Cross-Validation for spatial data

Cristina Chavez-Chong

Chapter 1: Cross-Validation for spatial data

February 7, 2025

1 Preliminaries

1.1 Regression problem in fixed design setting

The classical regression problem involves estimating an unknown function f based on observed data $\mathcal{D}_n = \{(\mathbf{X}_i, Y_i), i = 1, \ldots, n\}$. This problem can be considered under two different frameworks: the *fixed design* setting and the *random design* setting.

In the fixed design setting, the $\mathbf{X}_i$ are considered fixed and non-random, and the randomness arises solely from the noise in the observations. The model is defined as,

$$Y_i = f(\mathbf{X}_i) + \varepsilon_i, \quad i = 1, \ldots, n, \tag{1}$$

where $\mathbf{X}_i \in \mathbb{R}^{p+1}$ is the covariate vector for the i-th observation, which includes a constant term (intercept) and p fixed covariates, $Y_i \in \mathbb{R}$ are the realizations of the dependent variable, ε_i are independent and identically distributed (i.i.d.) noise terms with $\mathbb{E}[\varepsilon_i] = 0$ and $\mathbb{E}[\varepsilon_i^2] = \sigma^2$.

In contrast, the random design setting treats the covariates $\mathbf{X}_i$ as random variables drawn from a probability distribution P_X. Bach2020LearningTF discusses empirical risk minimization for regression problems for both random and fixed design. In this work, we focus on the fixed design setting. We are particularly interested in the case where f is linear in some parameters $\boldsymbol{\beta}$, and of the form $f(\mathbf{X}) = \mathbf{X}\boldsymbol{\beta}$, where $\boldsymbol{\beta} \in \mathbb{R}^{p+1}$ is a vector of coefficients to be estimated.

The linear regression model in equation (1) can be rewritten in matrix form,

$$\mathbf{Y} = \mathbf{X}\boldsymbol{\beta} + \boldsymbol{\varepsilon}, \tag{2}$$

where $\mathbf{Y} = (Y_1, Y_2, \ldots, Y_n)^\top \in \mathbb{R}^n$ is the vector of responses, $\mathbf{X} \in \mathbb{R}^{n \times (p+1)}$ is the design matrix, with the i-th row equal to $\mathbf{X}_i$ and $\boldsymbol{\varepsilon} = (\varepsilon_1, \varepsilon_2, \ldots, \varepsilon_n)^\top$ is the vector of noise terms.

Under these assumptions, we address the regression problem by obtaining an estimator $\hat{\boldsymbol{\beta}}$ of $\boldsymbol{\beta}$ by minimizing the fixed design *risk*,

$$\mathcal{R}(\boldsymbol{\beta}) = \mathbb{E}_{\mathbf{Y}} \left[\frac{1}{n} \sum_{i=1}^{n} \ell(Y_i, \mathbf{X}_i\boldsymbol{\beta}) \right] \tag{3}$$

where $\ell : \mathbb{R} \times \mathbb{R} \longleftarrow \mathbb{R}$ is called the *loss function*. In what follows we will consider the quadratic loss function $\ell(\mathbf{Y}, \mathbf{Y}') = (\mathbf{Y} - \mathbf{Y}')^2$. In the regression framework, the quadratic loss is particularly useful for several reasons. The quadratic loss is convex and differentiable, which simplifies the optimization problem and allows for its derivation of explicit solutions.

In this work, we are interested in estimating the predictive risk associated with an estimator $\hat{\boldsymbol{\beta}}$ obtained from the sample set $\mathcal{D}_n = \{(\mathbf{X}_i, Y_i), i = 1, \ldots, n\}$, focusing particularly on its performance in predicting new observations. This problem has been extensively studied in the statistical literature, with significant contributions by Breiman1992TheLB in the fixed design setting and more recently by rosset2020fixed in the random design context.

Suppose that $\mathbf{Y}^{\text{new}} = \mathbf{X}\boldsymbol{\beta} + \boldsymbol{\varepsilon}^{\text{new}}$ represents a new independent realization of the dependent variable, with the same fixed design matrix $\mathbf{X}$ and a new independent noise $\boldsymbol{\varepsilon}^{\text{new}}$, where $\boldsymbol{\varepsilon}^{\text{new}}$ has the same distribution as $\boldsymbol{\varepsilon}$ and is independent of it. The *predictive risk* associated with an estimator $\hat{\boldsymbol{\beta}}$ obtained from the data $\mathcal{D}_n$ is then defined as,

$$\mathcal{R}_{\text{pred}}(\hat{\boldsymbol{\beta}}) = \mathbb{E}_{\mathbf{Y}, \mathbf{Y}^{\text{new}}} \left[\frac{1}{n} \left\| \mathbf{X}\hat{\boldsymbol{\beta}} - \mathbf{Y}^{\text{new}} \right\|^2 \right], \tag{4}$$

where the subscripts on the expectation indicate the random variables over which the expectation is taken.

However, in practice, since the true parameter $\boldsymbol{\beta}$ and the noise terms are unknown, and we typically do not have access to new independent data $\mathbf{Y}^{\text{new}}$, we cannot compute $\mathcal{R}_{\text{pred}}(\hat{\boldsymbol{\beta}})$ directly. Instead, we often estimate the predictive risk using an empirical approximation based on the observed data.

$$\mathcal{R}_n(\hat{\boldsymbol{\beta}}) = \frac{1}{n} \left\| \mathbf{Y} - \mathbf{X}\hat{\boldsymbol{\beta}} \right\|^2. \tag{5}$$

The empirical risk $\mathcal{R}_n(\hat{\boldsymbol{\beta}})$ defined in equation (5) measures the average squared error between the observed responses $\mathbf{Y}$ and the predictions $\mathbf{X}\hat{\boldsymbol{\beta}}$ on the available data $\mathcal{D}_n$. From now on we will write $\mathcal{R}_n$ instead of $\mathcal{R}_n(\hat{\boldsymbol{\beta}})$ to simplify notations.

One common way to estimate the predictive risk is through cross-validation. Cross-validation involves partitioning the observed data into training and validation sets multiple times, effectively mimicking the process of predicting new independent observations. By training the model on one subset of the data and validating it on another, we can obtain an unbiased estimate of the model's predictive performance.

In the following section, we explore classical cross-validation techniques used for risk estimation in regression analysis.

1.2 Classical cross-validation

As stated by arlot2010survey the universality of the data splitting heuristics, attracts a major interest to cross-validation since this method only assumes

that data are identically distributed, and training and validation samples are independent.

We start with the so-called *hold-out* (HO) procedure. Let us divide the sample set $\mathcal{D}_n = \{(\mathbf{X}_i, Y_i), i = 1, \ldots, n\}$ into two separate independent non-empty subsets $\mathcal{D}_n^{\text{train}}$ and $\mathcal{D}_n^{\text{val}}$ such that $\mathcal{D}_n^{\text{train}} \cup \mathcal{D}_n^{\text{val}} = \mathcal{D}_n$ and $\mathcal{D}_n^{\text{train}} \cap \mathcal{D}_n^{\text{val}} = \emptyset$. $\mathcal{D}_n^{\text{train}}$ is called the training set and is used to estimate $\boldsymbol{\beta}$, while $\mathcal{D}_n^{\text{val}}$ is called the validation set and is used to estimate the risk. From now, we write $\mathcal{D}_n^{\text{t}}$ for $\mathcal{D}_n^{\text{train}}$, $\mathcal{D}_n^{\text{v}}$ for $\mathcal{D}_n^{\text{val}}$, and $\hat{\boldsymbol{\beta}}^{\text{t}}$ instead of $\hat{\boldsymbol{\beta}}^{\text{train}}$ to simplify notations.

We estimate $\boldsymbol{\beta}$ by $\hat{\boldsymbol{\beta}}^{\text{t}}$ using observations in $\mathcal{D}_n^{\text{t}}$, typically by minimizing:

$$\frac{1}{\text{Card}(\mathcal{D}_n^{\text{t}})} \sum_{(\mathbf{X}_i, Y_i) \in \mathcal{D}_n^{\text{t}}} (Y_i - \mathbf{X}_i \boldsymbol{\beta})^2.$$

Then we test the model's prediction ability induced by $\hat{\boldsymbol{\beta}}^{\text{t}}$ by computing the hold-out empirical risk:

$$\mathcal{R}_n^{\text{HO}} = \mathcal{R}_n^{\text{V}} = \frac{1}{\text{Card}(\mathcal{D}_n^{\text{v}})} \sum_{(\mathbf{X}_i, Y_i) \in \mathcal{D}_n^{\text{v}}} (Y_i - \mathbf{X}_i \hat{\boldsymbol{\beta}}^{\text{t}})^2. \tag{6}$$

We note here that the hold-out empirical risk depends on how the data has been split into the training and validation sets.

The *K-fold cross-validation* (KCV) algorithm is closely related to the hold-out method. It partitions $\mathcal{D}_n$ into K subsets $\{\mathcal{D}_1, \ldots, \mathcal{D}_K\}$, with observations randomly assigned to each subset, but ensuring that the sizes of the subsets are approximately equal, that is $\text{Card}(\mathcal{D}_m) = n_m$, with $\left\lfloor \frac{n}{K} \right\rfloor \leq n_m \leq \left\lceil \frac{n}{K} \right\rceil$. For $m = 1, \ldots, K$, successively, the hold-out risk $\mathcal{R}_n^{\text{V}_m}$ defined in (6) is calculated on $D_n^{\text{v}} = \mathcal{D}_m$, with $\boldsymbol{\beta}$ estimated by $\hat{\boldsymbol{\beta}}^{\text{t}_m}$ using $D_n^{\text{t}} = \bigcup_{i \neq m} \mathcal{D}_i$. The K-fold cross-validation empirical risk is the average of the K hold-out empirical risks:

$$\mathcal{R}_n^{\text{KCV}} = \frac{1}{K} \sum_{m=1}^{K} \mathcal{R}_n^{\text{v}_m}. \tag{7}$$

K-fold cross validation is usually preferred over the hold-out method because it trains the model on K different training-validation subsets splits. This provides a better indication of how well the model performs on unseen data. Moreover, it has been shown that the estimates obtained by minimizing (7) improve in terms of both bias and variance compared to those obtained via simple hold-out (see for instance james2013introduction).

When $K = n$, K-fold is called *leave-one-out* (LOO). In each iteration of LOO, one observation becomes the validation set and the remaining $n - 1$ observations are used to train the model. For $m \in \{1, \ldots n\}$, we have $\mathcal{D}_n^{\text{t}_m} =$

$\{(\mathbf{X}_m, Y_m)\}^c$, the complement of the singleton $\mathcal{D}_n^{\mathrm{v}_m} = \{(\mathbf{X}_m, Y_m)\}$. The leave-one-out cross-validation risk is then given by:

$$\mathcal{R}_n^{\mathrm{LOO}} = \frac{1}{n}\sum_{m=1}^{n} \mathcal{R}_n^{\mathrm{v}_m} = \frac{1}{n}\sum_{m=1}^{n} (Y_m - \mathbf{X}_m \hat{\boldsymbol{\beta}}^{\mathrm{t}_m})^2, \tag{8}$$

$\hat{\boldsymbol{\beta}}^{\mathrm{t}_m}$ being estimated in $\mathcal{D}_n^{\mathrm{t}_m}$.

1.3 Basic principle of spatial cross-validation

Classical cross-validation assumes that data are identically distributed and that training and validation samples are independent. However, in the context of spatial data, as noted by tobler1970computer, the behavior of a variable at a certain location is influenced by its neighbors due to spatial autocorrelation. Consequently, randomly splitting the data does not guarantee independence between training and validation sets, which can lead to overly optimistic risk estimates and inflated assessments of model performance.

Our illustration in Figure 1 displays the empirical risks under the following framework. We consider the general linear model $Y(\mathbf{s}) = \mathbf{X}(\mathbf{s})\boldsymbol{\beta} + \varepsilon(\mathbf{s})$ where $\mathbf{X}(\mathbf{s})\boldsymbol{\beta}$ represents the large-scale, deterministic mean structure of the process, and $\varepsilon(\mathbf{s})$ is the small-scale stochastic component modeling the spatial dependence among the data. We assume $\boldsymbol{\varepsilon} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{\Sigma})$. We generate 100 datasets and estimate $\boldsymbol{\beta}$ using the ordinary least squares procedure. Then we compute $\mathcal{R}_n^{\mathrm{KCV}}$ and its spatial version $\mathcal{R}_n^{\mathrm{SKCV}}$ with $K = 200$ and display the 100 results in boxplots diagrams.

Let us denote $\|\boldsymbol{\alpha}\|^2 = \boldsymbol{\alpha}^\top \boldsymbol{\alpha} = \sum_{i=1}^n \alpha_i^2$ for an $n-$dimensional vector $\boldsymbol{\alpha}$. The *ordinary least squares* (OLS) estimator is defined by:

$$\hat{\boldsymbol{\beta}}_{\mathrm{O}} = \underset{\boldsymbol{\beta}\in\mathbb{R}^{p+1}}{\arg\min} \left(\frac{1}{n} \|\mathbf{Y} - \mathbf{X}\boldsymbol{\beta}\|^2 \right) = \left(\mathbf{X}^\top \mathbf{X}\right)^{-1} \mathbf{X}^\top \mathbf{Y}, \tag{9}$$

provided that $\mathbf{X}^\top\mathbf{X}$ is invertible. When the noise terms are assumed to be normally distributed, minimizing the quadratic loss corresponds to maximum likelihood estimation.

The estimates are derived from 100 simulated datasets generated under the model $Y(\mathbf{s}) = \mathbf{X}(\mathbf{s})\boldsymbol{\beta} + \varepsilon(\mathbf{s})$, where the error terms $\boldsymbol{\varepsilon}$ are generated according to a zero-mean Gaussian spatial process with an isotropic exponential covariance function $C(h) = \sigma^2 \exp\left(-\frac{h}{\theta}\right)$. More details about the simulation design are provided in Section 2.4.

The left panel of Figure 1 displays results for $\theta = 3$, representing low spatial dependence, while the right panel shows results for $\theta = 12$, indicating high spatial dependence. In both cases, we observe that the risk estimates obtained using traditional K-fold cross-validation are lower than those obtained using Spatial K-fold cross-validation. This suggests that spatial cross-validation methods provide less optimistic risk estimates compared to traditional methods. However,

at this stage, the validity of these spatial cross-validation estimates remains to be determined.

These initial observations motivate the need to further investigate the effectiveness of spatial cross-validation methods in providing accurate and reliable risk estimates in the presence of spatial autocorrelation. Throughout this chapter, we will explore these methods and evaluate their performance through simulation studies, aiming to establish their validity and practical utility in spatial data analysis.

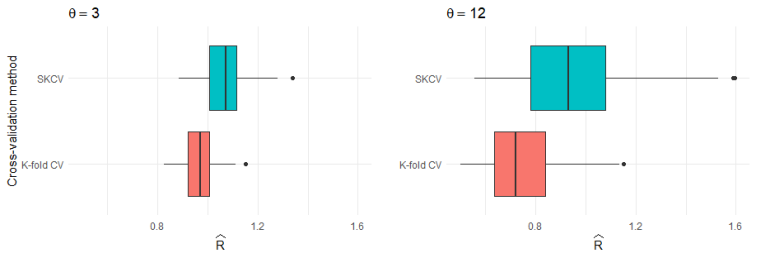


Figure 1: Risk estimates associated with the OLS estimator obtained using K-fold CV and Spatial K-fold CV (SKCV) methods, with $K = 200$.

Let us consider a spatial domain $\mathcal{S}_n = \{\mathbf{s}_1, \ldots, \mathbf{s}_n\} \subset \mathbb{R}^d$, where our sample data $\mathcal{D}_n = \{(\mathbf{X}(\mathbf{s}_i), Y(\mathbf{s}_i)) \,|\, \mathbf{s}_i \in \mathcal{S}_n\}$ are collected. We consider the following regression model:

$$Y(\mathbf{s}) = \mathbf{X}(\mathbf{s})\boldsymbol{\beta} + \varepsilon(\mathbf{s}), \tag{10}$$

where $\mathbf{X}(\mathbf{s})\boldsymbol{\beta}$ represents the large-scale, deterministic mean structure of the process, and $\varepsilon(\mathbf{s})$ is the small-scale stochastic component modeling the spatial dependence among the data. Typically, we assume $\boldsymbol{\varepsilon} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{\Sigma})$. Let us note that this general formulation allows for non-stationarity in the process, but it also encompasses the stationary case when $\mathbf{X}(\mathbf{s}) = \text{const}$.

Suppose we wish to estimate the predictive risk of an estimator of $\boldsymbol{\beta}$ obtained from our sample data $\mathcal{D}_n$. When the errors exhibit correlation, the Ordinary Least Squares (OLS) estimator remains unbiased but is no longer efficient, and standard error estimates may be inconsistent, leading to invalid inference. To account for the covariance structure of the errors, we can use the *Generalized Least Squares* (GLS) estimator. Assuming that the covariance matrix $\boldsymbol{\Sigma}$ is known, the GLS estimator is defined as:

$$\hat{\boldsymbol{\beta}}_{\text{GLS}} = \left(\mathbf{X}^\top \boldsymbol{\Sigma}^{-1} \mathbf{X}\right)^{-1} \mathbf{X}^\top \boldsymbol{\Sigma}^{-1} \mathbf{Y}. \tag{11}$$

In practice, $\boldsymbol{\Sigma}$ is often unknown and must be estimated from the data, leading to the *Feasible GLS* (FGLS) estimator where $\hat{\boldsymbol{\Sigma}}$ is a consistent estimate of the true covariance matrix,

$$\hat{\boldsymbol{\beta}}_{\text{FGLS}} = \left(\mathbf{X}^\top \hat{\boldsymbol{\Sigma}}^{-1} \mathbf{X}\right)^{-1} \mathbf{X}^\top \hat{\boldsymbol{\Sigma}}^{-1} \mathbf{Y}. \tag{12}$$

However, even with GLS, the issue of spatial dependence affects the evaluation of predictive performance through cross-validation. Spatial cross-validation involves redefining how we partition the data within the spatial domain $\mathcal{S}_n$. The key modification revolves around the principle of "point separation", which involves strategically removing observations from the training set to reduce dependence between the training and validation sets. This is achieved by delineating three spatial domains: the validation domain, the training domain, and a separating domain called the "buffer" or "dead zone", that provides a buffer zone to mitigate spatial dependence.

Let us consider the isotropic case, where spatial independence between observations can be achieved by establishing a buffer zone at the same distance in all directions. Let us denote h the necessary distance between the validation domain and the training domain that satisfies the independence assumption. For a chosen validation domain $\mathcal{S}_n^{\mathrm{v}} \subset \mathcal{S}_n$, we define a new subset of sites $\mathcal{S}_{n,h}^{\mathrm{D}}$. This set consists of all sites that are not part of the validation domain but are within a distance h of at least one site in the validation domain. Formally, the dead zone or *buffer* is defined as,

$$\mathcal{S}_{n,h}^{\mathrm{D}} = \{\mathbf{s}_j \notin \mathcal{S}_n^{\mathrm{v}} : \ \exists\ \mathbf{s}_i \in \mathcal{S}_n^{\mathrm{v}} : d(\mathbf{s}_j, \mathbf{s}_i) \leq h\}.$$

The training domain $\mathcal{S}_{n,h}^{\mathrm{t}}$ then comprises all sites not in $\mathcal{S}_n^{\mathrm{v}}$ or $\mathcal{S}_{n,h}^{\mathrm{D}}$, and is represented as

$$\mathcal{S}_{n,h}^{\mathrm{t}} = \mathcal{S}_n \setminus (\mathcal{S}_n^{\mathrm{v}} \cup \mathcal{S}_{n,h}^{\mathrm{D}}).$$

It is important to note that the choice of distance h is crucial as it determines the effectiveness of achieving independence between the training and validation domains. Therefore, determining h accurately is one of the primary objectives of this dissertation.

The process can be outlined algorithmically as follows. Assuming h has been determined, we proceed with the m-th iteration of the cross-validation procedure in three steps:

1. **Select the validation set:** We choose the validation domain $\mathcal{S}_n^{\mathrm{v}_m}$ for this iteration, and thus define the validation set $\mathcal{D}_n^{\mathrm{v}_m} = \{(\mathbf{X}(\mathbf{s}_i), Y(\mathbf{s}_i)) : s_i \in \mathcal{S}_n^{\mathrm{v}_m}\}$.

2. **Establish the "dead zone":** We define $\mathcal{S}_{n,h}^{\mathrm{D}_m}$.

3. **Define the training set:** The corresponding training set is $\mathcal{D}_{n,h}^{\mathrm{t}_m} = \{(\mathbf{X}(\mathbf{s}_i), Y(\mathbf{s}_i)) : \mathbf{s}_i \in \mathcal{S}_{n,h}^{\mathrm{t}_m}\}$, where $\mathcal{S}_{n,h}^{\mathrm{t}_m}$ represents the training domain for this iteration.

It is important to note that while the validation domain's specific structure is not detailed here, once h is determined, the process for selecting the dead zone and the training domain remains consistent. Then, different adaptations of principal cross-validation methods can be distinguished by how they define the validation domain.

The framework described before is particularly tailored for the isotropic case, where distances and separations are uniformly defined in all directions. Consequently, a single value of h is sufficient to delineate the minimal separation needed between the validation and training sets to ensure their independence.

However, in anisotropic scenarios where spatial dependence varies with direction, a simplistic approach with a single value of h may not be adequate. Instead, a more nuanced approach involving two or more values of h may be required to accurately capture spatial relationships. Figure 2 illustrates the dead zone taken around one point under anisotropic and isotropic conditions. On the left, the elliptical blue contour shows varying separation distances h_1 and h_2 in vertical and horizontal directions due to differing spatial dependencies in these directions. In contrast on the right, under isotropic conditions (depicted by the red circular contour), only a single radius h is necessary.

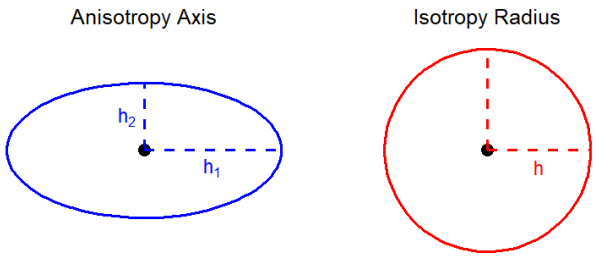


Figure 2: Examples of dead zone under anisotropic and isotropic conditions.

We now present adaptations of the main cross-validation procedures to the spatial context. To ease the presentation, we consider the isotropic framework in what follows.

2 Spatial cross-validation methods

2.1 Spatial Leave-One-Out

We begin by introducing Spatial Leave-One-Out (SLOO). This cross-validation procedure was initially presented by le2013accounting for model selection purposes. Back in 1994, burman1994cross introduced a modification of leave-one-out cross-validation tailored for general stationary data, known as h-block cross-validation, which laid the groundwork for Spatial Leave-One-Out. The primary goal of SLOO is to mitigate optimistic bias in the training set caused by spatial dependence. This is achieved by excluding data points "close" to the one held out for validation from the training set, as depicted in Figure 3. In this figure we represent $\mathcal{S}_n$, a lattice of size 80×60 with $n = 4800$ sites and its

partition into the three domains for one iteration of SLOO. The validation set is represented by a unique red point, a dead zone of radius 8 is removed around the validation domain, and the remaining locations colored in blue conform the training domain.

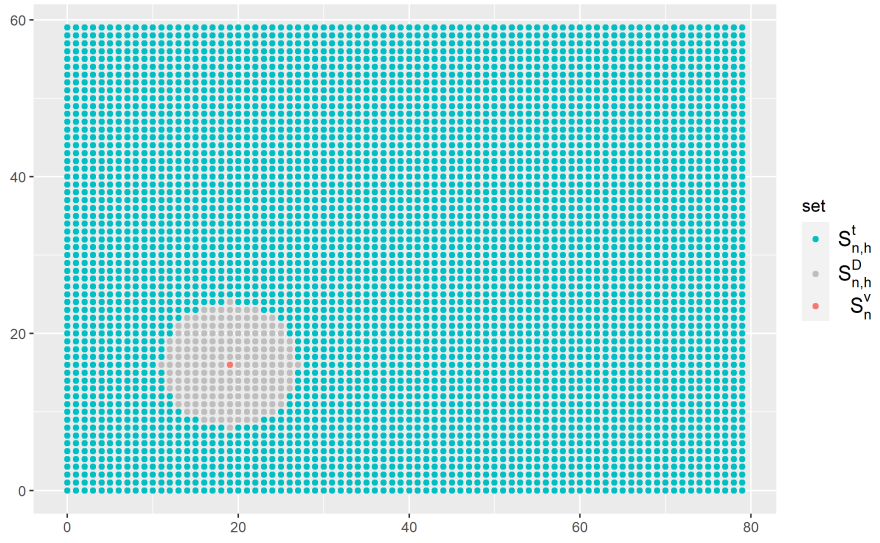


Figure 3: Dataset partition for one iteration of SLOO with a buffer of radius $h = 8$.

Let us precise the Spatial Leave-One-Out procedure within the regression framework, and assuming isotropy. We suppose that we know the distance h required to ensure independence between validation and training sets. At the $m-$th iteration of SLOO, one site is selected as the validation domain

$$\mathcal{S}_n^{\mathrm{v}_m} = \{\mathbf{s}_m\},$$

and

$$\mathcal{D}_n^{\mathrm{v}_m} = \{(\mathbf{X}(\mathbf{s}_m), Y(\mathbf{s}_m))\}$$

forms the validation set. The "dead zone" around $\{\mathbf{s}_m\}$ is defined as its neighbourhood of radius h,

$$\mathcal{S}_{n,h}^{\mathrm{D}_m} = \{\mathbf{s}_j \in \mathcal{S}_n : j \neq m \text{ and } d(\mathbf{s}_j, \mathbf{s}_m) \leq h)\}.$$

The remaining points constitute the training domain,

$$\mathcal{S}_{n,h}^{\mathrm{t}_m} = \{\mathbf{s}_j \in \mathcal{S}_n : d(\mathbf{s}_j, \mathbf{s}_m) > h\},$$

and the corresponding training set is

$$\mathcal{D}_n^{\mathrm{t}_m} = \{(\mathbf{X}(\mathbf{s}_j), Y(\mathbf{s}_j)) | \mathbf{s}_j \in \mathcal{S}_{n,h}^{\mathrm{t}_m}\}.$$

The predictive empirical risk is given by

$$\mathcal{R}_{n,h}^{\mathrm{SLOO}} = \frac{1}{n}\sum_{m=1}^{n} \mathcal{R}_{n,h}^{\mathrm{v}_m} = \frac{1}{n}\sum_{m=1}^{n} (Y(\mathbf{s}_m) - \mathbf{X}(\mathbf{s}_m)\hat{\boldsymbol{\beta}}_h^{\mathrm{t}_m})^2, \tag{13}$$

where we denote $\hat{\boldsymbol{\beta}}_h^{\mathrm{t}_m}$ the estimator computed on the training set $\mathcal{D}_{n,h}^{\mathrm{t}_m}$. The subscript h highlights the buffer or "dead zone" of radius that now needs to be considered.

For example, if we chose the OLS estimator, $\hat{\boldsymbol{\beta}}_h^{\mathrm{t}m}$ is defined by,

$$\hat{\boldsymbol{\beta}}_h^{\mathrm{t}m} = \left(\mathbf{X}_{m,h}^{\mathrm{T}}\mathbf{X}_{m,h}\right)^{-1}\mathbf{X}_{m,h}^{\mathrm{T}}\mathbf{Y}_{m,h}, \tag{14}$$

with $\mathbf{X}_{m,h}$ and $\mathbf{Y}_{m,h}$ representing the matrix $\mathbf{X}$ and vector $\mathbf{Y}$ whose elements related to $\mathcal{S}_n^{\mathrm{v}_m} \cup \mathcal{S}_{n,h}^{\mathrm{D}_m}$ have been deleted.

Spatial leave-one-out has proven effective in providing less biased estimates of predictive error. Research has shown its superiority over ordinary leave-one-out (LOO) cross-validation, particularly in spatial contexts. For example, karasiak2022spatial demonstrated that SLOO yielded less biased predictive error estimates across various sample sizes when compared to ordinary K-fold cross-validation and LOO. They trained a random forest model to classify two forest classes using images from the French department (area) "Herault". Subsequently, they compared the predictive error estimates derived from SLOO, LOO, and K-fold cross-validation against those obtained from a separate, independent dataset of images from two other French areas, which are sufficiently distant from the Herault department. Their findings indicated that the predictive error estimated by SLOO closely matched the error estimated using the independent data.

Similarly, misiuk2019spatially provided evidence of SLOO's advantages over LOO in estimating seabed sediment mapping using random forests. They compared the root mean square error, mean absolute error, and R^2 values in both interpolation and extrapolation areas, as estimated by SLOO and LOO, against the true values. Their results revealed that LOO tended to provide more optimistic estimates of performance metrics than SLOO.

2.1.1 Limitations

Spatial Leave-One-Out inherits two key limitations from the classical leave-one-out method. Firstly, it can be computationally expensive, particularly with large datasets or complex training models and learning algorithms. The iterative process of training and validating the model for each individual point in the dataset can demand substantial computational resources.

Secondly, SLOO tends to exhibit high variance in its hold-out estimations. This arises because the validation set in LOO and SLOO consists of only one data point, which leads to increased variability in performance metrics, compared to methods using larger validation sets.

As previously emphasized, the effectiveness of SLOO heavily relies on the accurate selection of the spatial buffer parameter h. Choosing an optimal h

value requires a delicate balance: it must be small enough to retain sufficient observations in the training set for effective model calibration, yet expansive enough to ensure the necessary independence from the validation set.

2.1.2 Extensions

Due to its popularity, several extensions and modifications have been proposed to enhance its performance and applicability across different scenarios. misiuk2019spatially introduced Spatially Resampled Leave-One-Out Cross-Validation (SR-LOO), which extends SLOO by incorporating spatial buffering procedures around both validation and training domains to prevent adjacent points from influencing model fitting or validation. This approach improves performance in handling spatial autocorrelation. The strategy was introduced to deal with random forests and builds upon the ideas from the variable scale selection procedure proposed by Holland2004DeterminingTS and the fact that separate "forests" can be combined and treated as a single model to make predictions (Liaw2007ClassificationAR). For each iteration of SR-LOO, besides the classical buffer around the point left out for validation, each training point is also spatially buffered so that no adjacent points are used for model fitting. To address the potential issue of significant data loss due to this strategy, they resample the training set multiple times and fit a small number of trees for each subsample. The final "forest" is constructed by combining the trees from the separate subsamples. As mentioned before, this method is particularly designed for random forests in scenarios where we aim to avoid overfitting on the training data because we wish to train a model that is good for extrapolation.

mila2022nearest proposed the *Nearest Neighbour Distance Matching Leave-One-Out Cross-Validation* method for map validation. This method uses nearest neighbor distance distribution functions to ensure that the spatial structure between training and validation sets matches that of the target area.

For the prediction target sites, a nearest neighbor distance distribution function is used to describe the distance between each prediction site and its nearest neighbor from the sample set. Specifically, the cross nearest neighbor cumulative distribution function $G_{i,j}(r)$ is defined as,

$$G_{i,j}(r) = P\left(d_{i,j} \leq r\right) = \frac{1}{N_i} \sum_{k=1}^{N_i} I\left(d_k^{(i,j)} \leq r\right),$$

where N_i is the number of prediction sites in dataset i, $d_k^{(i,j)}$ is the distance from the k-th prediction site in dataset i to its nearest neighbor in the sample set j and $I(\cdot)$ is the indicator function. This nearest neighbor distribution is replicated between the training and test sets using rejection sampling. The goal is to ensure that the distribution of distances from test sites to their nearest training sites matches the overall spatial structure of the target area.

This method only considers the spatial distribution of the data and does not take response variable or covariates into consideration. It refines SLOO by considering distances to the nearest neighbor at each iteration, which is

especially useful for irregularly distributed locations. At each iteration, the distance of the validation point to its nearest neighbor is considered. If there is minimal or no spatial autocorelation, only one site is placed in the test set and the method behaves like SLOO. If the nearest-neighbor distance is larger than the spatial autocorrelation range, the method behaves like spatial blocked CV, see section 2.3. One drawback is that the algorithm may eliminate too many data points from the training set when there is strong clustering of the sites.

In summary, while SLOO provides a spatially informed approach to cross-validation, it is essential to consider its computational challenges, the increased variance of hold-out estimations, and the meticulous selection of the spatial bandwidth parameter to ensure its efficacy in model evaluation and prediction.

2.2 Spatial K-fold cross-validation

Spatial K-fold cross-validation (SKVC) is a variant of K-fold cross-validation introduced by pohjankukka2017estimating.

We follow the notations of Section 1.3. In Spatial K-fold cross-validation, the domain is divided into K folds, denoted by $\{\mathcal{S}_n^{\mathrm{v}_1}, \ldots, \mathcal{S}_n^{\mathrm{v}_K}\}$, where each fold $\mathcal{S}_n^{\mathrm{v}_m}$ is defined such that its sites are selected randomly, and its cardinality satisfies $\mathrm{Card}(\mathcal{S}_n^{\mathrm{v}_m}) = n_m$, with $\left\lfloor \frac{n}{K} \right\rfloor \leq n_m \leq \left\lceil \frac{n}{K} \right\rceil$. It is important to note that here we refer to the cardinality of the spatial domains $\mathcal{S}_n^{\mathrm{v}_m}$ rather than the cardinality of data subsets, as it is typically done in classical K-fold cross-validation. However, since each site corresponds to a data point, the cardinality of each validation domain $\mathcal{S}_n^{\mathrm{v}_m}$ is equal to the cardinality of the corresponding validation data subset $\mathcal{D}_n^{\mathrm{v}_m}$. This means that $\mathrm{Card}(\mathcal{S}_n^{\mathrm{v}_m}) = \mathrm{Card}(\mathcal{D}_n^{\mathrm{v}_m})$, ensuring consistency between the spatial partitioning and the data used in validation. Besides, the partition satisfies $\cap_m \mathcal{S}_n^{\mathrm{v}_m} = \emptyset$ and $\cup_m \mathcal{S}_n^{\mathrm{v}_m} = \mathcal{S}_n$.
During the $m-$th step of SKCV, given the validation domain $\mathcal{S}_n^{\mathrm{v}_m}$, the set of dead points is defined as

$$\mathcal{S}_{n,h}^{\mathrm{D}_m} = \{\mathbf{s}_j \notin \mathcal{S}_n^{\mathrm{v}_m} : \exists\, \mathbf{s}_i \in \mathcal{S}_n^{\mathrm{v}_m} : d(\mathbf{s}_j, \mathbf{s}_i) \leq h\}$$

and the training domain is

$$\mathcal{S}_{n,h}^{\mathrm{t}_m} = \{\mathbf{s} \in \mathcal{S}_n : \mathbf{s} \notin \mathcal{S}_n^{\mathrm{v}_m} \cup \mathcal{S}_{n,h}^{\mathrm{D}_m}\}.$$

The SKCV predictive empirical risk is given by,

$$\mathcal{R}_{n,h}^{\mathrm{SKCV}} = \frac{1}{K} \sum_{m=1}^{K} \mathcal{R}_{n,h}^{\mathrm{v}_m}. \tag{15}$$

Figure 4 illustrates an example of SKCV applied to a lattice $\mathcal{S}_n$ of size $n = 80 \times 60$, with $K = 200$ folds of cardinality 24, and a dead zone of radius $h = 8$ around each point of the validation domain.

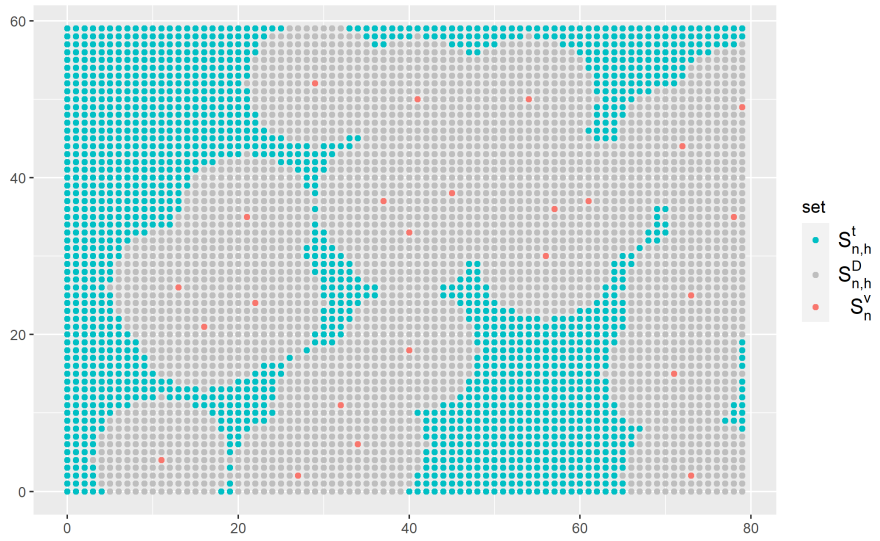


Figure 4: Dataset partition for one iteration of Spatial K-fold cross-validation with buffer radius $h = 8$ and $K = 200$.

2.2.1 Limitations

When employing spatial K-fold cross-validation, one potential drawback arises from the removal of a substantial number of training data points, which can introduce an additional pessimistic bias in prediction performance. To investigate this bias, pohjankukka2017estimating conducted a study where they randomly removed an equivalent number of points as those excluded during SKCV and compared the outcomes. Their findings indicate that random removal of training data points has minimal impact on prediction performance compared to spatial-based data exclusion.

Furthermore, the number of folds used in the SKCV procedure can exacerbate the aforementioned challenges. Depending on the size and spatial distribution of the data, a significant portion of the training data may be removed due to the cumulated effect of dead zones. Therefore, choosing an optimal number of folds is crucial to balance capturing spatial dependencies and maintaining an adequate sample size for robust model training.

Finally, similar to SLOO, the effectiveness of SKCV depends on the selection of an appropriate buffer radius. pohjankukka2017estimating noted that the estimated RMSE by SKCV increased as the size of the dead zone did so. However, no criteria for the choice of the optimal buffer was given. crosbyRTCV proposed a heuristic for determining the optimal buffer radius. In line with common recommendations in the literature, this heuristic suggests using the range of the variogram of the response variable as an upper bound for the buffer radius in the general case. Specifically, it defines an interval of potential values for the

buffer radius, ranging from zero to the variogram range $d_{\max}$, depending on the proximity between the prediction locations and the training points. The criteria for the buffer radius h are as follows,

$$h = \begin{cases} 0, & \text{if } \frac{\mu'_{tt}}{\mu'_{tr}} \leq 1, \quad \text{(interpolation scenario)} \\ \mu'_{tt}, & \text{if } \frac{\mu'_{tt}}{\mu'_{tr}} > 1 \text{ and } \mu'_{tt} < d_{\max}, \quad \text{(mixture scenario)} \\ d_{\max}, & \text{in any other case,} \quad \text{(extrapolation scenario),} \end{cases} \tag{16}$$

where μ'_{tt} is the average distance between each point in the training set and the corresponding prediction location, μ'_{tr} is the average distance within the training set and $d_{\max}$ is the range of the variogram of the response variable.

This method leverages information about the prediction locations (the new points where the model will be applied), making it adaptable to various spatial settings. In the pure interpolation scenario, where the prediction locations are well represented by the training set, the heuristic suggests using classical K-fold cross-validation with $h = 0$. For scenarios with both interpolation and extrapolation, the mixture scenario, it proposes a buffer radius equal to the average distance μ'_{tt}, provided that this distance is less than $d_{\max}$. In the extrapolation scenario, where prediction points are relatively far from the training data, the buffer radius defaults to the variogram range $d_{\max}$.

It is important to note that while this heuristic provides a systematic approach for buffer selection, the paper does not present any evidence for the optimality of the proposed buffer values.

In conclusion, although SKCV is designed to account for spatial autocorrelation by introducing buffer zones, it often results in the removal of an excessive number of data points. This significant reduction in the effective sample size can adversely affect model performance, leading to decreased statistical power and less reliable predictions. Consequently, SKCV may not always be the most efficient method for spatial cross-validation due to its tendency to discard more data than necessary. It is crucial to carefully evaluate the selection of the buffer radius and the number of folds, or consider alternative cross-validation strategies, to balance the need for spatial independence with the retention of sufficient data for robust spatial analysis.

2.3 Blocked cross-validation

Blocked cross-validation (BCV) is a well-established spatial cross-validation technique that differs from SKCV by using a structured block pattern instead of random location selection when determining the folds. This method is particularly advantageous when the goal is to identify causal predictors or forecast within unique spatial frameworks, as each block can encapsulate distinct spatial features. According to roberts2017cross, blocked cross-validation performs well in assessing a model's ability to extrapolate to independent data, making it a valuable tool in spatial analysis.

This method can be considered with and without spatial buffering around the block; in what follows we will refer to Spatial blocked cross-validation (SBCV) when considering buffering around the block (with buffer's width $h > 0$) and blocked cross-validation otherwise, with no buffer ($h = 0$).

Figure 4 illustrates an example of SBCV. Here, $\mathcal{S}_n$ is a lattice of size $n = 80 \times 60$, we consider $B = 200$ blocks of size 6×4 and $h = 8$. At this point it is worth mentioning that although the number of locations included in the validation set in figures 4 and 5 is the same, the cardinal of the dead zone $\mathcal{S}^{\mathrm{D}}_{n,h}$ is 324 for SBCV and 3074 for SKCV.

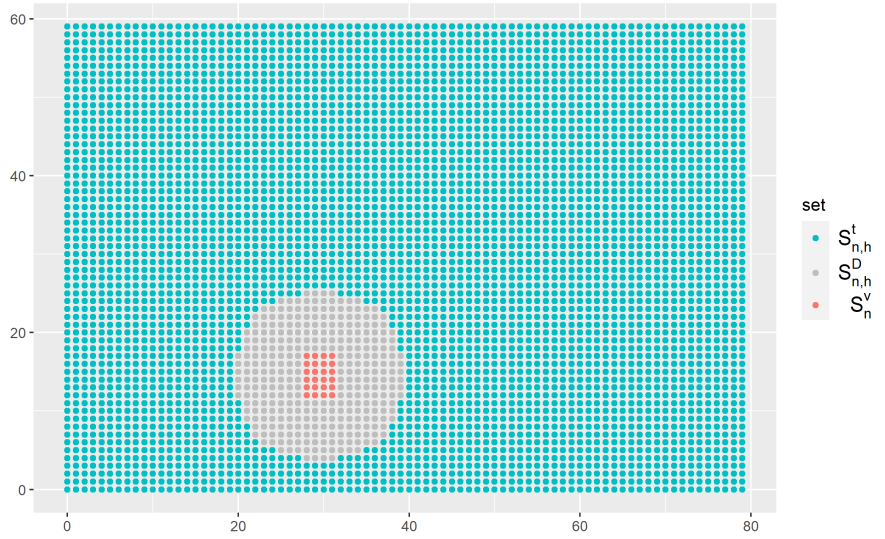


Figure 5: Dataset partition for one iteration of SBCV with buffer of radius $h = 8$ and square block of size 4×6.

Let B be the number of blocks, analogously to K in SKCV, but with blocks defined by neighboring locations rather than random selection. The risk estimate for SBCV is given by:

$$\mathcal{R}^{\mathrm{SBCV}}_{n,h} = \frac{1}{B} \sum_{m=1}^{B} \mathcal{R}^{\mathrm{v}_m}_{n,h}. \tag{17}$$

Here, each validation domain $\mathcal{S}^{V_m}_{n,h}$ is a block. It is important to note that unlike SBCV, the risk estimate $\mathcal{R}^{\mathrm{BCV}}_{n}$ does not depend on h, because it does not utilize a dead zone:

$$\mathcal{R}^{\mathrm{BCV}}_{n} = \frac{1}{B} \sum_{m=1}^{B} \mathcal{R}^{\mathrm{v}_m}_{n,h}. \tag{18}$$

Comparing SBCV with SKCV, the structuring of blocks in SBCV ensures that fewer points fall within the dead zone compared to SKCV, as seen in figures

4 and 5. They illustrate how far the cardinality of the buffers differs, though the validation domain in both methods includes an identical number of locations (24 in this example). Compared to SKCV, SBCV minimizes the impact of the dead zone, enhancing the validation process's integrity. Overall, blocked cross-validation is the most widespread spatial cross-validation method.

The review paper by JEMELJANOVA2024102634 on machine learning methods in environmental studies found 20 different examples of the use of some variant of blocked cross-validation up to November 2023. Blocked cross-validation methods offer flexibility through various block definitions, such as oblong blocks or checkerboard structures, as discussed in roberts2017cross. Variants include spatial blocking using clustering algorithms like k-means, see brenning2012spatial and rs10081277, or hierarchical clustering, see Stock2018MappingEI and WANG2023103364.

The choice of block definition can significantly impact the performance of BCV, and different methods may be more appropriate depending on the characteristics of the data and the objectives of the study. The most common practice is to define the blocks by applying a clustering algorithm on the spatial coordinates as seen in the works by brenning2012spatial, rs10081277 and Stock2018MappingEI. This approach can be particularly useful for irregularly spaced data, where regular grid-based blocks may not capture the spatial structure effectively.

However, for regularly spaced data, such as data collected on a lattice, clustering-based methods might result in irregularly shaped blocks, which may not be appropriate or may complicate the analysis. Figure 6 shows a partition defined by applying k-means clustering on the spatial coordinates in a 60×80 lattice. As can be seen, the resulting blocks are not regular, which may not align well with the underlying spatial structure of the data.

Moreover, WANG2023103364 argue that the data used for spatial prediction using machine learning models involve not only information about the spatial coordinates but also about the feature space, including the covariates and the response variable. Therefore, they suggest using a clustering approach that incorporates spatial coordinates, covariates, and the response variable to define the blocks. This method aims to create blocks that are more homogeneous in terms of both spatial location and feature space, potentially leading to better model validation.

The differences between these methods lie in how the blocks are defined and what aspects of the data are considered. Regular grid-based blocks may be appropriate for regularly spaced data and when the spatial structure aligns with the grid, offering simplicity and ease of implementation. Clustering-based methods may be more suitable for irregularly spaced data or when aiming to capture complex spatial dependencies. Including covariates and response variables in the clustering process, as suggested by WANG2023103364, may further improve the effectiveness of BCV by accounting for feature space similarity as well as spatial proximity. However, it's important to note that these methods might not be appropriate for regularly spaced data if they result in irregularly shaped blocks, as this can lead to uneven representation of spatial areas and complicate the analysis. Moreover, there is also the question of the optimal

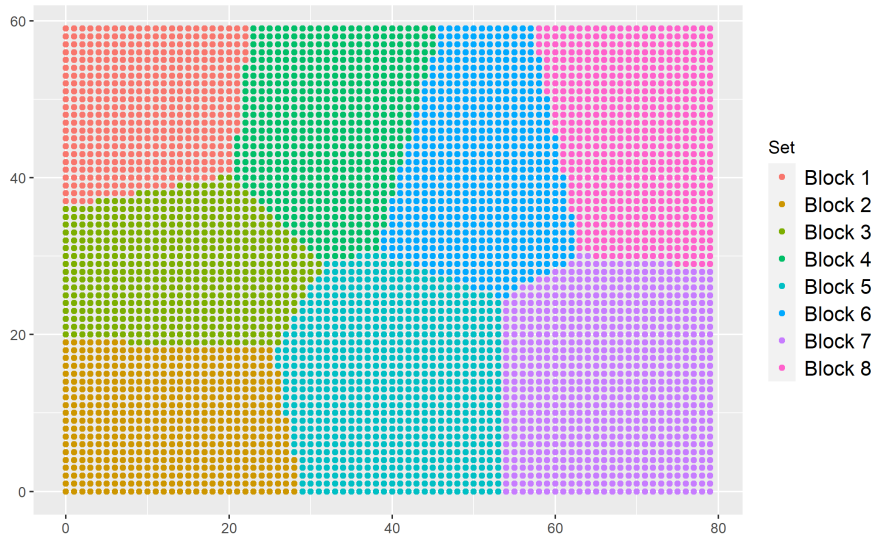


Figure 6: Dataset partition into blocks defined by k-means clustering on the spatial coordinates with $k = 8$.

number of clusters.

mahoney2023assessing explored three blocking techniques for spatial cross-validation: rectangular blocks, k-means clusters based on spatial locations, and the *Leave-one-disc-out* (LODO) method. LODO combines elements of SLOO and SBCV. The LODO method operates with as many iterations as there are points in the dataset, similarly to SLOO. However, instead of leaving out a single point for validation, LODO defines the validation set as a disc centered around each observation, including all points within a specified inclusion radius r. Additionally, it incorporates an exclusion radius h around the validation disc to ensure that nearby points are not included in the training set, thereby reducing spatial autocorrelation between training and validation sets.

Formally, for each observation $\mathbf{s}_m \in \mathcal{S}_n$, the validation set $\mathcal{S}_n^{\mathrm{v}_m}$ and training set $\mathcal{S}_n^{\mathrm{t}_m}$ are defined as,

$$\begin{aligned}
\mathcal{S}_n^{\mathrm{v}_m} &= \{\mathbf{s}_i \in \mathcal{S}_n : \|\mathbf{s}_m - \mathbf{s}_i\| \leq r\}, \\
\mathcal{S}_n^{D_m} &= \{\mathbf{s}_i \in \mathcal{S}_n : r \geq \|\mathbf{s}_m - \mathbf{s}_i\| \leq h\}, \\
\mathcal{S}_n^{\mathrm{t}_m} &= \{\mathbf{s}_i \in \mathcal{S}_n : \|\mathbf{s}_m - \mathbf{s}_i\| > r + h\}.
\end{aligned}$$

Leave-one-disc-out offers a flexible and localized approach to spatial cross-validation, combining features of both SLOO and SBCV. By adjusting the inclusion radius r and the exclusion radius (buffer size) h, practitioners can tailor the method to the specific spatial characteristics of their data. However, careful consideration

of parameter selection and computational resources is necessary to effectively implement LODO. In particular, LODO needs to consider the selection of both h and r. In contrast, for SLOO, the primary consideration is determining the appropriate buffer size h. Moreover, it's important to note that LODO does not offer computational gains compared to SLOO.

wang2023crossvalidation introduced a variation of SBCV that specifically addresses covariate shift during the data splitting process. Covariate shift refers to a situation where the distribution of the covariates differs between the training data and the validation data, while the conditional distribution of the response variable given the covariates remains the same. Formally, let $p_{\text{train}}(\mathbf{X})$ and $p_{\text{val}}(\mathbf{X})$ denote the distributions of the covariates in the training and validation sets, respectively, and $p(\mathbf{Y} \mid \mathbf{X})$ denote the conditional distribution of the response variable given the covariates. Covariate shift occurs when,

$$p_{\text{train}}(\mathbf{X}) \neq p_{\text{val}}(\mathbf{X}), \quad \text{but} \quad p_{\text{train}}(\mathbf{Y} \mid \mathbf{X}) = p_{\text{val}}(\mathbf{Y} \mid \mathbf{X}).$$

This discrepancy means that while the relationship between $\mathbf{X}$ and $\mathbf{Y}$ remains consistent, the way $\mathbf{X}$ is distributed changes between the training and validation sets. Covariate shift can lead to biased estimates of model performance because the model is trained on a distribution of covariates that is different from the one it encounters during validation or testing. Here the covariates are random variables. This places their work within the random design framework, where both $\mathbf{X}$ and $\mathbf{Y}$ are considered random, and the variability in $\mathbf{X}$ is explicitly accounted for.

2.3.1 Limitations

Employing blocking techniques requires careful consideration of potential implications on data characteristics. Grouping similar units may inadvertently remove unique features critical for accurate interpolation, see for example hoffiman2021.

An additional critical consideration when blocking involves determining the optimal block size for regular blocks. roberts2017cross recommend estimating the variogram of the dependent variable and using its range as the block size. This approach aims to capture the spatial autocorrelation present in the response variable, ensuring that observations within each block are sufficiently correlated.

Conversely, the `blockCV` package developed by valavi2019blockcv, which offers a range of functions for generating training and testing folds for spatial cross-validation, but omits SBCV, suggests selecting the block size based on the median range obtained from the variograms of the covariates. This method focuses on the spatial structure of the predictors, under the assumption that accounting for the spatial autocorrelation in the covariates will lead to more reliable model validation.

Both recommendations apply to regular blocks in SBCV but differ in their emphasis on the dependent variable versus the covariates. The choice between

these approaches may depend on the specific characteristics of the dataset and the objectives of the study. For instance, if the response variable exhibits strong spatial autocorrelation, using its variogram range may be more appropriate. On the other hand, if the covariates have significant spatial structure that influences the response, basing the block size on the covariates' variogram ranges might provide better cross-validation results.

It is important to note that selecting an appropriate block size is crucial because it affects the balance between bias and variance in the cross-validation estimates. Blocks that are too small may not effectively mitigate spatial autocorrelation between training and validation sets, leading to overly optimistic performance estimates. Conversely, blocks that are too large may result in high variance and less reliable estimates due to insufficient data in each fold.

Furthermore, defining effective regular blocks can be challenging when dealing with data clusters resulting from irregular sampling patterns. In such scenarios, alternative approaches can be considered. One solution involves using irregularly arranged but similarly sized blocks, which can help capture the spatial structure while accounting for the data distribution. Another approach is to employ irregularly shaped blocks, tailored to match the spatial clustering of the data, providing a more accurate representation of the underlying spatial patterns, see mahoney2023assessing. Choosing the most appropriate blocking method depends on the specific characteristics of the dataset and the goals of the study. Researchers should consider the spatial distribution of their data, the presence of spatial autocorrelation, and whether the inclusion of covariates and response variables in the blocking process would enhance model validation. Careful consideration of these factors will help ensure that the selected cross-validation method provides reliable and unbiased estimates of model performance.

2.4 Comparison of cross-validation strategies

Several studies have compared spatial cross-validation methods. In their seminal 2017 paper, roberts2017cross compared SBCV with SLOO and traditional cross-validation approaches within the framework of random forests. Their field of investigation is ecology, mainly species distribution Evans2011ModelingSD and random forest are widely used in this field. More recently, mahoney2023assessing conducted a study using the same data generation process as roberts2017cross, and compared different variants of SLOO and SBCV with classical cross-validation methods. Both studies benchmarked against an "ideal" root mean squared error (RMSE), with Roberts et al. employing visual comparisons of risk distributions, while Mahoney et al. focused on CV estimates falling within a specific percentile range of the ideal RMSE. Both concluded that classical cross-validation underestimates prediction errors in spatial contexts.

Ideal empirical risk

As in roberts2017cross and mahoney2023assessing, we consider the **ideal empirical risk**. The motivation behind introducing the ideal risk comes from the

need to have a benchmark against which to compare the cross-validation risk estimates.

The predictive risk of an estimator $\hat{\boldsymbol{\beta}}$ is defined as the expected loss when predicting new observations at unseen locations, as given in Equation (4). To approximate the expected predictive performance, roberts2017cross propose averaging the prediction errors over multiple independent realizations of the spatial process.

Given M independent realizations $\mathbf{Y}^{(1)}, \mathbf{Y}^{(2)}, \ldots, \mathbf{Y}^{(M)}$ of the spatial process $\mathbf{Y}$, we define the empirical ideal predictive risk for the l-th realization as,

$$\mathcal{R}_n^{I(l)} = \frac{1}{M-1} \sum_{\substack{j=1 \\ j \neq l}}^{M} \frac{1}{n} \sum_{\mathbf{s} \in \mathcal{S}_n} \left(\hat{Y}^{(l)}(\mathbf{s}) - Y^{(j)}(\mathbf{s}) \right)^2, \tag{19}$$

where,

- $\hat{Y}^{(l)}(\mathbf{s}) = \mathbf{X}(\mathbf{s})\hat{\boldsymbol{\beta}}^{(l)}$ is the predicted value at location $\mathbf{s}$ using the estimator $\hat{\boldsymbol{\beta}}^{(l)}$ obtained from the l-th realization,
- $Y^{(j)}(\mathbf{s})$ is the observed value at location $\mathbf{s}$ from the j-th realization.

This means that for each realization $Y^{(l)}(\mathbf{s})$ of the spatial process $\mathbf{Y}$, we obtain $\hat{\boldsymbol{\beta}}^{(l)}$ and then compute the ideal risk associated with this realization $\mathcal{R}_n^{I(l)}$ by considering the difference between the predicted values $\hat{Y}^{(l)}(\mathbf{s})$ and the observed values in the remaining $M-1$ independent realizations $Y^{(j)}(\mathbf{s})$ for all $j \neq l$.

This ideal risk measures the average prediction error of the model trained on $\mathbf{Y}^{(l)}$ when predicting independent realizations $\mathbf{Y}^{(j)}$ at the same locations. By averaging over multiple independent realizations, we obtain a more reliable estimate of the true predictive risk. The use of multiple independent realizations is a Monte Carlo approach, where we approximate expectations by averaging over a large number of random samples Rubinstein1981SimulationAT.

The comparison with the ideal risk allows us to evaluate the effectiveness of different cross-validation strategies in estimating the true predictive performance of the model. It helps in identifying methods that provide less biased and reliable risk estimates in the presence of spatial dependence.

Simulation design

To evaluate and compare classical cross-validation and spatial cross-validation strategies, we now present a simulation study.

Let $\mathcal{S}_n$ be a rectangular lattice of size 80×60, with $n = 4800$ sites. We consider the regression model,

$$Y(\mathbf{s}) = \mathbf{X}(\mathbf{s})\boldsymbol{\beta} + \varepsilon(\mathbf{s}), \tag{20}$$

where $\boldsymbol{\varepsilon}$ is generated according to a zero-mean Gaussian spatial process with an isotropic exponential covariance function defined by,

$$C(h) = \sigma^2 \exp\left(-\frac{h}{\theta}\right).$$

We choose this covariance function because it is a well-known model for spatial dependence. We set $\sigma^2 = 1$ and consider different values of $\theta \in \{3, 6, 9, 12, 15, 20\}$ to control the strength of spatial dependence.

We introduce a non-stationary trend linked to the spatial location within the domain. We denote the spatial location as $\mathbf{s} = (s_1, s_2) \in \mathcal{S}_n$, where s_1 and s_2 represent the coordinates in the horizontal and vertical directions, respectively. We define the design vector at location $\mathbf{s}$ as,

$$\mathbf{X}(\mathbf{s})^\top = [1,\, X_1(\mathbf{s}),\, X_2(\mathbf{s})],$$

where $X_1(\mathbf{s}) = s_1 - 40$ and $X_2(\mathbf{s}) = s_2 - 30$ are centered spatial coordinates serving as covariates, and the first element corresponds to the intercept.

We set the parameter vector $\boldsymbol{\beta} = (\beta_0, \beta_1, \beta_2)^\top$. Following the proceedure described by aldworth1999sampling, β_1 and β_2 are chosen in such a way that the deterministic trend exhibits a variance comparable to the error term, achieving a signal-to-noise ratio of $\tau = \dfrac{\sigma^2_{\text{trend}}}{\sigma^2_Y} = \dfrac{1}{2}$, where σ^2_{trend} is the variance of the trend component and σ^2_Y is the total variance of $Y(\mathbf{s})$.

To ensure a balanced contribution from $\mathbf{X}_1$ and $\mathbf{X}_2$ to the variance of the trend component, we aim for their scaled variances to be equal

$$\beta_1^2 \operatorname{Var}(\mathbf{X}_1) = \beta_2^2 \operatorname{Var}(\mathbf{X}_2).$$

Here, $\mathbf{X}_1$ and $\mathbf{X}_2$ are deterministic covariates defined over the spatial domain $\mathcal{S}_n$. The terms $\operatorname{Var}(\mathbf{X}_1)$ and $\operatorname{Var}(\mathbf{X}_2)$ represent the sample variances of $\mathbf{X}_1$ and $\mathbf{X}_2$ across all spatial locations, calculated as,

$$\operatorname{Var}(\mathbf{X}_i) = \frac{1}{n} \sum_{k=1}^{n} \left(X_i(\mathbf{s}_k) - \overline{X}_i\right)^2, \quad \text{for } i = 1, 2,$$

where $\overline{X}_i = \dfrac{1}{n} \sum_{k=1}^{n} X_i(\mathbf{s}_k)$ is the mean of $\mathbf{X}_i$ over the spatial locations $\mathbf{s}_1, \ldots, \mathbf{s}_n$.

Given that the variance of the trend component is,

$$\sigma^2_{\text{trend}} = \beta_1^2 \operatorname{Var}(\mathbf{X}_1) + \beta_2^2 \operatorname{Var}(\mathbf{X}_2),$$

the total variance of $Y(\mathbf{s})$ is the sum of the variance of the trend and the variance of the error term $\varepsilon(\mathbf{s})$,

$$\sigma^2_Y = \sigma^2_{\text{trend}} + \sigma^2,$$

where $\sigma^2 = \operatorname{Var}(\varepsilon(\mathbf{s}))$ is the variance of the stochastic error term.

This procedure leads to set $\beta_1 = \dfrac{1}{32}$ and $\beta_2 = \dfrac{1}{24}$.

Finally, the intercept β_0 is chosen to ensure that the coefficients of variation of the estimators of β_0, β_1, and β_2 are similar, balancing the estimation precision across parameters. So we set the parameter vector $\boldsymbol{\beta} = (\beta_0, \beta_1, \beta_2)^\top = \left(0.9, \frac{1}{32}, \frac{1}{24}\right)$.

Note that in our model, although the expectation $\mathbb{E}[Y(\mathbf{s})]$ varies spatially due to the trend, the covariance between $Y(\mathbf{s}_i)$ and $Y(\mathbf{s}_j)$ depends only on the distance $h = \|\mathbf{s}_i - \mathbf{s}_j\|$, because the residuals $\varepsilon(\mathbf{s})$ are generated with an isotropic covariance function.

Figure 7 shows a realization of $\mathbf{Y}$ with $\theta = 9$, illustrating the spatial variation captured by the model.

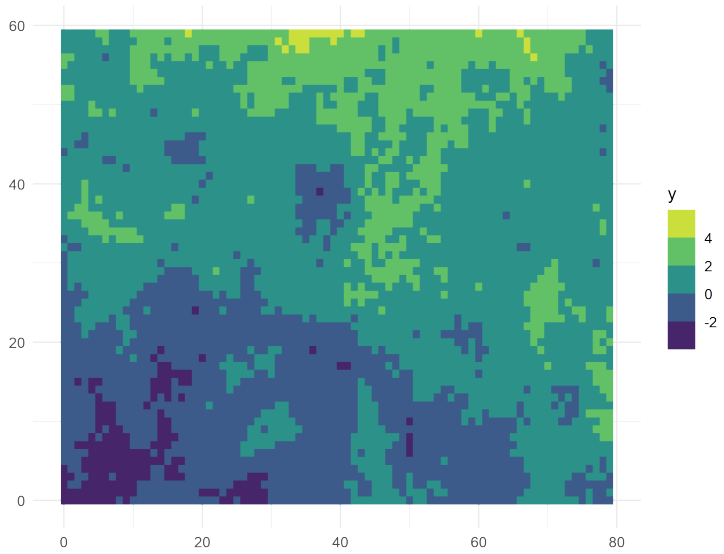


Figure 7: One simulated data generated following model of equation (20) with $\theta = 9$.

Our estimation scenario ignores the spatial dependence structure; we estimate the empirical risk associated with the ordinary least squares estimator using different cross-validation methods. Although Generalized Least Squares can account for spatial dependence by incorporating the covariance structure of the errors, it requires accurate specification of the spatial correlation model, which can be complex and computationally intensive. In many practical situations, practitioners may prefer ordinary least squares due to its simplicity and ease of interpretation, even when spatial dependence is present.

Moreover, since our primary interest is to estimate the risk of an estimator, the estimator itself does not necessarily need to be the best possible one. Studying spatial cross-validation methods with ordinary least squares allows us to focus on the effectiveness of these validation techniques in estimating the

predictive risk, regardless of whether the estimator fully accounts for spatial dependence. This approach provides valuable insights into how different cross-validation methods perform in practice, especially when the model does not explicitly model spatial autocorrelation.

Comparison of dead zone and training domain sizes for SBCV and SKCV

Let's first compare the SBCV and SKCV methods with respect to the sizes of the dead zone $\mathcal{S}_{n,h}^{\mathrm{D}}$ and the training domain $\mathcal{S}_{n,h}^{\mathrm{t}}$, while keeping the size of the validation domain fixed at $\mathrm{Card}(\mathcal{S}_n^{\mathrm{v}}) = 24$.

Table 1 presents the average number of points in the dead zone and the training domain for different buffer sizes h, along with the associated percentages relative to the total number of points $n = 4800$. The averages are calculated over all folds or blocks in the cross-validation procedure.

For example, for SKCV, the average size of the dead zone is computed as,

$$\mathrm{ave}(\mathrm{Card}(\mathcal{S}_n^{\mathrm{D}})) = \frac{1}{K}\sum_{m=1}^{K}\mathrm{Card}(\mathcal{S}_{n,h}^{\mathrm{D}_m}),$$

and the average size of the training domain is,

$$\mathrm{ave}(\mathrm{Card}(\mathcal{S}_n^{\mathrm{t}})) = \frac{1}{K}\sum_{m=1}^{K}\mathrm{Card}(\mathcal{S}_{n,h}^{\mathrm{t}_m}).$$

By averaging over all folds for SKCV and blocks for SBCV, we obtain a representative measure of the typical sizes of the dead zone and training domain across the entire cross-validation process. This allows for a fair comparison between SBCV and SKCV methods regarding how the buffer size h affects the amount of data excluded from the training set (dead zone) and the remaining data used for training.

The associated percentages are calculated by dividing the average sizes by the total number of points $n = 4800$ and multiplying by 100.

We observe that on average, for SKCV, a staggering 93.61% of the spatial domain is categorized as the dead zone for $h = 15$. This occasionally leads to scenarios where the training set is entirely vacated. Even at $h = 10$, the model is trained on merely an average of 25% of $\mathcal{S}_n$, a stark contrast to SBCV's utilization of approximately 91.15% of $\mathcal{S}_n$ for training, despite $\mathcal{S}_n^{\mathrm{v}}$ being equivalently sized as in SKCV. This disparity highlights the potential limitations of SKCV in spatial contexts and accentuates the efficiency of SBCV in leveraging available data for model training.

Method	h	ave(Card($\mathcal{S}_n^{\mathrm{D}}$)) (%)	ave(Card($\mathcal{S}_n^{\mathrm{t}}$)) (%)
BCV	0	0 (0%)	4776 (99.5%)
KCV		0 (0%)	4776 (99.5%)
SBCV	5	143.7 (2.99%)	4632.3 (96.51%)
SKCV		1502 (31.29%)	3272 (68.19%)
SBCV	10	400.96 (8.35%)	4375 (91.15%)
SKCV		3588 (74.5%)	1188 (24.75%)
SBCV	15	746.72 (15.56%)	4029.28 (83.94%)
SKCV		4493.14 (93.61%)	282.87 (5.86%)
SBCV	20	1157.04 (24.11%)	3618.96 (75.39%)
SKCV		4713.56 (98.20%)	62.45 (1.3%)

Table 1: Comparison of average dead zone (ave(Card($\mathcal{S}_n^{\mathrm{D}}$))) and training domain (ave(Card($\mathcal{S}_n^{\mathrm{t}}$))) sizes in number and percentages for SBCV and SKCV, with Card($\mathcal{S}_n^{\mathrm{v}}$) = 24.

Comparison of the empirical risks with the ideal risk

We now turn to examine the different cross-validation methods and how they behave with respect to one another; their performance is also compared to the ideal risk defined in (19), which we aim to reach.

We simulate $M = 100$ copies of Y under the design of equation (20). Let us recall that we consider $\theta \in \{3, 6, 9, 12, 15, 20\}$ to depict various schemes of dependence.

We consider the following cross-validation methods:

- Ordinary leave-one-out.
- Ordinary K-fold cross-validation, using $K = 200$ folds.
- Spatial leave-one-out, considering different buffer sizes with $h \in \{5, 10, 15, 20\}$.
- Spatial K-fold cross-validation, keeping $K = 200$ and $h \in \{5, 10, 15, 20\}$.
- Blocked cross-validation (without spatial buffering), using $B = 200$ blocks of size 6×4, aligning the number of folds and the size of the validation set in each iteration with those used in KCV.
- Spatial blocked cross-validation, adding spatial buffering around each block with $h \in \{5, 10, 15, 20\}$.

For each θ, we compute the ideal risk $\mathcal{R}_n^{I(l)}$ and the risk estimates $\mathcal{R}_{n,h}^{\mathrm{method}(l)}$ obtained from the methods described above for $l = 1$ to $M = 100$, considering different h values. $h = 0$ corresponds to the methods without spatial buffering.

Figure 8 displays the ideal risk and cross-validation risk estimates derived from 100 replications of the response variable Y across different cross-validation methods. For $h = 0$ SLOO, SKCV and SBCV become LOO, KCV and BCV.

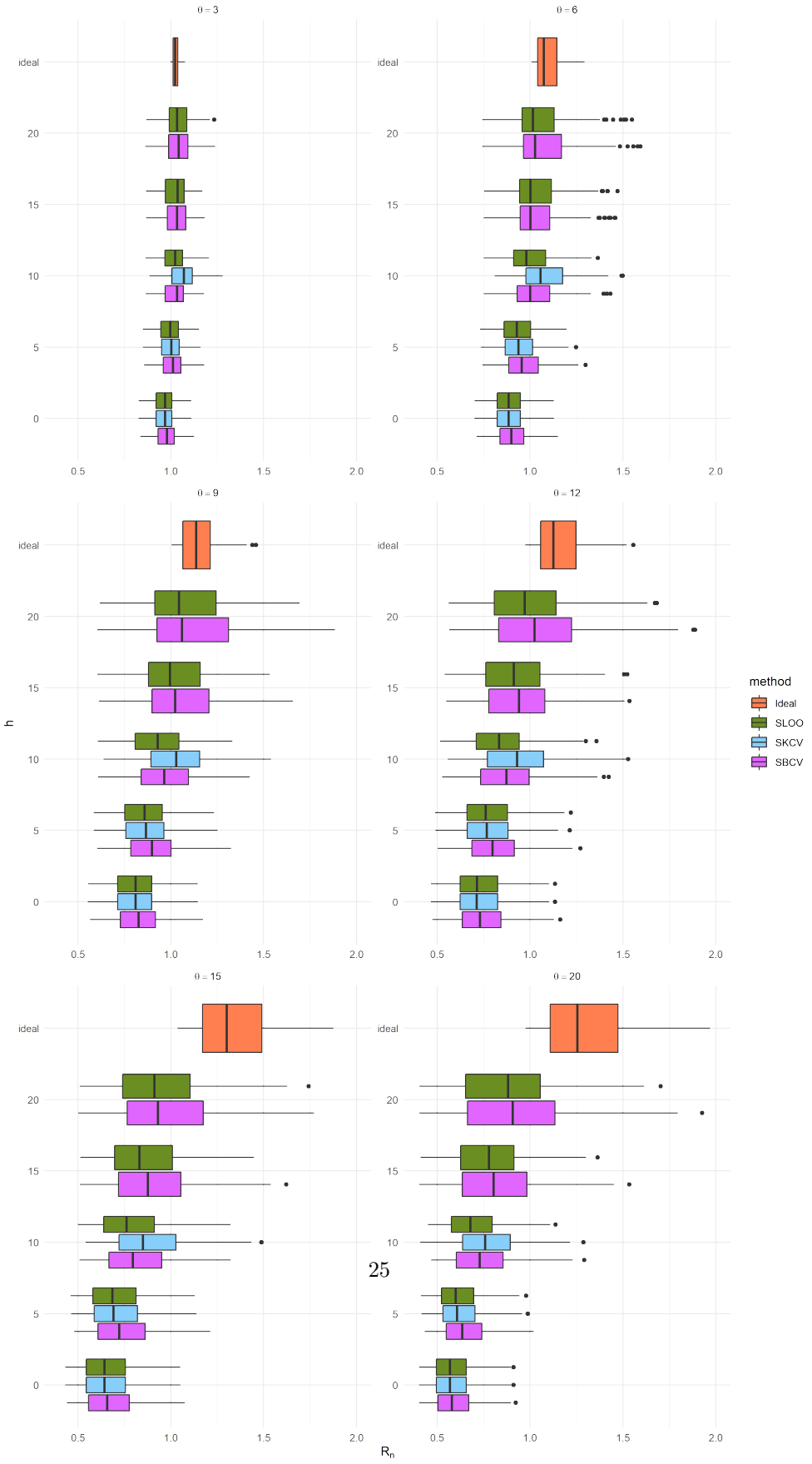
We see that across all examined values of θ, it is evident that the leave-one-out, K-fold cross-validation and Blocked cross-validation without spatial buffering methods yield the most optimistic risk assessments. This consistent optimism in risk estimates suggests a potential underestimation of the true risk in non-spatial cross-validation methods. This reinforces the idea that it is necessary to propose spatial adaptations of classic cross-validation methods.

For small values of h, the methods SBCV, SKCV and SLOO perform similarly, indicating that when the buffer size is minimal, the distinction between these spatially aware methods is less pronounced. As the buffer size h increases, differences among the methods become more apparent, reflecting how each method handles the exclusion of nearby spatial points differently.

As expected, the necessity for spatially aware cross-validation methods becomes more critical as the spatial dependence parameter θ increases. Larger values of θ correspond to stronger spatial autocorrelation, which means that observations are correlated over longer distances. Consequently, a larger buffer size h is required to adequately reduce the influence of spatial autocorrelation between training and validation sets.

However, at higher values of θ, we observe that none of the cross-validation methods manage to reach the ideal risk, underscoring a collective limitation in accurately estimating the predictive risk in the presence of strong spatial dependence. This suggests that as spatial autocorrelation intensifies, it becomes increasingly challenging for cross-validation methods to provide unbiased risk estimates.

To illustrate this, let us consider the case $\theta = 12$. The average value of the ideal risk $\mathcal{R}_n^{I(l)}$ is 1.2092. None of the spatial cross-validation methods achieve this value; the closest is SBCV with $h = 20$, which yields an average risk estimate of 1.1228, followed by SLOO with $h = 20$ at 1.0642.


θ = 3
θ = 6
θ = 9
θ = 12
θ = 15
θ = 20
ideal
20
15
10
5
0
0.5
1.0
1.5
2.0
h
R_n
method
Ideal
SLOO
SKCV
SBCV

While increasing the buffer size brings the risk estimates closer to the ideal risk, this improvement comes at the cost of increased variance in the risk estimates. Specifically, SBCV with $h = 20$ has a sample variance of 0.1753, and SLOO has a variance of 0.1282, whereas the sample variance of the ideal risk is much lower with a value of 0.0526. This indicates that larger buffer sizes introduce greater uncertainty into the risk estimates.

We could not consider larger buffer sizes because they would result in a substantial loss of data due to the expansion of the dead zone, leaving too few points for effective model training. Notably, we encountered computational limitations with SKCV at higher values of h, specifically $h = \{15, 20\}$, where the training set became empty after excluding the validation and dead zone areas. This highlights a practical constraint in applying spatial cross-validation methods with large buffer sizes, especially in datasets with limited sample sizes.

As θ increases, we see a corresponding rise in both the mean and variance of the ideal risk, signaling the mounting challenges in modeling the data with linear regression using OLS estimation amidst stronger spatial dependencies. Similarly, the risks estimated by both spatial and non-spatial cross-validation methods increase in variance with θ. Upon examining the spatial cross-validation methods more closely, we observe an upward trend in risk estimates as h increases. Additionally, the variability of these estimates grows with h, indicating greater predictive uncertainty at larger spatial separations for a constant θ.

Comparing SLOO and SBCV, we find that the results are generally similar. However, SLOO typically provides marginally lower risk estimates than SBCV for the same buffer size h, and exhibits slightly reduced variability. This suggests that both methods respond similarly to the spatial structure of the data for a given buffer size, but SLOO may offer a slight advantage in terms of both accuracy and precision.

These findings are supported by Figures 9 and 10, which illustrates the accuracy of risk estimates for SLOO and SBCV relative to the ideal risk. The accuracy is computed as,

$$\delta^2(h) = \left(\frac{1}{M} \sum_{l=1}^{M} \left(\mathcal{R}_{n,h}^{(l)} - \mathcal{R}_n^{I(l)} \right) \right)^2, \tag{21}$$

The variance of these estimates is also analyzed, calculated by,

$$\sigma^2_{\mathcal{R}_{n,h}} = \frac{1}{M} \sum_{l=1}^{M} \left(\mathcal{R}_{n,h}^{(l)} - \mathcal{R}_{n,h} \right)^2, \tag{22}$$

where $\mathcal{R}_{n,h} = \frac{1}{M} \sum_{l=1}^{M} \mathcal{R}_{n,h}^{(l)}$ is the average estimated risk over all simulation runs.

These two measures are presented in the spirit of a bias-variance decomposition, although $\delta^2(h)$ is not formally the squared bias. Specifically, $\delta^2(h)$ represents the squared difference between the average estimated risk and the average ideal risk, providing an indication of the accuracy (bias) of the cross-validation

method. The variance $\sigma^2_{\mathcal{R}_{n,h}}$ reflects the variability of the risk estimates across different simulations. Figures 9 and 10 present the curves of $\delta^2(h)$, $\sigma^2_{\mathcal{R}_{n,h}}$, and their sum $\delta^2(h) + \sigma^2_{\mathcal{R}_{n,h}}$, as functions of the buffer size h, for different values of the spatial dependence parameter θ, and for the risk estimates obtained using SLOO and SBCV methods.

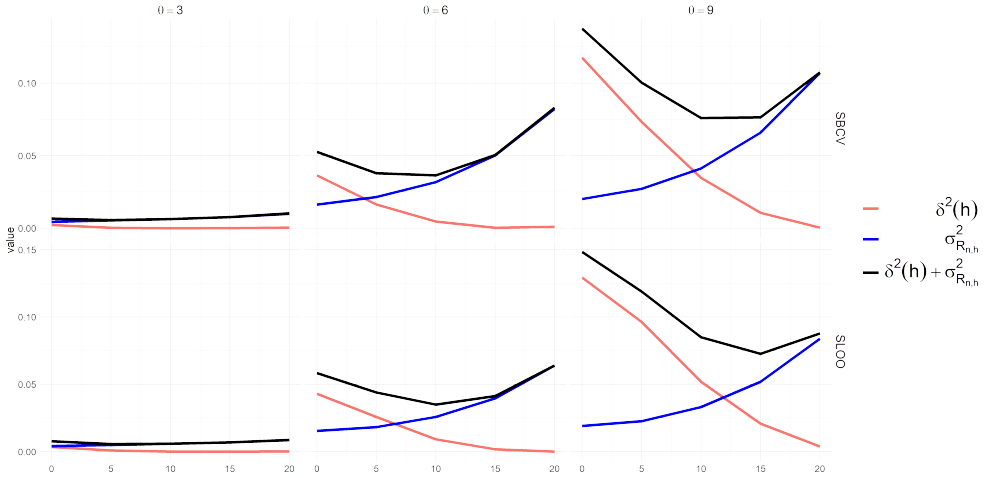


Figure 9: Comparative accuracy $\delta^2(h)$, variance $\sigma^2_{\mathcal{R}_{n,h}}$ of risk estimates, and their sum for SLOO and SBCV and increasing values of h for $\theta = (3, 6, 9)$.

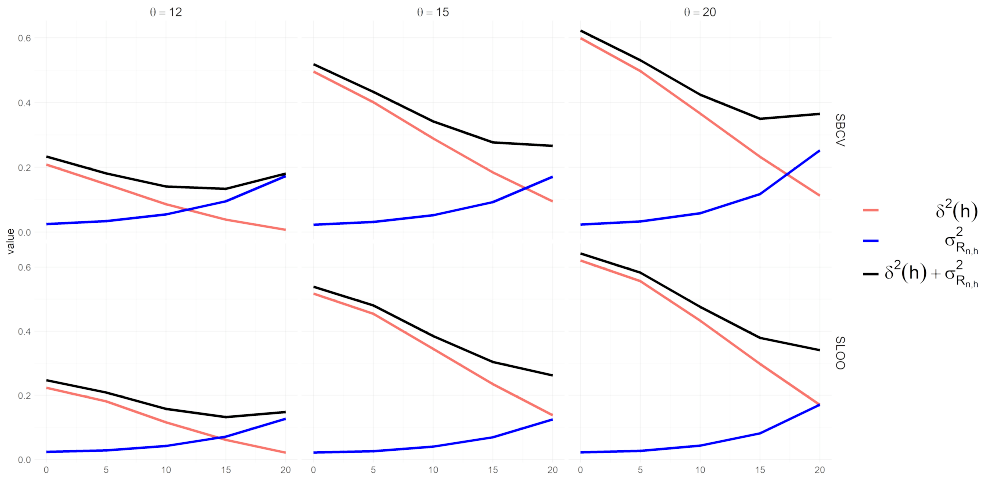


Figure 10: Comparative accuracy $\delta^2(h)$, variance $\sigma^2_{\mathcal{R}_{n,h}}$ of risk estimates and their sum for SLOO and SBCV and increasing values of h for $\theta = (12, 15, 20)$.

Starting with $\theta = 3$ on the left of Figure 9, representing low spatial dependence, we observe that increasing the buffer size h has little influence on both $\delta^2(h)$ and $\sigma^2_{\mathcal{R}_{n,h}}$. The accuracy measure $\delta^2(h)$ remains nearly constant and close to zero, indicating that the cross-validation estimates are already close to the ideal risk even without a buffer. This suggests that when spatial dependence is weak, the necessity for spatial buffering is minimal, and traditional cross-validation methods may suffice.

As the strength of spatial dependence increases (larger θ), we notice that the initial value of $\delta^2(0)$ is farther from zero, indicating a larger discrepancy between the estimated risk and the ideal risk when no buffer is used. This highlights the bias introduced by spatial autocorrelation when training and validation sets are not sufficiently separated. As the buffer size h increases, $\delta^2(h)$ decreases, approaching zero. This improvement in accuracy underscores the importance of increasing the distance between the training and validation sets to mitigate the influence of spatial autocorrelation.

However, for the two largest values of θ, representing strong spatial dependence, we observe that $\delta^2(h)$ does not reach zero within the range of buffer sizes we considered. This suggests that even with larger buffers, the cross-validation estimates cannot fully eliminate the bias introduced by strong spatial dependence, possibly due to practical limitations such as the finite size of the dataset or the maximum feasible buffer size before the training set becomes too small.

Simultaneously, we observe that as h increases, the variance of the risk estimates $\sigma^2_{\mathcal{R}_{n,h}}$ also increases. This is attributable to the reduction in the effective size of the training set as more data points are excluded by the buffer. A smaller training set leads to less stable model estimates and, consequently, higher variability in the risk estimates across simulations. The effect of increased variance is more pronounced with stronger spatial dependence (larger θ). Additionally, SBCV generally exhibits higher variance and lower accuracy than SLOO across different buffer sizes. This indicates that SLOO may be more reliable in terms of both precision (lower variance) and accuracy (lower bias) in estimating the predictive risk.

An important observation from Figures 9 and 10 is the presence of an apparent optimal buffer size that minimizes the total error, represented by the sum $\delta^2(h)+\sigma^2_{\mathcal{R}_{n,h}}$. Examining the curves for $\theta = 12$ in the left panels of Figure 10 , we observe that the accuracy improves for both SLOO and SBCV as the buffer size h increases, indicated by a decrease in $\delta^2(h)$. However, this improvement in accuracy comes at the cost of increased variance $\sigma^2_{\mathcal{R}_{n,h}}$. Initially, increasing h reduces the bias (improves accuracy) more than it increases the variance, leading to a net decrease in total error. Beyond a certain point, further increases in h lead to a smaller reduction in bias but a more substantial increase in variance, resulting in a net increase in total error. This demonstrates the classical bias-variance trade-off, where an optimal balance between bias and variance yields the most reliable risk estimates. Increasing h reduces bias due to mitigating spatial autocorrelation but increases variance due to a reduced effective sample size.

Overall, these results emphasize the importance of carefully selecting the buffer size h in spatial cross-validation methods. A larger buffer size effectively reduces the bias introduced by spatial autocorrelation between training and validation sets but can lead to higher variance in risk estimates due to fewer data points available for model training. Balancing this trade-off is crucial for obtaining reliable risk estimates.

Comparison of block sizes in SBCV

In examining SBCV, we explore the comparison of different block sizes. These blocks are crucial as they affect both the training and validation phases of the cross-validation process. Larger blocks encompass more spatial context, potentially capturing more spatial variability but at the cost of reduced granularity in local spatial patterns. Conversely, smaller blocks provide finer spatial resolution but may not adequately represent larger-scale spatial dependencies.

In practical terms, the optimal block size balances these considerations, aiming to capture significant spatial variability while maintaining sufficient sample size within each block for robust estimation. To further explore the influence of block size, we examine four distinct configurations: block sizes of 4×6, 8×12, 16×15, and 20×20, which respectively comprise 1%, 2%, 5%, and 8% of the total sample size.

Figure 11 illustrates the impact of different block sizes on risk estimates in SBCV, categorized by varying levels of spatial dependence. A consistent trend is observed where risk estimates increase with the enlargement of block size. Particularly in scenarios with minimal spatial dependence ($\theta = 3$), the risk estimates for the largest block size (20×20) are excessively pessimistic. Furthermore, the variance of these estimates tends to increase with block size. The necessity of using spatial buffering remains evident, especially as the strength of spatial dependence intensifies, even for the largest block size considered.

Conclusion

In summary, SKCV should not be considered due to the significant loss of observations it entails, which can adversely affect the reliability of risk estimates. In contrast, SBCV and SLOO emerge as superior methods for risk estimation, producing results that align more closely with the ideal risk. However, a crucial aspect in the application of these methods is the determination of the buffer size h, which directly influences the balance between bias and variance in the risk estimates.

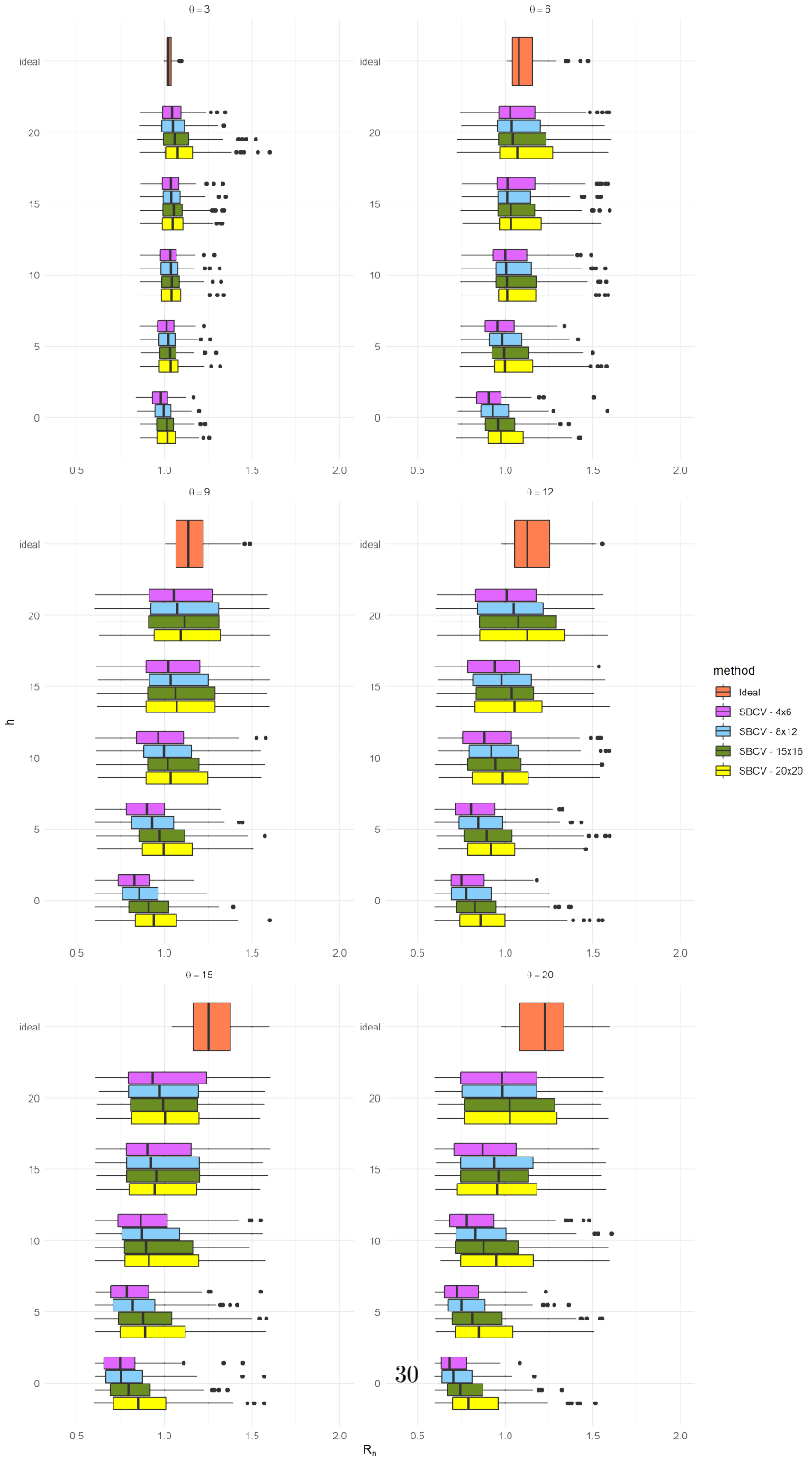


Figure 11: Risk estimates $\mathcal{R}_{n,h}^{\mathrm{SBCV}}$ based on 100 replications of the response variable Y, compared to the ideal risk across various block sizes.

When choosing between SBCV and SLOO, computational efficiency becomes a significant factor. SBCV offers a substantial reduction in computation time compared to SLOO, requiring only 200 iterations versus SLOO's 4800, thereby significantly reducing computational demands. However, SBCV introduces additional complexity in selecting the appropriate block shape and size, which requires careful consideration to ensure optimal performance.

Given these considerations, determining the optimal buffer size h becomes imperative for both SLOO and SBCV methods. The buffer size not only affects the degree of independence between training and validation sets but also impacts the effective sample size used for model estimation. An inappropriate choice of h can lead to biased risk estimates due to residual spatial autocorrelation or increased variance resulting from insufficient training data. The challenge lies in balancing the trade-off between reducing bias (by increasing h to minimize spatial autocorrelation between training and validation sets) and controlling variance (by keeping h small enough to retain an adequate number of training observations).

In the following chapter, we will address this critical issue by proposing methods for determining the optimal buffer size h for both SLOO and SBCV. We will explore empirical approaches, aiming to enhance the validity of risk estimates obtained from spatial cross-validation.