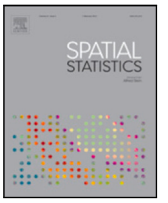
Spatial Statistics 60 (2024) 100822

Contents lists available at ScienceDirect

Spatial Statistics

journal homepage: www.elsevier.com/locate/spasta



Graph convolutional networks for spatial interpolation of correlated data

Marianne Abémgnigni Njifon [*,1], Dominic Schuhmacher

Institute for Mathematical Stochastics, University of Göttingen, Germany

ARTICLE INFO

Keywords:
Kriging
Graph convolution networks
Spatial correlation
Spatial interpolation
Mean square error

ABSTRACT

Several deep learning methods for spatial data have been developed that report good performance in a big data setting. These methods typically require the choice of an appropriate kernel and some tuning of hyperparameters, which are contributing reasons for poor performance on smaller data sets.

In this paper, we propose a mathematical construction of a graph-based neural network for spatial prediction that substantially generalizes the KCN model in [Appleby, Liu and Liu (2020). Kriging convolutional networks. In *Proc. AAAI Conf. AI* 34, pp. 3187–3194]. In particular, our model, referred to as SPONGE, allows for integrated learning of the convolutional kernel, admits higher order neighborhood structures and can make use of the distance between locations in the neighborhood and between labels of neighboring nodes. All of this yields higher flexibility in capturing spatial correlations.

We investigate in simulation studies including small, medium and (reasonably) large data sets in what situations and to what extent SPONGE comes close to or (if the conditions for optimality are violated) even beats universal Kriging, whose predictions incur a high computational cost if n is large. Furthermore we study the improvement for general SPONGE in comparison with the usual KCN.

Finally, we compare various graph-based neural network models on larger real world data sets and apply our method to the prediction of soil organic carbon in the southern part of Malawi.

1. Introduction

The spatial interpolation of data using covariates has numerous applications in all fields where data can be collected at various locations and is relevant for such diverse topics as canopy height prediction in forests, local geoid modeling, property prediction for materials, modeling of molecular fingerprints, surface data quality assessment, and prediction of soil pollution, to cite just a few (Akcin and Celik, 2013; Duvenaud et al., 2015; Xie and Grossman, 2018; Wang et al., 2019; Khouni et al., 2021; Gao et al., 2022).

Regardless of the field of application, the statistical discipline concerned with predictions based on observations in continuous space is often referred to as geostatistics (Sagar et al., 2018).

Suppose at spatial locations $s_i \in S \subset \mathbb{R}^d$ we observe data pairs (X_i, Y_i), $i = 1, \dots, n$, where $X_i \in \mathbb{R}^q$ denotes covariate information and $Y_i \in \mathbb{R}$ is the target quantity to be interpolated. From a neural network point of view we refer to these quantities also as *features* and *labels*, respectively (Bishop, 2006; Deisenroth et al., 2021). The goal is to predict Y_0 at a new location s_0 given the

* Corresponding author.
E-mail address: m.abemgnigninjifon@math.uni-goettingen.de (M. Abémgnigni Njifon).

[1] Supported by Deutsche Forschungsgemeinschaft, Germany GRK 2088.

https://doi.org/10.1016/j.spasta.2024.100822
Received 17 November 2023; Received in revised form 7 March 2024; Accepted 7 March 2024
Available online 18 March 2024

covariate information $X_0 \in \mathbb{R}^q$ at s_0, and in fact it is often desirable to predict at many different locations s_0, e.g., on a whole grid simultaneously.

A classic approach to do this is *universal Kriging (UK)*, see Cressie (2015), Chilès and Delfiner (2012), Wackernagel (2003) or Van Lieshout (2019). The term is used here in its more general sense that allows for external covariates beyond the geographical coordinates, i.e. what is often known in applied geostatistics as *Kriging with external drift* or *regression Kriging*, see e.g. Hengl et al. (2007). Suppose the response over all of S is modeled as a Gaussian random field (GRF) $\mathbb{Y} = (Y(s))_{s\in S}$ with mean function $\mu(s) = f(s)^\top \beta$ and covariance function $k : S\times S \to \mathbb{R}$. Here the function $f = (f^{(1)}, \dots, f^{(q)}) : S \to \mathbb{R}^q$ and the covariance function k are assumed to be known (or previously estimated), but the parameter $\beta \in \mathbb{R}^q$ is unknown. Set $Y = (Y_1 \ \dots \ Y_n)^\top = (Y(s_1) \ \dots \ Y(s_n))^\top \in \mathbb{R}^n$, $X = (X_1 \ \dots \ X_n)^\top = (f(s_1) \ \dots \ f(s_n))^\top \in \mathbb{R}^{n\times q}$ and write $v(s_0) = (k(s_0, s_i))_{1\le i\le n}$ and $\Sigma = (k(s_i, s_j))_{1\le i,j\le n}$. It is known (e.g. Stein, 1999) that

$$\hat{Y}(s_0) = f(s_0)^\top \hat{\beta} + v(s_0)^\top \Sigma^{-1}(Y - X\hat{\beta}), \tag{1}$$

where

$$\hat{\beta} = (X^\top \Sigma^{-1} X)^{-1} X^\top \Sigma^{-1} Y$$

minimizes the mean squared error (MSE), i.e. attains

$$\min_g \mathbb{E}(Y(s_0) - g(Y))^2,$$

where the minimum is taken over all measurable functions $g : \mathbb{R}^n \to \mathbb{R}$ such that $\mathbb{E}g(Y)^2 < \infty$ and $\mathbb{E}g(Y) = \mathbb{E}Y(s_0)$ (unbiasedness).

If we relax the Gaussianity of $\mathbb{Y}$ and only assume it to be an L^2 random field with mean and covariance functions as above, we still obtain (1) as the best (i.e. MSE minimizing) *linear* unbiased predictor (BLUP); see Stein (1999). Here, L^2 refers to the square integrability of the random variable $Y(s)$ for each $s \in S$.

Despite these optimality properties of UK, there may be good reasons to prefer other methods instead. For one, the computation of the predictor (1) is expensive for large n. It is often done in software packages by inverting the positive definite matrix Σ, which typically costs a hefty $O(n^3)$ elementary operations (e.g. via Cholesky decomposition, see Stoer and Bulirsch, 1993). This can be sped up by directly solving a system of linear equations, but even the best known theoretical complexity for this remains high at somewhere around $O(n^{2.37})$ (e.g. Coppersmith and Winograd, 1990) and algorithms that have real practical advantages for $n \le 10^{20}$ still require a much higher order, see Laderman et al. (1992). There are various techniques to mitigate the computational burden related to Kriging for large datasets, including covariance tapering (Furrer et al., 2006) and modeling by Gaussian Markov random fields (Rue and Held, 2005) to enforce sparsity of the covariance matrix or its inverse, continuous moving neighborhood approaches (Gribov and Krivoruchko, 2004; Rivoirard and Romary, 2011) that ignore all but neighboring data, and many other methods, such as fixed rank Kriging (Cressie and Johannesson, 2008) and stochastic partial differential equation approaches (Rue and Tjelmeland, 2002; Hrafnkelsson and Cressie, 2003; Lindgren et al., 2011). Another reason for looking for other method than Kriging is that some of the assumptions made above for optimality are often not directly satisfied for real data, so that transformations have to be applied and quantities in connection with the covariance structure have to be estimated beforehand, which may lead to departure from optimality in practice.

As an alternative, there has recently been a heightened interest in graph neural networks (GNN) based approaches in this and similar contexts (Appleby et al., 2020; Chen et al., 2020; Wu et al., 2020, 2021). The basic idea is that (local) graphs are built from the spatial data, which are then passed to a neural network to obtain predictions. GNNs have been studied in various contexts in Hamilton et al. (2017), Atwood and Towsley (2016), Defferrard et al. (2016), Kipf and Welling (2017), Veličković et al. (2017), Li et al. (2016), Ji et al. (2020), among others. They offer the flexibility to include the labels of neighboring points for the prediction of the label of the current point. However, capturing the spatial correlation within the spatial data remains a challenge.

While Kriging achieves this directly via variography, most of the GNN-based models do this indirectly by constructing weighted graphs (and their adjacency matrices) from distance-based kernels. Many models use Gaussian kernels with a fixed length scale which is either a direct function of the data (Wu et al., 2020 for example uses the standard deviation of distances within the data) or chosen by cross-validation (Appleby et al., 2020 for example). There are several limitations to these approaches.

On the one hand, there is no direct relationship between the data length scale and the appropriate kernel range as criticized in Wu et al. (2021). Instead, these authors suggest to combine multiple operations and aggregate the resulting kernels for more expressivity in the nodes. This is done in the same spirit as Chen et al. (2020), where several compactly supported Wendland correlation functions are used as kernels to capture spatial dependence. Unfortunately, this leads to a large amount of hyperparameters to choose and tune. Prior to these works, Monti et al. (2017) suggested Gaussian kernels with a learnable covariance matrix and mean vector, leading to a extremely complex network calling for regularization. To mitigate this effect, they considered only diagonal covariance matrices, which can be very restrictive in many scenarios. We note that other special kernels beside the Gaussian are also considered in the literature (Chen et al., 2020; Lei et al., 2021; Hammond et al., 2011)

On the other hand, the best cross-validation scheme to estimate length scales of chosen weight kernels for neural networks on spatially dependent data is not clear. In fact, standard cross-validation and generalized cross-validation techniques (Allen, 1974; Craven and Wahba, 1978; Geisser, 1975; Mallows, 1973; Stone, 1974) tend to choose smaller length scales, which are biased towards under-smoothing when the correlation within the spatial data is positive, and larger length scales, which are biased towards over-smoothing when the said correlation is negative (Altman, 1990; Chu and Marron, 1991). Therefore, some corrected methods were proposed (Altman, 1990; Lee et al., 2010), including some specific to long range dependence (Ray, 1997) and some aiming at

removing correlation before estimating kernel length (De Brabanter et al., 2010). Chu and Marron (1991) discuss the performance of two adjusted methods, namely the *leave-*$(2l+1)$*-out method* also referred to as modified cross-validation (MCV; Härdle et al., 1988, Györfi et al., 1989) and the partitioned cross-validation (PCV; Marron, 1987), while Hall et al. (1995) compared MCV and block bootstrap based methods. The first paper shows that the asymptotic effects hold for reasonable sample size, which is not the case in today's context where we have large size datasets available for analysis. The second shows that optimal bandwidths are only found provided that block length or leave-out number are appropriately chosen. Moreover, the implementation of these methods is not scalable to neural networks. That is why (Rodrigues et al., 2022) suggested a cross-validation method by resampling for neural networks. Not long ago, Wadoux et al. (2021) opened the discussion between standard cross-validation and spatial cross-validation (Pohjankukka et al., 2017).

Latest neural network methods try to avoid these problems. In the Bayesian framework for discrete spatial data studied in Semenova et al. (2022), for example, the Gaussian process prior is replaced by a trained variational autoencoder.

In the present paper, we give a precise mathematical construction of a substantial generalization of the Kriging Convolutional Network (KCN) model by Appleby et al. (2020). Among other things our model allows for integrated learning of the convolution kernel as a linear combination of a number of base kernels, admits higher order neighborhoods as in Zhu et al. (2020) and may include further adjustments depending on the target values at each node and the distances between the neighbors. To have an easily recognizable term, we refer to the new generalized model as Spatial Prediction On Neighborhood Graph Entities (SPONGE). The reader may think of it as a generalized GCN-based neural network for spatial prediction that includes the KCN as a special case (up to some very minor things we do differently). We refrained from including the word Kriging, as we see the Kriging solution to spatial prediction problem at best very weakly related to SPONGE. Based mainly on simulation studies, we investigate

(A) to what extent different variants of SPONGE, which by their nature are much faster than Kriging for moderate to large n, manage to come close to or (if the optimality conditions for Kriging are not fully satisfied) even beat Kriging with regard to prediction accuracy in terms of mean squared errors (MSEs).
(B) what improvements can be obtained for general SPONGE in comparison with the original KCN.

The rest of the paper is organized as follows: In Section 2, we present the new general SPONGE framework including the construction of weight matrices by linear combinations of matrices obtained from base kernels. Details on the process of finding appropriate length scales for these kernels are provided in Section 3. Controlled simulations for various settings in Section 4 allow us to discuss Questions A and B above in ample detail. Section 5 presents applications to several real world data sets and Section 6 gives a summary of our findings and provides an outlook.

2. Description of the network

We present the general SPONGE, starting with the definition of neighborhood graph entities. As we will see along the way this uses ideas from several previous neural networks combined and generalizes some of them, including (Appleby et al., 2020) and Li et al. (2017).

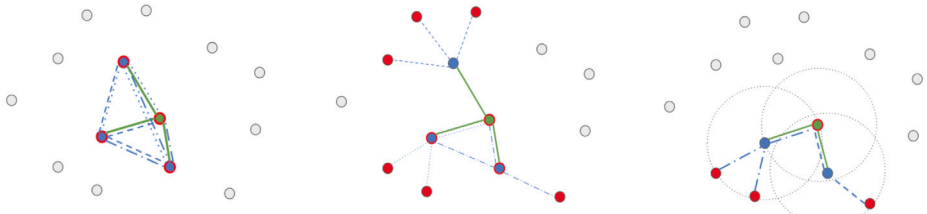
Set $[n] := \{1, \dots, n\}$ and denote by $\mathfrak{N}$ the space of finite simple counting measures on S or, by identification, finite subsets of S (the interpretation as counting measures is useful for questions of measurability and convergence, which we forgo in this paper). Furthermore, we denote by $\mathfrak{W}$ the set of edge-weighted graphs (V, E, w), where $V \in \mathfrak{N}$, $E \subset \mathcal{P}_2(V)$ (the set of two-element subsets of V), and $w : E \to (0, \infty)$. We think of missing edges as "edges having weight 0", in particular we identify w with the function $\mathcal{P}_2(V) \to [0, \infty)$ that assigns 0 to two-elements sets of vertices not in E.

For a fixed convolution order $r \in \mathbb{N}$ and fixed integers $0 = m_0 \le m_1 \le \cdots \le m_r \le n$, choose functions

$$\begin{aligned} &\Phi_\psi^{(t)} : S \times \mathfrak{N} \to \mathfrak{W} \\ &\qquad (s, \xi) \mapsto \Phi_\psi^{(t)}(s, \xi) = \big(V(s, \xi), E_\psi^{(t)}(s, \xi), w_\psi^{(t)}(s, \xi)\big), \end{aligned}$$

where $\psi \in \mathbb{R}^p$ is a learnable parameter and $t \in [\tau]$ indexes different variants (out of $\tau \in \mathbb{N}$), which we combine later on. The resulting graph $\Phi_\psi^{(t)}$ has a vertex set $V(s, \xi)$ containing (possibly multiple copies of each of) s and further points of ξ and $S \setminus \xi$. Its edge set $E_\psi^{(t)}(s, \xi)$ is such that at most m_l vertices from ξ have a shortest path distance $\le l$ to s (i.e. at most m_l vertices of $\Phi_\psi^{(t)}$ can be reached within l hops from s) for every $l \in [r]$. We refer to the images $\Phi_\psi^{(t)}(s, \xi)$ as *neighborhood graph entities (NGEs)*.

For convenience we set $m = m_r$ and renumber the vertices of $\Phi_\psi^{(t)}$ according to their shortest path distance to s as primary criterion. For every $l \in [r]$, we fill up the vertices from ξ having shortest path distance $\le l$ to s with isolated dummy vertices in $S \setminus \xi$ to a total of m_l. We refer to all of these vertices together as the l-th (order) neighborhood of s. Since the dummies are added purely for technical reasons, so that we are able to set up the GCN below, we use the same term for the vertices without the dummies. Thus in the new s-centered enumeration of vertices, which we will refer to as $s_0, \dots, s_m$ again where this does not lead to confusion with the original enumeration, we have first $s_0 = s$; then each of $s_1, \dots, s_{m_1}$ is a point in ξ that is one hop away from s or a point in $S \setminus \xi$ that is isolated; each of $s_{m_1+1}, \dots, s_{m_2}$ has minimal distance of two hops or is isolated (and not among the former isolated points) and so on up to $s_{m_{r-1}+1}, \dots, s_{m_r}$. It is reasonable to have a secondary order criterion to be applied for the new vertices added when going from the $(l-1)$-st to the l-th neighborhood, but a good choice depends on the concrete situation. Typically we would put the dummy points last and enumerate the points of ξ either according to their Euclidean distance to s or according to their Euclidean distance to the point they belong to in the $(l-1)$-st neighborhood. Whenever we do an s-centered renumbering of the locations we renumber the covariates $X_0, X_1, \dots, X_m \in \mathbb{R}^q$ and the target values $Y_0, Y_1, \dots, Y_m \in \mathbb{R}$ accordingly. Suppressing the indices t and ψ for now, typical examples of $\Phi(s, \xi) = (V, E, w)$ include the following.



(a) NGE in the first example: The neighborhood size is fixed by fixing the number of neighbors, and a complete graph is generated. The second order neighbors are chosen among the first order ones.

(b) NGE in the second example: The neighborhood size is fixed by fixing the number of neighbors, and an undirected graph is generated. Each point is connected to its closest neighbors.

(c) NGE in the third example: The neighborhood size is fixed by fixing its radius, and an undirected graph is generated. Points in the ball are connected to the center.

Fig. 1. Example of NGEs built from the observations: The center node is colored in green, its first-order neighbors in blue, the second-order neighbors in red and further points in gray. The thick green lines represent edges from first order neighbors to the center node, whereas the various types of blue lines are edges from second to first order neighbors. While edges in an NGE are set up as undirected, a natural direction is implied by the fact that the two vertices of any edge belong to neighborhoods of subsequent orders. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

1. (V, E) is the complete graph based on the vertex set V consisting of s and its k nearest neighbors among the points of $\xi \setminus \{s\}$, i.e. $E = \mathcal{P}_2(V)$. Here $r = 1$ and $m_1 = m = k$. The map w is given e.g. by $w(\{v, v'\}) = \exp\big(-\frac{1}{2\sigma^2}\|v - v'\|^2\big)$. This corresponds to the model in Appleby et al. (2020) (see Fig. 1(a)).
2. (V, E) is obtained by connecting s to its k nearest neighbors among the points of $\xi \setminus \{s\}$ and then connecting each newly connected point s' to its k nearest neighbors among $\xi \setminus \{s'\}$. The latter may include s or any of the first k nearest neighbors of s (except for s' itself), in which case we generate a copy of that point at the same location with individual edges. Here we have $r = 2$, $m_1 = k$ and $m_2 = m = k+k^2$, without the need for any dummy points. Again we choose e.g. $w(\{v, v'\}) = \exp\big(-\frac{1}{2\sigma^2}\|v-v'\|^2\big)$ (see Fig. 1(b)).
3. For some fixed $R > 0$, we obtain (V, E) by first connecting all points of $\xi \setminus \{s\}$ in the ball $\mathbb{B}(s, R) = \{s' \in S \mid \|s' - s\| \le R\}$ to s, and then for each newly connected point s' connecting all points of $\xi \setminus \{s'\}$ in $\mathbb{B}(s', R)$ (which may again generate copies of previous points). Here $R > 0$ has to be judiciously chosen, such that the graphs do not get too large but we typically have at least a few points in each ball. Denoting by $\tilde{k}$ the maximal number of points of ξ we can find in a ball of radius R with arbitrary center, we have $r = 2$, $m_1 = \tilde{k}$ and $m_2 = m = \tilde{k} + \tilde{k}^2$. Here we add isolated dummy vertices, so that formally there are exactly m_1 vertices among the first set of points $\neq s$ considered and m_2 vertices in the graph in total. Again we choose $w(\{v, v'\})$ as before (and there are many other possibilities as well). This corresponds to the choice in Li et al. (2017), Section 4 (see Fig. 1(c)).

In the above examples the weight function is largely arbitrarily chosen, but typically has a crucial effect on the performance of the model. This is why we introduce the parameter ψ and combine τ different weight functions below.

During the training phase, we shuffle the indices of our training data and subdivide them into batches. We then construct convolutional weight matrices for GCNs based on the NGEs $\boldsymbol{\Phi}_{\psi}^{(t)}(s, \xi)$ for each $s \in \xi$ in a batch. Note that the Y-values at $s \in \xi$ are known. During the test and deployment phases, we construct convolutional weight matrices based on $\boldsymbol{\Phi}_{\psi}^{(t)}(s, \xi)$ at new locations $s \in S$. The goal is to predict the unknown Y_s.

In the next two sections we set up the actual neural network. First we consider a special case that is essentially a direct generalization of Appleby et al. (2020). This requires us to pre-specify a single neighborhood graph function $\boldsymbol{\Phi}_{\psi}^{(t)}$ (so we suppress the indices t and ψ again) leading to a pre-specified convolutional filter. In Section 2.2 we learn the convolutional filter together with the other parameters of the model. It is an amalgam of various recent innovations in GCNs, using among others ideas of Monti et al. (2017). Finally, in Section 2.3, we discuss the choice of loss function and more general measures to evaluate the network.

2.1. Pre-specified graph weight matrices

Based on the NGE $\boldsymbol{\Phi}(s, \xi) = (V, E, w)$, let $W = (w_{ij})_{0 \le i,j \le m}$, where $w_{ij} = w(\{s_i, s_j\})$ (in local enumeration based on $s = s_0$; see above). We then consider the submatrices $W^{(l)} = (w_{ij})_{0 \le i \le m_{r-l}, 0 \le j \le m_{r-l+1}}$, $l \in [r]$, as our graph weight matrices.

Set $D^{(l)} = \mathrm{diag}(W^{(l)}\mathbf{1})$, which is the diagonal matrix of row sums of $W^{(l)}$ and consider the normalized matrices $\widetilde{W}^{(l)} = {D^{(l)}}^{-1} W^{(l)}$.

Given the covariate data $X = (X_1 \ \dots \ X_m)^\top \in \mathbb{R}^{m \times q}$ and the target value data $Y = (Y_1 \ \dots \ Y_m)^\top \in \mathbb{R}^m$ at the locations $s_1, \dots, s_m$ (all in local enumeration based on $s = s_0$) and new covariate data $X_0 \in \mathbb{R}^q$ at s_0, the input layer of the neural network is encoded as

$$H^{(0)} = \begin{pmatrix} 0 & 1 & X_0^\top \\ Y & 0 & X \end{pmatrix} \in \mathbb{R}^{(m+1)\times(q+2)}.$$

Our GCN can then be specified as follows. There are r convolutional layers obtained as

$$H^{(l)} = \sigma_l\big(\widetilde{W}^{(l)} H^{(l-1)} \Theta^{(l)}\big) \in \mathbb{R}^{(m_{r-l}+1)\times d_l}, \quad 1 \le l \le r, \tag{2}$$

where σ_l are activation functions and $\Theta^{(l)} \in \mathbb{R}^{d_{l-1}\times d_l}$ are the parameter matrices (matrices of "weights" between the layers in the network). We set $d_0 = q+2$, whereas the other $d_l \in \mathbb{N}$ are design parameters of the network that can be freely chosen.

We follow up the convolutional layers by a single dense layer, leading to the prediction

$$\hat{Y}(s_0) = H^{(r+1)} = H^{(r)} \Theta^{(r+1)} \in \mathbb{R}, \tag{3}$$

where $\Theta^{(r+1)} \in \mathbb{R}^{d_r \times 1}$ (note that $H^{(r)} \in \mathbb{R}^{1\times d_r}$).

2.2. Learning the graph weight matrices

Consider now the more general situation that we have several maps $\Phi^{(t)}_{\psi_t}$, where $\psi_t \in \mathbb{R}^p$ and $t \in [\tau]$. In this paper, ψ_t are typically the length scale parameters considered for the weight function of our graph. From a neural network perspective, ψ_t are hyperparameters and we later provide a procedure to select them for $t \in [\tau]$ (see Section 3.2). In general, ψ_t could be any learnable parameter of some parametric weight kernel function (see for example Monti et al., 2017).

In the same way as in the previous section this leads to the weight matrices $W^{(t)}_{\psi_t} \in \mathbb{R}^{(m+1)\times(m+1)}$ with (i,j)-th entries $w^{(t)}_{\psi_t}(s_i, s_j)$ (in local enumeration based on s_0) and their submatrices $W^{(t,l)}_{\psi_t} \in \mathbb{R}^{(m_{r-l}+1)\times(m_{r-l+1}+1)}$ for $t \in [\tau], l \in [r]$.

We aggregate the matrices $W^{(t)}_{\psi_t}$ (and thereby their submatrices) by taking linear combinations, where $\theta \in \mathbb{R}^\tau$ is another learnable parameter vector of the model (not to be confused with $\Theta^{(l)}$). By combining weight matrices obtained from a number of base kernels in this way, we obtain a quite flexible final weight matrix $W^{(l)}$ and free ourselves from any overly strong influence of the initial choice of one or several weight functions.

Considering now the normalized submatrices $\widetilde{W}^{(l)} = {D^{(l)}}^{-1} W^{(l)}$, we set up the network again by Eqs. (2) and (3).

2.3. Loss function and performance metrics

For the training of a SPONGE (just as for any artificial neural network), we subdivide our initial set of observations into two parts:

- a training set, over which we iterate a certain number of times (epochs) with data subdivided into batches in order to estimate the model parameters (θ and $\Theta^{(l)}$);
- a validation set, which we use to estimate hyperparameters and to monitor the updates of the model parameters.

The training cycles require minimization of a loss function; in the validation cycles we use a performance metric to evaluate the predictions. For both we use the mean squared error (MSE) defined by

$$\mathrm{MSE} = \frac{1}{n'} \sum_{i=1}^{n'} \big(\hat{Y}_i - Y_i\big)^2,$$

where $\hat{Y}_i$ denotes the predicted value at s_i for the current model (hyper)parameters and n' denotes the size of the data currently under consideration. During training n' is the batch size, during validation n' is the size of the validation set.

The reason why we choose the MSE for these tasks is because the corresponding MSE for an unknown response at a fixed location is the objective function minimized for UK. Since we are interested in assessing the performance gap between Kriging and SPONGE, we keep the most favorable setting for Kriging.

In order to evaluate the performance of the trained SPONGEs as well as of any other method investigated in this paper, we use a test set of another n locations $\bar{s}_i \in S$, $i \in [n]$, along with their covariates $\bar{X}_i$ (if any) and target values $\bar{Y}_i$. The performance metric is then the MSE of the $n' = n$ predicted values on this test set as compared to $\bar{Y}_i$.

If the goal is to produce a map of the target quantity on the whole set S, it is more natural to evaluate the performance with respect to the integrated squared error (MISE) defined by

$$\mathrm{MISE} = \int_S \big(\hat{Y}(s) - Y(s)\big)^2 \, ds. \tag{4}$$

In practice we approximate the MISE by computing an MSE over test locations forming a fine regular grid in S.

3. Weight functions and their length scales

As mentioned before, a challenge in prediction for spatial data using graph neural networks remains the choice of the weight functions $w^{(t)}_\psi = w^{(t)}_\psi(s,\xi) : E \to (0,\infty)$, which control the relative influence of (neighboring) labels according to their spatial distance and maybe further criteria. With the approach laid out in Section 2.2, the network can learn the best weight function as a linear combination of some base weight functions. The question remains, however, how to choose these base weight functions. We split up the discussion into choosing the global form of the weight functions and, what is often more crucial, picking their length scales.

3.1. Form of the weight functions

Basic constructions for $w(\{v, v'\}) = w_0(\|v - v'\|)$ include:

- Gaussian:

$$w_0(r) = \exp\left(-\frac{r^2}{2\psi^2}\right) \tag{5}$$

- Exponential:

$$w_0(r) = \exp\left(-\frac{r}{\sqrt{2}\psi}\right). \tag{6}$$

- Student's t kernel (Tang et al., 2017):

$$w_0(r) = \frac{1}{\left(1 + \frac{r^2}{\nu\psi^2}\right)^{\frac{\nu+d}{2}}}, \tag{7}$$

  where d is the dimensionality of the coordinates, ν is the degree of freedom.
- The rational quadratic kernel (Fadel et al., 2016):

$$w_0(r) = 1 - \frac{r^2}{r^2 + \psi}. \tag{8}$$

- B-splines of degree 3 : Let us assume we have a set of increasing hyperparameters values $(r_1, \ldots, r_{k+p+1})$ called knots, where k is the number of knots, p is the degree of the B-spline and its order is $p + 1$. The B-splines are defined through a recursive formula (de Boor, 1971):

$$B_{i,0}(r) = \begin{cases} 1 & \text{if } r_i \le r < r_{i+1} \\ 0 & \text{otherwise} \end{cases}$$

  if $p = 0$, and for $p \ge 1$:

$$B_{i,p}(r) = \frac{r - r_i}{r_{i+p} - r_i} B_{i,p-1}(r) + \frac{r_{i+p+1} - r}{r_{i+p+1} - r_{i+1}} B_{i+1,p-1}(r)$$

  where we define the following fraction: $\frac{0}{0} = 0$.

In the above description, ψ stands for the length scale, s and s' are two different positions. We may want to add a "nugget", see e.g. Van Lieshout (2019), to any of these kernels, i.e. set $w_0(0) = 1 + \alpha$ for some $\alpha > 0$. Depending on the scientific background of the data, it may also be relevant to include directional effects, information about the configuration of other locations, as well as the available feature and label information.

In the present paper we only consider a single variant, which is a combination of reweighing based on the difference in labels of neighboring nodes and reweighing based on the relative distances between locations in the same neighborhood.

Label-based reweighing.. Consider the difference of the labels for neighboring nodes to reweigh the weight function, so as to give more importance to neighboring nodes with similar measurement values. The map w is therefore updated according to the following scheme:

$$w(\{v, v'\}) = w(\{v, v'\}) \exp\left(-\frac{1}{\sigma'^2}(y - y')^2\right)$$

where y and y' are the target values associated with v and v', respectively.

Note that this reweighing scheme requires the knowledge of both of the labels at nodes v and v', respectively. Therefore, in the case where we do not have access to such information for the test set (typically at the node where one needs to do a prediction), we do not apply this update (even for the training set). Instead, we only apply the μ-closeness based reweighing as described below. This label reweighing is closely related to the notion of homo/heterophily introduced in Zhu et al. (2020). For simplicity, we always work with $\sigma' = 1$.

μ-Closeness-based reweighing.. Let $r \in \mathbb{R}$ be the distance between any two given points. We say that those points are μ-close, $\mu \in \mathbb{R}$, if $r \le \mu$.

The different methods enumerated to build subgraphs solely focus on the distance between the point of interest and its neighbors without taking into account the distance between the neighbors themselves. However, if two neighbors of the point of interest are very close, it is very likely that they carry the same information. Choosing another point in a different direction could therefore be more informative for the subgraph.

Here we consider the distances between points in the neighborhood and compute the median of those distances. If a distance is smaller than μ times this median, the two points at that distance are considered μ-close. We propose in that case to downweigh one of such $\mu-$close points, so as to give more importance to the points in the remaining directions. One example of such a downweighing function is given by $\frac{\exp(r)-1}{\exp(r)+1}$ where $r \in \mathbb{R}$ stands for the said distance. Without any other specification, we always work with $\mu = 0.2$.

Table 1
Process of choosing the appropriate length scales for Gaussian kernels. Tr MSE stands for the MSE on the training set, Val MSE stands for the MSE on the validation set and Test MSE stands for the MSE on the test set.

Length scales	θ	Tr MSE	Val MSE	Test MSE
$10^3, 10^2, 10^1, 10^0$	0., 0., 0., 0.042	2.549	2.401	3.013
$10^2, 10^1, 10^0, 10^{-1}$	0., 0., 0., 0.740	2.285	2.135	2.642
$10^1, 10^0, 10^{-1}, 10^{-2}$	0.005, 0.005, 0.06, 0.912	1.135	0.688	1.199
$10^0, 10^{-1}, 10^{-2}, 10^{-3}$	0., 0.098, 0.173, 2.470	1.202	0.747	1.127
$10^{-1}, 10^{-2}, 10^{-3}, 10^{-4}$	0.06, 0.048,1.558,1.978	1.004	0.736	**0.958**
$10^{-2}, 10^{-3}, 10^{-4}, 10^{-5}$	0.649, 0., 0., 0.	1.944	1.427	2.034
$10^{-3}, 10^{-4}, 10^{-5}, 10^{-6}$	0.078, 0.118, 0.118, 0.118	2.257	1.782	2.371

Table 2
Best sets of length scales and performances for various other kernels. Tr MSE stands for the MSE on the training set, Val MSE stands for the MSE on the validation set and Test MSE stands for the MSE on the test set.

Kernels	Best length scales	Tr loss	Val MSE	Test MSE
Exponential kernel	$10^{-1}, 10^{-2}, 10^{-3}, 10^{-4}$	0.632	0.389	0.645
Student's t kernel ($\nu = 1$)	$10^0, 10^{-1}, 10^{-2}, 10^{-3}$	1.062	0.713	1.094
Student's t kernel ($\nu = 4$)	$10^{-1}, 10^{-2}, 10^{-3}, 10^{-4}$	1.145	0.764	1.097
Rational quadratic kernel	$10^{-3}, 10^{-4}, 10^{-5}, 10^{-6}$	0.556	0.285	0.565

3.2. Finding the appropriate length scales

While the form of the weight functions should be left to the user, who can then decide to simply pick the "default" Gaussian kernel or make a more adaptive choice based on additional knowledge about the data, we would like to have the length scales picked more or less automatically by the model.

The following strategy is adopted:

1. Start with a set of length scales that are equally spaced on a scale of 10^b, $b \in \mathbb{Z}$, and as large as reasonably possible with respect to the size of the observation window S and essential range of the weight function (i.e. the distance where the weight function comes close to zero should be much larger than the diameter of the observation window).
2. Perform a cycle of training and validation with the current set of length scales. Note the MSE obtained on the validation set, remove the left most length scale value and update the initial set of fixed length scales with the next lower of the form 10^b.
3. Do this until the validation MSE plateaus or keeps getting worse.

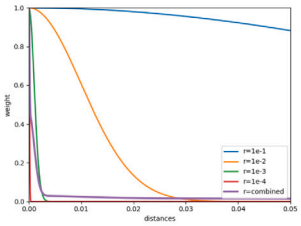
We exemplify this procedure for one of the simulated data sets of 2500 observations on $S = [0, 1]^2$ from Section 4.1. In Table 1 the different sets of length scales are given in sequence, with their weight vectors obtained by SPONGE when using a Gaussian kernel, and the corresponding MSEs on the training, validation and test data sets, respectively. Table 2 shows the final length scales and their performances for other kernels.

We observe empirically that this process allows us to find nearly optimal length scales within few iterations. As a rule of thumb, 10^2 or 10^3 is a good large value to start with. When the length scale values are too large for the data, not all the points considered for predictions are necessary and some may distort the signal of the most useful ones. This explains why at the beginning of the process where the length scale values are still too large, the θ vector only relies on the smallest available one (first and second row of Table 1). As we reduce the length scale values, the network gradually assigns bigger weights to the most influential ones. Length scales that are so small that their kernel values become virtually zero for all of the distances observed in the data, have θ-entries that are irrelevant and do not seem to be distinguished in the SPONGE model (last two rows of Table 1).

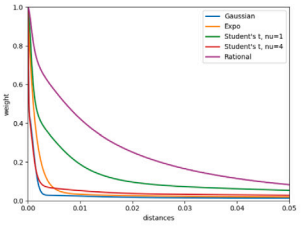
Depending on the data, some kernels can be more efficient in capturing the influences of nearby observations. It is therefore worth trying different kernels. We note that the optimal set of length scales is (of course) not the same for all the kernels (see Table 2), among other things because their essential ranges are quite different.

Fig. 2(a) displays the kernel functions associated with each of the individual optimal length scales found in Table 1 (Gaussian), together with the resulting combined kernel used for prediction. Fig. 2(b) displays the resulting learned functions of various kernel maps (exponential, Student's t and rational quadratic). The combined weight functions obtained from exponential and Gaussian kernels are similar. We speculate that the similarity of the combined weight function for the Student's t kernel with $\nu = 4$ to the two former is due to the fact that the Student's t-density converges to the Gaussian as $\nu \to \infty$. The largest weights were learned by the rational quadratic kernel. Note, however, that the absolute values of the combined weight functions have no meaning, since they are normalized to sum to one for the individual distances between neighbors for each row of our weight matrices. They are informative inasmuch as each function for itself displays the relative influence of labels located at various distances.

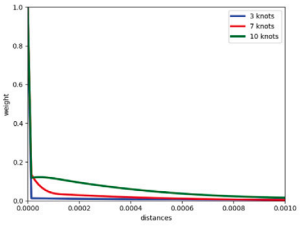
We adopt a similar strategy for B-splines, noting that the number and positions of the knots influence the flexibility but also the complexity of the combined weight function. A higher concentration of knots within a certain range of distances allows the combined weight function to adapt more flexibly to local changes in influence on a point by neighbors located at these distances. We choose the number of interior knots $k' \geq 1$ and place them at the fixed positions $0, 10^{-(k')}, 10^{-(k'-1)}, \ldots, 10^{-1}, 1$ (measured in units of the maximum distance between points in the dataset). See Table 3 for the performance of various k'. We use the implementation



(a) Individual and combined Gaussian kernels.



(b) Resulting combined functions for various weight functions.



(c) Resulting combined functions for B-spline weight functions

Fig. 2. Example of learned kernel maps.

Table 3
Process of choosing the appropriate number of knots for B-spline kernels. Tr MSE stands for the MSE on the training set, Val MSE stands for the MSE on the validation set and Test MSE stands for the MSE on the test set. The knot values in the second column are in unit of the maximum distance.

# interior knots	Knot values	Tr MSE	Val MSE	Test MSE
3 knots	$0, 10^{-3}, 10^{-2}, 10^{-1}, 1$	1.024	0.66	1.074
7 knots	$0, 10^{-7}, \ldots, 10^{-2}, 10^{-1}, 1$	0.948	0.865	0.966
10 knots	$0, 10^{-10}, \ldots, 10^{-2}, 10^{-1}, 1$	0.589	0.506	0.625
11 knots	$0, 10^{-11}, \ldots, 10^{-2}, 10^{-1}, 1$	1.024	0.981	1.079

in the Python library patsy (Smith et al., 2018), where we have excluded the intercept at one and used all the remaining available basis function. The resulted combined weight functions are shown in Fig. 2(c). When comparing these functions to each other and to the functions in Fig. 2(b), note again that only the relative values within each function are meaningful.

The rational quadratic kernel has the best test MSE, followed by the B-spline kernel.

4. Simulation study

In this section we study a number of simulation scenarios to address the two questions asked in the introduction, i.e. to what extent can SPONGE catch up with Kriging in terms of MSE performance (or even beat it in situations where the conditions for Kriging are not fully satisfied) and to what extent do the generalizations we propose yield an improvement over KCN.

In Section 4.1 we describe the setup and the main simulation procedures and Section 4.2 provides details on the implementation of SPONGE and the other methods. The remaining sections present and discuss the simulation results in different settings (Kriging-friendly and Kriging-unfriendly) and present two follow-up studies.

4.1. Data generation process

For our main experiments, we generate data that is *well-specified* for Kriging, i.e. satisfies all the conditions under which Kriging is optimal with respect to the MSE. More precisely, we assume that the random field $\mathbb{Y} = (Y(s))_{s\in S}$ is Gaussian with mean function μ of the form $\mu(s) = f(s)^\top \beta$ for some $\beta \in \mathbb{R}^q$ and covariance function k, where the true f and k are known to us. We first simulate, as a ground truth, a stationary Gaussian random field (GRF) on $S = [0,1]^2$ with mean $\mu = 0$, variance $\sigma^2 = 3$ and Gaussian correlation function with length scale $r = 0.01$. For this the GStools package (Mueller et al., 2021) is used. We then generate our data (without covariates) by picking n i.i.d. locations $s_1, \ldots, s_n$ in S at which the GRF is evaluated to obtain the corresponding labels $Y_1, \ldots, Y_n$. In all our experiments we consider three different sizes of data: $n = 50$ (small), $n = 250$ (medium), and $n = 2500$ (large). These sizes and terminologies are also chosen with a focus on Kriging and more traditional geostatistics in mind rather than a neural network setting. As per our question A asked in the introduction, we want to study here a situation where Kriging is the gold standard, because it is computationally feasible and theoretically optimal, and we investigate how close can we get with neural network models.

In the same spirit we generate a data set with covariates as

$$Y_i = aX_{i1} + bX_{i2} + \tilde{Y}_i, \quad i \in [n]. \tag{9}$$

Here, $\tilde{Y}_i$ are generated as observations from a zero mean stationary Gaussian random field on $S = [0,1]^2$ with variance $\sigma^2 = 3$ and Gaussian correlation function with length scale $r = 0.1$. Then we have added covariate information generated from independent centered GRFs with Gaussian correlation functions. X_1 is sampled from a GRF with variance $\tilde{\sigma}_1^2 = 2$ and length scale $r_1 = 0.03$ and X_2 from a GRF with variance $\tilde{\sigma}_2^2 = 2$ and length scale $r_2 = 0.17$. Furthermore $a = 2.5$ and $b = 0.5$. We thus have three components influencing the final labels. A first covariate making up a large part of the total variance (contribution is $(2.5)^2 \cdot 2$) with small patches of similar values ($r_1 = 0.03$), a second covariate making up a rather small part of the total variance (contribution is $(0.5)^2 \cdot 2$) with

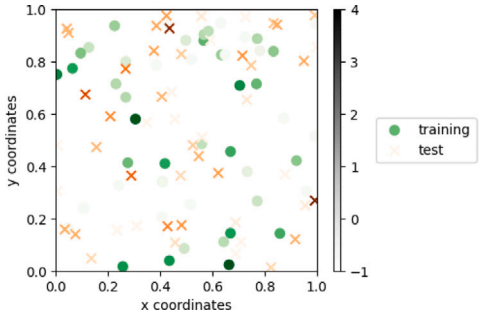


Fig. 3. Sampled training and test data from a GRF. The grayscale bar indicates how the value (brightness) of the color, with different hues for training and test data, translates to the label value. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

rather large patches of similar values ($r_1 = 0.17$) and a noise field making up a medium part of the total variance with medium correlation.

We typically evaluate the various models on test data obtained by picking another n i.i.d. observations from the same realization of the random fields for which the model is fitted (Fig. 3). Each experiment is replicated 100 times to give a stable reproducible result. We report the mean and the standard deviation of the MSE measured on each of the test sets.

In Section 4.4, we consider also contaminated and mis-specified data. By the latter we mean data that cannot be accurately described by a Kriging model.

4.2. Implementation details

We use the Python environment (Van Rossum and Drake, 1995) for our experimentation all the libraries used are in Python as well. In particular, the code for the SPONGE uses the TensorFlow 2 library (Abadi et al., 2015). Our code is publicly available on the GitHub repository https://github.com/ANMarianne/SPONGE. All experiments with SPONGE are performed using the ADAM Optimizer (Kingma and Ba, 2015) with a learning rate of 0.01 and a batch size of 64. The learning rate value here is the default one proposed in the TensorFlow2 library, which in general is a reasonable choice when not performing any hyperparameter tuning. A rule of thumb to choose batch sizes is to start with relatively small values (usually 32 or 64). In our case we started with 64 and the training seemed stable enough.

In what follows, we refer to the SPONGE built simply from the first k nearest neighbors as *1-hop model*. This is the $r = 1$ version of Example 2 in the list in Section 2, which is represented by the green graph with the blue nearest-neighbor nodes in Fig. 1(b). The SPONGE constructed from two k-nearest neighborhoods as described in Example 2 is referred to as *2-hop model*. In both cases we set $k = 5$ and the weight function is Gaussian as long as nothing else is mentioned. By the *reweigh 2-hop* models we mean the corresponding model with an additional reweighing based on labels (if applicable) and on μ-closeness.

We have implemented a TensorFlow 2 version of the original KCN to reduce possible implementation differences from TensorFlow 1. We validated our implementation with the original KCN data and were able to recover most of the experimental values reported in Appleby et al. (2020). For each dataset, we performed a grid search over several hyperparameter combinations just as described in Appleby et al. (2020) to find the best setting.

Except from the 1-hop model which trains a 1-layer graph neural networks with a hidden size of 10, we always train a 2-layer graph neural networks with respectively 5 and 10 as hidden size for each layer, and combine $\tau = 4$ length scales for our SPONGE.

Among the various classic spatial prediction methods, we have implemented inverse distance weighting (IDW) (Shepard, 1968), Nadaraya–Watson kernel regression (NW) (Nadaraya, 1964; Watson, 1964; Whang, 1998) and Kriging. We also add a naive nearest neighbor (NN) predictor, which simply returns the value of the label at the nearest observed location. Both for IDW and NW regression we perform modified cross-validation for each dataset as described in Chu and Marron (1991) in order to obtain their power and length scale parameter respectively (using a Gaussian kernel for NW). All the code for Kriging in this work has been freshly implemented for finer control, but follows PyKrige implementation (Murphy, 2014). We note that (except for the contaminated data) Kriging was always applied with the correct (Gaussian) class of covariance functions, but the parameters were estimated for each dataset based on variography (see for example Van Lieshout, 2019), minimizing the mean squared deviation between the empirical variogram and the estimated variogram on the interval $[0, 0.8]$ in units of the maximum distance between points in the training dataset.

4.3. Simulation results in a perfect Kriging world

We first consider our main experiments with well-specified data for Kriging, i.e. where all the conditions for Kriging are satisfied (see Section 4.1). In the setting without covariates (but an unknown mean of the GRF), the procedure is commonly known as *ordinary Kriging (OK)*, see Van Lieshout (2019). This is nothing else than UK with $q = 1$, constant "covariate function" $f \equiv 1$ and, consequently, $X = \mathbf{1} \in \mathbb{R}^{n\times 1}$.

Table 4
Well-specified setting for Kriging, *without* covariates. The numbers specify means and standard deviations of test MSEs over 100 replications each for the three sizes of data sets. Data are simulated from a isotropic GRF with variance $\sigma^2 = 3$ and Gaussian correlation function with length scale $r = 0.01$. Numbers highlighted in blue correspond to the best ones across all the methods evaluated, while the numbers in bold are the best across the SPONGE methods.

	Small	Medium	Large
NN	5.93 ± 1.34	5.47 ± 0.58	3.01 ± 0.17
IDW	3.29 ± 0.83	3.06 ± 0.39	2.15 ± 0.25
NW	3.23 ± 0.72	3.00 ± 0.27	2.99 ± 0.12
OK	3.27 ± 0.7	2.90 ± 0.28	2.12 ± 0.26
KCN	4.36 ± 1.14	3.05 ± 0.28	$\mathbf{2.25 \pm 0.16}$
1-hop	3.78 ± 0.92	3.20 ± 0.32	2.64 ± 0.23
2-hop	3.56 ± 0.31	3.67 ± 0.38	2.9 ± 0.51
reweigh 2-hop	$\mathbf{3.33 \pm 0.78}$	$\mathbf{3.04 \pm 0.29}$	2.65 ± 0.15

The results are given in Table 4.

We observe without much surprise that the NN prediction displays the worst performance. As the number of training samples increase, the test MSE gets considerably better because each test sample has a higher probability of having a training sample within a range of reasonably strong correlation. However, as opposed to all the other methods, NN prediction neither directly nor indirectly evaluates this correlation range. It only serves as a naive baseline here and any of the other method should and does yield better performance.

There have been many articles comparing IDW and Kriging (Schloeder et al., 2001; Setianto and Triandini, 2015; Wu and Hung, 2016; Ibrahim and Nasser, 2017 to cite a few). They agree on the fact that overall Kriging performs better than IDW. Our results from Table 4 are consistent with this observation.

Kang and Joseph (2016) compare Kriging and NW interpolation, and enumerated cases where the latter could have a better performance than Kriging. As we laid out in the introduction, Kriging makes MSE-optimal predictions *if* the correct variogram is known and used (see also Yakowitz and Szidarovszky, 1985). However, here we only used the correct variogram class but estimated the parameters using variography, which explains why NW can have a better performance, especially for small data sets. This is also confirmed by Fengping et al. (2016), who explain that NW tends to have a good performance on small training sets, but that their performance becomes weaker on large training sets. The similarity in performance of NW and NN for large data set is in agreement with the findings of Kang and Joseph (2016).

For this first comparison, all of our SPONGE models use Gaussian kernels as in Eq. (5) as their weight functions. Note that, in contrast with OK, where we use additional information that is typically not available when performing the statistical analysis to supply the method with the right class of Gaussian covariance functions, the default use of Gaussian kernels for the distance-based weight functions does not constitute any direct favoritism of SPONGE since the real relations are more complex. We observe in Table 4 that KCN does not work well for small data sets. As the number of training samples increase the MSE of KCN gets gradually better but never outperforms Kriging.

The other SPONGE models yield a decent performance for small data sets. However, it should be noted that the simple kernel-based linear combination of labels computed by NW with a bandwidth from cross-validation does considerably better at this size. For the large data set many of the SPONGEs leave NW behind, but still do not quite manage to catch up to Kriging.

Among all the SPONGEs, the 1-hop model is the smallest in size and is therefore considerably faster in practice. It carries a reduction of time complexity by a factor of KL in comparison to KCN, where K is the size of the neighborhood and L is the total number of hidden layers for KCN. We notice moreover the improvement it brings on the small dataset in comparison to KCN. For the medium and large dataset the 1-hop MSE is smaller than the 2-hop ones, but becomes worse than the KCN MSE. The latter can be attributed to the simplicity of the 1-hop model. Additionally, we note that adapting the size of the receptive field of the 1-hop allows to improve its performance for larger size dataset. For example, if we reduce the number of neighbors to 3, we obtain an MSE of 2.51 on the large dataset.

The outcome for the 2-hop model is mixed. In comparison to KCN, the MSE of the 2-hop model on the small dataset is considerably smaller, but bigger for large datasets. Among all the SPONGE models in this scenario, the best model is the reweighed one, which always perform similarly or (sometimes much) better than the unreweighed counterpart.

Turning to the setting with covariates, (Table 5, still with well-specified data for Kriging), we first notice that the MSE gap between Kriging and the SPONGEs becomes much larger. This is no surprise, because the two covariates make the data structure much more complex for the SPONGEs but only partly so for Kriging, which is custom-tailored for the linear form of (9).

Among the SPONGE methods, the best performance is mostly obtained by a form of unreweighed 2-hop model. It appears that, in contrast with Table 4, the simple 2-hop model performs better when there are more features to learn. First of all, we note the consistent decrease in MSEs for the 2-hop model as the training data size increase. The reweigh 2-hop model does best on the small data set as was already the case in Table 4, but has a catastrophic performance on the large data set indicating that our reweighing strategy is not universal.

We further use this data set to compare the performance of different weight functions from Section 3 for the 2-hop model. In the small sample case, the exponential, the rational quadratic and the Student's t function are more sensitive, but, as the training

Table 5
Well-specified setting for Kriging, *with* covariates. The numbers specify means and standard deviations of test MSEs over 100 replications each for the three sizes of data sets. The covariates are sampled from GRFs with variance $\sigma_1^2 = 2$ and $\sigma_2^2 = 3$ and ranges $r_1 = 0.01$ and $r_2 = 0.17$, respectively. The response Y is generated according to Eq. (9). Numbers highlighted in blue correspond to the best ones across all the methods evaluated, while the numbers in bold are the best across the SPONGE methods.

	Small	Medium	Large
UK	1.39 ± 0.61	0.15 ± 0.29	$3.23 \times 10^{-3} \pm 9.71 \times 10^{-3}$
KCN	5.21 ± 3.83	3.00 ± 2.16	1.63 ± 0.26
1-hop	$\mathbf{4.16 \pm 2.37}$	2.22 ± 1.05	1.04 ± 0.47
2-hop (Gauss)	5.39 ± 4.00	2.77 ± 2.15	0.96 ± 0.42
2-hop (Expo)	6.03 ± 4.72	2.64 ± 2.05	0.93 ± 0.29
2-hop (Rational Quadratic)	6.79 ± 4.88	2.84 ± 1.42	0.60 ± 0.28
2-hop (Student's T)	5.77 ± 4.38	2.89 ± 2.89	0.90 ± 0.32
2-hop (B-splines)	4.85 ± 3.08	$\mathbf{2.09 \pm 1.0}$	$\mathbf{0.596 \pm 0.3}$
reweigh 2-hop (Gauss)	4.63 ± 2.77	2.90 ± 2.24	3.12 ± 0.32

Table 6
Contaminated data without covariates. Means and standard deviations of the test MSEs over 100 replicated data sets. The response variable is generated by Eq. (10). Numbers highlighted in blue correspond to the best ones across all the methods evaluated, while the numbers in bold are the best across the SPONGE methods.

	Small	Medium	Large
OK	37.12 ± 243.69	7.19 ± 37.37	6.68 ± 29.45
KCN	16.99 ± 92.92	5.96 ± 40.31	1.78 ± 6.92
1-hop	16.26 ± 90.52	3.83 ± 19.35	1.94 ± 7.45
2-hop	17.35 ± 93.52	8.04 ± 45.24	2.18 ± 8.4
reweigh 2-hop	19.27 ± 96.66	10.12 ± 48.15	6.02 ± 28.9

size increases, both the mean and the standard deviation become better than for the Gaussian kernel. The model based on a B-spline weight function has the highest flexibility and gives overall the best performance. For the large data set, it is tied with the model based on the rational quadratic function however. The 1-hop model gives the best performance on the small dataset here again. It is consistently better than KCN across all the sizes, and even better than most of the 2-hops for the medium dataset. Therefore, adding more informative features is also a plus for the 1-hop model.

4.4. Simulation results in a Kriging-unfriendly world

We explore the performance of the predictors if the data is mis-specified for Kriging, meaning that a considerable part of the optimality conditions mentioned at the beginning of Section 4.1 is violated. For this, we investigate two rather natural causes of mis-specification: (1) the data is locally contaminated (has drastically different distribution on small patches of the observation window) and (2) the mean and covariance functions are globally not of the expected form.

In the first setting we generate a GRF $(Z(s))_{s\in[0,1]^2}$, say, this time with mean zero, variance $\sigma^2 = 3$ and Gaussian correlation functions with length scale 0.4. The locations s_i are still generated uniformly at random in $S = [0,1]^2$, but this time we designate the area

$$C = [0.1, 0.2]^2 \cup [0.45, 0.55]^2 \cup [0.8, 0.9]^2 \cup [0.1, 0.2] \times [0.8, 0.9] \cup [0.8, 0.9] \times [0.1, 0.2]$$

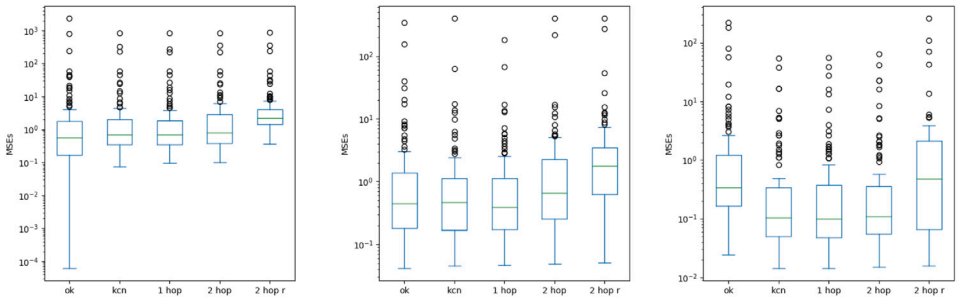
(forming the shape of a 5 on a die) as contaminated. We then set

$$Y_i = \begin{cases} Z(s_i) & \text{if } s_i \in C^c, \\ \exp(Z(s_i)) & \text{if } s_i \in C. \end{cases} \tag{10}$$

This artificially generated data set could be an example of data collected in an area that is locally polluted by chemical spills, and we assume the ordinary measurements to be normally distributed, whereas the spilled sites are log-normally distributed.

In Table 6, we see that OK displays mostly (much) higher mean test MSEs than the SPONGE methods. This comes from a few catastrophic errors due to the contaminated data, which OK can deal with (even) less well than the SPONGEs.

Fig. 4 shows the box plots of the MSEs on a logarithmic scale, revealing in particular a large number of outliers at the top. While this is the case for all of the methods, we can see that OK tends to have overall more and higher extreme outliers than the SPONGEs. We ignore here and in what follows the reweigh 2-hop model, which again performs badly for the large data set. While the central behavior (the median and the interquartile box) for OK are among the best for the small and medium data set, it also gets clearly beaten by the remaining SPONGEs for the large data set.



(a) small (b) medium (c) large

Fig. 4. Contaminated data set without covariates: Box plots of the MSEs on a log-scale.

Table 7
Mis-specified data with covariates. Means and standard deviations of the test MSEs over 100 replicated data sets. The response variable is generated by Eq. (11). Numbers highlighted in blue correspond the best ones across all the methods evaluated, while the numbers in bold are the best across the SPONGE methods.

	Small	Medium	Large
UK	10.52 ± 3.92	5.68 ± 2.14	3.91 ± 1.43
KCN	13.43 ± 4.75	6.21 ± 3.7	1.09 ± 0.21
1-hop	12.76 ± 11.22	4.58 ± 2.38	0.93 ± 0.24
2-hop	**12.56 ± 3.81**	7.73 ± 5.73	0.96 ± 0.29
reweigh 2-hop	13.31 ± 4.59	7.70 ± 4.84	1.67 ± 0.28

The one very small MSE of around 10^{-4} visible in Fig. 4(a) is due to the fact that by chance no points fall into the contaminated area C in this data set. Thus OK is applied under perfect conditions again, with a high correlation length scale of 0.4.

The box plots also seems to show that 1-hop is overall slightly less affected by outliers than the other methods. For the small and medium datasets, this is captured in Table 6 by the smaller means and standard deviations.

For the second Kriging-unfriendly setting, we consider a situation with covariates that have a non-linear influence on the target values and a mis-specified correlation function.

We generate three spatial covariates X_1, X_2, X_3 from zero mean isotropic GRFs having Gaussian covariance functions with variances $2, 2, 1$ and length scales $0.03, 0.17, 0.2$, respectively. We combine the three covariates non-linearly to obtain data (observed at $s_i \in S$ as usual)

$$Y_i = X_{i1} \sin\left(X_{i3}\right) + X_{i2} \cos\left(X_{i3}\right) + \tilde{Y}_{i1} + \tilde{Y}_{i2}, \quad i \in [n], \tag{11}$$

where the unobserved components ("error terms") come from two further independent centered GRFs $\tilde{Y}_1$, $\tilde{Y}_2$ having Gaussian covariance functions with variances 3, 10 and length scales 0.1 and 0.2, respectively. The sum of the two may be interpreted as a single centered GRF with a non-Gaussian covariance function that is the sum of the covariance functions of $\tilde{Y}_1$ and $\tilde{Y}_2$.

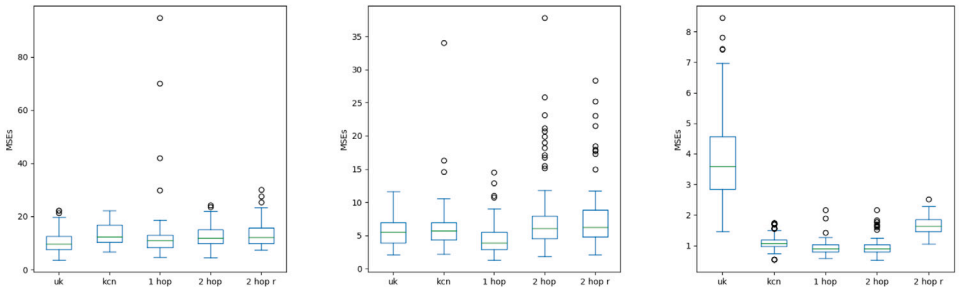
We run the models with covariates X_1, X_2, X_3, so from the point of view of Kriging the mis-specification is two-fold: the linear dependence on X_1, X_2, X_3 and the Gaussian covariance function that we continue to fit in the variography step. The results are given in Table 7.

As in the well-specified case with covariates, we observe again that the best performances among the SPONGE are achieved by one of our proposed generalizations, which even obtained the smallest MSE overall in the case of the medium and large sample sizes. The distribution of test MSEs are shown in Fig. 5. Despite some outliers recorded for the 1-hop model if the data size is small, we note that the overall performance remains among the best.

4.5. Evaluation of map drawing task

Rather than predicting the Y-values at individual new locations, it is often of interest in geostatistics to produce a map of the prediction over the whole area S. For completeness we evaluate the well-specified simulation data without covariates from Section 4.3 also on this map drawing task. For this we approximate the MISE (4) between the prediction and the true map by computing the MSE on a fine grid of 100×100 test locations within $S = [0, 1]^2$.

The results are given in Table 8, which should be compared with Table 4. It comes without surprise that the means in the two tables are very similar and have moved only more or less within their standard errors. The standard deviations of the MISEs have become considerably smaller than those in Table 4, because we have essentially removed the sampling error of the test set.



(a) *small* (b) *medium* (c) *large*

Fig. 5. Mis-specified data set with covariates: Box plots of MSEs.

Table 8
Well-specified setting for Kriging, *without* covariates. Means and standard deviations of (approximated) test MISEs over 100 replications each. Data from Section 4.3. Numbers highlighted in blue correspond the best ones across all the methods evaluated, while the numbers in bold are the best across the SPONGE methods.

	Small	Medium	Large
NN	6.03 ± 0.78	5.42 ± 0.35	3.01 ± 0.13
IDW	3.63 ± 0.27	3.13 ± 0.22	2.13 ± 0.11
NW	3.14 ± 0.16	3.00 ± 0.11	2.99 ± 0.1
OK	3.23 ± 0.26	2.88 ± 0.13	1.88 ± 0.07
KCN	3.44 ± 0.75	$\mathbf{3.03 \pm 0.14}$	$\mathbf{2.23 \pm 0.14}$
1-hop	3.72 ± 0.46	3.27 ± 0.79	2.61 ± 0.25
2-hop	3.53 ± 0.45	3.67 ± 0.29	2.9 ± 0.51
reweigh 2-hop	$\mathbf{3.32 \pm 0.4}$	$\mathbf{3.03 \pm 0.13}$	2.65 ± 0.1

5. Applications to real data

In this section we display the performance of SPONGE on some real world datasets.

5.1. Data sets studied in (Appleby et al., 2020)

We consider some real world data sets of much larger size to see whether the improvements of our other SPONGE over KCN prevail. These data sets were originally prepared and presented in Appleby et al. (2020). We thank the authors for making them available to us.

1. Bird count data set: a curated subset of wood thrush counts in June 2014 originating from the eBird project (Fink et al., 2010). The data set contains 107,246 records, 2D spatial coordinates and 19 covariates.
2. Precipitation data set: log-values of the average precipitation levels for the month of May from 1981 through 2010. Measurements were made across 8832 locations and the elevation is used as a covariate.
3. Restaurant rating data set: extracted from the business ratings on Yelp. A total of 188,586 restaurants with 31 covariates and 2D spatial coordinates were considered.

For each of these data sets, we use 50% of the locations for training and 50% for test. We run the released KCN version of the authors available on their GitHub page and perform hyperparameter tuning on each data set. Except for the precipitation dataset, the results obtained here conform with the ones reported by the authors.

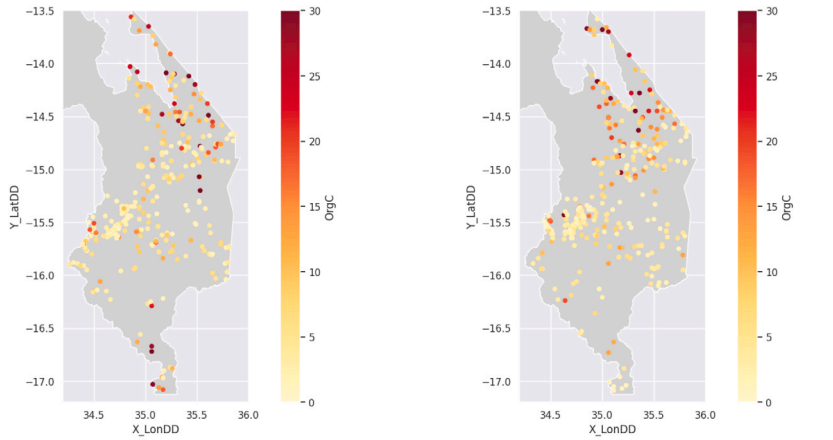
We see in Table 9 that the new SPONGE models mostly improve the performance of the baseline KCN. The restaurant rating data set is described in Appleby et al. (2020) as having uninformative features and weak spatial correlations, which makes it plausible that we should obtain more or less the same performance as with KCN. The small gains we obtain for the reweighed 2-hop and the 1-hop models with this data set may be real or they may be a statistical artifact of running three models against one.

5.2. Soil organic carbon prediction in Southern Malawi

We finally apply our SPONGE methods for the prediction of Soil Organic Carbon (SOC) with data obtained from the Africa Soil Profiles Database of the AFSIS (Africa Soil Information Service) project (Leenaars et al., 2014b). The database contains 14,197 georeferenced and standardized samples from 40 countries, collected between 1941 and 2011. Among the available countries, Malawi presents the highest number of data soil profiles, with the peak of collection in 1988. Therefore we only consider the data collected in 1988 for Malawi. This yielded 656 samples, which were randomly split into training and test sets of equal size; see Fig. 6. Seven samples were taken out of the training set to serve as a validation set.

Table 9
Test MSEs of different methods on some available real world datasets. Besides the test MSEs we also provide relative improvement in blue (decrease in MSE). In red we show the relative decrease in performance (increase in MSE).

	Bird count	Precipitation ($\times 10^{-2}$)	Restaurant rating
original KCN	0.49	3.64	1.00
1-hop	0.49	2.95 ▼**−18.96**%	0.97 ▲**−2**%
2-hop	0.44 ▼**−10.2**%	3.14 ▼ − 13.74%	1.02 ▲ + 2%
reweigh 2-hop	0.45 ▼ − 8.16%	3.00 ▼ − 17.58%	0.98 ▼ − 2%



(a) Training set *(b) Test set with true SOC values*

Fig. 6. Percentage of soil organic carbon in Southern Malawi (1988).

Table 10
Test MSEs of different methods on the SOC Malawi test set.

	KCN	1-hop	2-hop	reweigh 2-hop
MSE (test set)	40.03	38.05	37.10	39.4

SOC is an excellent indicator for soil quality. Stevens et al. (2013) highlights the difficulties of effectively assessing the SOC. In the following analysis, we use the amount of exchangeable calcium, magnesium, sodium, potassium, the base saturation and the (water based) pH of the soil as covariates; see Leenaars et al. (2014a).

The smallest test MSE on the SOC Malawi dataset is obtained by the 2-hop model (see Table 10). To keep the analysis simple, we will provide further visualizations for the top two models only, i.e. the 1-hop and the 2-hop.

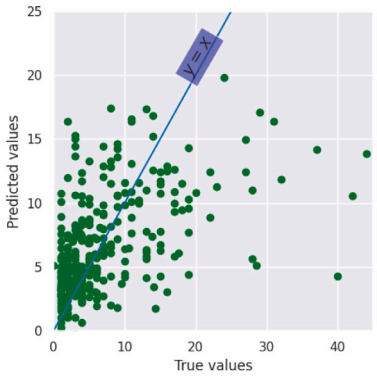
We observe on the scatter plots that the 1-hop model (Fig. 7(a)) tends to smooth out large values more than the 2-hop (Fig. 7(b)). In particular, we observe no predictions of 1-hop above 20. On the other hand, the 1-hop model generates also many smaller predictions for values that both models underestimate, which may well be the cause for the larger overall MSE.

The higher amount of smoothing in the large values is also visible in the maps of predicted SOC in Southern Malawi, especially in the northern half (Fig. 7(c) versus Fig. 7(d)).

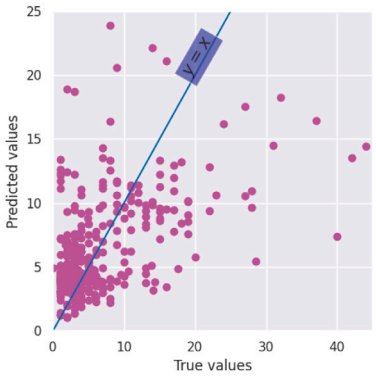
6. Conclusion

Many GCNs methods have been proposed in the literature that aim to achieve good performance for prediction with spatial data. We have presented in this paper a general formalism that we introduced as SPONGE (Spatial Prediction On Neighborhood Graph Entities), which describes and generalizes existing GCNs, including the possibility to:

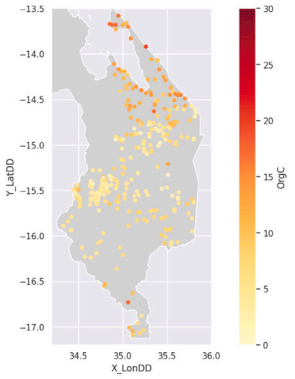
- choose the structure and the order of the neighborhood (Section 2);
- adjust the weight values of various graph edges by label/μ-closeness based reweighing (Section 2.2);
- capture spatial correlation within the data through a kernel which combines various length scales and whose weights are updated via backpropagation. A methodology to find optimal length scales is also proposed (See Section 3).



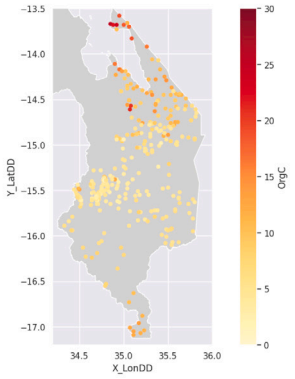
(a) Scatter plots of true vs predicted SOC values : 1-hop model.



(b) Scatter plots of true vs predicted SOC values : 2-hop model.



(c) Prediction map of SOC: 1 hop model.



(d) Prediction map of SOC: 2-hop model.

Fig. 7. Performance overview of top 2 SPONGE Models on the Malawi test set. We indicate the best MSE obtained across the various methods in blue.

While Kriging is the MSE minimizing linear unbiased predictor, it comes at a high computational cost of up to $O(n^3)$, whereas SPONGEs are much more easily scalable to large data sets. A major goal of this study was therefore to critically compare the prediction performance of SPONGE against Kriging. As our original point of departure was KCN (as one particular GCN based method), another major goal was to assess the gain obtained when using a SPONGE as opposed to the original KCN. By simulating what we call small, medium and large datasets in both a regime where all the Kriging conditions are satisfied (perfect Kriging world) and a regime where the Kriging model is mis-specified, we obtain the following findings:

1. In a perfect Kriging world, the MSE performance gap is naturally in favor of Kriging. Nonetheless, SPONGEs systematically decrease the MSEs of the original KCN and therefore come closer to the Kriging performance, especially when the number of covariates increases. We specifically point out the improvement obtained for the small dataset size, a context where neural networks are known to struggle (see Section 4.3).
2. As long as mis-specifications are not too extreme, the performance of Kriging remains halfway reasonable (see for example the second mis-specified example in Section 4.4). In small data examples it may still perform very well, because there is not enough data for a method that does not rely on wrong assumptions to do something better. For larger datasets, where other methods can recognize the structure within the data, MSEs recorded by SPONGEs outperform those obtained by Kriging in the predictive tasks.
   With more extreme mis-specifications on the other hand, it is much more easily possible to observe poor performance of Kriging and a clear MSE gain for the SPONGEs (see the first example of Section 4.4).
   In both cases, the MSE advantage of SPONGE versus Kriging is more or less equally observed in SPONGE versus KCN.

The scenarios in the simulation experiments were set up to cover some realistic situations both in a perfect Kriging world and in a Kriging-unfriendly world. In all of these scenarios we chose very specific parameters in a more or less ad-hoc way to create prediction problems of average difficulty and with average effect sizes (choosing e.g. Kriging-unfriendly effects, such as contamination, to be just about large enough to produce a desired result while remaining at a realistic level of here 5%). Of course many other combinations of parameters in the existing scenarios (and many more scenarios) are possible, and while we expect other settings of similar flavor to be consistent with the results presented here, we would like to warn against undue generalization.

In all of our simulation experiments the training data locations are sampled uniformly at random. Note that, if the locations are strongly inhomogeneous, using simple k-nearest-neighbor NGEs as in Section 4 may not be a good idea. In the extreme case of highly clustered sampling locations it may happen that the k nearest neighbors are always close-by for every training NGE but far away for predicting a label at a location far away from the observations, which would then become a very hard task.

One solution would be to adapt the prediction strategy of SPONGE, for instance, by not training only with nearest neighborhood data but also with NGEs based on a number of far-away observations. Another possibility could be to inform the network of the sparsity level around a test location within the data, e.g. by considering the smallest distance between the said test point and any neighboring point, and adjusting predictions for those with sparsity levels higher than a given threshold. Finally, one could think about an appropriate data augmentation technique during training with strongly inhomogeneously sampled data sets. One possibility is adding dummy observations at locations in sparsely sampled regions, with labels equal to the mean of response variables of the entire data sets / closest clusters, such that any time a test location falls within those regions, it uses the dummy observation to improve its prediction.

Case studies on real world datasets, whose sizes are typically (much) bigger than the simulated ones, mostly display a performance gain for SPONGE models in comparison to KCN (see Section 5), an observation which agrees with the findings on simulated data. We also highlight in particular the satisfactory results achieved by 1-hop, which is by far the smallest and most efficient of the SPONGEs. Future research should focus on evaluating the performance of SPONGE on data with many covariates, since they tend to perform better with a larger amount of informative features. In a similar vein, predicting multivariate labels, a task traditionally performed by multivariate Kriging, could be a task where SPONGE perform well.

Acknowledgment

We thank two anonymous reviewers for their pertinent comments that led to an improvement of the paper.

References

Abadi, M., Agarwal, A., Barham, P., Brevdo, E., Chen, Z., Citro, C., Corrado, G.S., Davis, A., Dean, J., Devin, M., Ghemawat, S., Goodfellow, I., Harp, A., Irving, G., Isard, M., Jia, Y., Jozefowicz, R., Kaiser, L., Kudlur, M., Levenberg, J., Mané, D., Monga, R., Moore, S., Murray, D., Olah, C., Schuster, M., Shlens, J., Steiner, B., Sutskever, I., Talwar, K., Tucker, P., Vanhoucke, V., Vasudevan, V., Viégas, F., Vinyals, O., Warden, P., Wattenberg, M., Wicke, M., Yu, Y., Zheng, X., 2015. TensorFlow: Large-scale machine learning on heterogeneous systems. Software available from tensorflow.org.

Akcin, H., Celik, C., 2013. Performance of artificial neural networks on Kriging method in modeling local geoid. Bol. Ciênc. Geod. 19, 84–97.

Allen, D.M., 1974. The relationship between variable selection and data agumentation and a method for prediction. Technometrics 16 (1), 125–127.

Altman, N.S., 1990. Kernel smoothing of data with correlated errors. J. Amer. Statist. Assoc. 85 (411), 749–759.

Appleby, G., Liu, L., Liu, L.-P., 2020. Kriging convolutional networks. In: Proceedings of the AAAI Conference on Artificial Intelligence. Vol. 34, pp. 3187–3194, (04).

Atwood, J., Towsley, D., 2016. Diffusion-convolutional neural networks. In: Lee, D., Sugiyama, M., Luxburg, U., Guyon, I., Garnett, R. (Eds.), Advances in Neural Information Processing Systems. Vol. 29, Curran Associates, Inc..

Bishop, C.M., 2006. Pattern Recognition and Machine Learning. Springer.

Chen, W., Li, Y., Reich, B.J., Sun, Y., 2020. Deepkriging: Spatially dependent deep neural networks for spatial prediction. arXiv preprint arXiv:2007.11972.

Chilès, J.-P., Delfiner, P., 2012. second ed. Geostatistics: Modeling Spatial Uncertainty, vol. 713, John Wiley & Sons.

Chu, C.-K., Marron, J.S., 1991. Comparison of two bandwidth selectors with dependent errors. Ann. Statist. 19 (4).

Coppersmith, D., Winograd, S., 1990. Matrix multiplication via arithmetic progressions. J. Symbolic Comput. 9, 251–280.

Craven, P., Wahba, G., 1978. Smoothing noisy data with spline functions. Numer. Math. 31 (4), 377–403.

Cressie, N., 2015. Statistics for Spatial Data. John Wiley & Sons.

Cressie, N., Johannesson, G., 2008. Fixed rank kriging for very large spatial data sets. J. R. Stat. Soc. Ser. B Stat. Methodol. 70 (1), 209–226.

de Boor, C., 1971. Subroutine package for calculating with B-splines.

De Brabanter, K., De Brabanter, J., Suykens, J., De Moor, B., 2010. Kernel regression with correlated errors. IFAC Proc. Vol. 43 (6), 13–18.

Defferrard, M., Bresson, X., Vandergheynst, P., 2016. Convolutional neural networks on graphs with fast localized spectral filtering. Adv. Neural Inf. Process. Syst. 29.

Deisenroth, M.P., Ong, C.S., Faisal, A.A., 2021. Mathematics for Machine Learning. Cambridge University Press.

Duvenaud, D.K., Maclaurin, D., Iparraguirre, J., Bombarell, R., Hirzel, T., Aspuru-Guzik, A., Adams, R.P., 2015. Convolutional networks on graphs for learning molecular fingerprints. In: Cortes, C., Lawrence, N.D., Lee, D.D., Sugiyama, M., Garnett, R. (Eds.), Advances in Neural Information Processing Systems 28. Curran Associates, Inc., pp. 2224–2232.

Fadel, S., Said, G., Abdallah, M., Abu, H., 2016. Investigating the effect of different kernel functions on the performance of SVM for recognizing arabic characters. Int. J. Adv. Comput. Sci. Appl. 7.

Fengping, L., Yuqing, Z., Wei, X., 2016. Parameter optimization for Nadaraya-Watson kernel regression method with small samples. Int. J. Adv. Res. Artif. Intell. 5 (5).

Fink, D., Hochachka, W.M., Zuckerberg, B., Winkler, D.W., Shaby, B., Munson, M.A., Hooker, G., Riedewald, M., Sheldon, D., Kelling, S., 2010. Spatiotemporal exploratory models for broad-scale survey data. Ecol. Appl. 20 (8), 2131–2147.

Furrer, R., Genton, M.G., Nychka, D., 2006. Covariance tapering for interpolation of large spatial datasets. J. Comput. Graph. Statist. 15 (3), 502–523.

Gao, B., Stein, A., Wang, J., 2022. A two-point machine learning method for the spatial prediction of soil pollution. Int. J. Appl. Earth Obs. Geoinf. 108, 102742.

Geisser, S., 1975. The predictive sample reuse method with applications. J. Amer. Statist. Assoc. 70 (350), 320–328.

Gribov, A., Krivoruchko, K., 2004. Geostatistical mapping with continuous moving neighborhood. Math. Geol. 36, 267–281.

Györfi, L., Härdle, W., Sarda, P., Vieu, P., 1989. Regression estimation and time series analysis. In: Nonparametric Curve Estimation from Time Series. pp. 15–51.

Hall, P., Lahiri, S.N., Polzehl, J., 1995. On bandwidth choice in nonparametric regression with both short- and long-range dependent errors. Ann. Statist. 23 (6).

Hamilton, W., Ying, Z., Leskovec, J., 2017. Inductive representation learning on large graphs. Adv. Neural Inf. Process. Syst. 1024–1034.

Hammond, D.K., Vandergheynst, P., Gribonval, R., 2011. Wavelets on graphs via spectral graph theory. Appl. Comput. Harmon. Anal. 30 (2), 129–150.

Härdle, W., Hall, P., Marron, J.S., 1988. How far are automatically chosen regression smoothing parameters from their optimum? J. Amer. Statist. Assoc. 83 (401), 86.

Hengl, T., Heuvelink, G.B., Rossiter, D.G., 2007. About regression-kriging: From equations to case studies. Comput. Geosci. 33 (10), 1301–1315.

Hrafnkelsson, B., Cressie, N., 2003. Hierarchical modeling of count data with application to nuclear fall-out. Environ. Ecol. Stat. 10, 179–200.
Ibrahim, D.M., Nasser, R.H.A., 2017. Comparison between Inverse Distance Weighted (IDW) and Kriging.
Ji, F., Yang, J., Zhang, Q., Tay, W.P., 2020. GFCN: A new graph convolutional network based on parallel flows. In: ICASSP 2020 - 2020 IEEE International Conference on Acoustics, Speech and Signal Processing. ICASSP, pp. 3332–3336.
Kang, L., Joseph, V.R., 2016. Kernel approximation: From regression to interpolation. SIAM/ASA J. Uncertain. Quantif. 4 (1), 112–129.
Khouni, I., Louhichi, G., Ghrabi, A., 2021. Use of GIS based Inverse Distance Weighted interpolation to assess surface water quality: Case of Wadi El Bey, Tunisia. Environ. Technol. Innov. 24, 101892.
Kingma, D.P., Ba, J., 2015. Adam: A method for stochastic optimization. In: Bengio, Y., LeCun, Y. (Eds.), 3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings.
Kipf, T.N., Welling, M., 2017. Semi-supervised classification with graph convolutional networks. In: Proceedings of the 5th International Conference on Learning Representations. ICLR '17.
Laderman, J., Pan, V., Sha, X.-H., 1992. On practical algorithms for accelerated matrix multiplication. Linear Algebra Appl. 162, 557–588.
Lee, Y.K., Mammen, E., Park, B.U., 2010. Bandwidth selection for kernel regression with correlated errors. Statistics 44 (4), 327–340.
Leenaars, J.G.B., Kempen, B., van Oostrum, A., Batjes, N., 2014a. Africa Soil Profiles database, version1.2. A compilation of georeferenced and standardised legacy soil profile data for Sub-Saharan countries - ISRIC Report 2014/01.
Leenaars, J.G.B., van Oostrum, A., Gonzalez, M.R., 2014b. Africa Soil Profiles database, version1.2. A compilation of georeferenced and standardised legacy soil profile data for Sub-Saharan countries (with dataset).
Lei, H., Akhtar, N., Mian, A., 2021. Spherical kernel for efficient graph convolution on 3D point clouds. IEEE Trans. Pattern Anal. Mach. Intell. 43 (10), 3664–3680.
Li, Y., Tarlow, D., Brockschmidt, M., Zemel, R.S., 2016. Gated graph sequence neural networks. In: Bengio, Y., LeCun, Y. (Eds.), 4th International Conference on Learning Representations, ICLR 2016, San Juan, Puerto Rico, May 2-4, 2016, Conference Track Proceedings.
Li, Y., Yu, R., Shahabi, C., Liu, Y., 2017. Diffusion convolutional recurrent neural network: Data-driven traffic forecasting. arXiv preprint arXiv:1707.01926.
Lindgren, F., Rue, H., Lindström, J., 2011. An explicit link between Gaussian fields and Gaussian Markov random fields: the stochastic partial differential equation approach. J. R. Stat. Soc. Ser. B Stat. Methodol. 73 (4), 423–498.
Mallows, C.L., 1973. Some comments on C P. Technometrics 15 (4), 661.
Marron, J.S., 1987. Partitioned cross-validation. Econometric Rev. 6 (2), 271–283.
Monti, F., Boscaini, D., Masci, J., Rodola, E., Svoboda, J., Bronstein, M.M., 2017. Geometric deep learning on graphs and manifolds using mixture model cnns. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 5115–5124.
Mueller, S., Schueler, L., Zech, A., Hesse, F., 2021. GSTools v1.3: A toolbox for geostatistical modelling in Python.
Murphy, B.S., 2014. PyKrige: Development of a kriging toolkit for Python. In: AGU Fall Meeting Abstracts. Vol. 2014, pp. H51K–0753.
Nadaraya, E.A., 1964. On estimating regression. Theory Probab. Appl. 9 (1), 141–142.
Pohjankukka, J., Pahikkala, T., Nevalainen, P., Heikkonen, J., 2017. Estimating the prediction performance of spatial models via spatial K-fold cross validation. Int. J. Geogr. Inf. Sci. 31 (10), 2001–2019.
Ray, B., 1997. Bandwidth selection for kernel regression with long-range dependent errors. Biometrika 84 (4), 791–802.
Rivoirard, J., Romary, T., 2011. Continuity for kriging with moving neighborhood. Math. Geosci. 43, 469–481.
Rodrigues, B.P., Rofatto, V.F., Matsuoka, M.T., Assunção, T.T., 2022. Resampling in neural networks with application to spatial analysis. Geo-Spat. Inf. Sci. 1–12.
Rue, H., Held, L., 2005. Gaussian Markov Random Fields: Theory and Applications. CRC Press.
Rue, H., Tjelmeland, H., 2002. Fitting Gaussian Markov random fields to Gaussian fields. Scand. J. Stat. 29 (1), 31–49.
Sagar, D., Cheng, Q., Agterberg, F. (Eds.), 2018. Handbook of Mathematical Geosciences: Fifty Years of IAMG. Springer Nature.
Schloeder, C., Zimmerman, N., Jacobs, M., 2001. Comparison of methods for interpolating soil properties using limited data. Soil Sci. Am. J. 65 (2), 470–479.
Semenova, E., Xu, Y., Howes, A., Rashid, T., Bhatt, S., Mishra, S., Flaxman, S., 2022. PriorVAE: encoding spatial priors with variational autoencoders for small-area estimation.
Setianto, A., Triandini, T., 2015. Comparison of Kriging and inverse distance weighted (IDW) interpolation methods in lineament extraction and analysis. J. Appl. Geol. 5 (1).
Shepard, D., 1968. A two-dimensional interpolation function for irregularly-spaced data. In: Proceedings of the 1968 23rd ACM National Conference. ACM '68, Association for Computing Machinery, New York, NY, USA, pp. 517–524.
Smith, N.J., Wardrop, M., Hudon, C., broessli, Quackenbush, P., Seabold, S., Portnoy, A., Davidson-Pilon, C., Kibirige, H., Leinweber, K., Humber, M., Hudson-Doyle, M., Korenciak, M., Gates, T., 2018. pydata/patsy: v0.5.1.
Stein, M.L., 1999. Linear prediction. In: Interpolation of Spatial Data: Some Theory for Kriging. Springer New York, New York, NY, pp. 1–14.
Stevens, A., Nocita, M., Tóth, G., Montanarella, L., van Wesemael, B., 2013. Prediction of soil organic carbon at the European scale by visible and near infrared reflectance spectroscopy. PLoS One 8 (6), e66409.
Stoer, J., Bulirsch, R., 1993. Introduction to Numerical Analysis, second ed. Springer.
Stone, M., 1974. Cross-validatory choice and assessment of statistical predictions. J. R. Stat. Soc. Ser. B Stat. Methodol. 36 (2), 111–133.
Tang, Q., Niu, L., Wang, Y., Dai, T., An, W., Cai, J., Xia, S.-T., 2017. Student-t process regression with student-t likelihood. In: Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence, IJCAI-17. pp. 2822–2828.
Van Lieshout, M., 2019. Theory of Spatial Statistics: A Concise Introduction, first ed. Taylor & Francis.
Van Rossum, G., Drake, Jr., F.L., 1995. Python Reference Manual. Centrum voor Wiskunde en Informatica Amsterdam.
Veličković, P., Cucurull, G., Casanova, A., Romero, A., Liò, P., Bengio, Y., 2017. Graph attention networks. In: 6th International Conference on Learning Representations.
Wackernagel, H., 2003. Multivariate Geostatistics: An Introduction with Applications. Springer Science & Business Media.
Wadoux, A., Heuvelink, G., de Bruin, S., Brus, D., 2021. Spatial cross-validation is not the right way to evaluate map accuracy. Ecol. Model. 457, 109692.
Wang, H., Guan, Y., Reich, B., 2019. Nearest-neighbor neural networks for geostatistics. In: 2019 International Conference on Data Mining Workshops. ICDMW, pp. 196–205.
Watson, G.S., 1964. Smooth regression analysis. Sankhyā: Indian J. Stat. Ser. A (1961-2002) 26 (4), 359–372.
Whang, Y.-J., 1998. Topics in advanced econometrics: estimation, testing and specification of cross-section and time series models. Econom. Theory 14 (3), 369–374.
Wu, Y.-H.E., Hung, M.-C., 2016. Comparison of spatial interpolation techniques using visualization and quantitative assessment. In: Hung, M.-C. (Ed.), Applications of Spatial Statistics. IntechOpen, Rijeka, chapter 2.
Wu, Y., Zhuang, D., Labbe, A., Sun, L., 2020. Inductive graph neural networks for spatiotemporal kriging. arXiv preprint arXiv:2006.07527.
Wu, Y., Zhuang, D., Lei, M., Labbe, A., Sun, L., 2021. Spatial aggregation and temporal convolution networks for real-time kriging. arXiv preprint arXiv:2109.12144.
Xie, T., Grossman, J.C., 2018. Crystal graph convolutional neural networks for an accurate and interpretable prediction of material properties. Phys. Rev. Lett. 120, 145301.
Yakowitz, S., Szidarovszky, F., 1985. A comparison of kriging with nonparametric regression methods. J. Multivariate Anal. 16 (1), 21–53.
Zhu, J., Yan, Y., Zhao, L., Heimann, M., Akoglu, L., Koutra, D., 2020. Beyond homophily in graph neural networks: Current limitations and effective designs. In: Proceedings of the 34th International Conference on Neural Information Processing Systems. NIPS '20, Curran Associates Inc., Red Hook, NY, USA.