(\mathbf{x}) &= \frac{1}{K_q(\mathbf{x})} \sum_{j=1}^{n} K_q(\mathbf{x}, \mathbf{x}_j) \cdot \{\mathbf{y}_j + \mathbf{G}_{MLR}(\mathbf{x}) \cdot (\mathbf{x} - \mathbf{x}_j)\} \\ &= \mathbf{f}_{sKNR}(\mathbf{x}) + \mathbf{G}_{MLR}(\mathbf{x}) \cdot (\mathbf{x} - \overline{\mathbf{x}}_{sKNR})\end{aligned} \tag{18}$$

for $\mathbf{f}(\mathbf{x})$ at the arbitrary input $\mathbf{x} \in \mathbf{X}$; here $\overline{\mathbf{x}}_{sKNR} = \frac{1}{K_q(\mathbf{x})} \sum_{j=1}^{n} K_q(\mathbf{x}, \mathbf{x}_j) \cdot \mathbf{x}_j$.

3.3 Estimation of Unknown Function at Arbitrary Point

The estimators $\mathbf{f}_{MLR}(\mathbf{x})$ (18) and $\mathbf{G}_{MLR}(\mathbf{x})$ (17) use the stationary kernels $K_q(\mathbf{x}, \mathbf{x}')$ in (18) and $K_p(\mathbf{Z}, \mathbf{Z}')$ in $\Sigma(\mathbf{Z}^*|K_p)$ (7), respectively; here we introduce their non-stationary analogues.

Let $\mathrm{L} = \mathrm{Span}(Q)$ and $\mathrm{L}' = \mathrm{Span}(Q')$ be q-dimensional linear spaces in $\mathbb{R}^p$ whose orthonormal bases are the columns of the $p \times q$ orthogonal matrices Q and Q', respectively. Considering them as elements of the Grassmann manifold $\mathrm{Grass}(p, q)$, let us denote by

$$d_{BC}(\mathrm{L}, \mathrm{L}') = \{1 - \mathrm{Det}^2[Q^{\mathrm{T}} \cdot Q']\}^{1/2} \quad \text{and} \quad K_{BC}(\mathrm{L}, \mathrm{L}') = \mathrm{Det}^2[Q^{\mathrm{T}} \cdot Q']$$

the Binet-Cauchy metric and the Binet-Cauchy kernel on the Grassmann manifold, respectively [37,38]. Note that these quantities do not depend on a choice of the orthonormal bases Q and Q'. Let us introduce another Grassmann kernel depending on the threshold τ as

$$K_{G,\tau}(\mathrm{L}, \mathrm{L}') = \mathrm{I}\{d_{BC}(\mathrm{L}, \mathrm{L}') \le \tau\} \cdot K_{BC}(\mathrm{L}, \mathrm{L}').$$

The final mMLR estimators are constructed by modification of the Steps 1–5 above using the introduced non-stationary kernels. For $\mathbf{Z}, \mathbf{Z}' \in \mathbf{Z}_{(n)}$, we introduce the non-stationary kernel

$$K_{p,MLR}(\mathbf{Z}, \mathbf{Z}') = K_{p,\varepsilon,\rho}(\mathbf{Z}, \mathbf{Z}') \cdot K_{G,\tau}(\mathrm{L}_{PCA}(\mathbf{Z}), \mathrm{L}_{PCA}(\mathbf{Z}')). \tag{19}$$

Step 6 (modified Step 1). The columns of the orthogonal $p \times q$ matrices $Q_{mPCA}(\mathbf{Z}_i)$ at sample points consist of the eigenvectors of the matrices $\Sigma(\mathbf{Z}_i|K_{p,MLR})$ (8) corresponding to its q largest eigenvalues, $i = 1, 2, \ldots, n$.

When calculating the covariance matrices $\Sigma(\mathbf{Z}_i|K_{p,MLR})$ we use the non-stationary kernels $K_{p,MLR}$ (19) at the sample points.

Step 7 (modified Step 2). Using (17) with the matrices $\{Q_{PCA}(\mathbf{Z}_i)\}$ replaced by the matrices $\{Q_{mPCA}(\mathbf{Z}_i)\}$ we compute the modified $m \times q$ matrices $\{\mathbf{G}_{mMLR}(\mathbf{x}_i)\}$.

Step 8 (modified Step 3). The value $\mathbf{f}^{**}(\mathbf{x})$ at the arbitrary input $\mathbf{x} \in \mathbf{X}$ is computed by (16) with the matrices $\{\mathbf{G}_{MLR}(\mathbf{x}_i)\}$ replaced by the matrices $\{\mathbf{G}_{mMLR}(\mathbf{x}_i)\}$.

Step 9 (modified Step 4). We compute the $p \times q$ matrix $Q_{mPCA}(\mathbf{Z})$ at the point $\mathbf{Z} = \mathbf{F}_{mMLR}(\mathbf{x}) = (\mathbf{x}, \mathbf{f}^{**}(\mathbf{x}))$ with arbitrary input $\mathbf{x} \in \mathbf{X}$. Columns of this matrix are the eigenvectors of the matrix $\Sigma(\mathbf{F}_{mMLR}(\mathbf{x})|K_{p,MLR})$ (8) corresponding to its q largest eigenvalues with the non-stationary kernel $K_{p,MLR}(\mathbf{Z}, \mathbf{Z}')$ (19), $\mathbf{Z}, \mathbf{Z}' \in \mathbf{Z}_{(n)}$.

Let us denote $\mathrm{L}_{mPCA}(\mathbf{F}_{mMLR}(\mathbf{x})) = \mathrm{Span}(Q_{mPCA}(\mathbf{F}_{mMLR}(\mathbf{x})))$. For the arbitrary inputs $\mathbf{x}, \mathbf{x}' \in \mathbf{X}$ we introduce the non-stationary kernel

$$K_{q,MLR}(\mathbf{x}, \mathbf{x}') = K_{q,\varepsilon,\rho}(\mathbf{x}, \mathbf{x}') \cdot K_{G,\tau}(\mathrm{L}_{mPCA}(\mathbf{F}_{mMLR}(\mathbf{x})), \mathrm{L}_{mPCA}(\mathbf{F}(\mathbf{x}))). \quad (20)$$

Step 10 (modified Step 5). We compute the final estimators $\mathbf{G}_{mMLR}(\mathbf{x})$ for $\mathbf{J}_f(\mathbf{x})$ at the arbitrary input $\mathbf{x} \in \mathbf{X}$ by the formula (17), where $Q_{PCA}(\mathbf{F}_{MLR}(\mathbf{x}))$ is replaced by $Q_{mPCA}(\mathbf{F}_{mMLR}(\mathbf{x}))$.

After that, we compute the final estimators $\mathbf{f}_{mMLR}(\mathbf{x})$ for $\mathbf{f}(\mathbf{x})$ at the arbitrary input $\mathbf{x} \in \mathbf{X}$ by the formula (18) in which $\mathbf{G}_{MLR}(\mathbf{x})$ is replaced by $\mathbf{G}_{mMLR}(\mathbf{x})$, the KNR-estimators $\mathbf{f}_{sKNR}(\mathbf{x})$ and $\overline{\mathbf{x}}_{sKNR}$ with the stationary kernel K_q are replaced by the KNR-estimators $\mathbf{f}_{nsKNR}(\mathbf{x})$ and $\overline{\mathbf{x}}_{nsKNR}$ with the non-stationary kernel $K_{q,MLR}$ (20), respectively.

4 Conclusion

The initially proposed Manifold Learning Regression (MLR) method was based on the GSE-solution to the Tangent Bundle Manifold Learning problem, which is very computationally expensive. The paper proposes a modified version of the MLR method, which does not require to use the most of GSE/MLR steps (such as constructing the aligned bases at the estimated tangent spaces, the embedding and the recovery mappings, the reparameterization mapping, etc.). As a result the modified estimator has significantly smaller computational complexity while preserving its accuracy.

References

1. Seber, G., Wild, C.: Nonlinear Regression. Wiley, Hoboken (2003)
2. Vapnik, V.N.: Statistical Learning Theory. Wiley-Interscience, Hoboken (1998)
3. Loader, C.: Local Regression and Likelihood. Springer, New York (1999). https://doi.org/10.1007/b98858

4. Belyaev, M., Burnaev, E., Kapushev, E., et al.: GTApprox: surrogate modeling for industrial design. Adv. Eng. Softw. **102**, 29–39 (2016)
5. Rasmussen, C.E., Williams, C.K.I.: Gaussian Processes for Machine Learning. Adaptive Computation and Machine Learning. The MIT Press, Cambridge (2005)
6. Calandra, R., Peters, J., Rasmussen, C., Deisenroth, M.: Manifold Gaussian processes for regression. CoRR abs/1402.5876v4 (2014)
7. Wasserman, L.: All of Nonparametric Statistics. Springer, Berlin (2007). https://doi.org/10.1007/0-387-30623-4
8. Bishop, C.M.: Pattern Recognition and Machine Learning. Information Science and Statistics. Springer, New York (2006)
9. Burnaev, E., Vovk, V.: Efficiency of conformalized ridge regression. CoRR arXiv:abs/1404.2083 (2014)
10. Bernstein, A., Kuleshov, A., Yanovich, Y.: Manifold learning in regression tasks. In: Gammerman, A., Vovk, V., Papadopoulos, H. (eds.) SLDS 2015. LNCS (LNAI), vol. 9047, pp. 414–423. Springer, Cham (2015). https://doi.org/10.1007/978-3-319-17091-6_36
11. Bernstein, A.V., Kuleshov, A.P., Yanovich, Y.: Statistical learning via manifold learning. In: 14th IEEE International Conference on Machine Learning and Applications, ICMLA 2015, Miami, FL, USA, 9–11 December 2015, pp. 64–69 (2015)
12. Burnaev, E.V., Panov, M.E., Zaytsev, A.A.: Regression on the basis of nonstationary Gaussian processes with Bayesian regularization. J. Commun. Technol. Electron. **61**(6), 661–671 (2016)
13. Burnaev, E., Nazarov, I.: Conformalized kernel ridge regression. In: 15th IEEE International Conference on Machine Learning and Applications, ICMLA 2016, Anaheim, CA, USA, 18–20 December 2016, pp. 45–52 (2016)
14. Burnaev, E., Panov, M.: Adaptive design of experiments based on Gaussian processes. In: Gammerman, A., Vovk, V., Papadopoulos, H. (eds.) SLDS 2015. LNCS (LNAI), vol. 9047, pp. 116–125. Springer, Cham (2015). https://doi.org/10.1007/978-3-319-17091-6_7
15. Belyaev, M., Burnaev, E., Kapushev, Y.: Gaussian process regression for structured data sets. In: Gammerman, A., Vovk, V., Papadopoulos, H. (eds.) SLDS 2015. LNCS (LNAI), vol. 9047, pp. 106–115. Springer, Cham (2015). https://doi.org/10.1007/978-3-319-17091-6_6
16. Yang, Y., Dunson, D.: Bayesian manifold regression. CoRR abs/1305.0167v2 (2014)
17. Genton, M.G.: Classes of kernels for machine learning: a statistical perspective. JMLR **2**, 299–312 (2001)
18. Burnaev, E., Zaytsev, A., Spokoiny, V.: The Bernstein-von Mises theorem for regression based on Gaussian processes. Russ. Math. Surv. **68**(5), 954–956 (2013)
19. Zaitsev, A.A., Burnaev, E.V., Spokoiny, V.G.: Properties of the posterior distribution of a regression model based on Gaussian random fields. Autom. Remote Control **74**(10), 1645–1655 (2013)
20. Krige, D.: A statistical approach to some basic mine valuation problems on the witwatersrand. J. Chem. Metall. Mining Eng. Soc. South Affrica **52**(6), 119–139 (1951)
21. Sacks, J., Welch, W., Mitchell, T., Wynn, H.: Design and analysis of computer experiments. Stat. Sci. **4**(4), 409–435 (1989)
22. Simpson, T.W., et al.: Metamodels for computer-based engineering design: survey and recommendations. Eng. Comput. **7**(2), 129–150 (2001)
23. Wang, G., Gary, S.S.: Review of metamodeling techniques in support of engineering design optimization. J. Mech. Des. **129**(3), 370–381 (2007)

24. Forrester, A., Sobester, A., Keane, A.: Engineering Design via Surrogate Modelling. A Practical Guide. Wiley, New York (2008)
25. Xiong, Y., Chen, W., Apley, D., Ding, X.: A non-stationary covariance-based kriging method for metamodelling in engineering design. Int. J. Numerical Methods Eng. **71**(6), 733–756 (2006)
26. Toal, D.J., Keane, A.J.: Non-stationary kriging for design optimization. Eng. Optim. **44**(6), 741–765 (2012)
27. Sampson, P.D., Guttorp, P.: Nonparametric estimation of nonstationary spatial covariance structure. J. Am. Stat. Assoc. **87**(417), 108–119 (1992)
28. Bernstein, A., Kuleshov, A.: Low-dimensional data representation in data analysis. In: El Gayar, N., Schwenker, F., Suen, C. (eds.) ANNPR 2014. LNCS (LNAI), vol. 8774, pp. 47–58. Springer, Cham (2014). https://doi.org/10.1007/978-3-319-11656-3_5
29. Kuleshov, A., Bernstein, A.: Incremental construction of low-dimensional data representations. In: Schwenker, F., Abbas, H.M., El Gayar, N., Trentin, E. (eds.) ANNPR 2016. LNCS (LNAI), vol. 9896, pp. 55–67. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-46182-3_5
30. Kuleshov, A.P., Bernstein, A.: Nonlinear multi-output regression on unknown input manifold. Ann. Math. Artif. Intell. **81**(1–2), 209–240 (2017)
31. Bernstein, A.V., Kuleshov, A.P.: Tangent bundle manifold learning via grassmann & stiefel eigenmaps. CoRR abs/1212.6031 (2012)
32. Bernstein, A., Kuleshov, A.: Manifold learning: generalization ability and tangent proximity. Int. J. Softw. Inf. **7**(3), 359–390 (2013)
33. Jost, J.: Riemannian Geometry and Geometric Analysis. Springer, Heidelberg (2002). https://doi.org/10.1007/978-3-642-21298-7
34. Zhang, Z., Zha, H.: Principal manifolds and nonlinear dimension reduction via local tangent space alignment. SIAM J. Sci. Comput. **26**(1), 313–338 (2005)
35. Belkin, M., Niyogi, P.: Laplacian eigenmaps for dimensionality reduction and data representation. Neural Comput. **15**, 1373–1396 (2003)
36. Tyagi, H., Vural, E., Frossard, P.: Tangent space estimation for smooth embeddings of riemannian manifold. CoRR abs/1208.1065v2 (2013)
37. Hamm, J., Lee, D.D.: Grassmann discriminant analysis: a unifying view on subspace-based learning. In: Koller, D., Schuurmans, D., Bengio, T., Bottou, L. (eds.) The 25th NIPS Conference, Advances in Neural Information Processing Systems 21, pp. 376–383. MIT Press, Cambridge (2009)
38. Wolf, L., Shashua, A.: Learning over sets using kernel principal angles. J. Mach. Learn. Res. **4**, 913–931 (2003)