GEO ESSENTIAL ERA-PLANET

Essential Variables workflows for resource efficiency and environmental management

Deliverable 1.3
Modelling and processing services and tools

Creator	*Department of Mathematics Tullio Levi-Civita, University of Padova. ** Forschungszentrum Jülich, Institute of Bio- and Geosciences: Agrosphere (IBG-3). *** Dipartimento di Salute della Donna e del Bambino, University of Padova. **** Dipartimento di Ingegneria Civile, Edile e Ambientale – ICEA, University of Padova.
Creation date	March 22, 2019
Due date	August 30, 2019
Last revision date	October 04, 2019
Status	Final
Type	Report
Description	This deliverable proposes tools to model data and give predictions on the evolution of several quantities of interest.
Right	Public
Language	English

Citation E. Perracchione*, M. Polato*, W. Erb*, F. Piazzon*, F. Marchetti***, F. Aiolli*, B. Bayat**, A. Botto****, S. De Marchi*, S. Kollet**, C. Montzka**, A. Sperduti*, M. Vianello*, M. Putti*, 2019. Modelling and processing services and tools.

Grant agreement GEOEssential Deliverable 1.3., ERA-PLANET No 689443

Table of Contents

Introduction

The objective of this report is to provide a survey of the mathematical foundation of the efficient and robust reduced order modelling based on compression/approximation methods. This task aims at delivering value-added services to Essential Variables (EVs) within the context of the GeoEssential project (McCallum, et al., 2019). The realm of the EVs of interest encompasses a large number of Earth observation data types, especially 2D spaceborne measurements, that necessarily calls for a drastic synthesis. To this aim we here investigate the use of compression techniques based on least squares Caratheodory-Tchakaloff subsampling and discontinuous kernel-based interpolation. Once the data is compressed, we are then able to apply useful machine learning tools for forecasting the dynamics of the EVs.

The huge amount of available satellite data needs model reduction schemes for delivering the sought services such as, e.g., short term real-time predictions via machine learning schemes. Effective synthesis can be achieved only via suitable reduced order models that must be at the same time efficient and accurate to ensure meaningful analysis of the EVs of interest. Novel theories arising in the field of numerical analysis, and more specifically approximation theory, provide an opportunity to accomplish these relevant tasks. This document intends to demonstrate the effectiveness of this combination by developing the necessary theoretical foundation of approximation algorithms and shows, via simple examples, the applicability of these modelling tools for the purposes of the GEOEssential project.

More specifically, we focus on two aspects of data-based modelling:

- Compression (refer to Section Data compression): reduce the dimensions for satellite data via either Caratheodory-Tchakaloff or Padova points subsampling (Bos, et al., 2006) (Bos, et al., 2017) (Piazzon, et al., 2017) (Piazzon, et al., 2017).
- Reconstruction (refer to Section Approximation with VDSKS): accurately approximate the data with the reduced number of points via either polynomial least squares or discontinuous kernel-based methods (Bozzini, et al., 2015) (De Marchi, et al., 2017) (De Marchi, et al., 2019).

These data based reduced models are used as inputs for machine learning schemes. The latter allow us to predict the time evolution of the considered EVs (refer to Section Time prediction). This results in an accurate and truly efficient tool for inferring on the dynamics of the quantities of interest. Precisely, in the following we focus on Support Vector Regression (SVR) (Shawe-Taylor & Cristianini, 2004) that takes as training examples the reduced data models. Comparisons with the well-known Ensamble Kalman Filter (EnKF), briefly reviewed in the Appendix, are also carried out.

After providing a theoretical framework, we test in Sections Simulations for compression algorithms-Simulations for time predictions and further discuss in Section Conclusions the proposed techniques for satellite data consisting of the Level 2 soil moisture product obtained by the NASA Soil Moisture Active Passive (SMAP) satellite (Entekhabi at al., 2014) and simulated data obtained via the TERRestrial SYStem Modelling Platform (TerrSysMP) (Kollet

& Maxwell, 2008) (Shrestha, et al., 2014). Such experiments were carried out with a Python software which is freely available for the scientific community at https://github.com/emmaA89/vlabprediction.

Data compression

The schemes we present in this section represent the background for efficiently dealing with huge data (Perracchione, et al., 2019). To introduce them, we need some mathematical background recalled in what follows.

Caratheodory-Tchakaloff subsampling

Tchakaloff's theorem is the basis for quadrature theory. It states that for every compactly supported measure there exists a positive algebraic quadrature formula with a cardinality that do not exceed the dimension of the exactness polynomial space. This specifically allows us to compress data, meaning that we are able to extract a reduced number of points such that the approximation error in reconstructing the original underlying function via Polynomial Least Squares (POLSs) can be bounded and controlled. Such a strategy is also independent of the problem geometry and thus enables us to consider specific polygonal regions.

To explain the algorithm we first need to recall the following theorem (Caratheodory, 1911); see also (Piazzon, et al., 2017) (Piazzon, et al., 2017) (Sommariva & Vianello, 2017).

Theorem 1. *Let μ be a multivariate discrete measure supported at a finite set $X_N = \{x_i, i = 1, \dots, N\} \subseteq \Omega,\ \Omega \in \mathbb{R}^d$, with correspondent positive weights (masses) $\Lambda_\mathrm{N} = \{\lambda_i, i = 1, \dots, N\}$ and let $S = \mathrm{span}(\phi_1 \dots, \phi_L)$ a finite dimensional space of d-variate functions defined on $\Omega \supseteq X_N$, with $P = \dim(S_{|X_N}) \leq L$. Then, there exists a quadrature formula with nodes $T_M = \{\boldsymbol{t}_i, i = 1, \dots, M\} \subseteq X_N$ and positive weights $W_M = \{w_i, i = 1, \dots, M\}$, with $M \leq P$, such that*

$$I_\mu(f) = \sum_{i=1}^{N} \lambda_i f(x_i) = \sum_{i=1}^{M} w_i f(t_i), \qquad \forall f \in S_{|X_N}.$$

In what follows we consider $S = P_p^d(\Omega)$, i.e., the subspace of d-variate real polynomials of total degree not exceeding p. If Ω is an algebraic variety of $\mathbb{R}^d$, then

$$\dim(S_{|X_N}) \leq \dim(S_{|X_N}) < L = \dim\left(P_p^d(\Omega)\right).$$

In other words, such theorem shows that the original sampling set can be replaced by a smaller one. Moreover in (Piazzon, et al., 2017), the authors prove that this can be done keeping practically invariant the least squares approximation estimates. To solve the problem outlined in Theorem 1, i.e. the one of computing weights and nodes, one can use quadratic programming, namely the classical Lawson-Hanson active set method for NonNegative Least Squares (NNLS) or also Linear Programming (LP) via the classical simplex method.

Indeed, restricting our search to polynomials of degree p, we need to solve the following quadratic minimum problem

$$\begin{cases} \min \left\| V^T u - b \right\|_2, \\ \qquad u \geq 0, \end{cases}$$

where V is the classical Vandermonde matrix and $b = V^T \lambda$. Then, the nonvanishing components of such a solution give the weights $W_M = \{w_i, i = 1, \dots, M\}$ and the indexes of the nodes $T_M = \{t_i, i = 1, \dots, M\}$.

For the linear programming approach, we instead have to find

$$\begin{cases} \min s^T u, \\ V^T u = b, \\ \quad u \geq 0, \end{cases}$$

where the constraints identify a polytope (the feasible region) and the vector s is chosen to be linearly independent from the rows of V^T.

This method enables us to consider only a few points to reconstruct a given function f. The so-constructed algorithm has already been used in (Piazzon, et al., 2017) and we refer to as CATCH scheme. Precisely, let us consider two subsets: $X_N = \{x_i, i = 1, \dots, N\} \subseteq \Omega,\ \Omega \subset \mathbb{R}^d$, the set of distinct data points (or data sites or nodes), arbitrarily distributed on Ω and $F_N = \{f_i, i = 1, \dots, N\} \subseteq \mathbb{R}^d$, the associated set of data values (or measurements or function values). We model the function f by considering M points extracted via Theorem 1 and then we apply some approximation schemes, such as in this case, POLSs.

Remark 1. Note that, for the spatial data we can think of X_N as the set of pixels with the associated values F_N. Then, we construct the reduced model and by evaluating it, we can reconstruct the image at each pixel via the approximated values, namely $y_k, k = 1, \dots, N$.

We also stress the fact that the data might be characterized by steep gradients or discontinuities and thus ad hoc meshfree methods might be of interest. Therefore, as a second reconstruction scheme after the compression we propose kernel-based methods constructed via (possibly) discontinuous bases as an alternative to POLSs. This approach is outlined in the next section and enables us to partially overcome the Gibbs phenomenon; refer to (Fornberg & Flyer, 2008) (Gottlieb & Shu, 1997) (Jung, 2007) for a general overview.

As a final remark on this section, we point out that if there are no special needs of focusing on a particular region of interest, but the whole (square or rectangular) image needs to be compressed and reconstructed, then a good choice for reducing samples is the one of considering the so-called *Padova points*; refer e.g. to (Bos, et al., 2006). Indeed, since on those nodes the Lebesgue constant grows logarithmically with respect to the number of nodes, the polynomial approximation turns out to be stable.

Padova points subsampling

The Padova points belong to four families which only differ because of the position of the Padova points on two consecutive vertices of the square $[-1,1]^2$. In particular, for each family of Padova points, two nodes lie on consecutive vertices of the square $[-1,1]^2$, $2p-1$ points lie on the edges of the square, and the remaining points lie on the self-intersections of a generating curve inside the square.

The one we consider here are so generated: letting p a natural number, we define

$$T_M = \left\{ \varphi\left(\frac{k\,\pi}{p(p+1)}\right), k = 0, \dots, p(p+1) \right\},$$

where

$$M = \frac{(p+1)(p+2)}{2},$$

and

$$\varphi(t) = \big(-\cos((p+1)t), -\cos(pt) \big), \quad t \in [0,\pi].$$

Note that φ is a closed parametric curve in the interval $[0,2\pi]$, and is a special case of Lissajous curves; see (Bos, et al., 2017) (De Marchi, et al., 2017) (Erb, 2016).

Approximation with VSDKs

The second reconstruction scheme we propose is based on interpolating the function values via kernel-based methods briefly reviewed below and then extended to work with discontinuous bases.

Kernel-based methods

After compressing the image as explained in the previous section, we want to find an interpolating function $P_f \colon \Omega \to \mathbb{R}$ such that:

$$P_f(t_i) = f_i, \quad i \in \{1, \dots, M\}.$$

Since the extracted nodes are scattered, meshfree or meshless methods turn out to be particularly suitable. Indeed, they are easy to implement in any dimension.

In RBF interpolation, we suppose to have a univariate function $\phi\colon [0,\infty) \to \mathbb{R}$ that provides, for $t, z \in \Omega$, the real symmetric strictly positive definite kernel

$$K(t,z) = \phi\left(||t-z||_2 \right).$$

The interpolating function P_f on the reduced nodes can be written as

$$P_f(t) = \sum_{k=1}^{M} c_k K(t, t_k), \quad t \in \Omega$$

To find the coefficients, we simply have to solve the linear system of equations (Fasshauer, 2007) (Fasshauer, 2007)

$$Ac = f,$$

where $c = (c_1, \dots, c_M)^T$, $c = (f_1, \dots, f_M)^T$, and

$$(A)_{ik} = K(t_i, t_k), \quad i, k = 1, \dots, M.$$

Since we suppose K symmetric and strictly positive definite, this system has exactly one solution.

Variably scaled discontinuous kernels

We consider now a scenario in which piecewise continuous functions are approximated with discontinuous kernels. Note that this framework is the one that characterizes satellite data over land surfaces, where no measurements are over the oceans and/or there are regions masked out due to lakes, built-up areas or huge forests. We consider the following general setting (refer to (De Marchi, et al., 2019)):

Assumption 1. We assume that:

1. The bounded set $\Omega \subset \mathbb{R}^d$ is the union of n pairwise disjoint sets Ω_i, $i \in \{1, \dots, n\}$.
2. The subsets Ω_i satisfy an interior cone condition and have a Lipschitz boundary.
3. We define an auxiliary function $\psi : \Omega \to \Sigma$, $\Sigma \subset \mathbb{R}$ so that it is constant on the subsets Ω_i, i.e., $\psi(t) = \alpha_i$, for $t \in \Omega_i$. If, for $i \neq j$, $\overline{\Omega_\iota} \cap \overline{\Omega_\iota} \neq \emptyset$ holds true, then also $\alpha_i \neq \alpha_j$. In other words, the piecewise constant function ψ is discontinuous at the boundaries of Ω_i.

The function ψ is used to mimic the discontinuities of the original function/image and thus, we consider now the following variably scaled discontinuous kernels (VSDKs) see (De Marchi, et al., 2019) and also refer to (Bozzini, et al., 2015) for the general definition of VSKs.

Definition 1. Assume that all prerequisites in Assumption 1 are given and let K be a continuous strictly positive definite kernel on $\Omega \times \Sigma \subset \mathbb{R}^{d+1}$ defined by a radial basis function ϕ. We define a VSDK K_ψ on Ω as

$$K_\psi(t, z) := K\Big(\big(t, \psi(t)\big), \big(z, \psi(z)\big)\Big) = \phi\left(\sqrt{||\mathrm{t} - \mathrm{z}||_2^2 + \big(\psi(t) - \psi(z)\big)^2}\right),$$

for $t, z \in \Omega$.

We now use the following notation $\hat{t} = (t, \psi(t))$ and we define the on intepolant $\Omega \times \Sigma$, as

$$P_f(\hat{t}) = \sum_{k=1}^{M} c_k K(\hat{t}, \hat{t}_k).$$

Then the VSDK interpolant V_f on Ω is given by

$$V_f(t) = P_f(\hat{t}) = \sum_{k=1}^{M} c_k K_\psi(t, t_k), \quad t \in \Omega.$$

The computation of the VSDK interpolant turns out to be trivial. Indeed, for the given node set $T_M \subset \Omega$, we consider the associated node set $\widehat{T}_M = \{\hat{t}_1, \dots, \hat{t}_M\}$ and the coefficients $c_1, \dots, c_M$ of the interpolant $V_f(t)$ are computed by solving

$$A_\psi c = f,$$

where

$$A_\psi = \begin{pmatrix} K(\hat{t}_1, \hat{t}_1) & \cdots & K(\hat{t}_1, \hat{t}_M) \\ \vdots & \ddots & \vdots \\ K(\hat{t}_M, \hat{t}_1) & \cdots & K(\hat{t}_M, \hat{t}_M) \end{pmatrix}.$$

For details about the error bounds please refer to the Appendix.
As a final tool, we will also present SVR. Indeed, the reduced models for the images enable us to give predictions in time which turn out to be accurate and efficient.

Time prediction

Machine learning (see (Shawe-Taylor & Cristianini, 2004) for a general overview) is a field of computer science which aims to discover patterns from data. In other words, its goal consists in understanding which relation links inputs and outputs. In particular, SVR can be used in our context in order to give reliable predictions on the future dynamics. We thus suppose to have a set of T grids, with same pixels and an underlying function which varies in time, i.e. a set of pixels $X_N = \{x_i, i = 1, \dots, N\} \subseteq \Omega$ and the function values at each time step, i.e. $F_N^j = \{f_i^j, i = 1, \dots, N\} \subseteq \mathbb{R}, j = 1, \dots, T$. To infer on the dynamics of the process, one could apply, e.g. regression tools, pixel by pixel. This would be truly expensive and therefore, we propose to study the time evolution only for the reduced set of points selected via Padova of Caratheodory-Tchakaloff subsampling. Then, once we get the estimations for the time step $T+1$, i.e. $\hat{y}_i^{T+1}, i = 1, \dots, M$, we reconstruct the predicted image on the whole domain by means of VSDKs and we produce an estimation of the image evolution, i.e. we produce the set $y_i^{T+1}, i = 1, \dots, N$, which approximate the true solution f_i^{T+1}. About reduced models for machine learning we also refer the reader to (Aminian Shahrokhabadi, et al., 2019).

To point out how SVR works, we focus on the i-th point, i.e. the index i is fixed if not elsewhere noted. Given $T_T^i = \{t_i^j, j = 1, \dots, T\}$, and $F_T^i = \{f_i^j, j = 1, \dots, T\}$, we formally define the training set $D_i = T_T^i \times F_T^i$, in which we assume there exists a function (i.e., relation) such that

$$g_i(t_i^j) \approx f_i^j,$$

for $(t_i^j, f_i^j) \in D_i$. Again, a learning algorithm tries to find a function, i.e. a mode, that approximates the unknown function g_i as tightly as possible.

To this aim, we consider kernel-based methods which are one of the most used machine learning approaches. They take advantage of the so-called kernel trick which allows to implicitly compute vector similarities (defined in terms of dot-product).

Kernel-based methods

Focusing on strictly positive and symmetric kernels, we know that $K(t, z)$ admits the following expansion

$$K(t, z) = \Phi^{\mathrm{T}}(t)\Phi(z),$$

where $\Phi: \Omega \rightarrow G$ is a mapping function from Ω to the embedding feature space G (Shawe-Taylor & Cristianini, 2004). Kernel machines are a particular family of machine learning methods that try to minimize the following problem:

$$\arg\min_{g_i \in G} L(g_i(t_i^1), \dots, g_i(t_i^T)) + \Lambda||g_i||_G,$$

where L is the so-called loss function associated to the empirical error, Λ is a trade-off parameter also known as regularization parameter. We assume that the solution of such kind of problem can be expressed as

$$g_i(t) = w_i^T \Phi(t) = \sum_{j=1}^{T} c_j K(t_i^j, t),$$

which means that g_i can be expressed as a hyperplane in a feature space defined by the function Φ, and hence it can be seen as a weighted sum of kernels between the input vector and the vectors in the training set. The intuition behind this result is that, by using the embedding function Φ, the data are projected into a (usually higher dimensional) space in which the hypothesis we are looking for becomes a linear function that can be expressed in terms of training examples.

We now focus on support vector machine, which is the most famous kernel method and represents the state of the art performances in many learning tasks.

Support vector regression

Support vector machine is usually used for classification tasks, but it can be easily adapted to regression (SVR) (Cortes & Vladimir, 1995). The SVR problem in its primal form is given by

$$\min_{w_i, b_i, \xi_i, \xi_i^*} \frac{1}{2} ||w_i||_2^2 + C \sum_{j=1}^{T} \xi_{ij} + \xi_{ij}^*,$$

subject to

$$\begin{gathered} f_i^j - w_i^T \Phi(t_i^j) - b \leq \epsilon + \xi_{ij}, \\ -f_i^j + w_i^T \Phi(t_i^j) + b \leq \epsilon + \xi_{ij}^*, \\ \xi_{ij}^*, \xi_{ij} \geq 0. \end{gathered}$$

for $j = 1, \dots, T$, where the objective function aims to minimize the squared norm of the hypothesis in order to get a smooth function, while maintaining the number of errors as low as possible. C represents the trade-off hyper-parameter. The hyper-parameter ϵ indicates the width of the tube in which the examples can fall into without being counted as errors.

This problem is usually solved in its dual form defined as

$$\min_{c_i, c_i^*} \frac{1}{2} \sum_{k,j=1}^{T} (c_{i,k} - c_{i,k}^*)(c_{j,k} - c_{j,k}^*) K(t_i^k, t_i^j) + \sum_{k=1}^{T} (c_{i,k} + c_{i,k}^*)\epsilon - \sum_{k,j=1}^{T} (c_{i,k} - c_{i,j}^*) f_i^k,$$

and

$$\begin{gathered} \sum_{k=1}^{T} (c_{i,k} + c_{i,k}^*) = 0, \\ c_{i,k}, c_{i,k}^* \in [0, C]. \end{gathered}$$

Then, the final regression model reads as follows:

$$g_i(t) = \sum_{j \in SV} (c_{i,j} - c_{i,j}^*) K(t, t_i^j) + b,$$

where SV is the set of training examples t_i^j such that the corresponding $c_{i,j}$ or $c_{i,j}^*$ are not both zero, and they are called support vectors.

Concerning the implementation, we make use of the Python package scikit-learn freely available on the Github repository at https://github.com/scikit-learn/scikit-learn.

As a comparison with SVR, in the experiments we also consider the so-called Kalman Filter. Once we construct the reduced model, in the context of data assimilation, the aim is to incorporate new observations and predictions into the model state of a numerical model. To this aim, the well-known scheme based on the so-called Ensemble Kalman Filter (EnKF) is usually employed. For more details on that, we refer the reader to the Appendix and e.g. to (Gillijns, et al., 2006) (Johns & Mandel, 2008) (Kalman, 1960). In the following test it is implemented via the Python package filterpy freely available on the Github repository at https://github.com/rlabbe/filterpy. Refer also to the Appendix.

Simulations for compression algorithms

In the numerical experiments that follow, we consider two test data sets representing the soil moisture over Europe. Soil moisture is a key variable for hydrology which turns out to be meaningful for many applications, such as modeling and forecast of climate variability and water resources management. Moreover, soil moisture is now recognized as an essential variable (EV), playing an important role in addressing the UN's SDGs, either directly or indirectly. For instance, soil moisture content may influence access to available water (targeting SDG no. 6), control land-atmosphere feedback and interactions (targeting SDG no. 13), affect land surface properties and ecosystem functioning (targeting SDG no. 15), and crop biomass and yield productions (targeting SDG no. 2). Such links may even be of great importance in arid and semi-arid regions facing water shortage and intensive droughts.

The first grid we consider is plotted in Figure 1 (left) and consists of soil moisture data taken by NASA Soil Moisture Active Passive (SMAP) mission in April 2015 (Entekhabi at al., 2014). The space segment has been launched on January 31, 2015, and the mission is designed to principally measure soil moisture globally. In what follows we might refer to such grid as raw data image. The second test grid is plotted in Figure 1 (right) and it consists of simulated soil moisture data obtained via the TERRestrial SYStem Modelling Platform (TerrSysMP) that was developed to simulate the interaction between lateral flow processes in river basins with the lower atmospheric boundary layer (Kollet & Maxwell, 2008) (Shrestha, et al., 2014). The Centre for High-Performance Scientific Computing in terrestrial systems (HPSC TerrSys) is operating TerrSysMP in a forecasting setup over North Rhine-Westphalia and Europe. The model results are made available for the scientific community daily as videos via the YouTube Channel of HPSC TerrSys https://www.youtube.com/channel/UCGio3ckQwasR5a_kJo1GdOw.

Starting with the two grids of soil moisture reported in Figure 1 we extract two sub-regions used in our simulations and plotted in Figure 2. After compression, the reconstruction scheme is carried out in two ways: 1) POLSs and 2) VSDKs. The second method turns out to be meaningful when discontinuities appear, indeed it reduces the Gibbs effect.

The two approximation methods are compared in terms of Compress Ratio (CR) and Mean Square Error (MSE).

$$\mathrm{MSE} = \frac{1}{N}\sum_{i=1}^{N}(f_i - y_i)^2.$$

Letting S the number of coefficients that need to be computed for POLSs and VSDKs, the CR is given by

$$\text{CR} = \frac{N}{S},$$

where for VSDK interpolation $S = M$, while for the POLSs scheme it is linked to the polynomial degree denoted by p.
Furthermore, the accuracy of the fit is also compared in terms of the Peak Signal to Noise Ratio (PSNR)

$$\text{PSNR} = 20 \log_{10}\left(\frac{255}{\text{MSE}}\right).$$

To validate our models, we also compute the so-called variograms (Bricio Hernandez, 1995). We use them for two scopes: 1) check that the spatial correlation of the extracted data via subsampling is comparable to the one of the original image, 2) check that the spatial correlation of the reconstructed image is comparable to the one of the original image.

To briefly review the variogram theory, let us represent a given random process by the model (Oliver & Webster, 2014)

$$Z(t) = \mu + \epsilon(t)$$

where μ is the mean of the process, $t \in \Omega$ and $\epsilon(t)$ is a random quantity with zero mean and a covariance, $\text{COV}(l)$, given by

$$\text{COV}(l) = \mathrm{E}[\epsilon(\mathrm{t})\epsilon(\mathrm{t}+\mathrm{l})].$$

In these equations l is the separation between samples in both distance and direction and E denotes the expectation. If the mean is not constant, we make the following assumption:

$$\mathrm{E}[Z(t) - Z(t+l)] = 0$$

Then, the covariance is replaced by the semivariance. In this way, we are able to define the so-called variogram whose formula is given by

$$\gamma(l) = \frac{1}{2}\text{VAR}[Z(t) - Z(t+l)].$$

Concerning the experimental variogram, i.e. the one computed via the samples, we use the method of moments (Matheron, 1965) for which

$$\hat{\gamma}(l) = \frac{1}{2m(l)}\sum_{i=1}^{m(l)}[f(t_i) - f(t_i + l)]^2,$$

where $m(l)$ is the number of paired comparisons at lag l. By computing such quantity, we are able to check if the spacial correlation is preserved in the approximation process. Usually, aside the graphical results one tries to fit the empirical variograms with known models. Here we consider one of the most appreciated models in geostatistics, i.e. the spherical fit. Thus, letting $l_1 = ||l||_2$, we define

$$\hat{\gamma}(l_1) = \begin{cases} \alpha_0 + \alpha \left(\frac{3l_1}{2r} - \frac{l_1^3}{2r^3} \right), \; 0 < l_1 \leq r, \\ \alpha_0 + \alpha, \quad l_1 > r, \\ 0, \; l_1 = 0, \end{cases}$$

where the parameters to fit are α_0, the nugget variance, α the spatially correlated variance, and r the range, which is the limit of spatial correlation. Moreover, the quantity $\alpha_0 + \alpha$ estimates the variance of the random process and is known as the *sill*. We also remark that the nugget variance represents the uncorrelated variation at the scale of sampling (Oliver & Webster, 2014).

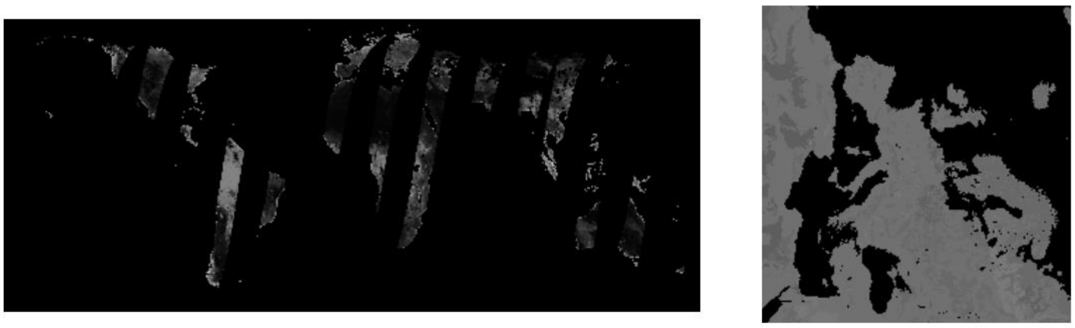

Figure 1: Left: Image taken by SMAP satellite on April 2015. The image size is 1624×3856. Right: Image obtained via the terrestrial system modelling platform (TerrSysMP) satellite on May 2018. The image size is 436×424.

Test with Padova points

At first, we test our technique on these soil moisture data sampled over the whole domain Ω, i.e. on rectangular grids, and then we focus on particular polygonal areas.

Being the domain a rectangle we reduce the grid size taking a few samples at the Padova points. An example of their distribution is plotted in Figure 3. The polynomial interpolation/approximation over those points takes advantage of a very slow growth of the Lebesgue constant. Further, they cluster at the boundary where usually the error is higher. For both images, we fist compress the data via Padova points and then we reconstruct the image for testing the accuracy of our procedure.

After compressing the two images (raw and simulated data, respectively composed by $N = 27504$ and $N = 29998$ pixels) via the Padova points, the two reconstruction schemes are compared in terms of CR and MSE. The results, obtained by varying the polynomial degree and consequently M, are reported in Tables 1 and 2 for respectively raw and simulated images. The CPU times for the raw data needed for the reconstruction algorithms are also shown. We omit further tests in this direction, indeed similar results hold for all the considered tests.

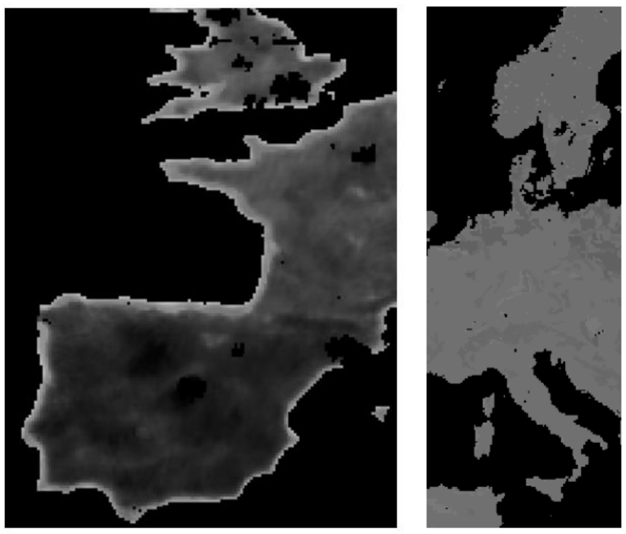

Figure 2: Left: the selected image for tests with SMAP satellite; its size is $N = 191 \times 144$. *Right: the selected image for tests with TerrSysMP; its size is* $N = 106 \times 283$.

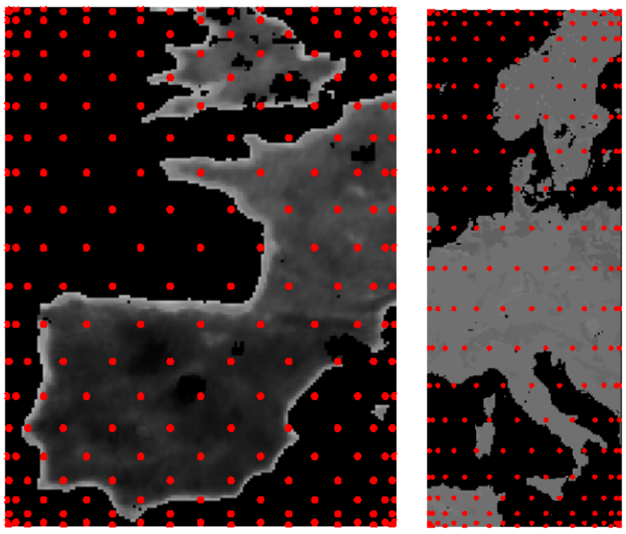

Figure 3: Examples of extracted Padova points.

Method	p	M	$\lfloor CR \rfloor$	PSNR	CPU	MSE
POLSs	20	--	119	66.9	1.07	1.29E-02
VSDKs	--	231	199	71.0	0.98	1.29E-02
POLSs	30	--	55	67.8	4.52	1.05E-02
VSDKs	--	496	55	72.9	1.52	3.26E-03
POLSs	40	--	31	68.7	18.7	8.66E-03
VSDKs	--	861	31	73.2	3.14	3.04E-03
POLSs	50	--	20	75.6	5.02	1.75E-03
VSDKs	--	1326	20	69.8	157	6.72E-03
POLSs	60	--	14	77.0	7.45	1.26E-03
VSDKs	--	1891	14	70.3	379	6.00E-03
POLSs	70	--	10	77.9	10.6	1.04E-03
VSDKs	--	2556	10	69.3	57.3	7.57E-03
POLSs	80	--	8	70.3	904	5.49E-03
VSDKs	--	3321	8	78.8	15.5	8.52E-04
POLSs	90	--	6	71.0	2156	5.10E-03
VSDKs	--	4186	6	79.5	19.8	7.27E-04

Table 1: Results for the compression algorithms used for the raw data image.

Method	p	M	$\lfloor CR \rfloor$	PSNR	MSE
POLSs	20	--	129	67.4	1.17E-02
VSDKs	--	231	129	85.2	1.92E-04
POLSs	30	--	60	69.0	8.10E-03
VSDKs	--	496	60	85.4	1.86E-04
POLSs	40	--	34	69.8	6.68E-03
VSDKs	--	861	34	85.6	1.76E-04
POLSs	50	--	22	70.5	5.70E-03
VSDKs	--	1326	22	86.2	1.53E-04
POLSs	60	--	5	71.0	5.06E-03
VSDKs	--	1891	15	86.7	1.37E-04
POLSs	70	--	11	71.4	4.66E-03
VSDKs	--	2556	11	87.0	1.28E-04
POLSs	80	--	9	71.7	4.31E-03
VSDKs	--	3321	9	87.1	1.23E-04
POLSs	90	--	6	72.0	4.08E-03
VSDKs	--	4186	6	87.4	1.17E-04

Table 2: Results for the compression algorithms used for the simulated data image.

As evident from Tables 1 and 2, the compression via Padova points is particularly effective. Nevertheless, the reconstruction via POLSs suffers from the Gibbs phenomenon. Kernels in a varying scale setting show their robustness in dealing with the Gibbs phenomenon.
To have graphical feedback, we report several reconstruction results in Figures 4 and 5. We can graphically note that when M grows while VSDKs are truly performing, the POLSs scheme suffers from the Gibbs phenomenon.

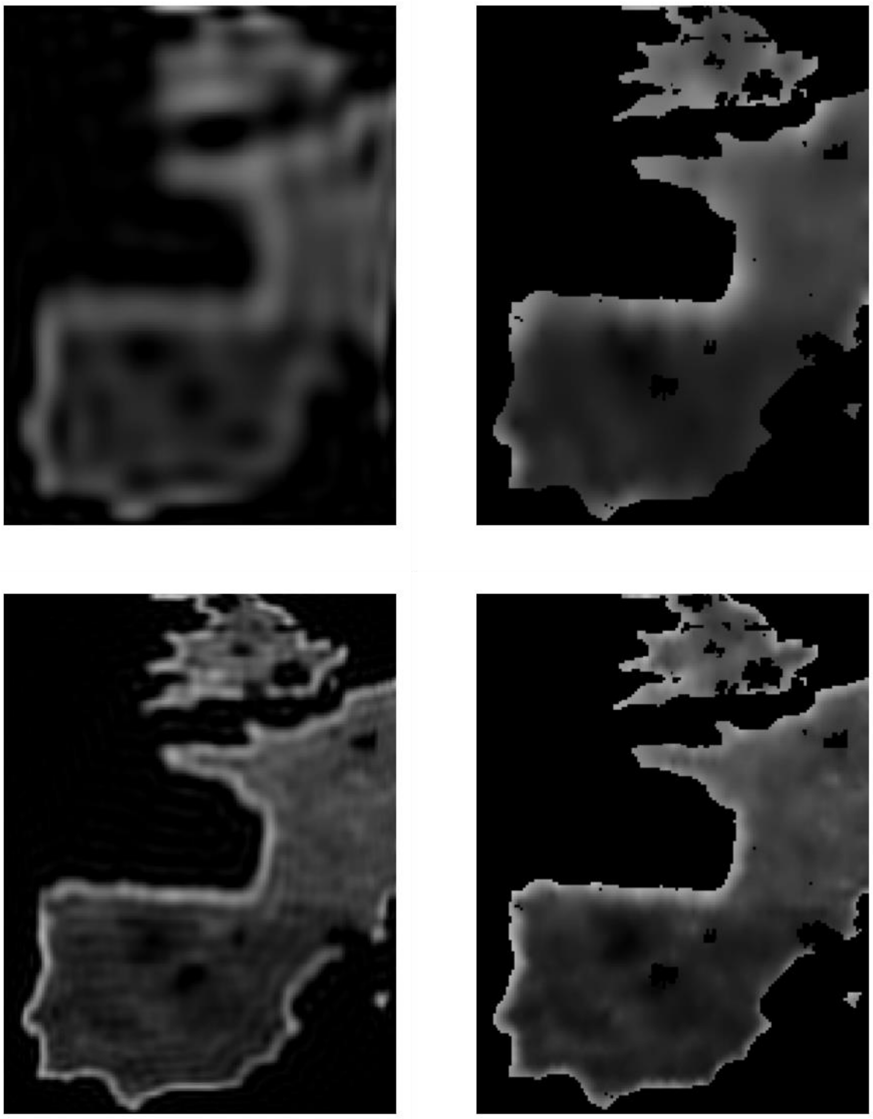

Figure 4: Reconstruction of the raw data image via POLSs and VSDKs interpolation (left and right, respectively) for $p = 30$ and 90 (top and bottom respectively).

Finally, in Figure 6 we show the empirical variograms for the raw data image. They are computed via the Python package PyKrige freely available on the Github repository at https://github.com/bsmurphy/PyKrige. We should note from the two frames on the top of Figure 6 that the empirical variograms computed on a few extracted Padova points (with the known values at those points) are similar to the one computed for the original image. This is a confirmation about the fact that approximating on Padova points turns out to be reliable. However, the variogram on the extracted data shows a moderate hole-effect. This is due to the particular distribution of Padova points that cluster at the boundary. Nevertheless, since the error at the boundary is usually higher, this is certainly not a drawback. Concerning the other variograms, they are computed on the reconstructed grids in the same framework of Figures 4. As expected, while VSDKs maintain the spatial correlation, POLSs show their difficulties.

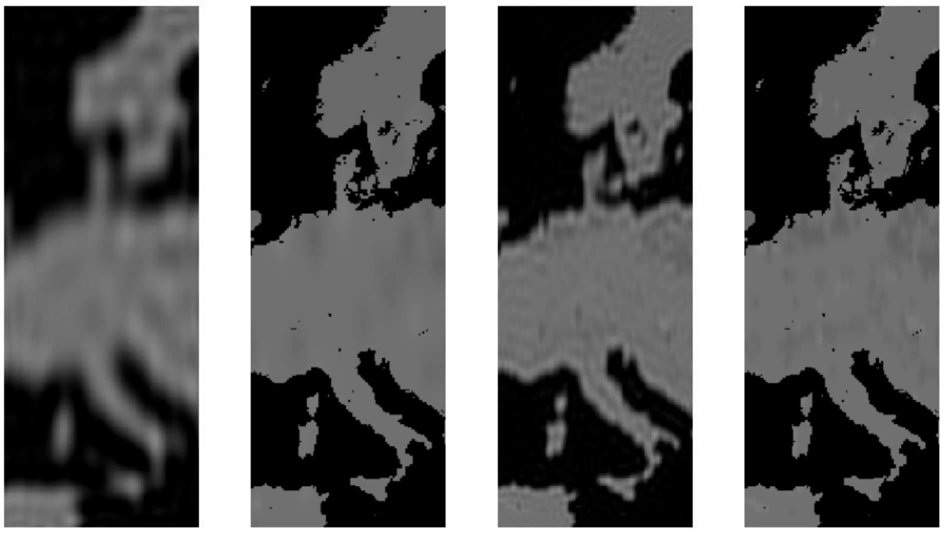

Figure 5: Reconstruction of the simulated data image via POLSs and VSDKs interpolation (left and right, respectively) for $p = 30$ and 90 (top and bottom respectively).

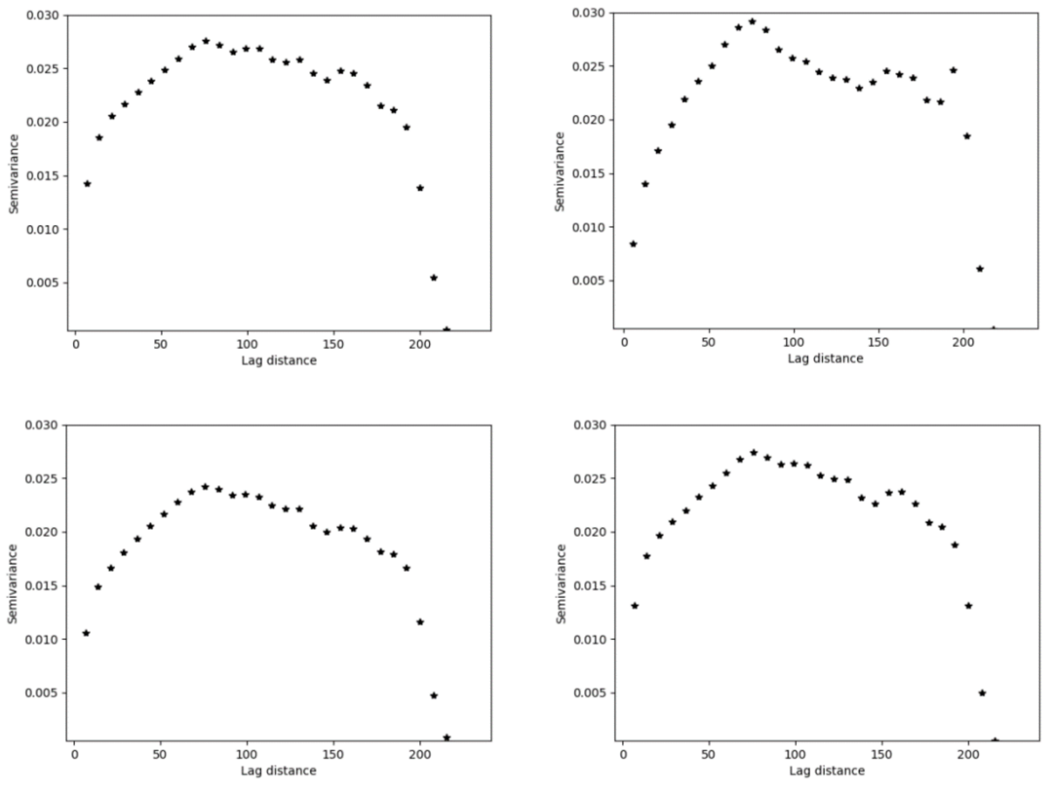

Figure 6: This figure refers to the raw data image. From left to right, top to bottom. 1) The variogram of the original image. 2) The variogram on a few extracted Padova points for $p = 90$ (with the known values at those points). 3) The variogram of the reconstructed image with POLSs ($p = 90$). 4) The variogram of the reconstructed image with VSDKs ($p = 90$).

Test with Caratheodory-Tchakaloff

Since it might be of interest focusing the attention on particular areas and/or (partially) removing points lying on the sea or on masked regions, extracting points on polygons is meaningful. The reduced nodes are thus extracted via the CATCH scheme. Tests are carried out via the images plotted in Figure 7.

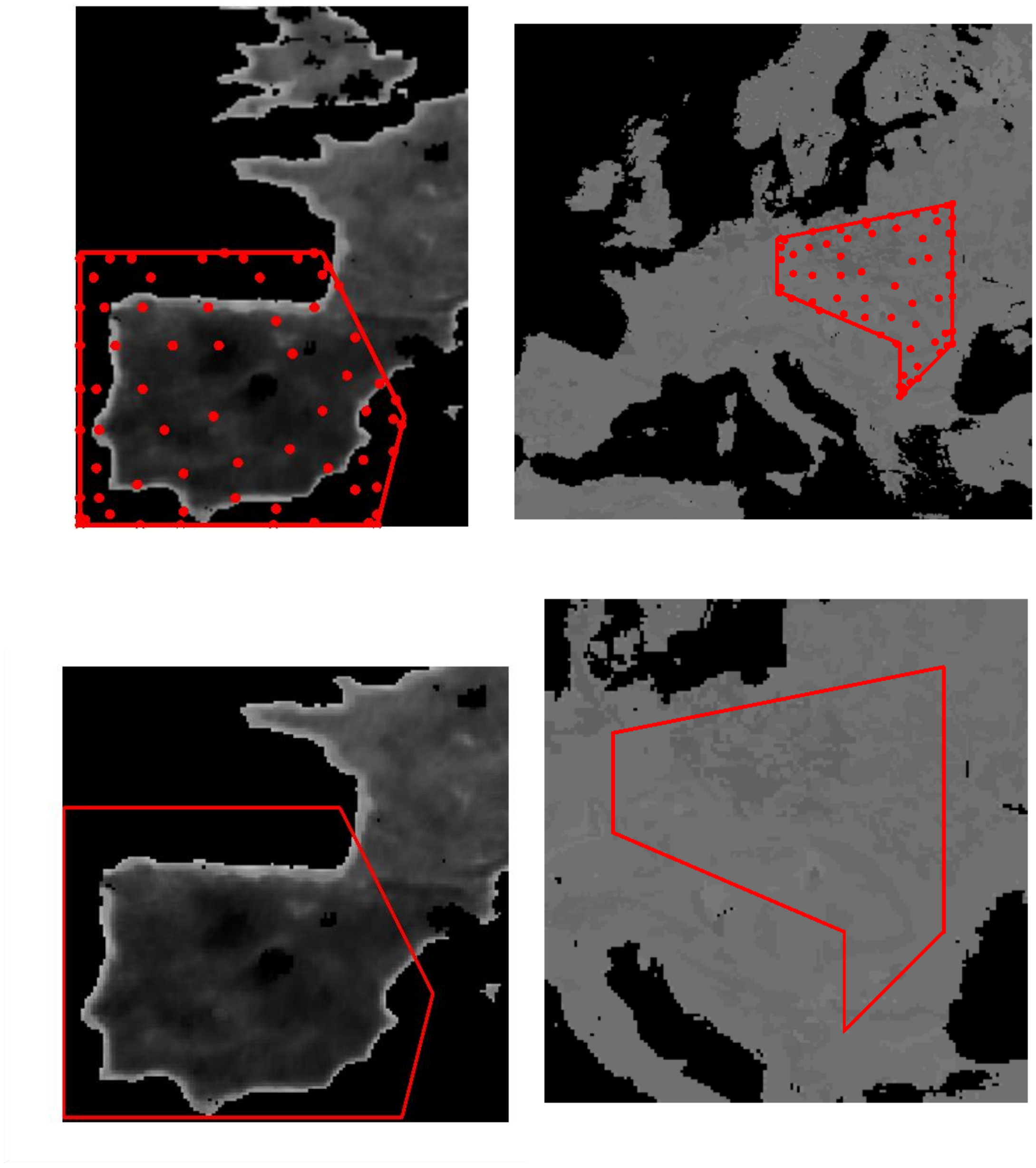

Figure 7: Top Left: the selected polygonal image for tests with SMAP satellite; its size is $N = 10880$ Right: the selected polygonal image for tests with TerrSysMP; its size is $N = 6440$. Examples of Caratheodory-Tchakaloff points are plotted in red. Bottom: zoom of the two figures.

The results of the compression and reconstruction via both POLSs and VSDKs are reported in Tables 3 and 4. We note that in this case the two methods apparently behave similarly. However, VSDKs are not completely suitable for the simulated data type. In this case, being the field to reconstruct smooth, POLSs outperform VSDKs.

Method	p	M	$\lvert CR \rvert$	MSE
POLSs	10	--	164	1.48E-02
VSDKs	--	66	164	8.03E-03
POLSs	15	--	80	1.27E-02
VSDKs	--	136	80	4.44E-03
POLSs	20	--	47	1.08E-02
VSDKs	--	231	47	4.59E-03
POLSs	25	--	30	9.60E-03
VSDKs	--	351	30	2.93E-03
POLSs	30	--	21	8.96E-03
VSDKs	--	496	21	2.41E-03
POLSs	35	--	16	7.99E-03
VSDKs	--	666	16	1.35E-03
POLSs	40	--	12	7.83E-03
VSDKs	--	861	12	1.16E-03
POLSs	45	--	10	7.60E-03
VSDKs	--	1080	10	9.75E-04

Table 3: Results for the compression algorithms used for the raw data polygonal image.

Method	p	M	$\lvert CR \rvert$	MSE
POLSs	10	--	97	3.21E-04
VSDKs	--	66	97	3.80E-04
POLSs	15	--	47	2.93E-04
VSDKs	--	136	47	3.99E-04
POLSs	20	--	27	2.82E-04
VSDKs	--	231	27	2.93E-04
POLSs	25	--	18	2.54E-04
VSDKs	--	351	18	3.25E-04
POLSs	30	--	13	2.28E-04
VSDKs	--	495	13	2.88E-04
POLSs	35	--	9	2.27E-04
VSDKs	--	662	9	2.75E-04
POLSs	40	--	7	2.17E-04
VSDKs	--	854	7	2.68E-04
POLSs	45	--	6	2.29E-04
VSDKs	--	1059	6	2.52E-04

Table 4: Results for the compression algorithms used for the simulated data polygonal image.

To have a graphical feedback, in Figures 8 and 9 we plot the reconstruction of the two grids for different polynomial degrees. For the raw data grid, we can note that the Gibbs phenomenon is evident for large number of data sites. Such effect is mitigated via VSDKs. For the simulated data image instead, smooth approximations via POLSs are preferable.

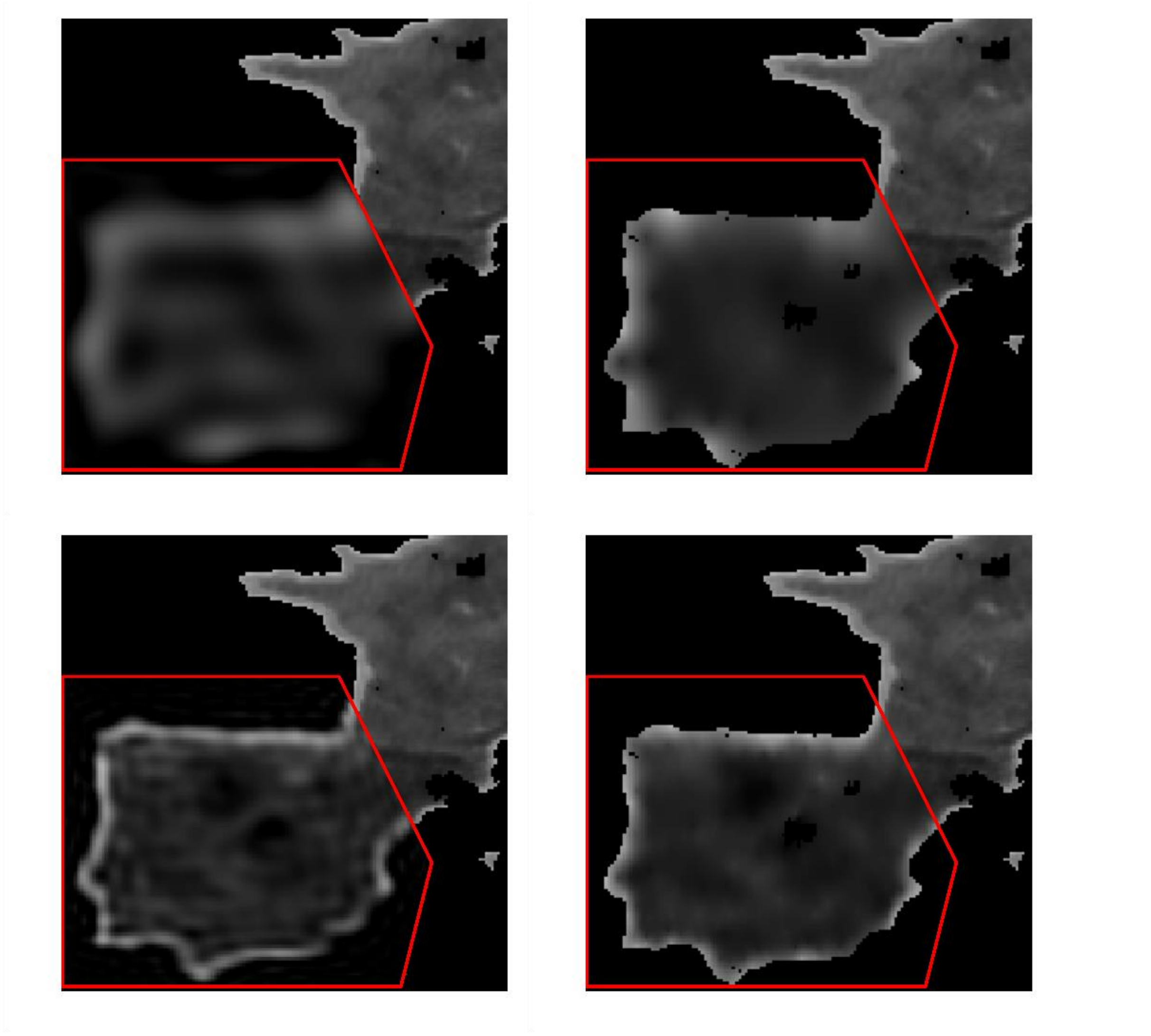

Figure 8: Reconstruction of the raw data polygonal image via POLSs and VSDKs interpolation (left and right, respectively) for $p = 15$ and 45 (top and bottom respectively).

Finally, in Figures 10 and 11 we show the empirical variograms. We should note from the two frames on the top of Figures Figures 10 and 11 that the empirical variograms computed on a few extracted Caratheodory-Tchakaloff points (with the known values at those points) are similar to the one computed for the original image, even if we register a modest loss in the space correlation. Concerning the other variograms, they are computed on the reconstructed images in the same framework of Figures 8 and 9. As expected, both POLSs and VSDKs maintain the spatial correlation.

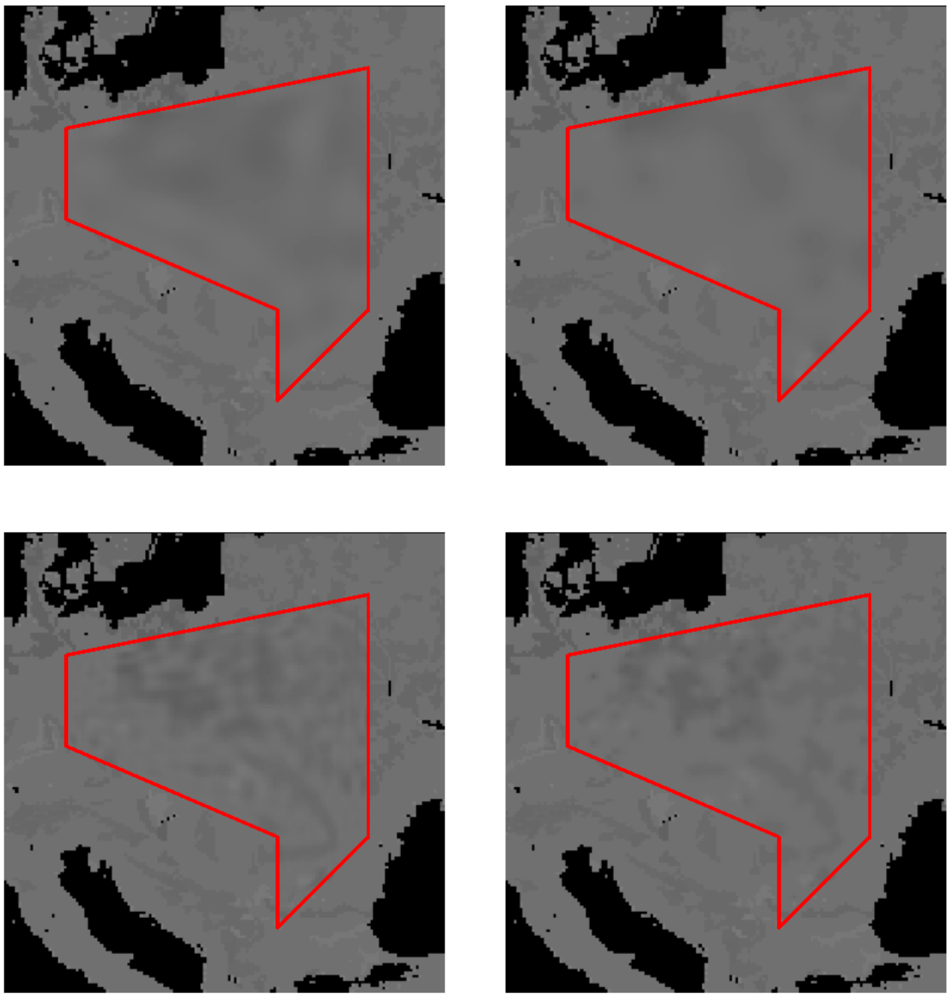

Figure 9: Reconstruction of the simulated data polygonal image via POLSs and VSDKs interpolation (left and right, respectively) for $p = 15$ and 45 (top and bottom respectively).

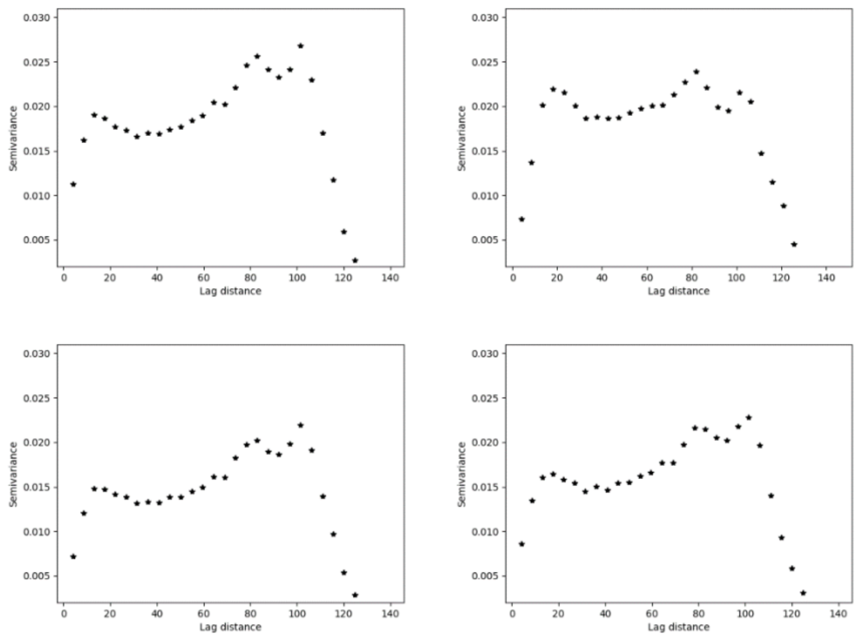

Figure 10: his figure refers to the polygonal raw data image. From left to right, top to bottom. 1) The variogram of the original image. 2) The variogram on a few extracted Caratheodory-Tchakaloff points for p=45 (with the known values at those points). 3) The variogram of the reconstructed image with POLSs (p=45). 4) The variogram of the reconstructed image with VSDKs (p=45).

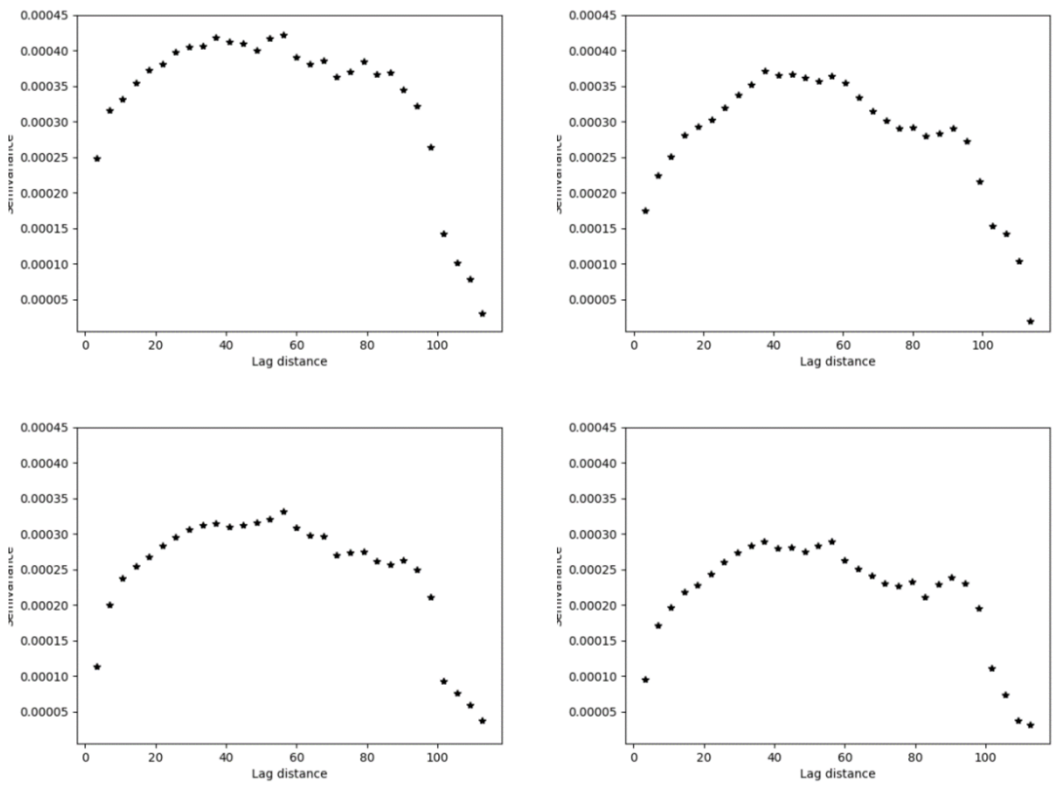

Figure 10: his figure refers to the polygonal raw data image. From left to right, top to bottom. 1) The variogram of the original image. 2) The variogram on a few extracted Caratheodory-Tchakaloff points for p=45 (with the known values at those points). 3) The variogram of the reconstructed image with POLSs (p=45). 4) The variogram of the reconstructed image with VSDKs (p=45).

Simulations for time predictions

In this section we point out how we can produce efficient prediction on the dynamics of the considered quantities. The following soil moisture products are processed by the ParFlow system. We take 91 data takes sampled at different time steps (1h) between 10/04/19 and 12/04/19. During that period a perturbation passed over Greece and thus we focus on that area. The training set consists of the first 90 images and pixel by pixel one could perform both SVR and EnKF. But taking all pixels would be truly inefficient. Thus, we only take into account 1891 Padova points and we reconstruct the final figure via VSDKs.

The original image at the 91st time step and related variogram are plotted in Figure 11.
The graphical results of our prediction via both SVR and EnKF are displayed in Figure 12 and for both methods the MSE is about $2E-03$.

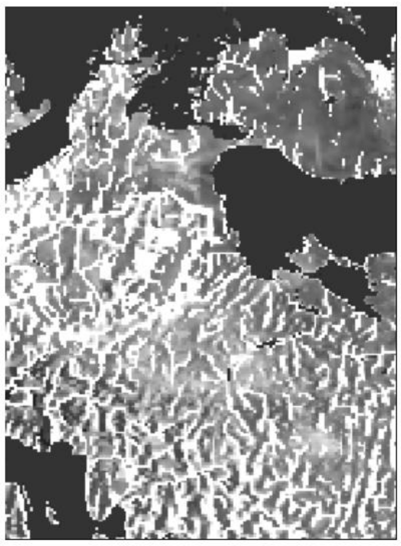

Figure 11: The original image whose size is 206×151.

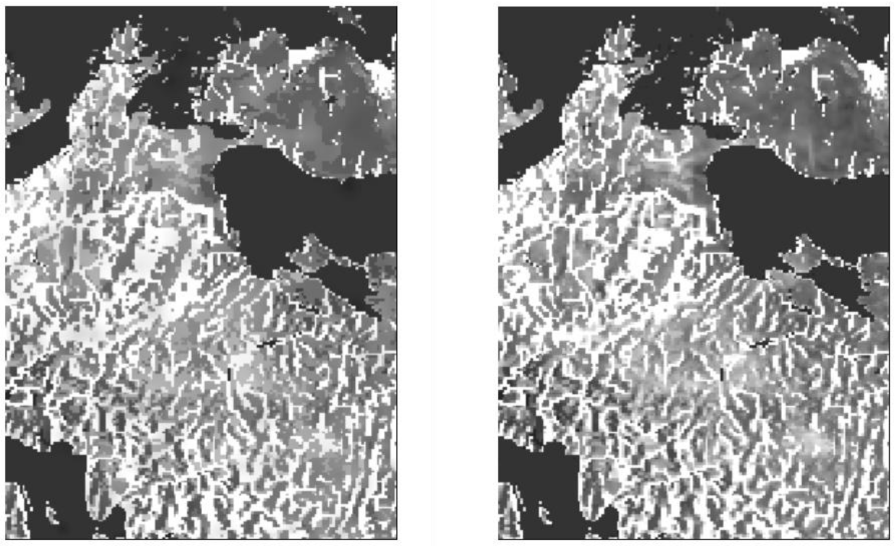

Figure 12: The image predicted via SVR and EnKF (top, left and right respectively).

Conclusions

In this work we presented an effective tool for data compression. Points are extracted via approximation techniques which gained much attention in the last years. Because of the specific application considered here, i.e. approximating satellite data, aside POLSs a robust reconstruction scheme, based on VSDKs which take advantage of reducing the Gibbs phenomenon, has been extensively studied and tested. Numerical results are promising and show that, provided the field is smooth, POLSs are preferable. Otherwise switching to VSDKs is preferable. Finally, we also provided promising examples devoted to investigate the dynamics of the considered quantities. Work in progress consists in investigating data fusion algorithms for effectively extract features from the given data products.

Appendix: Error bounds

To introduce later error bounds for VSDKs, we start with standard kernels by defining the space (Wendland, 2005)

$$H_K \;=\; \mathrm{span}\{K(\cdot,t), \quad t \in \Omega\},$$

with an associated bilinear form $(\cdot,\cdot)_H$ that makes H_K an inner product space with reproducing kernel K; see (Wendland, 2005). The native space N_K of the kernel K is then defined as the completion of H_K with respect to the norm $||\cdot||_H \;=\; \sqrt{(\cdot,\cdot)_H}$.

The power function P_{K,T_M}will help us to find an upper bound for the interpolation error. Setting

$$\kappa^T(t) \;=\; \big(K(t,t_1),\dots,K(t,t_M)\big),$$

we have (Fasshauer & McCourt, 2015)

$$P_{K,T_M} \;=\; ||K(\cdot,t) - \kappa^T(t)A^{-1}\,\kappa(t)||_{N_K} = \; K(t,t) - \kappa^T(t)A^{-1}\,\kappa(t),$$

where the last equality is a consequence of the *reproducing property* (Wendland, 2005), i.e.

$$\big(f, K(\cdot,t)\big)_{N_K} \;=\; f(t), \quad f \in \; N_K.$$

Finally, we need to introduce the so-called *fill distance*, see (Fasshauer, 2007)

$$h_{T_M} = \sup_{t\in\Omega}\min_{t_k\in T_M} ||\mathrm{t}-\mathrm{t_k}||_2.$$

It is an indicator of how Ω is filled out by points and is related to error bounds (refer e.g. to Theorem 14.5 in (Fasshauer, 2007) p. 121).

We now provide error bounds for VSDKs in terms of the power function P_{K_ψ,T_M} and fill distance; see (De Marchi, et al., 2019). For further details on the native spaces induced by the discontinuous kernels we refer the reader to (De Marchi, et al., 2019). In that paper, an error bound (more strict than the one reported here for Sobolev kernels) is shown. However, due to its technicality, we decide to propose the following and more general one that in particular offers a computable error bound. Refer to also to (Fasshauer & McCourt, 2015) for further details.

Theorem 2. Assume that there exists a constant δ (not *too large)* so that $||f - V_f||_{N_{K_\psi}} \leq \delta||V_f||_{N_{K_\psi}}$. Let $K \in C^{2k}((\Omega \times \Sigma) \times (\Omega \times \Sigma))$ be a strictly positive definite kernel. Then there exist positive constants h_0 and C, such that

$$\left| f(t) - V_f(t)\right| \leq \delta\, C\, h_{\hat{T}_M}^k \sqrt{C_K(\hat{t})\, f^t A_\psi^{-1}\, f},$$

provided that $0 < h_{\hat{T}_M} \leq h_0$ and $f \in N_{K_\psi}$ where

$$C_K(\hat{t}) \max_{|\beta|=2k} \max_{\hat{v},\hat{w}\in(\Omega\times\Sigma)\cap B\left(\hat{t},C_2 h_{\hat{T}_M}^k\right)} |D_2^\beta K(\hat{v},\hat{w})|,$$

with C_2 from Theorem 14.4 (Fasshauer, 2007) p. 120, and where $B\left(\hat{t}, C_2 h_{\hat{T}_M}^k\right)$ denotes the ball of radius $C_2 h_{\hat{T}_M}^k$ centered at $\hat{t}$.

Proof. In the varying scale setting, the coefficients $c = A_\psi^{-1}\, f$ are so that

$$V_f(t) = \kappa_\psi^T(t) A_\psi^{-1}\, f,$$

or, equivalently, because of the symmetry of the kernel matrix and because trivially

$$V_f(t) = f^T A_\psi^{-1}\, \kappa_\psi(t), \tag{1}$$

Because of the reproducing property, we have that

$$\begin{aligned} f^T &= \left(\left(f, K_\psi(\cdot,t_1)\right)_{N_{K_\psi}}, \dots, \left(f, K_\psi(\cdot, t_{1M})\right)_{N_{K_\psi}}\right), \\ &= \left(\left(f, (K_\psi(\cdot,t_1),\dots,K_\psi(\cdot,t_M)\,)\right)\right)_{N_{K_\psi}} = \left(f, \kappa_\psi^T(\cdot)\right)_{N_{K_\psi}}. \end{aligned}$$

By plugging this into (1), we obtain

$$V_f(t) = \left(f, \kappa_\psi^T(\cdot)\right)_{N_{K_\psi}} A_\psi^{-1}\, \kappa_\psi(t) = \left(f, \kappa_\psi^T(\cdot) A_\psi^{-1}\, \kappa_\psi(t)\right)_{N_{K_\psi}}.$$

Thus,

$$\begin{aligned} \left| f(t) - V_f(t)\right| &\leq ||f||_{N_{K_\psi}} \left|\left|K_\psi(\cdot,t) - \kappa_\psi^T(\cdot) A_\psi^{-1}\, \kappa_\psi(t)\right|\right|_{N_{K_\psi}}, \\ &= ||f||_{N_{K_\psi}} \left|\left|K(\cdot,\hat{t}) - \kappa^T(\cdot) A_\psi^{-1}\, \kappa(\hat{t})\right|\right|_{N_K} \\ &= ||f||_{N_{K_\psi}} P_{K,\hat{T}_M}(\hat{t}). \end{aligned}$$

Then, as well-known, see also Th. 14.5 in (Fasshauer, 2007)

$$P_{K,\hat{T}_M}(\hat{t}) \le C\, h^k_{\hat{T}_M} \sqrt{C_K(\hat{t})}\,.$$

And finally,

$$\left| f(t) - V_f(t) \right| \le \delta\, C\, h^k_{\hat{T}_M} \sqrt{C_K(\hat{t})\, f^t A_\psi^{-1}\, f}.$$

■

Of course, we have that $h^k_{T_M} \le h^k_{\hat{T}_M}$ and this a drawback for the VSK setting, indeed the error decreases according to the fill distance, i.e. according to the number of points if they are quasi-uniform. However, we have an improvement in terms of stability, which is meaningful for our purposes of reducing oscillations in the solution. Indeed, also the separation distance, which decreases according to the smallest eigenvalue of the kernel matrix, never decreases in the VSK setting.

Appendix: Kalman Filter

To introduce the Ensemble Kalman Filter (EnKF), we need to recall the basic features of the Extended Kalman Filter. Let us take a discrete-time nonlinear system with dynamics

$$s_{k+1} = f(s_k, u_k) + w_k,$$

and measurements

$$l_k = g(s_k) + v_k,$$

where in general, $s_k, w_k \in \mathbb{R}^d$, $u_k \in \mathbb{R}^p$, $l_k, v_k \in \mathbb{R}^q$. We assume that w_k and v_k are stationary zero-mean white noise processes with covariance matrices Q_k and Z_k, respectively. Furthermore, let s_0, w_k and v_k be uncorrelated. The scope is to construct estimates s_k^a of the state s_k using the measurements so that

$$\mathrm{tr}(\mathrm{E}[\delta_k^a (\delta_k^a)^t]),$$

is minimized, where $\delta_k^a = s_k - s_k^a$.
When the dynamics is linear, i.e.

$$f(s_k, u_k) = B_k s_k + C_k u_k.$$

$$g(s_k) = D_k s_k.$$

we define the analysis state error covariance $P_k^a \in \mathbb{R}^{d\times d}$ as $P_k^a = \mathrm{E}[\delta_k^a (\delta_k^a)^T]$. Furthermore, we introduce the forecast state error covariance $P_k^f \in \mathbb{R}^{d\times d}$, defined by

$$P_k^f = = \mathrm{E}\left[\delta_k^f \left(\delta_k^f\right)^T\right],$$

and

$$P^f_{s,l_k} = \mathrm{E}\left[\delta^f_k (l_k - l^f_k)^T\right] = P^f_k D^T_k, \qquad P^f_{l,l_k} = \mathrm{E}\left[(l_k - l^f_k)(l_k - l^f_k)^T\right] = D_k P^f_k D^T_k + Z_k,$$

where $l^f_k = Ds^f_k, \delta^f_k = l_k - l^f_k$. Then, the Kalman filter iterations can be summarized in the following two steps:

1. Analysis step:

$$K_k = P^f_{s,l_k}\left(P^f_{l,l_k}\right)^{-1}, \qquad P^a_k = (I - K_k D_k)P^k_f, \qquad s^a_k = s^f_k + K_k\left(l_k - D_k s^f_k\right).$$

2. Forecast step:

$$s^f_{k+1} = B_k s^a_k + C_k u_k, \qquad P^f_{k+1} = B_k P^a_k B^T_k + Q_k.$$

In case the dynamics in nonlinear, we drive our attention towards the Extended Kalman Filter (EKF), where in the forecast step:

$$s^f_{k+1} = f(s^a_k, u_k), \qquad P^f_{k+1} = B_k P^a_k B^T_k + Q_k,$$

and for the data assimilation we have:

$$s^a_k = s^f_k + K_k\left(l_k - g\left(s^f_k\right)\right), \qquad K_k = P^f_k D^T_k\left(D_k P^f_k D^T_k + Z_k\right)^{-1},$$
$$P^a_k = P^f_k - P^f_k D^T_k\left(D_k P^f_k D^T_k + Z_k\right)^{-1} D_k P^f_k,$$

where $B_k \in \mathbb{R}^{d\times d}$ and $D_k \in \mathbb{R}^{q\times d}$ are given by

$$B_k = \frac{\partial f(s,u)}{\partial s}\Big|_{s=s^a_k}, \qquad D_k = \frac{\partial g(s)}{\partial s}\Big|_{s=s^a_k}.$$

While when the dynamics is linear the Kalman filter produces optimal estimates of the state, the EnKF for non-linear model is a suboptimal estimator, where the statistical errors are predicted by producing an ensemble $S^f_k = \left(s^{f_1}_k, \dots, s^{f_r}_k\right)^T$ and f_i refers to the i –th forecast ensemble member. Then, we define the ensemble mean $\bar{s}^f_k \in \mathbb{R}^d$ as

$$\bar{s}^f_k = \frac{1}{r}\sum_{i=1}^{r} s^{f_i}_k.$$

To approximate the state s_k, we first introduce the ensemble error matrix $S^f_k \in \mathbb{R}^{d\times r}$

$$S^f_k = \left[t^{f_1}_k - \bar{t}^f_k, \dots, t^{f_r}_k - \bar{t}^f_k\right],$$

and the ensemble of output error $S^a_{l_k} \in \mathbb{R}^{d\times r}$

$$S_{l_k}^{a} = [l_k^{f_1} - \bar{l}_k^f, \dots, l_k^{f_r} - \bar{l}_k^f].$$

Taking into account the notation previously introduced we approximate P_k^f by $\hat{P}_k^f$, P_{t,l_k}^f by $\hat{P}_{t,l_k}^f$ and P_{l,l_k}^f by $\hat{P}_{l,l_k}^f$, with

$$\hat{P}_k^f = \frac{1}{r-1} S_k^f \left(S_k^f\right)^T, \qquad \hat{P}_{t,l_k}^f = \frac{1}{r-1} S_k^f \left(S_{l_k}^f\right)^T, \qquad \hat{P}_{l,l_k}^f = \frac{1}{r-1} S_{l_k}^f \left(S_{l_k}^f\right)^T.$$

Therefore, the spread of the ensemble members around the mean is the error between best estimate and actual state, while we can see the ensemble mean as the best forecast estimate of the state. Then, for each $i = 1, \dots, r$, we define

$$s_k^{a_i} = s_k^{f_i} + \hat{K}\left(l_k^i - g\left(s_k^{f_i}\right)\right),$$

and the perturbed observations $l_k^i = l_k + v_k^i$, where v_k^i is a zero mean random variable with normal distribution and covariance Z_k. Then, letting

$$\bar{s}_k^a = \frac{1}{r} \sum_{i=1}^{r} s_k^{a_i}.$$

the analysis error covariance P_k^a is approximated by

$$\hat{P}_k^a = \frac{1}{r-1} S_k^a (S_k^a)^T, \quad \text{with} \quad S_k^a = \left[r_k^{a_1} - \bar{r}_k^a, \dots, r_k^{a_r} - \bar{s}_k^a\right].$$

Finally, in agreement with the linear Kalman filter, we get $s_{k+1}^{f_i} = f\left(s_k^{a_i}, u_k\right) + w_k^i$, where w_k^i are from a normal distribution with zero average and covariance Q_k.

After introducing $\hat{K}_k = \hat{P}_{s,l_k}^f \left(\hat{P}_{l,l_k}^f\right)^{-1}$, we summarize the two main steps as follows:

1. Analysis step:

$$s_k^{a_i} = s_k^{f_i} + K_k\left(l_k + v_k^i - g\left(s_k^{f_i}\right)\right).$$

2. Forecast step:

$$s_{k+1}^{f_i} = f\left(s_k^{a_i} + v_k^i\right).$$

In our case, the model is constructed via POLSs or VSDKs on the reduced number of basis, as explained before. Then, we apply the EnKF on this reduced number of points. In this way, we are able to give reliable and efficient previsions on the future dynamics.

References

Aminian Shahrokhabadi, M., Neisy, A., Perracchione, E. & Polato, M., 2019. Learning with subsampled kernel-based methods: Environmental and financial applications. *Dolomites Res. Notes Approx.*, pp. 12:17-27.

Bos, L. et al., 2006. Bivariate Lagrange interpolation at the Padua points: the generating curve approach. *J. Approx. Theory*, pp. 143:15-25..

Bos, L., De Marchi, S. & Vianello, M., 2017. Polynomial approximation on Lissajous curves in the d-cube. *Appl. Numer. Math.*, pp. 116:47--56.

Bozzini, M., Lenarduzzi, L., Rossini, M. & Schaback, R., 2015. Interpolation with variably scaled kernels. *IMA J. Numer. Anal.*, pp. 35:199-219..

Bricio Hernandez, D., 1995. *Lectures on Probability and Second Order Random Fields.* s.l.:World Scientific.

Caratheodory, C., 1911. Uber den Variabilittsbereich der Fourierschen Konstanten von positiven harmonischen Funktionen. *Rend. Circ. Mat. Palermo*, pp. 32:193-217.

Cortes, C. & Vladimir, V. N., 1995. Support-vector networks. *Machine Learning*, pp. 20:273-297.

De Marchi, S., Erb, W. & Marchetti, F., 2017. Spectral filtering for the reduction of the Gibbs phenomenon for polynomial approximation methods on Lissajous curves with applications in MPI. *Dolomites Res. Notes Approx.*, p. 10.

De Marchi, S. et al., 2019. Shape-Driven Interpolation with Discontinuous Kernels: Error Analysis, Edge Extraction and Applications in MPI. *Preprint.*

De Marchi, S., Marchetti, F. & Perracchione, E., 2019. Jumping with Variably Scaled Discontinuous Kernels (VSDKs). *Preprint.*

Entekhabi at al., D., 2014. *SMAP Handbook-Soil Moisture Active Passive.* s.l.:JPL Publication; Pasadena, CA, 2014.

Erb, W., 2016. Bivariate Lagrange interpolation at the node points of Lissajous. *Appl. Math. Comput.* , pp. 289:409-425..

Fasshauer, G. E., 2007. *Meshfree Approximations Methods with Matlab.* s.l.:World Scientific, Singapore.

Fasshauer, G. E. & McCourt, M. J., 2015. *Kernel-based Approximation Methods Using Matlab.* s.l.:World Scientific, Singapore.

Fornberg, B. & Flyer, N., 2008. *The Gibbs Phenomenon for radial basis functions.* Potsdam, Jerri, A.J., pp. 201-224..

Gillijns, S. et al., 2006. What is the ensemble Kalman filter and how well does it work?. *American Control Conference*, pp. 1-6.

Gottlieb, D. & Shu, C. W., 1997. On the Gibbs phenomenon and its resolution. *SIAM Review*, pp. 39:644-668.

Johns, C. J. & Mandel, J., 2008. A Two-Stage Ensemble Kalman Filter for Smooth Data Assimilation. *Environmental and Ecological Statistics*, pp. 15: 101-110.

Jung, J. H., 2007. A note on the Gibbs phenomenon with multiquadric radial basis functions. *Appl. Num. Math.*, pp. 57:213-219..

Kalman, R. E., 1960. A new approach to linear filtering and prediction problems. *Transactions of the ASME – Journal of Basic Engineering. Series D.*, pp. 82:35-45.

Kollet, S. & Maxwell, R. M., 2008. Capturing the influence of groundwater dynamics on land surface processes using an integrated, distributed watershed model. *Water Resour. Res.*, p. 44:W02402.

Matheron, G., 1965. Les variables régionalisées et leur estimation. *Masson, Paris.*

McCallum, I. et al., 2019. Developing food, water and energy nexus workflows. *Int. J. Digit. Earth, DOI: 10.1080/17538947.2019.1626921*.

Oliver, M. A. & Webster, R., 2014. A tutorial guide to geostatistics: Computing and modelling variograms and kriging. *Catena* , pp. 113:56-69..

Perracchione, E. et al., 2019. Modelling and processing services and tools. p. GEOEssential Deliverable 1.3.

Piazzon, F., Sommariva, A. & Vianello, M., 2017 . Caratheodory-Tchakaloff Subsampling. *Dolomites Res. Notes Approx.* , pp. 5-14.

Piazzon, F., Sommariva, A. & Vianello, M., 2017. Caratheodory-Tchakaloff Least Squares. *Sampling Theory and Applications 2017, IEEE Xplore Digital Library*, p. 12017.

Shawe-Taylor, J. & Cristianini, N., 2004. *Kernel Methods for Pattern Analysis.* New York, NY, USA: Cambridge University Press.

Shrestha, P. et al., 2014. A scale-consistent terrestrial systems modeling platform based on COSMO, CLM, and ParFlow. *Mon. Weather Rev.* , pp. 142:3466-3483.

Sommariva, A. & Vianello, M., 2017. Nearly optimal nested sensors location for polynomial regression on complex geometries. *Sampl. Theory Signal Image Process.*, pp. 17:95-101.

Wendland, H., 2005. *Scattered Data Approximation.* s.l.:Cambridge Monogr. Appl. Comput. Math., vol. 17, Cambridge Univ. Press, Cambridge, 2005..