# *Model-based geostatistics for global public health using R*

*Emanuele Giorgi*
*Claudio Fronterre*

# *Contents*

# *Preface*

Its companion book "Model-based geostatistical for global public health'' by P. J. Diggle and Giorgi (2019) is a strongly recommended complementary read, as you work your way through this book.

# *Author*

An author bio.

# 1

# *Introduction*

The book shows how to carry out model-based geostatistical analysis of public health data using the `RiskMap` R package. In this introductory chapter, we explain what are the pre-requisites for using this book and its learning objectives. We also explain what software should be installed and how. Finally, we give a brief overview of the class of models covered in this book, and the examples that will be used to illustrate the methods and use of software.

## 1.1 Objectives of this book

The overall aim of this book is to provide you with the skills to perform a geostatistical analysis of a data-set using the R software environment. As you work your way through the book, you will learn to:

- explore geostatistical data-sets using graphical procedures and summary statistics;
- formulate and fit geostatistical models using the maximum likelihood estimation method;
- carry out prediction of health outcomes at different spatial scales;
- visualize and interpret the results from geostatistical models;
- model the relationships between spatially referenced risk factors and the health outcome of interest;
- validate the assumptions of geostatistical models and assess their predictive performance.

Although the focus of this book is on public health, the statistical ideas, as well as the software used, can also be applied for the analysis of geostatistical data-sets arising from other scientific fields.

## 1.2 Pre-requisites for using this book

To effectively understand and use the material presented in this book, it is expected that you should possess prior knowledge of basic probability theory, foundational topics in statistical modelling and R programming. Below we provide a more detailed explanation of the pre-requisites for each of these three fields.

### 1.2.1 Topics in probability

Basics probability theory is important to fully understand the content of this book. In particular, you should have knowledge of: the general definition and properties of continuous and discrete distribution; how the describe the properties of probability distributions through their mean, variance and skeweness; the concepts of stochastic dependence and correaltion; the distinction between marginal and conditional distributions; the basic properties of the Gaussian, Binomial and Poisson distributions; the definition and properties of the multivariate Gaussian distribution. The redear can find an extensive explanation and illustrations with examples of all these topics in Ross (2013).

### 1.2.2 Topics in statistics

Likelihood-based inference (whether frequensist or Bayesian) provides the theoretical bedrock for the estimation of almost any statistical model. In this book will focus on maximum likelihood estimation methods of inference. Extensive use of the notions of point and interval estimates obtained using the maximum likelihood estimation methods will be made through the book. Recommended readings include chapters 1, 2 and 4 of Pawitan (2001).

Good prior knowledge of Generalized linear models (GLMs) is essential, as the geostatistical modelling framework builds on these as an extension. Before embarking on the use of this book, we thus encourage you to review the basic theory of GLMs and, in particular, how these are applied and interpreted. In this book, we will cover examples that will model continuously measured outcomes and counts. Hence, good understanding of linear regression modelling and modelling of counts data using Binomial and Poisson regression should be the main focus of the review. For comprehensive overview of GLMs and their implementation in R, we refer you to Dobson and Barnett (2008).

### 1.2.3 Topics in R programming

Although this book does not require to possess advanced skills in R programming, it is important you have good knowledge in the following topics: creation

and manipulation of vectors and matrices; logical vectors; character vectors; handling of lists and data frame objects; reading data into R; graphical procedures. A very large amount of freely available material covering these topics can be found online. Our recommendation is to start from the manual "An introduction to R" of the Comprehensive R Archive Network available at this link, available at R manual.

## 1.3 Obtaining and running the R packages

It is advised that you obtain the latest 64-bit version of R in order to run the R code of this book. To install R, go to the R website, where you can download the installer packages for Windows and Mac, and find instructions for Linux, using binary files.

- Windows
- Mac
- Linux

The list of the R packages used in this book is provided in Table 1.1.

Table 1.1: List of the R packages that will be used in the book with a description of their use in the data analysis. The packages marked by (E) are essential for the geostatistical analysis. Those instead marked by (R) are recommended and can be helpful to overcome issues as described under the column "Used for".

| R packages | Used for |
|---|---|
| `RiskMap` (E) | Estimating of geostatistical models and spatial prediction |
| `sf` (E) | Handling of spatial data in R |
| `terra` (E) | Handling of raster files in R |
| `ggplot2` (E) | Creating maps and exploratory plots |
| `crsuggest` (R) | Guessing a coordinate reference systems when unknown |

To install packages in R for the first time, you can use the command `install.packages` in the R console, as shown below for the `RiskMap` package.

```
install.packages("RiskMap")
```

## 1.4 Data-sets used in the book

The geostatistical data-sets described in this section will be used throughout the book to illustrate the use of the R packages mentioned in the previous sections.

Each of the examples data-sets can be loaded from the `RiskMap` package, using the command

```
data(NAME_OF_THE_DATASET)
```

where in place of `NAME_OF_THE_DATASET` you should type of the name of one of the data-sets listed in Table 1.2.

Table 1.2: List of data-sets available from the `RiskMap` package. Data-sets listed as "Example" are used throughout the book to illustrate the use of R functions. Data-sets listed as "Case study" are analysed in Chapter 5.

| Names of the data-set | Short description | Used in this book as |
|---|---|---|
| `galicia` | Lead concentration m from moss samples collected in Galicia, Northern Spain | Example |
| `liberia` | Prevalence data on river-blindness from Liberia | Example |
| `malkenya` | Malaria prevalence data from a community and school survey conducted in Western Kenya | Example |
| `italy_sim` | Simulated geostatistical data-set within the Italian national boundaries | Example |
| `malnutrition` | Data on stunting among children in Ghana | Case study |

### 1.4.1 Lead concentration in Galicia

Lead is a heavy metal which, in high concentrations, can cause chronic damage to living organisms over a long period of time. For this reason its spread and source must be regularly monitored. To assess the extent of the contamination

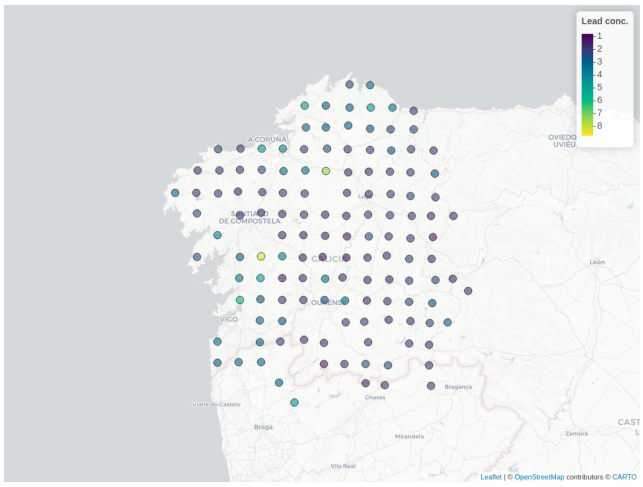



Figure 1.1: Data on the measured lead concentration (in micrograms per gram dry weight) in moss samples collected in Galicia, North-West of Spain.

in an area, measurements of lead are often taken from plants. The data here considered (Figure 1.1) consist of 132 locations of moss samples collected in 2000, in and around Galicia, a region in the North-Western part of Spain. One of the objectives of this survey was to establish the spatial pattern of lead concentration in Galicia so as to better identify possible sources of contamination; fore more information, see Fernández, Rey, and Carballeira (2000).

In this case, geostatistical modelling can be used to predict the lead concentration across Galicia and allows to disentangle variation which is purely random, possibly due to measurement error, and genuine spatial variation, which is our main object of interest.

This data-set will be used in this book to show how to carry out the spatial analysis of continuously measured variables using linear geostatistical models.

### 1.4.2 River-blindness in Liberia

In low-resource settings, where disease registries are typically absent, cross-sectional surveys are an essential monitoring tool that enables the estimation of the disease burden in a population of interest. The data considered in this example (Figure 1.2) have been collected as part of an Africa-wide initiative called the Rapid Epidemiological Mapping of Onchocerchiasis (REMO) carried out in 2011 in 20 African countries (Zouré et al. 2014). The goal of REMO is to identify areas where river-blindness (or onchocerchiasis), a disease transmitted by black flies who breed along fast flowing rivers, is still a public health problem. In this context, it is especially of interest to identify communities

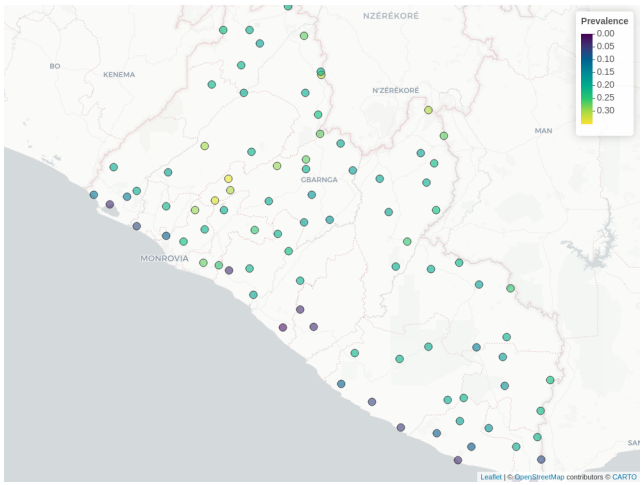



Figure 1.2: River-blindness data from a cross-sectional survey carried out in Liberia.

with a prevalence above 20% where treatment is urgently needed.

In this book, we will use data collected from Liberia to model nodule prevalence, which is based on a alternative and cheaper diagnostic technique for river-blindness. In the analysis of this data-set, we will illustrate how to formulate and fit Binomial geostatistical models, and how these can be used to predict prevalence within a region of interest.

### 1.4.3 Malaria in the Western Kenyan Highlands

Malaria is one of deadliest diseases that affects populations living in tropical and subtropical countries. It is caused by a parasite of the genus Plasmodium which is transmitted through the infectious bite of female Anopheles mosquitoes. In the following chapters, we shall analyse a data-set (Figure 1.3) from a cross-sectional community survey carried out in July 2010 in Nyanza Province, in the Western Highlands of Kenya (Stevenson 2013).

What distinguishes this from the other examples data-sets is that the data contain both individual-level and household-level information. The outcome of interest is the result from a rapid diagnostic test for malaria. In the book, we will illustrate how to account for the the hierarchical structure of the data and the binary nature of the outcome at each of the stages of the geostatistical analysis.

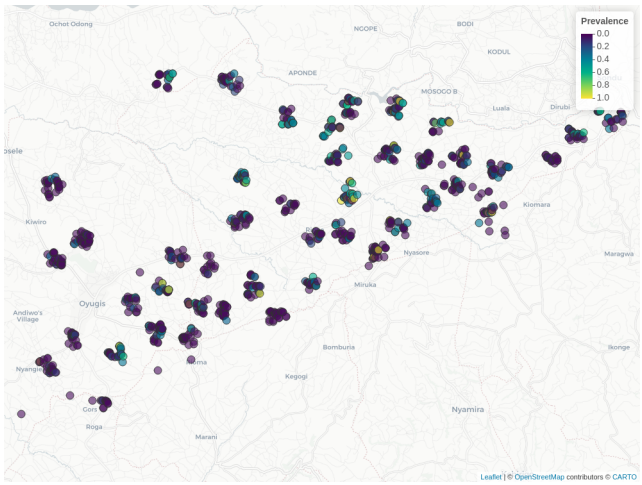



Figure 1.3: Malaria prevalence data from a cross-sectional survey carried out in Nyanza Province, in the Western Highlands of Kenya.

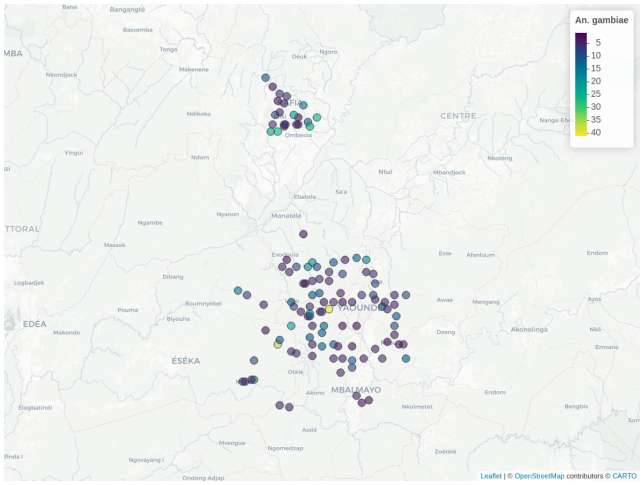



Figure 1.4: Map of the collected number of *Anopheles gambiae* mosquitoes in an area of Southern Cameroon.

### 1.4.4 *Anopheles gambiae* mosquitoes in Southern Cameroon

In studies of vector-borne and zoonotic diseases, understanding the vector distribution can help to better guide the decision-making process for the implementation, monitoring and evaluation of control programmes. *Anopheles gambiae* mosquitoes are one of the main vectors for malaria transmission in sub-Saharan Africa. Their distribution over space is affected by several environmental and climatic factors, including temperature, humidity and vegetation.

The data-set on mosquitoes (Figure 1.4) that we will use in the book consists of a sub-set taken from a larger database (Tene Fossog et al. 2015). This was assembled in order to understand how the environment affects the distribution of different species of *Anopheles* mosquitoes in sub-Saharan Africa. This example data-set will be used to illustrate the application of Poisson geostatistical models for mapping mosquitoes abundance.

### 1.4.5 Simulated-dataset

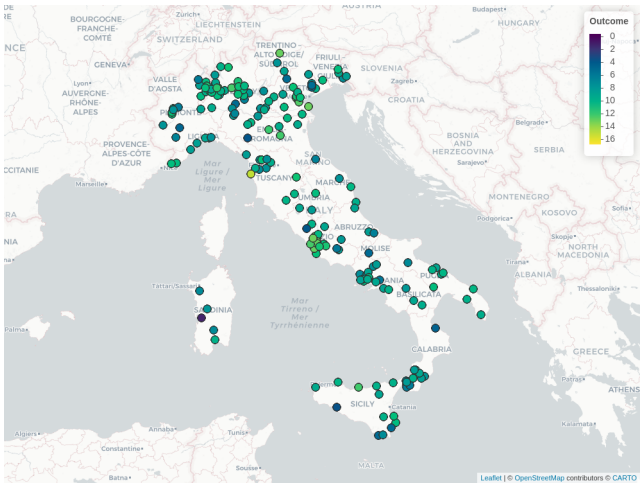



Figure 1.5: Map of the locations of the simulated data-set generated over Italy for a continuous outcome.

The data-set reported in Figure 1.5 was generated using a geostatistical model, with the addition of unstructured random effects at provincial and regional level. More details on how this data-set was generated will be provided in Section 3.2. Whilst this data-set does not have any scientific relevance like the other data-sets used in this book, it will serve us to illustrate some of the more advanced features of the package that enable the inclusion of random effects, in addition to the latent Gaussian process that is common to all geostatistical models. The skills you will aquire through the analysis of this data-set will be

useful for the analysis of data-sets presented as case studies in Chapter 5.

## 1.5 Geostatistical problems and geostatistical models

What the examples of the previous section have in common is that, in each case, the goal of the statistical analysis is to draw inferences on an unobserved spatially continuous surface using data collected from a finite set of locations. The lead concentration in Galicia, the prevalence for river-blindness in Liberia and the abundance of *A. gambiae* mosquitoes in Cameroon can all be represented as spatially continuous processes that originate from the combined effects of environmental factors. We denote this class of inferential problems as *geostatistical problems* for which a solution can be found through the development and application of suitable *geostatistical models*, which are the subject of this book.

As one can soon realize, geostatistical problems are not unique to global health but arise in many other fields of science, including economics, physics, biology, geology and others. It thus comes to no surprise that geostatistics was initially developed in the South African mining industry in the 1950s (Krige 1951). This was then further developed as a self-contained discipline by Georges Matheron and other researchers at Fontainebleau, in France (Matheron 1963; Chilès and Delfiner 2016). In Watson (1971) and Watson (1972) a first connection is drawn between geostatistics and the prediction of stochastic processes. However, it is only with Ripley (1981) and then N. A. C. Cressie (1991) that geostatistics is explicitly brought into a classical statistical framework for the analysis of spatially referenced data. P. J. Diggle, Tawn, and Moyeed (1998) coined the term *model-based geostastics* and introduced this as belonging to the general class of generalized linear mixed models (Breslow and Clayton 1993), while emphasizing the use of likelihood-based methods of inference. As in P. J. Diggle, Tawn, and Moyeed (1998), also in this book, we advocate the application of model-based geostistical models as a class of parametric statistical models on which inference can be carried out using either maximum likelihood estimation or Bayesian methods.

More precisely, our attention will be directed at the class of *generalized linear geostatistical models*, or GLGM. To formally specify this, we first define the random variables $S$, a spatial stochastic process, and the random variable $Y = (Y_1, \dots, Y_n)$ which correspond to the outcome observed at a set of locations $X = (x_1, \dots, x_n)$. Let us use $[A]$ to denote "the distribution of the random variable $A$". To formulate a GLGM, we should then specify the joint distribution of $S$ and $Y$, which we write as

$$[Y, S] = [S][Y|S]. \tag{1.1}$$

On the right-hand side of the equation above, we have factorized the joint distribution of $Y$ and $S$, as the product between the marginal distribution of $S$ and the conditional distribution of $Y$ given $S$. Hence, the formulation of a GLGM can be break down into the tasks of formulating $[S]$ and $[Y|S]$.

In defining $[S]$, throughout the book, we shall assume that this is a zero-mean stationary and isotropic Gaussian process. In other words, these assumptions impose that the joint distribution of $S(X) = (S(x_1), ..., S(x_n))$, i.e. the process $S$ at the sampled locations $x_1, ..., x_n$, is invariant with respect to rotations and translations of the locations $X$. In practical terms, the main implication of this is that, for any pair of locations $x_i$ and $x_j$ the correlation function $\rho(\cdot)$ between $S(x_i)$ and $S(x_j)$ is purely a function of the Euclidean distance, $u_{ij}$, between $x_i$ and $x_j$, i.e.

$$\text{cov}\{S(x_i), S(x_j)\} = \sigma^2 \rho(u_{ij}), \tag{1.2}$$

where $\sigma^2$ is the variance of $S(x)$ for all $x$. Below we provide more details on the type of covariance functions that we consider in this book. Furthermore, the fact that we assume the process $S$ to have mean zero is because this process acts as a residual term in our modelling of $Y$. This aspect will be reiterated several times in the following chapters, as it as important implications for the interpretation of the other components of a geostatistical model, as well understanding the results of the analysis.

Finally, we model $[Y|S]$, i.e. the distribution of $Y$ given $S$, as a set of mutually independent distributions which belong the exponential family, as defined in classical generalized linear modelling framework (Nelder and Wedderburn 1972). It then follows that, we can write $[Y|S]$ as

$$[Y|S] = \prod_{i=1}^{n} [Y_i|S(x_i)]. \tag{1.3}$$

The final step then consists of specifying a distribution for $[Y_i|S(x_i)]$. Table 1.3 gives the range, mean and variance of the three specifications for $[Y_i | S(x_i)]$ which we will consider in this book. In Table 1.3, the *canonical function*, say $g(\cdot)$, denotes the natural transformation of the mean component $\mu_i$ that allows us to introduce both covariates and the spatial process $S(x_i)$ into the model so as to explain the variation in $\mu_i$ as

$$g(\mu_i) = d(x_i)^\top \beta + S(x_i). \tag{1.4}$$

where $d(x_i)$ is a vector of spatially referenced covariates with associated regression coefficients $\beta$. Finally, the quantity $m_i$, which appears in the formulation

of the Binomial and Poisson distributions, is an offset quantity and is used to account for the number of *tests* or the population size at a given location $x_i$.

Table 1.3: Type of outcomes $Y_i$ considered in this book.

| Distribution | Range of $Y_i$ | Mean of $[Y_i\|S(x_i)]$ | Variance of $[Y_i\|S(x_i)]$ | Canonical link |
|---|---|---|---|---|
| Gaussian | $(-\infty, +\infty)$ | $\mu_i$ | $\tau^2$ | $g(\mu_i) = \mu_i$ |
| Binomial | $1, ..., m_i$ | $m_i\mu_i$ | $m_i\mu_i(1-\mu_i)$ | $g(\mu_i) = \log\{\mu_i/(1-\mu_i)\}$ |
| Poisson | $1, 2, ..., \infty$ | $m_i\mu_i$ | $m_i\mu_i$ | $g(\mu_i) = \log\{\mu_i\}$ |

Based on the formulation in (1.4), we can see that $S(x_i)$ quantifies residual spatial effects on $\mu_i$ that have not been accounted for by the covariates $d(x_i)$. In an ideal scenario, the covariates $d(x_i)$ should explain all the spatial variation without the need for $S(x_i)$. Although this is often unrealistic, in practice we may be able to explain most of the variation in $\mu_i$ through $d(x_i)$ and, hence, reduce $S(x_i)$ to a negligible component. In Chapter 2, we will show how a thorough exploratory analysis can help to understand whether we have come close to that ideal scenario or, if instead, we need the use of GLGM to model the data.

The model described in (1.4) can be seen as the most basic GLGM that can be used for a geostatistical analysis. As we will see in the analysis of some of the examples and, in Chapter 6, for the case studies, extensions of this model will be required to accommodate the intrinsic non-spatial random variation of the data which is not captured by the covariates.

The types of problems that statistical models are applied to can be distinguished into three main categories: prediction problems; explanatory problems; problems of hypothesis testing. Most of the times, geostatistical problems tend to fall under the first category, where the goal is make predictive inferences on the process $S(x)$ at location $x$, which is usually outside of the set of sampled locations. However, as we will illustrate in the later chapters, geostatistical models play an important role also in the other two types of problems. In particular, we will show that spatial correlation can have a substantial impact on the point estimates and standard errors for $\beta$. Hence, if the goal of the analysis is to explain the relationship between a covariate $d(x)$ with the mean component $\mu$.

### 1.5.1 The Matern family of correlation functions

Throughout the book, we shall consider the Matern (2013) family of correlation functions to model the spatial correlation of the Gaussian process $S(x)$. This is defined as

$$\rho(u;\phi,\kappa) = \{2^{\kappa-1}\Gamma(\kappa)\}^{-1}(u/\phi)^{\kappa}K_{\kappa}(u/\phi), \tag{1.5}$$

where $\phi > 0$ and $\kappa > 0$ are parameters and $K_{\kappa}(\cdot)$ is the modified Bessel function of the third kind of order $\kappa$. The parameters $\phi$ and $\kappa$ regulate how fast the spatial correlation decays to zero for increasing distance and the smoothness of the process, respectively. A special case of Matern family of correlation functions, which is obtained when $\kappa = 0.5$, will be of particular relevance to the applications considered in this book. This is the exponential correlation function which we write as

$$\rho(u;\phi) = \exp\{-u/\phi\}. \tag{1.6}$$

Another special case, which we don't consider in this book but has often been used in machine learning applications, is the Gaussian correlation function obtained as a limiting case for $\kappa \to +\infty$ the possible smoothest process arising from the Matern family.

To better understand how $\phi$ and $\kappa$ affect the spatial correlation and the pattern of the spatial surface, we now consider some examples.

Figure 1.6 shows six different Matern correlation functions. In panel (a), we have kept $\kappa$ fixed to 0.5 and varied $\phi$ over the values 0.05, 0.1 and 0.2. As expected, for larger values of $\phi$ the correlation function has a slower decay to zero. Panels (a) to (c), in Figure 1.7, show three realizations of a Gaussian process from each of these correlation functions. The mean of the Gaussian process was set to zero and variance to 1. We can observe that spatial correlations with larger scales are associated with longer spatial trends, whilst smaller scales exhibit a patchier pattern. This is because, as $\phi$ takes values that are closer to zero, the spatial surface will tend to show a less structured pattern and will revert towards its zero mean more rapidly. In our examples in the book, we will often use the so called *practical range* to interpret our estimates for $\phi$. The practical range is defined as the distance at which spatial correlation reaches 0.05, hence one can interpret this as the distance beyond which observations can be considered approximately independent. In the case of the exponential correlation function, the practical range is $\log(20) \times \phi \approx 3\phi$.

Finally, let us consider the correlation functions, shown in the panel (b) of Figure 1.6. Here, we have varied $\kappa$ over the values 0.5, 1.5 and 2.5, whilst $\phi$ has been fixed in order to force all three correlation functions to reach 0.05 for distance 0.3. In this way, we can better observe the effect of different values of $\kappa$ on the spatial surface for process that have approximately the same range for the spatial correlation. In Figure 1.7, we observe three realizations

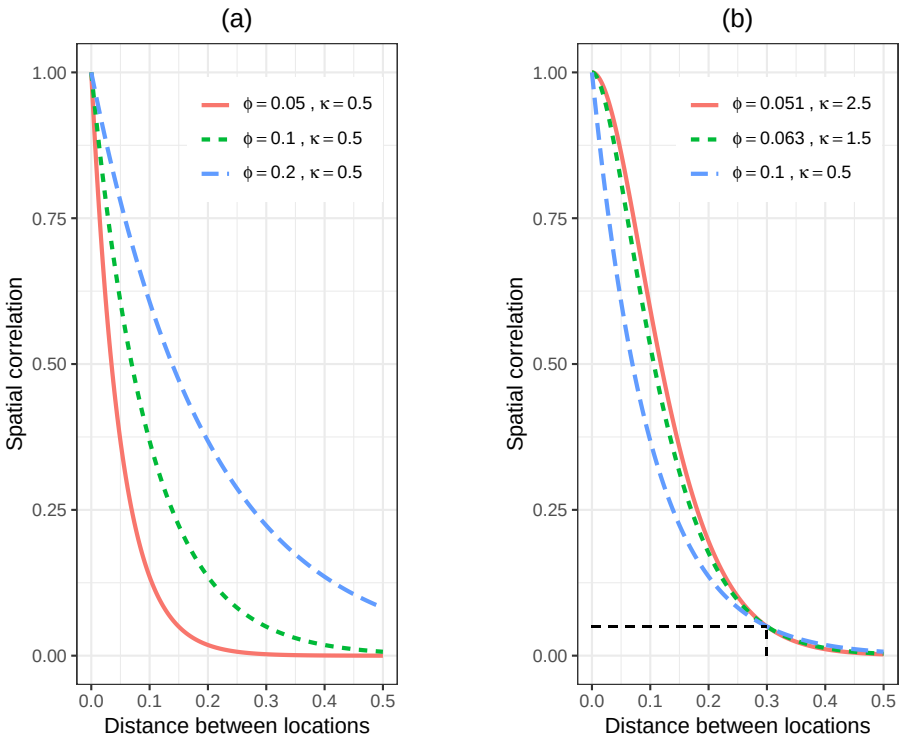



Figure 1.6: Examples of stationary and isotropic Matern correlation functions. Panel (a) shows three different correlations functions that have the same smothness parameter of $\kappa = 0.5$, while varying the scale parameter $\phi$ over $0.05, 0.1, 0.2$. In panel (b) the scale of the spatial correlation $\phi$ is chosen so that each of the three functions reaches 0.05 at distance 0.3 (as also shown by the horizontal and vertical black dashed segments).

from these correlation functions. We observe that the differences between the different surfaces are determined by the small spatial scale behavior; $\kappa = 0.5$ correspond to a rougher and less regular spatial pattern, whilst $\kappa = 2.5$ shows the smoothest surface among the three processes considered. These properties of the spatial surface are related to the so called differentiability of the the Guassian process, which determines its local behavior. If you are interested in delving into these theoretical aspects, we suggest reading Chapter 2 of Stein (1999).

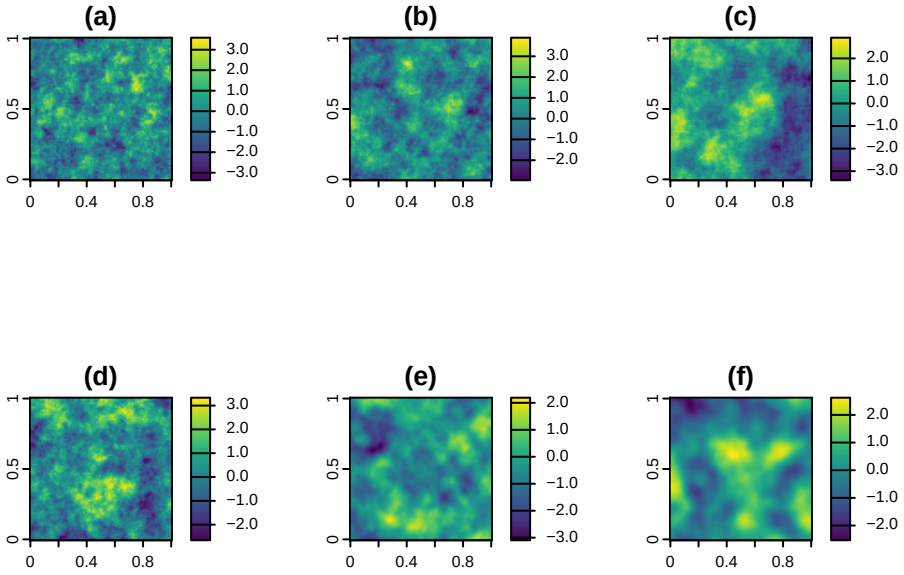



Figure 1.7: Simulated spatial surface using the three correltion functions shown in Figure 1.6. Panels (a), (b) and (c) correspond to the correlation functions from panel (a) in Figure 1.6 and in order these are: $\phi = 0.05$ and $\kappa = 0.5$; (b) $\phi = 0.1$ and $\kappa = 0.5$; (c) $\phi = 0.2$ and $\kappa = 0.5$. Panels (c), (d) and (e) correspond to the correlation functions from panel (b) in Figure 1.6 and in order these are: $\phi = 0.1$ and $\kappa = 0.5$; (b) $\phi = 0.063$ and $\kappa = 1.5$; (c) $\phi = 0.051$ and $\kappa = 2.5$.

The flexibility provided by the Matern correlation function in capturing different forms of spatial correlations has made it one of, if not the most widely used correlation function in model-based geostatistics (Stein 1999). For this reason, in this book we will consider the Matern correlation function. We will consider estimation issues related to the Matern correlation in Chapter 3.

## 1.6 Workflow of a statistical analysis and structure of the book

Figure 1.8 shows the different stages that will follow in carrying the geostatistical analysis of the examples introduced in Section 1.4. The exploratory analysis of the data is an essential first step that is used to understand the empirical associations between risk factors and the the health outcome of interest. In our case, this first stage is also used to justify the use of geostatistical models by questioning the underlying assumptions of standard generalized linear models. Based on the results obtained from the exploration of the data, we then formulate a suitable statistical model and estimate its parameters using likelihood based methods of inference. These also allows us to obtain uncertainty measures about the strength of associations of regression relationships and the other model parameters that define the shape of the spatial correlation in the data. Following the estimation of the model, we then proceed to validate its underlying assumptions using suitable diagnostics that assess whether the model can later be sufficiently trusted to represent the observed variation in the modeled outcome. At this stage, if the diagnostics checks yield results that indicate the incompatibility of the model with the data, we go back to the stage of model formulation and address the issues arisen from the validation stage. If instead, we do not find any evidence against the fitted model we can proceed to carry out spatial prediction. At this stage, it is important to define suitable predictive targets that can help us to better answer the original research question and better assist the decision making process. The final step of visualization of uncertainty plays an important role in geostatistical analysis in order to convey the main findings of the study in an effective and easy-to-understand way for a wider audience which also consists of non-experts.

In the remainder of this book, each chapter focuses on a specific stage as shown in Figure 1.8. We treat visualization of uncertainty together with spatial prediction in Chapter 4.

Chapter 2 will provide an overview of how to handle spatial data in R, in particular raster and vector data (both points and polygons). The skills learned in this chapter will be applied throughout the book, and will especially be useful in Chapter 4 and Chapter 5 for generating predictive maps of the modeled outcome.

Chapter 3 focuses on the model building process and estimation of geostatistical models. This chapter will show how to carry out initial exploratory analyses of the data to inform the formulation of suitable geostatistical models and how these can be fitted using maximum likelihood estimation methods.

Chapter 4 shows how geostatistical models can be used to carry out spatial

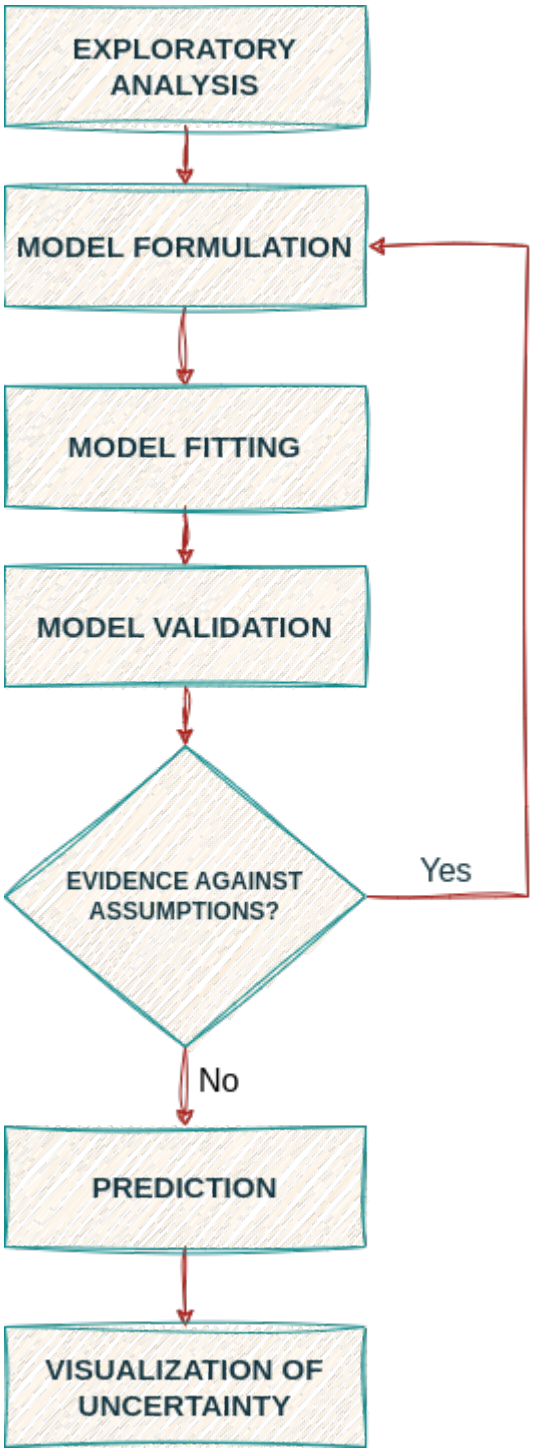



Figure 1.8: Stages of a statistical analysis

prediction of a health outcome of interest both on a spatially continuous and spatially aggregated scales.

Finally, Chapter 5 presents the application of all the methods illustrated in the previous chapters to three additional data-sets. This chapter offers a summary of the content of the book by putting together all the stages in the geostatistical analyses for each of the three case studies, and illustrates additional functionalities of the `RiskMap` R package not covered in the previous chapters.

# 2

# *Handling of spatial data in R*

## 2.1 Introduction

Geostatistical analysis primarily deals with **point-referenced data**. However, geostatistical modeling often requires more than just point data, **raster** and **areal (polygon)** data are frequently used to build essential covariates, enhance spatial understanding, or provide environmental context. For instance, population density, climate data, land use, and elevation, often key covariates in many spatial analyses, usually come in raster or polygon formats. Efficient handling of these different data types is critical for successful model-building. This chapter provides a comprehensive guide to the various stages of managing spatial data in R, using the `sf` and `terra` packages, to support a complete geostatistical workflow.

## 2.2 Spatial data handling in geostatistical analysis

Throughout a geostatistical analysis, handling spatial data takes place at multiple key stages:

1. **Retrieving External Spatial Data Sources**: Before starting the modeling process, one often needs to acquire spatial datasets from external sources to improve model accuracy. These could include satellite-derived environmental variables, population data from WorldPop, or climate data from WorldClim. Acquiring these datasets and ensuring they are in compatible formats and resolutions is an essential early step in spatial analysis.
2. **Importing and Standardizing Spatial Data**: Once external data is obtained, it must be imported into R. This involves handling various file formats such as shapefiles or geopackages for vector data and GeoTIFFs for raster data. Moreover, different datasets might use different **Coordinate Reference Systems (CRS)**, requiring

CRS standardization to ensure that all datasets align correctly. Failure to standardize CRS can result in spatial misalignments and incorrect results.

3. **Extracting Covariate Information for Modeling**: Covariate extraction is an essential step in geostatistical modeling. After importing spatial data, it is necessary to extract relevant covariate information both for the **sampled locations** (where we have observed data) and the **prediction locations** (where we wish to make predictions). This step involves linking raster or polygonal covariates, such as climate, population, or land cover data, to the geostatistical data points.
4. **Prediction and Creation of a Spatial Grid**: For predictive geostatistical models, a regular grid of points is often created over the study region. Covariates must then be assigned to each grid point to make predictions across the entire region of interest. Creating predictive grids and linking them to the necessary covariates is key to generating continuous spatial predictions from geostatistical models.
5. **Visualizing Spatial Data**: Visualization is crucial for exploring spatial data, interpreting model results, and communicating findings. Whether working with point-referenced data, polygons, or rasters, clear and effective visualization helps reveal patterns that inform the modeling process. Effective visualization can also help highlight covariate trends, spatial clusters, and uncertainty in predictions.

## 2.3 Accessing covariates for disease mapping

Covariates can play a crucial role in understanding and modeling the spatial variation in disease risk. In geostatistical analysis, incorporating environmental, demographic, and climatic variables might improve the predictive power of models or at least reduce the level of uncertainty in the predictions. These covariates can influence factors such as the spread of infectious diseases, the distribution of disease vectors, and the socio-economic conditions that impact health outcomes.

A wide range of open-access spatial datasets provide covariates for disease mapping. These include population density, climate, land cover, and human infrastructure data, which often come in raster or polygon formats. Alongside these environmental covariates, administrative boundaries are also crucial for public health analysis, as they help organize and aggregate data at different

levels (e.g., country, region, or district). For example, aggregating health outcomes or covariate data by administrative units allows researchers to identify geographic disparities and allocate resources accordingly. Fortunately, open-source platforms such as geoBoundaries provide easily accessible administrative boundary data for many countries at various levels of granularity. This data is available in formats compatible with geospatial analysis tools in R, making it easy to integrate into geostatistical workflows.

Table 2.1 below summarizes key sources of covariates useful in disease mapping. Each dataset offers specific types of data, from satellite-derived environmental variables to gridded population estimates and administrative boundaries, which can be accessed through various R packages or APIs.

Table 2.1: Some key sources of covariates useful in disease mapping and R packages to retrieve them.

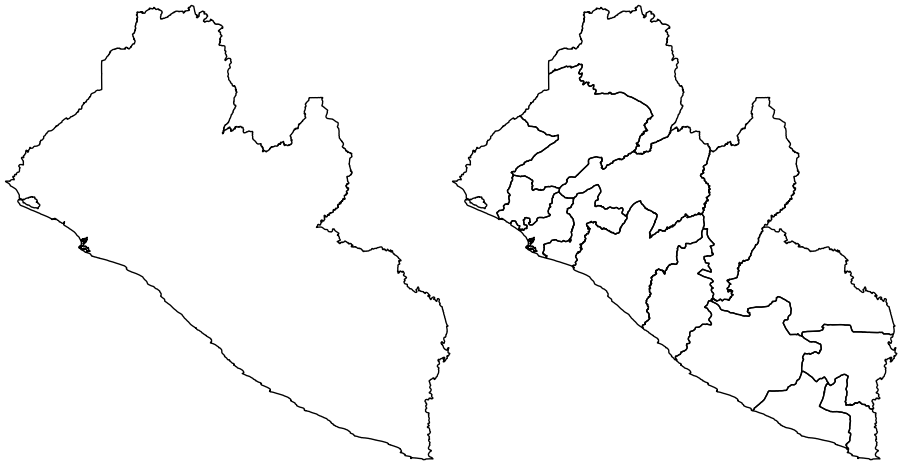
| Source | Data Type | Description | R Package |
|---|---|---|---|
| **WorldClim** | Climate (temperature, rainfall) | Global climate data | `geodata` |
| **MODIS** | Remote Sensing | Satellite imagery (e.g., vegetation indices) | `MODIStsp` |
| **OpenStreetMap (OSM)** | Human settlements, roads | Global geographic features | `osmdata` |
| **Google Earth Engine** | Satellite imagery | Large-scale environmental data analysis | `rgee` |
| **WorldPop** | Population data | Gridded population density estimates | `wpgpDownloadR` |

### 2.3.1 Example: Downloading administrative boundaries

Administrative boundaries provide an essential spatial structure for many types of geostatistical analyses, particularly in disease mapping where data is often aggregated by administrative units such as regions, districts or provinces. The geoBoundaries datasets, which are accessible via the `rgeoboundaries` package, provide openly available administrative boundary data for nearly every country, allowing researchers to integrate these boundaries into their analyses seamlessly.

```
par(mar = c(0, 0, 0, 0))  #| hide_line
# Load the rgeoboundaries package
library(rgeoboundaries)

# Download administrative boundaries for Liberia (level 0:
↪  country)
```

```
liberia_admin0 <- gb_adm0("Liberia")

# Do the same for level 1: regions
liberia_admin1 <- gb_adm1("Liberia")

# Plot the administrative boundaries
plot(liberia_admin0$geometry)
plot(liberia_admin1$geometry)
```



(a) Adimin level 0 (b) Admin level 1

Figure 2.1: Administrative boundaries for Liberia retrieved using the `rgeoboundaries` package.

### 2.3.2 Example: Downloading population data

Population density is a key covariate in geostatistical models for public health research. In this section, we demonstrate how to retrieve high-resolution population data for Liberia from WorldPop using the `wpgpDownloadR` package. This package provides easy access to the WorldPop datasets, which offers gridded population estimates at various spatial resolutions.

Before downloading the data, we will search for available datasets for Liberia. The function `wpgpListCountryDatasets()` helps in retrieving a list of all available datasets for a specified country.

```
library(wpgpDownloadR)

# Search for datasets available for Liberia
# usign the ISO3 country code
```

```
lbr_datasets <- wpgpListCountryDatasets(ISO3 = "LBR")
```

We can check the description column to see what datasets are available. Let's download the population data for Liberia for the year 2014 at a 100m resolution. The `wpgpGetCountryDataset` function will then download a raster dataset based on ISO3 code and covariate name.

```
lbr_pop_url <- wpgpGetCountryDataset(ISO3 = "LBR", covariate =
 ↪  "ppp_2014")
```

This will download a raster file locally in a temporary directory. The path to the downloaded file is contained in the `lbr_pop_url` variable and when we introduce the `terra` package in the next sections we will show how to upoload the population raster into R. It is also possible to specify the directory where we want the raster to be downloaded using the `destDir` argument.

### 2.3.3 Example: Dowloading data using Google Earth Engine

Google Earth Engine (GEE) is a powerful platform that hosts petabytes of environmental and geospatial data, including climate, topography, and land cover datasets. The `rgee` R package provides a convenient interface to GEE from within R.

**Getting started with `rgee`**: To use `rgee`, you need a Google account and must complete a one-time setup process. Follow the official guide here: https://r-spatial.github.io/rgee/. For further tutorials and use cases, see: https://csaybar.github.io/rgee-examples/.

In this example, we retrieve elevation data for Liberia using the SRTM 90m dataset. The steps include initializing Earth Engine, access the desired data product with the `ee$Image("ASSET_ID")` function (the `ASSET_ID` can be found on the dataset page under *Earth Engine Snippet*), converting the boundary of Liberia into a format readable by GEE, clipping the global elevation raster to that boundary, and downloading the result as a raster object in R. This process demonstrates how `rgee` can be integrated into a geostatistical workflow to enrich your data with high-quality environmental covariates.

> **Note**
>
> This example assumes you have already completed `rgee` setup and authenticated using `ee_Initialize()`.

```
# Load required packages
library(rgee)
```

```r
# Initialize Earth Engine (first time will prompt browser login)
ee_Initialize(quiet = T)

# Convert Liberia boundary to Earth Engine Geometry
liberia_ee <- sf_as_ee(liberia_admin0)

# Access the global SRTM elevation dataset from Earth Engine
elev <- ee$Image("CGIAR/SRTM90_V4")

# You can check general info about the data produc with ee_print
ee_print(elev)
```

```
------------------------------------------------------------
Earth Engine Image --
Image Metadata:
 - Class                       : ee$Image
 - ID                          : CGIAR/SRTM90_V4
 - system:time_start           : 2000-02-11
 - system:time_end             : 2000-02-22
 - Number of Bands             : 1
 - Bands names                 : elevation
 - Number of Properties        : 25
 - Number of Pixels*           : 62208576001
 - Approximate size*           : 17.53 GB
Band Metadata (img_band = elevation):
 - EPSG (SRID)                 : WGS 84 (EPSG:4326)
 - proj4string                 : +proj=longlat +datum=WGS84
 +no_defs
 - Geotransform                : 0.000833333333333 0 -180 0
 -0.000833333333333 60
 - Nominal scale (meters)      : 92.76624
 - Dimensions                  : 432001 144001
 - Number of Pixels            : 62208576001
 - Data type                   : INT
 - Approximate size            : 17.53 GB
 ------------------------------------------------------------
 ------------------
 NOTE: (*) Properties calculated considering a constant
 geotransform and data type.
```

```r
# Clip the elevation data to Liberia
elev_liberia <- elev$clip(liberia_ee)

# Download clipped elevation data as a raster using Google Drive
elev_rast <- ee_as_rast(
```

```
  image = elev_liberia,
  region = liberia_ee$geometry(),
  via = "drive",
  scale = 1000, # Aggregate to 1000 meters
  quiet = T
)

# Replace 0s with NAs as these are usually elevation values
# where no data exists due to clipping or ocean masking.
elev_rast[elev_rast == 0] <- NA

# Plot the result
terra::plot(elev_rast, main = "Elevation in Liberia (m)")
plot(liberia_admin0$geometry, add = T)
```

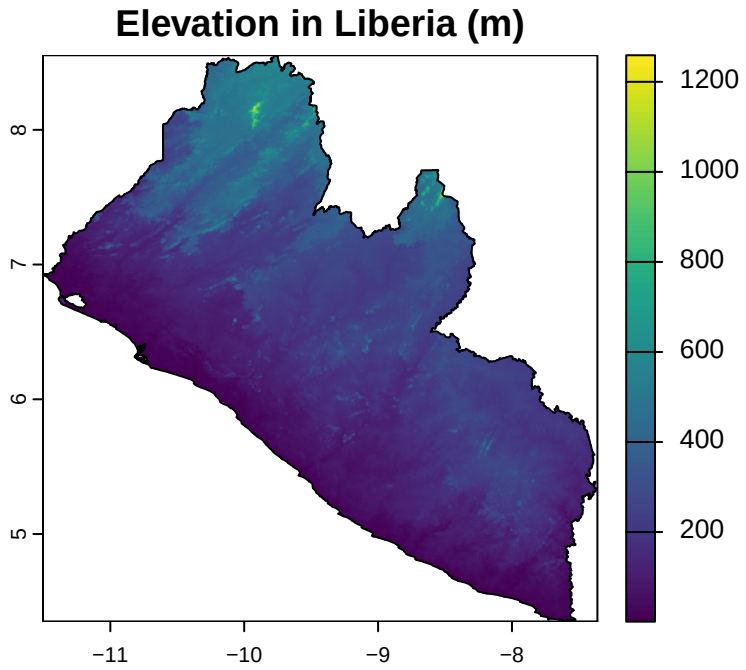



Figure 2.2: Elevation (m) in Liberia retrieved via Google Earth Engine (SRTM).

## 2.4 Importing and standardizing spatial data

In geostatistical analysis, importing and standardizing spatial data is a critical step to ensure that data from different sources align and can be used effectively. Spatial data, whether it's vector data (points, lines, polygons) or raster data (grids), can come in various formats and may use different **Coordinate Reference Systems (CRS)**. To perform accurate spatial analyses, it's essential to import data correctly and ensure consistency in terms of projection and format. This section will cover how to import vector and raster data into R, explain the concept of CRS and sensure that different datasets align properly for subsequent geostatistical analysis.

### 2.4.1 Importing vector data

Vector data are typically stored in formats such as **shapefiles** (`.shp`) or **geopackages** (`.gpkg`). The `sf` (simple features) package in R is the most common tool for handling vector data.

```
# Load the sf package
library(sf)

# Import a shapefile (e.g., administrative boundaries)
admin <- st_read("path_to_your_shapefile/Admin_Boundaries.shp")

# Inspect the data
print(admin)
plot(admin)  # Basic plot of the shapefile
```

The `st_read()` function reads various spatial data formats, automatically recognizing file types.

### 2.4.2 Importing raster data

Raster data consists of a grid of cells, where each cell holds a value representing a spatial attribute such as elevation or temperature. The `terra` package in R is designed to work with raster data and has superseded the older `raster` package due to better performance and greater functionality. Let's see now how to import a GeoTIFF file using `terra`. We can upload the population raster for Liberia that we have downloaded in Section 2.3.2.

```
# Load the terra package
library(terra)

# Import a raster file
```

```
lbr_pop_100 <- rast(lbr_pop_url)

# Inspect the raster data
print(lbr_pop_100)
```

```
class       : SpatRaster
size        : 5040, 4945, 1  (nrow, ncol, nlyr)
resolution  : 0.0008333333, 0.0008333333  (x, y)
extent      : -11.48625, -7.365417, 4.352084, 8.552084  (xmin,
xmax, ymin, ymax)
coord. ref. : lon/lat WGS 84 (EPSG:4326)
source      : lbr_ppp_2014.tif
name        : lbr_ppp_2014
min value   :  0.006426936
max value   : 92.716874581
```

```
# Basic plot of the raster (log-scale)
plot(log(lbr_pop_100), main = "Population per 100m (log-scale)")
plot(liberia_admin0$geometry, add = T)
```

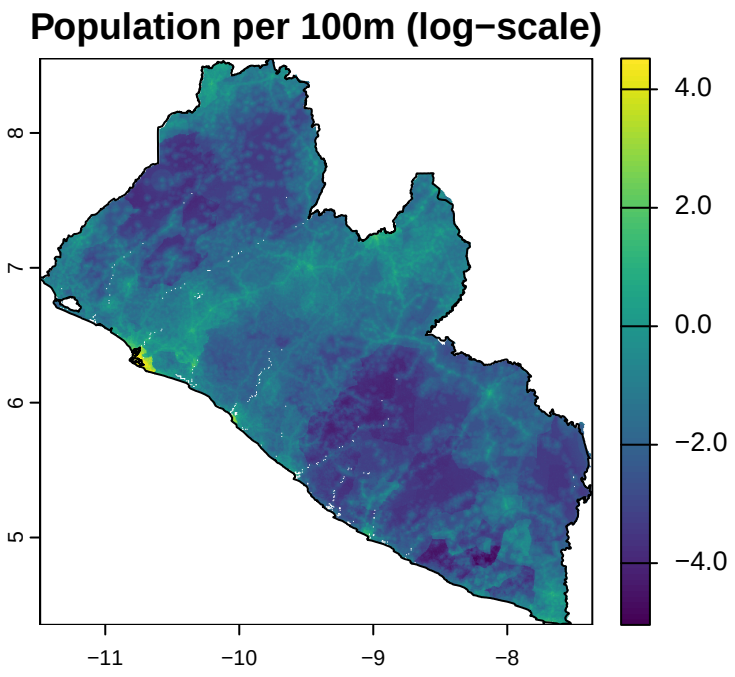



Figure 2.3: Liberia population count for 2014 at 100m resolution (log-scale).

The output of `print()` provides a detailed summary of the raster's properties.

The `rast()` function reads raster files in formats like GeoTIFF, ASCII grid, or other common raster formats and then we used `plot()` as a quick way to visualize raster data.

### 2.4.3 Understanding coordinate reference systems (CRS)

When working with spatial data, especially from different sources, one of the most critical tasks is ensuring that all datasets share the same **Coordinate Reference System (CRS)**. A CRS defines how the two-dimensional map data corresponds to locations on the three-dimensional Earth. If different layers (e.g., raster and vector data) have different CRSs, they may not align correctly when plotted or analyzed together, leading to inaccurate analyses or visualizations.

CRSs can either be:

- **Geographic**: These use latitude and longitude coordinates to represent locations on the Earth's surface. The most common example is **WGS84** (EPSG:4326), the default CRS used by GPS and global datasets.
- **Projected**: These convert the Earth's curved surface to a flat map and preserve certain properties like area, distance, or direction. Examples include **Universal Transverse Mercator (UTM)** or **Albers Equal Area** projections.

In many cases, using a projected rather than a geographic CRS is preferred, especially when any summary statistic or parameter is distance-related. In a geographic CRS, distances between two points are calculated using angular coordinates (degrees), which do not translate easily into linear units like meters or kilometers. This makes interpreting distances challenging, as the length of a degree of latitude differs from the length of a degree of longitude. In contrast, a projected CRS uses a linear coordinate system (usually meters), ensuring that distances are accurately represented on a flat surface. This is important when computing spatial variograms, covariance functions, or some parameters in geostatistical models, where distance between sampling locations is a key factor. In this context, we assume that the default distance metric is Euclidean distance. However, an important exception to our recommendation arises when analyzing data across large, global-scale regions. In such cases, it is more appropriate to use a geographic CRS along with spherical distances, as these better reflect the curved nature of the Earth's surface.

### 2.4.4 EPSG codes

An **EPSG code** is a unique identifier that defines a CRS. These codes, managed by the **European Petroleum Survey Group (EPSG)**, are widely used in geographic information systems (GIS) to simplify the use of specific projections, ensuring that spatial data is correctly aligned and interpreted.

Each EPSG code corresponds to a unique CRS or map projection, making it easier to standardize and manage spatial data from different sources.

Some key and often used EPSG codes are:

- **EPSG:4326**: This code represents **WGS84**, the most commonly used geographic CRS, which uses latitude and longitude to describe locations on the Earth's surface. It is the default CRS for global datasets and GPS systems.
- **EPSG:326XX**: These codes represent the **UTM (Universal Transverse Mercator)** projection, which divides the world into zones. Each zone is optimized to preserve local distances and areas. For example (e.g. EPSG:32629: UTM Zone 29N, covering parts of Western Africa, including Liberia)
- **EPSG:3857**: This code is for the **Web Mercator projection**, which is widely used for web mapping services, including **Google Maps**, **OpenStreetMap**, and **Bing Maps**. This projected CRS uses meters as the unit of distance and is optimized for visualizing maps on a 2D plane, though it distorts area and distance, especially at high latitudes. It is well-suited for interactive online mapping but not ideal for precise distance-based geostatistical analyses.

For a more in-depth discussion on how to choose an appropriate CRS, we recommend referring to the excellent treatment in *Geocomputation with R* (Lovelace, Nowosad, and Muenchow 2020), especially Section 7.6: Which CRS?.

### 2.4.5 Convert a data frame to an `sf` object

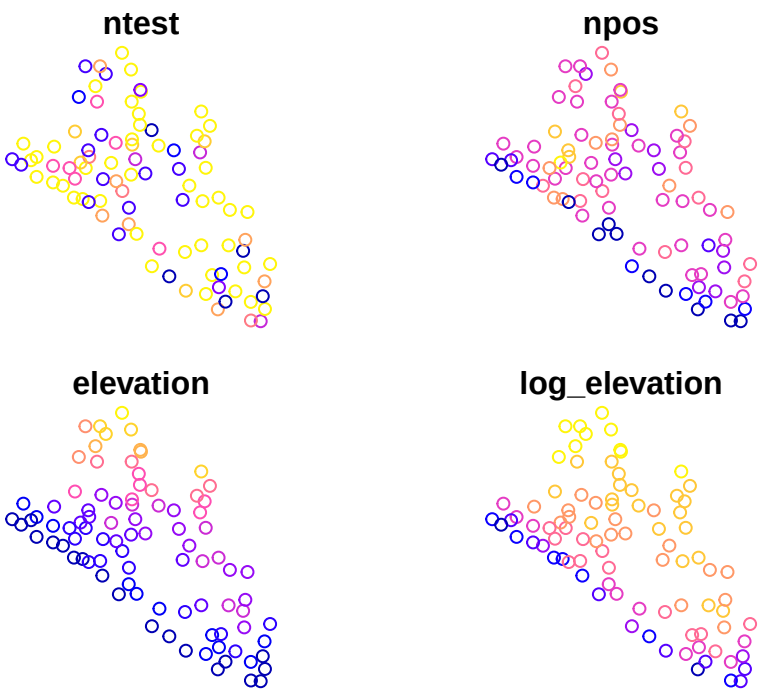
In geospatial analysis, data is often provided in tabular formats like CSV files that contain spatial coordinates (e.g., latitude and longitude). To use these data effectively in R, it is necessary to convert the data frame into an `sf` object, which is the standard format for working with spatial data in R. Here we show how to achieve this, we can use the Liberia data available in the `RiskMap` package as it is a data frame.

```
# Load the RiskMap package
library(RiskMap)

# Load the Liberia data set
data("liberia")

# Convert the data frame to an sf object
liberia_sf <- st_as_sf(liberia,
                       coords = c("long", "lat"),
                       crs = 4326)

# Inspect the new sf object
```

```
liberia_sf
```

```
Simple feature collection with 90 features and 4 fields
Geometry type: POINT
Dimension:     XY
Bounding box:  xmin: -11.40335 ymin: 4.42552 xmax: -7.4928
ymax: 8.46911
Geodetic CRS:  WGS 84
First 10 features:
   ntest npos elevation log_elevation                    geometry
1     50   14  17.82621      2.880670    POINT (-10.4671 6.2878)
2     46   10 104.85070      4.652537   POINT (-10.45111 6.5725)
3     43   11 119.09543      4.779925  POINT (-10.02615 6.56667)
4     50   10 144.10921      4.970571  POINT (-10.28775 6.73333)
5     48    9  19.03508      2.946283 POINT (-10.03595 6.016667)
6     50   13  16.99954      2.833186   POINT (-10.335 6.266666)
7     43    9 137.75360      4.925467   POINT (-9.63333 6.13541)
8     50   11 101.75000      4.622519   POINT (-10.06875 6.2375)
9     47   11 147.12867      4.991308    POINT (-9.73333 6.3869)
10    43    0  26.61054      3.281308     POINT (-9.7856 5.7354)
```

```
par(mar = c(0, 0, 0, 0))  #| hide_line
plot(liberia_sf)
```



The `st_as_sf()` function converts the data frame into an `sf` object. The `coords` argument specifies which columns contain the spatial coordinates and the `crs` argument assigns the CRS that in this case we know being WGS84 (EPSG:4326). The `sf` object can now be used for operations such as spatial joins, distance calculations, and mapping with other spatial layers. Note that the columns containing the spatial coordinates have been replaced by a `geometry` column, which now stores this information. If you would like to retain the original coordinate columns in the output, you can set the `remove` argument to `FALSE` when converting the data frame to an `sf` object.

### 2.4.6 Working with CRSs in R

When working with data from multiple sources, such as environmental layers, population data, or administrative boundaries, ensuring that all datasets share the same CRS is essential for accurate spatial analysis. This section covers the core tasks involved in managing CRSs in R: checking the CRS of spatial data to ensure datasets are compatible and reprojecting spatial data into a common CRS when necessary.

#### 2.4.6.1 Checking the CRS of Spatial Data

Before performing any spatial operation, it's crucial to check the CRS of your spatial datasets. Knowing whether your data uses geographic coordinates (e.g., WGS84) or a projected coordinate system (e.g., UTM) helps ensure that they are aligned and ready for analysis. Both `sf` and `terra` provide functions to retrieve and inspect the CRS, ensuring datasets are spatially aligned before analysis. The `st_crs()` function retrieves the CRS information for vector data.

```
# Check the CRS of the vector data
st_crs(liberia_sf)
```

```
Coordinate Reference System:
  User input: EPSG:4326
  wkt:
GEOGCRS["WGS 84",
    ENSEMBLE["World Geodetic System 1984 ensemble",
        MEMBER["World Geodetic System 1984 (Transit)"],
        MEMBER["World Geodetic System 1984 (G730)"],
        MEMBER["World Geodetic System 1984 (G873)"],
        MEMBER["World Geodetic System 1984 (G1150)"],
        MEMBER["World Geodetic System 1984 (G1674)"],
        MEMBER["World Geodetic System 1984 (G1762)"],
        MEMBER["World Geodetic System 1984 (G2139)"],
        ELLIPSOID["WGS 84",6378137,298.257223563,
            LENGTHUNIT["metre",1]],
```

```
        ENSEMBLEACCURACY[2.0]],
    PRIMEM["Greenwich",0,
        ANGLEUNIT["degree",0.0174532925199433]],
    CS[ellipsoidal,2],
        AXIS["geodetic latitude (Lat)",north,
            ORDER[1],
            ANGLEUNIT["degree",0.0174532925199433]],
        AXIS["geodetic longitude (Lon)",east,
            ORDER[2],
            ANGLEUNIT["degree",0.0174532925199433]],
    USAGE[
        SCOPE["Horizontal component of 3D system."],
        AREA["World."],
        BBOX[-90,-180,90,180]],
    ID["EPSG",4326]]
```

The same can be achieved for raster data with the `crs()` function from the `terra` package.

```
# Check the CRS of the raster data
crs(lbr_pop_100, proj = TRUE, describe = TRUE)
```

```
    name authority code  area             extent
1 WGS 84      EPSG 4326 World -180, 180, -90, 90
                                  proj
1 +proj=longlat +datum=WGS84 +no_defs
```

#### 2.4.6.2 Reprojecting spatial data to a common CRS

If your datasets have different CRSs or if you want to change CRS (e.g. from geographical to projected) you will need to reproject one or more datasets so they can be spatially aligned. This ensures that they can be overlaid and analyzed together. For vector data, `st_transform()` reprojects the data into a specified CRS. This example transforms the liberia point data from WGS84 into UTM. To know what's the correct UTM zone and hence EPSG code for Liberia we can use the `propose_utm` function from the `RiskMap` package.

```
# Obtain EPSG code for UTM for Liberia
propose_utm(liberia_sf)
```

```
[1] 32629
```

```
# Reproject the vector data
liberia_sf_utm <- st_transform(liberia_sf, crs =
 ↪  propose_utm(liberia_sf))
```

Reprojecting raster data is more complex than reprojecting vector data due to

the continuous nature of raster grids. The process involves recalculating cell values to fit a new grid based on the new CRS, which can lead to challenges like resampling, distortion, and data loss. When reprojecting a raster, the grid must adjust to the new CRS, often requiring resampling of cell values. The method you choose depends on the data type: **nearest neighbor** is best for categorical data like land use while **bilinear or cubic interpolation** is good for continuous data like temperature, where smooth transitions are needed.

The function `project()` from the `terra` package can be used to project a raster.

```
lbr_pop_100_utm <- project(lbr_pop_100,
                           crs(liberia_sf_utm),
                           method = "bilinear")
```

Reprojecting a raster may alter its resolution. For example, reprojecting from geographic (degrees) to projected (meters) CRS can result in a mismatch between the original and new cell sizes. Moreover, distortion can occur when converting between projections, especially at high latitudes. Some cells may be stretched or compressed, leading to potential loss of information or edge artifacts. These distortions arise because the Earth is not flat, and projecting the curved surface of the Earth onto a flat plane (or vice versa) leads to trade-offs. For example, the Mercator projection preserves angles and shapes but distorts area, particularly near the poles.

For these reasons, it's often better to reproject vectors rather than rasters when both data types are used together and avoid as much as possible to change the CRS of rasters. One way to achieve this is to work with WGS84 when performing all spatial operations like extraction of covariates from rasters and then transform only the point data to a projected CRS before fitting the model.

## 2.5 Extracting covariate data

In geostatistical models, the inclusion of relevant covariates (environmental, demographic, or climatic) can potentially enhance predictive accuracy. Covariate data often come from raster or polygon sources, and extracting these values for point locations is essential to link spatial context to point-referenced data. These covariates could include variables like temperature, elevation, land cover, or population density, which influence the spatial distribution of diseases. In this section, we will cover how to extract covariates at point locations from both polygon and raster layers.

#### 2.5.0.1 Extracting covariates from polygon layers

Polygon layers represent discrete spatial features such as administrative boundaries, land cover categories, or protected areas. These layers typically include associated attributes (e.g. region names, population totals, land type), which can serve as covariates in geostatistical models. A common task in spatial data analysis is to assign attributes from a polygon layer to a set of point-referenced data, for example, determining the administrative region or land use category that each survey location falls within. This is done using a **spatial join**, which overlays the point and polygon layers and attaches the attributes of the enclosing polygon to each point. Below is an example where we assign administrative region names (admin1) to a set of point locations in Liberia:

```
# Perform a spatial join to transfer polygon attributes to the
 ↪  points
points_with_admin <- st_join(liberia_sf,
 ↪  liberia_admin1["shapeName"])

# View the results, points now include covariates from the
 ↪  polygon layer
head(points_with_admin)
```

```
Simple feature collection with 6 features and 5 fields
Geometry type: POINT
Dimension:     XY
Bounding box:  xmin: -10.4671 ymin: 6.016667 xmax: -10.02615
ymax: 6.73333
Geodetic CRS:  WGS 84
  ntest npos elevation log_elevation   shapeName
     geometry
1    50   14  17.82621      2.880670     Margibi    POINT
(-10.4671 6.2878)
2    46   10 104.85070      4.652537 Montserrado   POINT
(-10.45111 6.5725)
3    43   11 119.09543      4.779925     Margibi  POINT
(-10.02615 6.56667)
4    50   10 144.10921      4.970571     Margibi  POINT
(-10.28775 6.73333)
5    48    9  19.03508      2.946283 Grand Bassa POINT
(-10.03595 6.016667)
6    50   13  16.99954      2.833186 Grand Bassa   POINT
(-10.335 6.266666)
```

This spatial join enriches each point with the corresponding polygon attribute, in this case, the admin1 name (shapeName). These attributes can then be

used as categorical covariates in geostatistical models to account for regional variation in disease risk or programmatic coverage.

### 2.5.1 Extracting covariates from raster layers

Raster data provides continuous spatial information, such as elevation, climate data, or population density. Covariate values from raster layers can be extracted for specific points using the **`extract()`** function from the `terra` package. Each point will receive the value of the raster cell it overlaps.

```
# Extract raster values at the point locations
covariate_values <- extract(lbr_pop_100, liberia_sf)

# Combine the extracted values with the point data
liberia_sf$pop_total <- covariate_values[, 2]

# View the updated dataset
head(liberia_sf)
```

```
Simple feature collection with 6 features and 5 fields
Geometry type: POINT
Dimension:     XY
Bounding box:  xmin: -10.4671 ymin: 6.016667 xmax: -10.02615
ymax: 6.73333
Geodetic CRS:  WGS 84
  ntest npos elevation log_elevation                   geometry
  pop_total
1    50   14  17.82621      2.880670    POINT (-10.4671 6.2878)
0.2914019
2    46   10 104.85070      4.652537   POINT (-10.45111 6.5725)
0.6087928
3    43   11 119.09543      4.779925  POINT (-10.02615 6.56667)
0.2044306
4    50   10 144.10921      4.970571  POINT (-10.28775 6.73333)
0.5078438
5    48    9  19.03508      2.946283 POINT (-10.03595 6.016667)
0.1767686
6    50   13  16.99954      2.833186   POINT (-10.335 6.266666)
1.6693381
```

In this example, the `extract()` function assigns the raster value from the population density layer to each point in the dataset. This allows the point data to include population density as a covariate in the analysis.

Instead of extracting values for exact point locations, it can sometimes be useful to aggregate covariate values within a defined area around each point.

This is often done by creating a buffer around each point and calculating summary statistics (e.g., mean, sum) of the raster values within that buffer. For instance, you might want to calculate the average population density or temperature within a 5 km radius around each point to smooth out fine-scale variation.

```
# Create buffers around each point (e.g., 5 km radius)
buffered_points <- st_buffer(liberia_sf_utm, dist = 5000)

# Plot the buffers for visualization
par(mar = c(0, 0, 0, 0))  #| hide_line
plot(st_geometry(buffered_points), border = "black")
plot(st_geometry(liberia_sf_utm), add = T, pch = 19, col =
 ↪  "red", cex = .2)
```

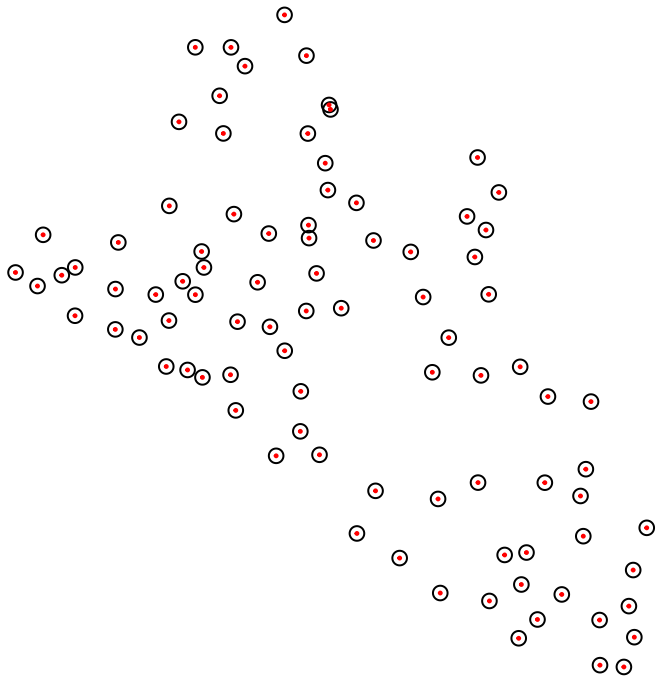


Figure 2.4: Survey locations (red dots) surrounded by a 5 km buffer (dark circles).

```
# Extract raster values within the buffer areas and calculate
 ↪  the
# mean or sum. Note that since we used the utm data to work on
 ↪  the
```

```
# meter scale we need to convert them back to WGS84
mean_pop_density <- extract(lbr_pop_100,
                            st_transform(buffered_points, crs =
                             ↪  4326),
                            fun = mean, na.rm = TRUE)

# Add the averaged values to the points dataset
liberia_sf$pop_mean5km <- mean_pop_density[,2]

# View the updated dataset
head(liberia_sf)
```

```
Simple feature collection with 6 features and 6 fields
Geometry type: POINT
Dimension:     XY
Bounding box:  xmin: -10.4671 ymin: 6.016667 xmax: -10.02615
ymax: 6.73333
Geodetic CRS:  WGS 84
  ntest npos elevation log_elevation                   geometry
  pop_total
1    50   14  17.82621      2.880670    POINT (-10.4671 6.2878)
0.2914019
2    46   10 104.85070      4.652537   POINT (-10.45111 6.5725)
0.6087928
3    43   11 119.09543      4.779925  POINT (-10.02615 6.56667)
0.2044306
4    50   10 144.10921      4.970571  POINT (-10.28775 6.73333)
0.5078438
5    48    9  19.03508      2.946283 POINT (-10.03595 6.016667)
0.1767686
6    50   13  16.99954      2.833186   POINT (-10.335 6.266666)
1.6693381
  pop_mean5km
1   0.6015397
2   0.5565529
3   0.2488039
4   0.5604327
5   0.2877019
6   2.0974484
```

You can modify the `fun` argument to calculate other summary statistics, such as the sum, min or max of the raster values within the buffer. This approach is particularly useful when the phenomenon being modeled (e.g., disease transmission) is influenced by broader spatial factors around the observation point, rather than just the value at the exact point location.

## 2.6 Creating a predictive grid

A predictive grid is a regularly spaced set of points or cells that spans the study region. This grid serves as the basis for predictions made by your model. The density of the grid (i.e., the distance between grid points) affects both the resolution of the prediction and the computational cost. For point-based predictions, we can generate a grid of points over a polygon (e.g., administrative boundary) using the `sf` package and the `st_make_grid` function.

```
# First we convert the Liberia boundaries to the UTM CRS
# because we want our grid in meters
liberia_admin0_utm <- liberia_admin0 |>
  st_transform(crs = propose_utm(liberia_sf))

# Generate prediction grid at 5km resolution
pred_locations <- st_make_grid(liberia_admin0_utm,
                               cellsize = 5000,
                               what = "centers")

# Exclude locations that fall outside the study area
pred_locations <- st_intersection(pred_locations,
 ↪  liberia_admin0_utm)

# Visualize the result
par(mar = c(0, 0, 0, 0))  #| hide_line
plot(liberia_admin0_utm$geometry, col = "white")
plot(pred_locations, cex = .01, col = "red", pch = 19, add = T)
```

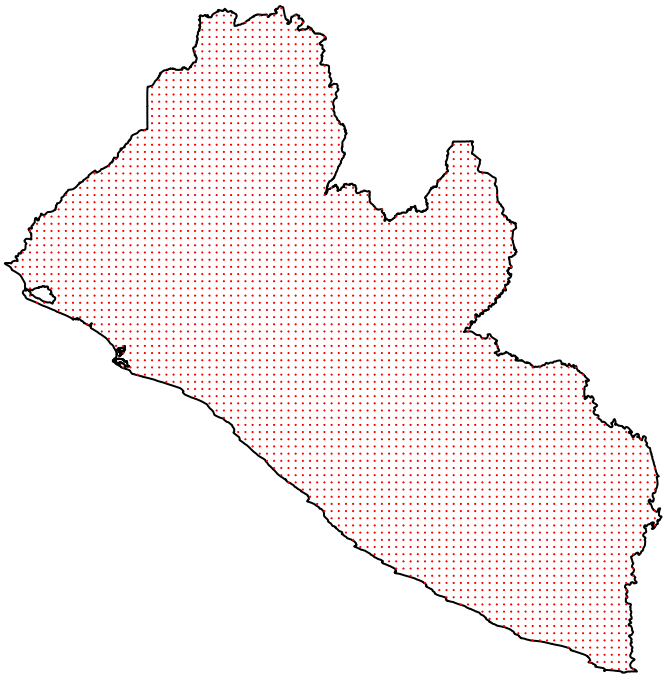


Figure 2.5: Prediction grid (5 km spacing) created within the boundaries of Liberia. Points are spaced regularly in UTM coordinates and filtered to remain within the national boundary.

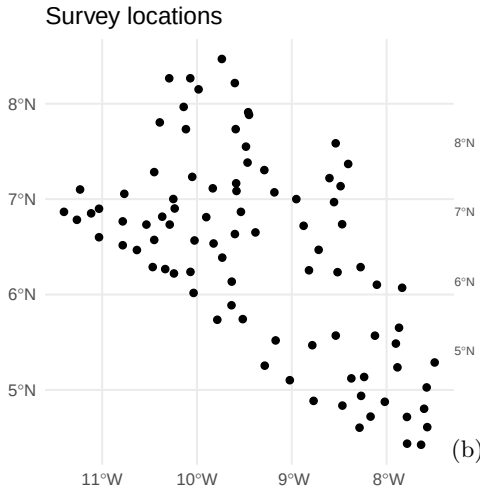
## 2.7 Visualizing spatial data

Visualization is a key part of spatial data analysis, as it allows you to explore and communicate spatial patterns and relationships effectively. R provides a lot of functionalities to visualize spatial data and create very beautiful maps. Until now we have used basic plotting functions. Here we introduce the `ggplot2` package that allows to combine different types of geographic data in a map. The `ggplot2` package in R provides a flexible and powerful framework for creating both simple and complex visualizations, including maps of point data, polygons, and rasters. With the help of extensions like `geom_sf()` and `geom_raster()`, `ggplot2` makes it easy to visualize spatial data, whether you're working with point locations, polygons, or continuous raster data.

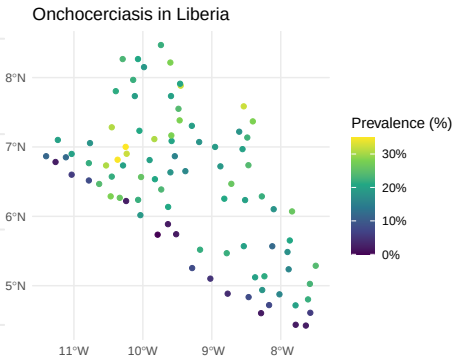
### 2.7.1 Visualizing point data

Point data often represents the locations of observations (e.g., disease cases, sampling sites). `ggplot2` allows you to plot these points and optionally color them by a covariate (e.g., disease prevalence or population density). Here is an example that uses the Liberia data.

```
# Load necessary libraries
library(ggplot2)

# Create a new variable with prevalence in the dataset
liberia_sf$prevalence <- liberia$npos / liberia$ntest

# Plot only the locations
ggplot(data = liberia_sf) +
  geom_sf(col = "black") +
  theme_minimal() +
  labs(title = "Survey locations")

# Color the points according to prevalence
ggplot(data = liberia_sf) +
  geom_sf(aes(color = prevalence)) +
  scale_color_viridis_c(labels = scales::label_percent()) +
  theme_minimal() +
  labs(title = "Onchocerciasis in Liberia",
       color = "Prevalence (%)")
```




(a) Survey locations.




(b) Survey locations colored by prevalence.

Figure 2.6: Using `geom_sf()` to plot point data.

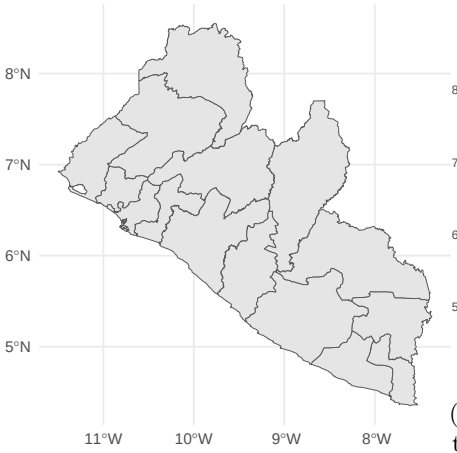
Here `geom_sf()` is used to plot the spatial points. `aes(color = prevalence)` specifies that the points should be colored based on the `prevalence` covariate, providing a visual representation of spatial variation in disease risk. The `scale_color_viridis_c()` function applies the Viridis color scale, which is well-suited for continuous data and is friendly for those with color blindness. The `labels = scales::label_percent()` argument ensures that the color scale's labels are displayed as percentages (e.g., 5, 10%) rather than raw decimal values. To make the plot visually clean and minimal, `theme_minimal()` is applied, stripping away unnecessary background elements and keeping the focus on the data. Finally, the `labs()` defines the plot title and the color legend label.

### 2.7.2 Visualizing polygon data

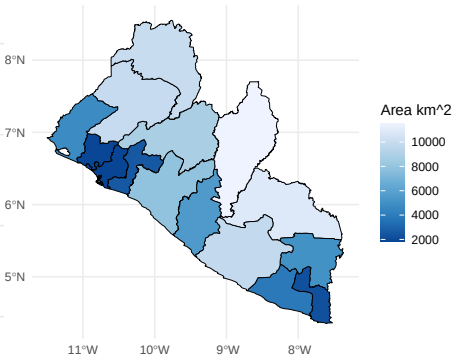
Polygon data typically represents administrative boundaries, land use, or other regional divisions. We can still use `geom_sf()` to create maps of polygons, optionally filling them by a covariate.

```
# Plot Liberia admin 1 level boundaries
ggplot(data = liberia_admin1) +
  geom_sf() +
  theme_minimal() +
  labs()

# We compute the area of each polygon
liberia_admin1$area <- as.numeric(st_area(liberia_admin1) /
 ↪  1000 ^ 2)

# Color the polygons according tis new variable
ggplot(data = liberia_admin1) +
  geom_sf(aes(fill = area), color = "black") +
  scale_fill_distiller(direction = -1) +
  theme_minimal() +
  labs(fill = "Area km^2")
```

In this code, `aes(fill = area)` is used to fill each polygon with colors corresponding to its area. The `color = "black"` argument outlines the polygons in black, and you could set `fill = NA` to make the polygons transparent while still displaying the borders. The `scale_fill_distiller(direction = -1)` function applies a color gradient from ColorBrewer, with the `direction = -1` argument reversing the gradient (e.g., darker colors for larger areas).




(a) Admin 1 units for Liberia.




(b) Admin 1 units colored according to their area.

Figure 2.7: Using `geom_sf()` to plot polygon data.

### 2.7.3 Visualizing raster data

In `ggplot2`, you can visualize raster data by converting it into a data frame of coordinates and values. You can convert raster data into a format that `ggplot2` can handle by using the `as.data.frame()` function from `terra`.

```
# Convert the raster to a data frame for ggplot2
raster_df <- as.data.frame(log(lbr_pop_100), xy = TRUE)

# Plot raster using ggplot2
ggplot(data = raster_df) +
  geom_raster(aes(x = x, y = y, fill = lbr_ppp_2014)) +
  scale_fill_viridis_c() +
  coord_sf(crs = 4326) +
  theme_minimal() +
  labs(title = "Population per 100m (log-scale)",
       fill = "Density", x = "", y = "")
```

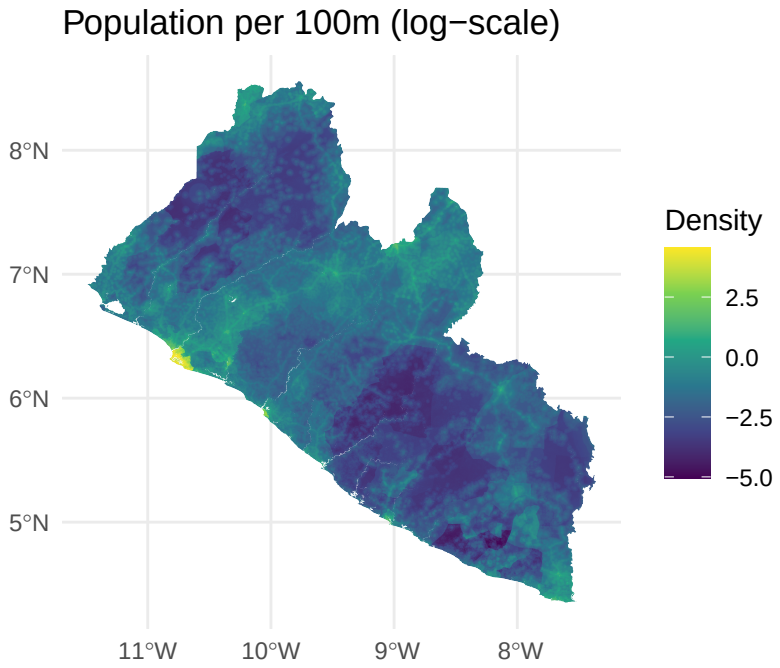



Figure 2.8: Log-population in Liberia (2014) visualized as a raster using `geom_raster()` with Viridis color scale.

In this example `as.data.frame()` converts the raster into a data frame with x and y coordinates and their corresponding raster values and `geom_raster()` is used to plot the raster cells, coloring them based on the population density.

### 2.7.4 Combining Multiple Spatial Data Types

In many cases, it's useful to combine different spatial data types (points, polygons, and rasters) in a single visualization. `ggplot2` allows you to overlay these layers, providing a more comprehensive view of your spatial data.

```
# Combine points, polygons, and raster data in one plot
combined_map <- ggplot() +
  geom_raster(data = raster_df, aes(x = x, y = y, fill =
   ↪  lbr_ppp_2014)) +
  geom_sf(data = liberia_admin1, fill = NA, color = "black") +
  geom_sf(data = liberia_sf, shape = 21, col = "black", fill =
   ↪  "white") +
  scale_fill_viridis_c() +
  theme_minimal() +
  labs(title = "Combined Spatial Data: Points, Polygons, and
   ↪  Raster",
```

```
       fill = "Population Density",
       x = "", y = "")

combined_map
```

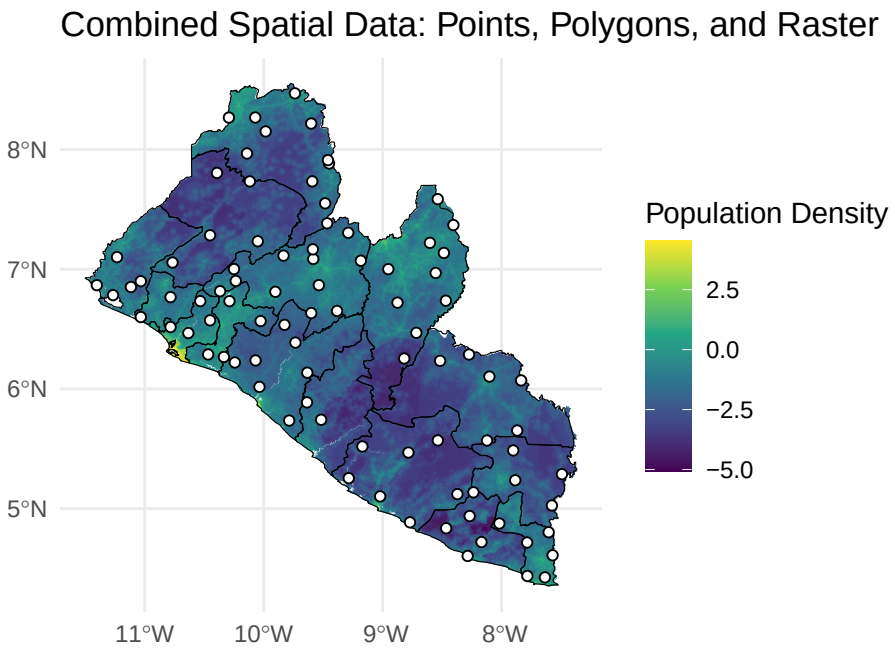



Figure 2.9: Overlay of raster (population density), polygons (admin-1 boundaries), and points (survey locations) in a single `ggplot2` map.

To enhance spatial visualizations in `ggplot2`, adding a scale bar and a north arrow improves map readability and professionalism. The `ggspatial` package offers tools to easily integrate these elements into your maps. Below is an example that demonstrates how to use `ggspatial` to add a scale bar and north arrow to a map that includes raster, polygon, and point data.

```
# Load necessary libraries
library(ggspatial)

# Add scale bar and north arrow
combined_map +
  annotation_scale(location = "bl", width_hint = 0.25) +
  annotation_north_arrow(location = "tr", which_north = "true",
                         height = unit(.5, "cm"), width =
                          ↪  unit(.5, "cm"))
```

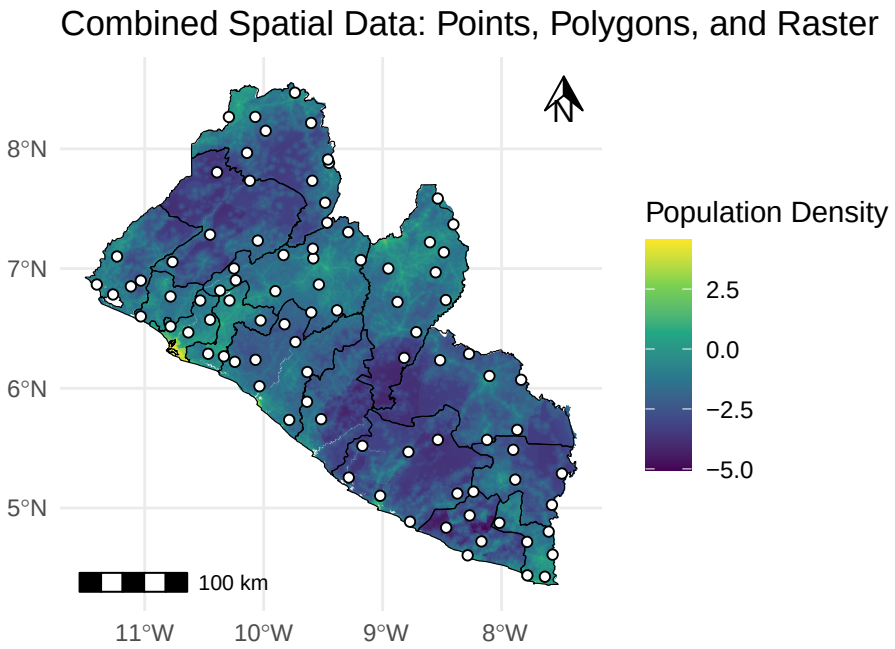



Figure 2.10: Enhanced map with scale bar and north arrow added using the `ggspatial` package.

## 2.8 Review Questions

- What are the main differences between raster and vector data, and in which scenarios is each type commonly used in geostatistical analysis?
- Why is it important to ensure that spatial datasets share the same coordinate reference system (CRS)? What problems can arise if CRSs are mismatched?
- What are the advantages of using a projected CRS (e.g., UTM) over a geographic CRS (e.g., WGS84) in geostatistical modeling?
- Describe the purpose of creating a predictive grid. How does the resolution of this grid affect the model?
- What is the difference between extracting raster values at a point location versus using a buffer around a point?

## 2.9 Exercises

1. **Import and visualize shapefiles**: Download a shapefile of administrative boundaries for a country of your choice using the `rgeoboundaries` package. Visualize the boundaries using both base R and `ggplot2`.
2. **Download and plot raster data**: Use either the `wpgpDownloadR` or `rgee` package to obtain raster data (e.g., elevation or population) for the same country. Plot it using `terra::plot()` and `ggplot2`.
3. **Coordinate transformation**: Convert the CRS of both vector and raster datasets to UTM. Check the CRS before and after transformation and plot the results to confirm proper alignment.
4. **Covariate extraction**: Sample 50 random points within your country boundary. Extract elevation or population values at each point and add them to the point data.
5. **Buffered extraction**: Create a 2 km buffer around each point. Extract and summarize (e.g., mean) raster values within each buffer, then compare with the point-level values.
6. **Create a predictive grid**: Build a 5 km prediction grid over your country and plot it along with the raster background. Prepare the grid for input into a geostatistical model by extracting covariates at each grid point.

# 3

# *Model formulation and parameter estimation*

**List of the main functions used in the chapter**

| Function | R Package | Used for |
|---|---|---|
| `lmer` | `lme4` | Fitting linear mixed models |
| `glmer` | `lme4` | Fitting generalized linear mixed models |
| `glgm` | `RiskMap` | Fitting generalized linear geostatistical models |
| `s_variogram` | `RiskMap` | Computing the empirical variogram and carrying out permutation test for spatial independence |

## 3.1 Exploratory analysis

As illustrated in Figure 1.8, exploratory analysis is the first step that should be carried out in a statistical analysis. This stage is essential to inform how covariates should be introduced in the model and, in our case, whether the variation unexplained by those covariates exhibits spatial correlation.

In the exploratory analysis of count data, we will also look at how overdispersion, which is a necessary, though not sufficient, condition for residual spatial correlation.

### 3.1.1 Exploring associations with risk factors using count data

Assessment of the association between the health outcome of interest and non-categorical (i.e. continuous) risk factors can be carried using graphical tools, such scatter plots. The graphical inspection of the empirical association between the outcome and the covariates is especially useful to identify non-linear patterns in the relationship which should then be accounted for in the model formulation.

In this section, we look more closely at the case when the observed outcome is a count which requires a different treatment from continuously measured outcomes, which are generally covered by most statistics textbooks (see, for example, Chapter 1 of Weisberg (2014)).

#### 3.1.1.1 When the outcome is an aggregated count

Let us first consider the example of the river-blindness data in Liberia (Section 1.4.2), and examine the association between prevalence and elevation. We first generate a plot of the prevalence against the measured elevation at each of the sample locations

```
liberia$prev <- liberia$npos/liberia$ntest

ggplot(liberia, aes(x = elevation, y = prev)) + geom_point() +
  labs(x="Elevation (meters)",y="Prevalence")
```

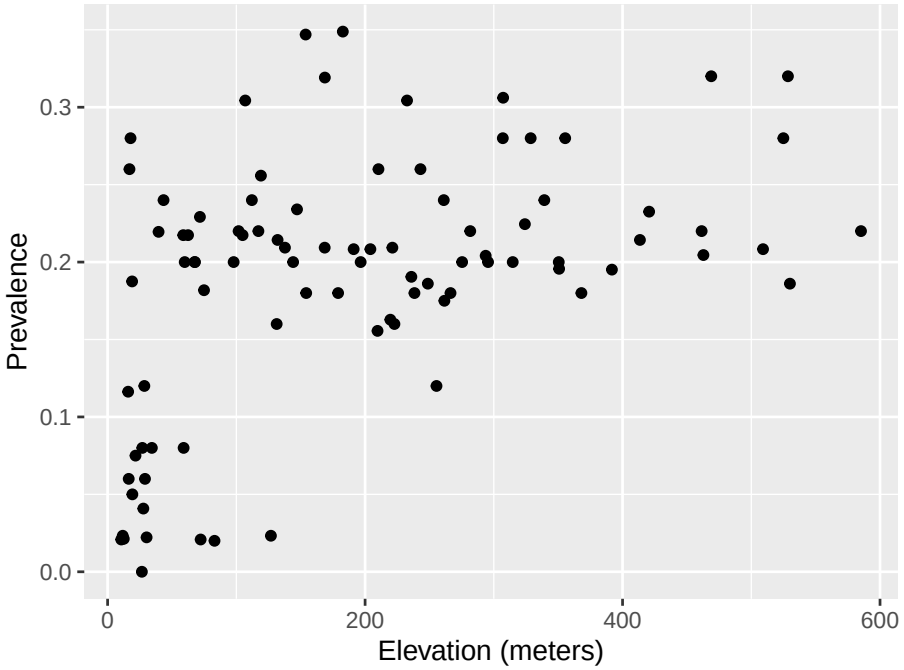



Figure 3.1: Scatter plot of the empirical prevalence for river-blindess against elevation, measured in meters.

The plot shown in Figure 3.1 shows that, as elevation increases from 0 to around 150 meters, prevalence rapidly increases to around 0.25 and, for larger values in elevation than 150 meters, the relationship levels off. This begs the question of how we can account for this in a regression model. To answer this question rigorously, however, the plot in Figure 3.1 cannot be used. This is because, when the modelled outcome is a bounded Binomial count, regression relationships are specified on the logit-transformed prevalence (log-odds) scale; see Table 1.3 in Section Section 1.5 . To explore regression relationships in the case of prevalence data, it is convenient to use the so-called empirical logit in place of the empirical prevalence. The empirical logit is defined as

$$l_i = \log\left\{\frac{y_i + 1/2}{n_i - y_i + 1/2}\right\} \tag{3.1}$$

where $y_i$ are the number of individuals who tested positive for river-blindness and $n_i$ is the total number of people tested at a location. The reason for using the empirical logit, rather than the standard logit transformation applied directly to the empirical prevalence, is that it allows to generate finite values for empirical prevalence values of 0 and 1, for which the standard logit transformation is not defined.

```
# The empirical logit
liberia$elogit <- log((liberia$npos+0.5)/
                      (liberia$ntest-liberia$npos+0.5))

ggplot(liberia, aes(x = elevation, y = elogit)) + geom_point() +

  # Adding a smoothing spline
  labs(x="Elevation (meters)",y="Empirical logit") +
  stat_smooth(method = "gam", formula = y ~ s(x),se=FALSE)+

  # Adding linear regression fit with log-transformed elevation
  stat_smooth(method = "lm", formula = y ~ log(x),
              col="green",lty="dashed",se=FALSE) +

  # Adding linear regression fit with change point in 150 meters
  stat_smooth(method = "lm", formula = y ~ x + pmax(x-150, 0),
              col="red",lty="dashed",se=FALSE)
```

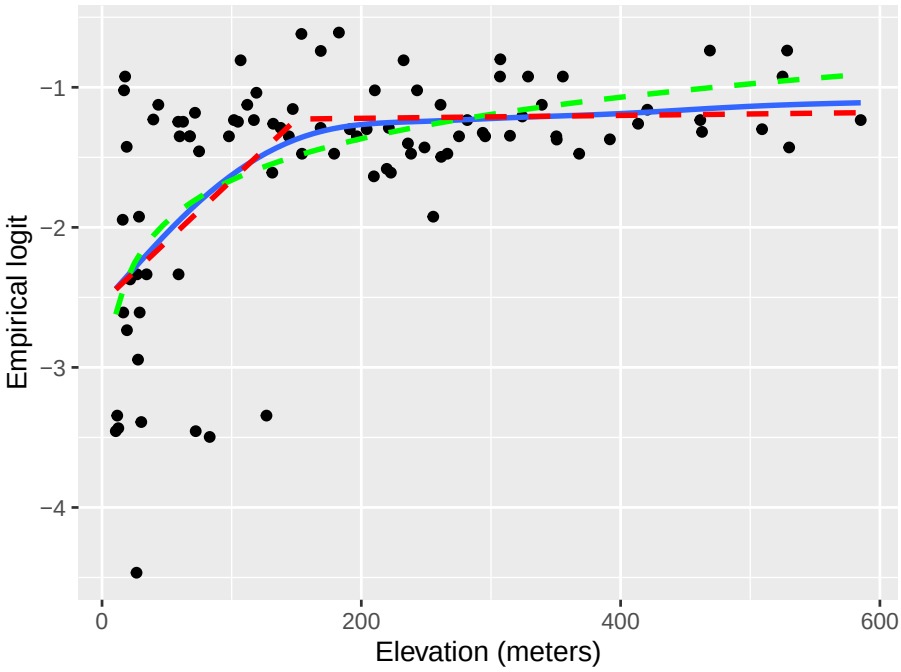



Figure 3.2: Scatter plot of the empirical prevalence for river-blindess against elevation, measured in meters.

Figure 3.2 shows the scatter plot of the empirical logit against elevation. In this plot, we have also added three lines though the `stat_smooth` from the `ggplot2` package. Using this function, we first pass the term `gam` to `method` to add a penalized smoothing spline (Hastie, Tibshirani, and Friedman 2001), represented by the blue solid line. The smoothing spline allows us to better discern how the type of relationship and how to best capture it using a standard regression approach. As er can see from Figure 3.2, the smoothing spline corroborates our initial observation of a positive relationship up to about 150 meters, followed by a plateau.

To capture this non-linear relationship, we can use the two following approaches. The first is based on a simple log-transformation of elevation and is represented in Figure 3.2 by the green line. If were to express this relationship using a standard Binomial regression model, this would take the form

$$\log\left\{\frac{p(x_i)}{1-p(x_i)}\right\} = \beta_0 + \beta_1 \log\{e(x_i)\} \tag{3.2}$$

where $p(x_i)$ and $e(x_i)$ are the river-blindness prevalence and elevation at sampled location $x_i$, respectively.

Alternatively, the non-linear effect of elevation on prevalence could be captured using a linear spline. Put in simple terms, we want to fit a linear regression model that allows for a change in slope above 150 meters. Formally, this is expressed in a Binomial regression model as

$$\log\left\{\frac{p(x_i)}{1-p(x_i)}\right\} = \beta_0 + \beta_1 e(x_i) + \beta_2 \max\{e(x_i) - 150, 0\}. \tag{3.3}$$

Based on the equation above, the effect of elevation below 150 meters is quantified by the parameter $\beta_1$. Above 150 meters, instead, the effect of elevation becomes $\beta_1 + \beta_2$. Note that the function `pmax` (and not the standard base function `max`) should be used in R when the computation of the maximum between a scalar value and each of the components of a numeric vector is required.

Before proceeding further, it is important to explain the differences between the use of the logarithmic transformation (Equation 3.2) and the linear spline (Equation 3.3). We observe that both curves provide a similar fit to the data, with larger differences observed for larger values in elevation, where the log-transformed elevation models yields larger values for the predicted prevalence. This also suggests that if we were to extrapolate the predictions beyond 600 meters in elevation the implied pattern by the model with the log-transformed elevation would predict an increasingly larger elevation, which is unrealistic, since the fly that transmits the diseases cannot breed at those altitudes. The linear spline model instead would generate predictions that would be very similar to those observed between 150 and 600 meters. From this point view, the linear spline model would thus have more scientific validity than the other

model. However, which of the two approaches should be chosen to model the effect of elevation is a question that closely depends on the research question to be addressed.

If the interest of the study was in better understanding the association between elevation and prevalence, the linear spline model does not only provide a more credible explanation but also its regression parameters can be more easily interpreted. In fact, for a unit increase in elevation, the multiplicative change in the odds for river-blindness is $\exp\{\beta_1\}$, if elevation is below 150 meters, and $\exp\{\beta_1 + \beta_2\}$, if elevation is above 150 meters. When instead we use the log-transformed elevation, the interpretation of $\beta_1$ in Equation 3.2 is slightly more complicated, as it is based on the multiplicative increase in elevation by the same amount given by the base of the algorithm, which is about $e \approx 2.718$[1]. To avoid this, one could rescale the regression coefficient as, for example, $\beta_1 / \log_2(e)$ which would be interpreted as the multiplicative change in the odds for river-blindness for a doubling in elevation. However, a doubling in elevation is less meaningful when considering larger values of elevation.

When the goal of statistical analysis is instead in developing a predictive model for the outcome of interest, the explanatory power and interpretability of the model may be of less concern. For this reason, the model with the log-transformed model could be preferred over the model with the linear spline, if it shown to yield more predictive power. We will come back to this point again in Chapter 4, where will show how to assess and compare the predictive performance of different geostatistical models.

The other type of aggregated count data that we consider are unbounded counts. The Anopheles mosquitoes data-set (Section 1.4.4) is an example of this, since there is no upper limit to the number of mosquitoes that can be trapped at a location. Let us consider the covariate represented by elevation. In this case, the simplest model that can be used to analyse the data is a Poisson regression, where the linear predictor is defined on the log of the mean number of mosquitoes (Table 1.3). Hence, exploratory plots for the association with covariates should be generated using the log transformed counts of mosquitoes. In this instance, to avoid taking the log of zero, we can add 1 to the reported counts, if required. The variable of the `An.gambiae` in the `anopheles` data-set does not contain any 0, hence we simply apply the log tranformation without adding 1.

```
anopheles$log_counts <- log(anopheles$An.gambiae)
ggplot(anopheles, aes(x = elevation, y = log_counts)) +
 ↪  geom_point() +
```

[1]The letter $e$ stands for the so called Euler's number and represents the base of the natural logarithm. In the book, we write $\log(\cdot)$ to mean the "natural logarithm of $\cdot$".

```
# Adding a smoothing spline
labs(x="Elevation (meters)",y="Log number of An. gambiae
 ↪  mosquitoes") +
stat_smooth(method = "lm", formula = y ~ x, se=FALSE)
```

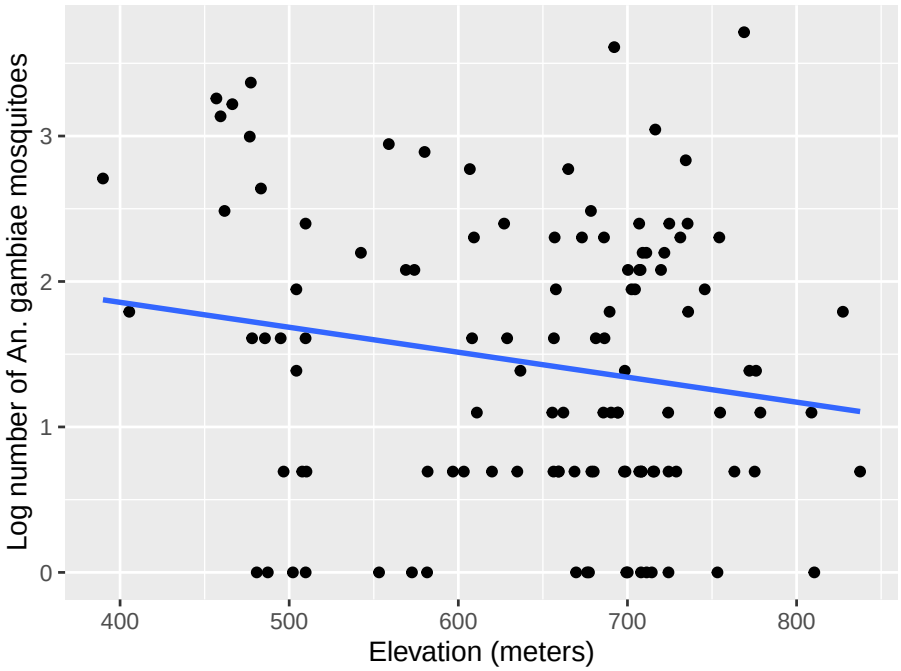



Figure 3.3: Scatter plot of the log tranformed number of *Anopheles gambiae* mosquitoes against elevation, measured in meters. The blue line is generated using the least squares fit.

The scatter plot of Figure 3.3 shows that there is a negative, though weak, association, with the average number of mosquitoes decreasing for increasing elevation. In this instance, the assumption of a linear relationship with elevation would be a reasonable choice.

#### 3.1.1.2 When the outcome is an invidual-level binary indicator

We now consider the malaria data from Kenya (Section 1.4.3) where the main outcome is the result from a rapid diagnostic test (RDT) for malaria from individuals within households. In this case, because the outcome only takes two values, 1 for a positive RDT test result and 0 otherwise, the direct application of the empirical logit from Equation 3.1 would not help us to generate informative scatter plots. Throughout the book, we will consider the data from the

community survey only, hence we work with a subset of the data which we shall name `malkenya_comm`

```
malkenya_comm <- malkenya[malkenya$Survey=="community", ]
```

To show how this issue can be overcome, let us consider the variables age and gender. To generate a plot that can help us understand between the relationship with malaria prevalence and the two risk factors, we proceed as follows.

```
# Grouping of ages into classes defined through "breaks"
malkenya_comm$Age_class <- cut(malkenya_comm$Age,
                               breaks = c(0, 5, 10, 15, 30, 40, 50,
                                ↪  100),
                               include.lowest = TRUE)
```

Using the `cut` function, we first split age (in years) into classes through the argument `breaks`. The classification of age into $[0, 5]$, $(5, 10]$ and $(10, 15]$ is common in many malaria epidemiology studies, as children are one of the groups at highest risk malaria. The choice of the other classes of age reflects instead the need to balance the number of observations falling in each of the classes.

```
# Computation of the empirical logit by age groups and gender
age_class_data <- aggregate(RDT ~ Age_class + Gender,
                                    data = malkenya_comm,
                                    FUN = function(y)
                                    log((sum(y)+0.5)/(length(y)
                                     ↪  -sum(y)+0.5)))
```

We then compute the empirical logit, using the total number of cases within age group and by gender. For a given age group and gender, which we denote as $\mathcal{C}$, the empirical logit in Equation 3.1, now takes the form

$$l_{\mathcal{C}} = \log\left\{\frac{\sum_{i\in\mathcal{C}} y_i + 0.5}{|\mathcal{C}| - \sum_{i\in\mathcal{C}} y_i + 0.5}\right\} \tag{3.4}$$

where $y_i$ are the individual binary outcomes and $i \in \mathcal{C}$ is used to indicate that the sum is carried out over all the individuals who belong the class $\mathcal{C}$, identified by a specific age group and gender. Finally, $|\mathcal{C}|$ is the number of individuals who fall within $\mathcal{C}$. In the code above, the empirical logit in Equation 3.4 is computed using the `aggregate` function. An inspection of the object `age_class_data`, a data frame, shows that the empirical is found in the column named `RDT`.

```
# Computation of the average age within each age group
age_class_data$age_mean_point <- aggregate(Age ~ Age_class +
 ↪  Gender,
```

```
                                  data = malkenya_comm,
                                  FUN = mean)$Age


# Number of individuals within each age group, by gender
age_class_data$n_obs <-  aggregate(Age ~ Age_class + Gender,
                          data = malkenya_comm,
                          FUN = length)$Age
```

In order to generate the scatter-plot, we compute the average age within each age group by gender, and use these as our values for the x-axis. Note that since we only need to obtain the average age from this output, we use `$Age` to extract this only and allocate to the column `age_mean_point`. Finally, we also compute the number of observations within each of classes and place this in `n_obs`.

```
ggplot(age_class_data, aes(x = age_mean_point, y = RDT,
                           size = n_obs,
                           colour = Gender)) +
  geom_point() +
  labs(x="Age (years)",y="Empirical logit")
```

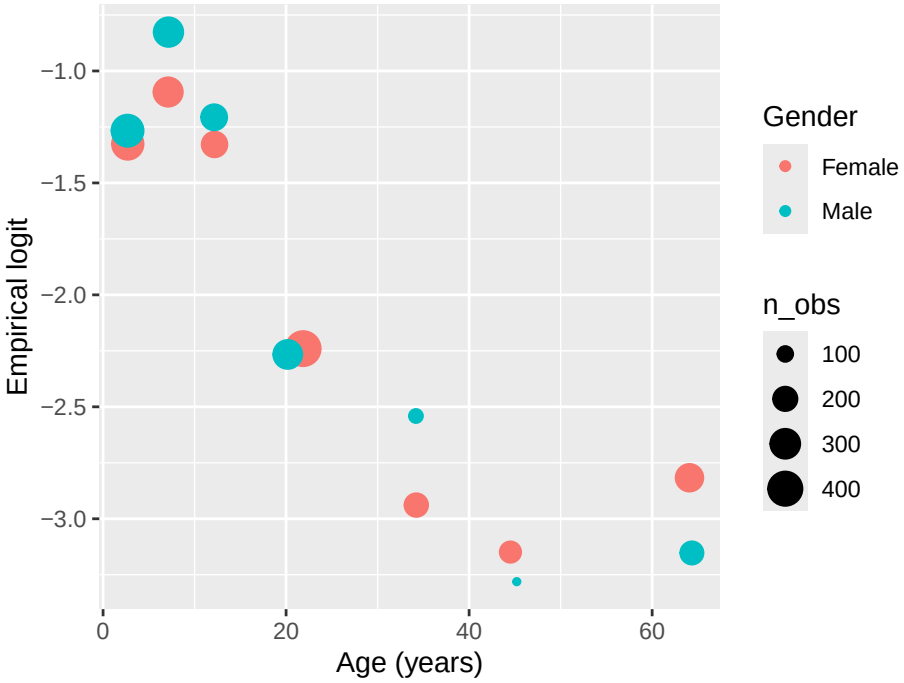



Figure 3.4: Plot of the empirical logit against age, for males and females. The size of each solid point is rendered proportional to the number of individuals within age group, as indicated in the legend.

The resulting plot in Figure 3.4 shows the empirical logit against age by gender, with the size of each of the points proportional to the number of observations falling within each class. The observed patterns are explained by the fact that young children, especially those under the age of five, are particularly vulnerable to severe malaria infections. This is primarily due to their immature immune systems and lack of acquired immunity. As individuals grow older, they generally develop partial immunity to malaria through repeated exposure to the disease. This acquired immunity can provide some level of protection against severe malaria. At the same time, gender roles and activities can influence exposure to malaria-carrying mosquitoes. For example, men may spend more time outdoors for work or other activities, increasing their exposure to mosquito bites and thus their risk of infection. In addition, there are also biological factors to consider. Hormonal and genetic differences between males and females may also contribute to variations in immune responses to malaria infection. The interaction between age and gender is complex and may vary depending on the specific context and population being studied. A 2020 report from the Bill & Melinda Gates foundation provides a detailed overview of this and other aspects related to gender and malaria (Katz and Bill & Melinda

Gates Foundation 2020).

To account for age in a model for malaria prevalence, several approaches are possible, some of which have been developed using biological models (Smith et al. 2007). To model the patterns observed in Figure 3.4, we can follow the same approach used in the previous section to model the relationship between elevation and river-blindness prevalence. First, let us consider age without the effect of gender. Let $p_j(x_i)$ denote the probability of a positive RDT for the $j$-th individual living in a household at location $x_i$. Assuming that malaria risk reaches its peak at 15 years of age, we can capture the non-linear relationship using a linear spline with two knots, one at 15 years and a second one at 40 years. This is expressed as

$$\log\left\{\frac{p_j(x_i)}{1-p_j(x_i)}\right\} = \beta_0 + \beta_1 a_{ij} + \beta_2 \times \max\{a_{ij} - 15, 0\} + \beta_3 \max\{a_{ij} - 40, 0\} \tag{3.5}$$

where $a_{ij}$ is the age, in years, for the $j$-th individual at household $i$. Based on this model the effect of age on RDT prevalence is $\beta_1$, for $a_{ij} < 15$, $\beta_1 + \beta_2$, for $15 < a_{ij} < 40$, and $\beta_1 + \beta_2 + \beta_3$ for $a_{ij} > 40$.

Figure 3.4 indicates that age may interact with gender, meaning that the effect of gender on RDT prevalence changes across age, with larger differences observed between males and females for ages above 20 years. To assess such differences using a standard Binomial regression model, the linear predictor for RDT prevalence can be formulated as

$$\log\left\{\frac{p_j(x_i)}{1-p_j(x_i)}\right\} = \beta_0 + (\beta_1 + \beta_1^* g_{ij}) \times a_{ij} + (\beta_2 + \beta_2^* g_{ij}) \times \max\{a_{ij} - 15, 0\} + (\beta_3 + \beta_3^* g_{ij}) \times \max\{a_{ij} - 40, 0\} \tag{3.6}$$

where $g_{ij}$ is the indicator for gender, with 1 corresponding to male and 0 to female. The coefficients $\beta_1^*$, $\beta_2^*$ and $\beta_3^*$ thus quantify the differences in risk between the two genders for ages below 15 years, betwee 15 and 40 years, and above 40 years, respectively. If all of those coefficients were 0, the model in Equation 3.5 would be recovered.

```
glm_age_gender_interaction <- glm(RDT ~ Age + Gender:Age +
                                    pmax(Age-15, 0) +
                                     ↪  Gender:pmax(Age-15, 0) +
                                    pmax(Age-40, 0) +
                                     ↪  Gender:pmax(Age-40, 0),
                                  data = malkenya_comm, family =
                                   ↪  binomial)

summary(glm_age_gender_interaction)
##
```

```
## Call:
## glm(formula = RDT ~ Age + Gender:Age + pmax(Age - 15, 0) +
 ↪  Gender:pmax(Age -
##     15, 0) + pmax(Age - 40, 0) + Gender:pmax(Age - 40, 0),
 ↪  family = binomial,
##     data = malkenya_comm)
##
## Coefficients:
##                              Estimate Std. Error z value
 ↪  Pr(>|z|)
## (Intercept)                 -1.05835    0.10245 -10.331  <
 ↪  2e-16 ***
## Age                         -0.03384    0.01310  -2.584
 ↪  0.00978 **
## pmax(Age - 15, 0)           -0.03975    0.02356  -1.687
 ↪  0.09162 .
## pmax(Age - 40, 0)            0.09170    0.02482   3.695
 ↪  0.00022 ***
## Age:GenderMale               0.01428    0.01221   1.170
 ↪  0.24202
## GenderMale:pmax(Age - 15, 0) -0.03625    0.03145  -1.153
 ↪  0.24908
## GenderMale:pmax(Age - 40, 0)  0.02451    0.04320   0.567
 ↪  0.57052
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' '
 ↪  1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 2875.8  on 3351  degrees of freedom
## Residual deviance: 2673.8  on 3345  degrees of freedom
## AIC: 2687.8
##
## Number of Fisher Scoring iterations: 5
```

The code above shows how to fit the model specified in Equation 3.6. The terms `Age`, `pmax(Age-15, 0)` and `pmax(Age-40, 0)` respectively correspond to $\beta_1$, $\beta_2$ and $\beta_3$, whilst the `Gender:Age`, `Gender:pmax(Age-15, 0)` and `Gender:pmax(Age-40, 0)` to $\beta_1^*$, $\beta_2^*$ and $\beta_3^*$, respectively. In the summary of the fitted model, we observe that the interaction coefficients are non-statistically significant. However, removing the interaction based on the fact that each of the coefficients have each p-values larger than the conventional level of 5% would be wrong. Instead we should carry out the likelihood ratio

test, as shown below.

```
glm_age_gender_no_interaction <- glm(RDT ~ Age +  pmax(Age-15,
 ↪  0) + pmax(Age-40, 0),
                                data = malkenya_comm, family =
                                 ↪  binomial)

anova(glm_age_gender_no_interaction, glm_age_gender_interaction,
 ↪  test = "Chisq")
## Analysis of Deviance Table
##
## Model 1: RDT ~ Age + pmax(Age - 15, 0) + pmax(Age - 40, 0)
## Model 2: RDT ~ Age + Gender:Age + pmax(Age - 15, 0) +
 ↪  Gender:pmax(Age -
##     15, 0) + pmax(Age - 40, 0) + Gender:pmax(Age - 40, 0)
##   Resid. Df Resid. Dev Df Deviance Pr(>Chi)
## 1      3348     2675.6
## 2      3345     2673.8  3   1.8051   0.6138
```

To carry out the likelihood ratio test to assess the null hypothesis that $\beta_1^* = \beta_2^* = \beta_3^* = 0$, we first fit the simplified nested model under this null hypothesis. The likelihood ratio test can then be carried out using the `anova` command as shown. The p-value indicates that the we do not find evidence against the null hypothesis, hence in our analysis of the data we might favour the simplified model that does not assumes an interaction between the two genders.

The approach just illustrated, can also be applied to explore the association with other continuous variables that are a property of the household and not of the individual. Let us, for example, consider the variable `elevation` from the `malkenya` data-set.

```
malkenya_comm$elevation_class <- cut(malkenya_comm$elevation,
                              breaks = quantile(malkenya_comm$
                               ↪  elevation, seq(0, 1, by =
                               ↪  0.1)),
                              include.lowest = TRUE)
```

Following the same approach used for age, we first split elevation into classes. To define these, we use the deciles of the empirical distribution of `elevation` which we calculate using the `quantile` function above. In this way we also ensure that the number of observations falling within each class of elevation is approximately the same.

```
# Computation of the empirical logit by classes of elevation
elev_class_data <- aggregate(RDT ~ elevation_class,
                                    data = malkenya_comm,
                                    FUN = function(y)
```

```
                                    log((sum(y)+0.5)/(length(y)
                                     ↪  -sum(y)+0.5)))


# Computation of the average elevation within each class of
 ↪  elevation
elev_class_data$elevation_mean <- aggregate(elevation ~
 ↪  elevation_class,
                                    data = malkenya_comm,
                                    FUN = mean)$elevation
```

We then compute the empirical logit and the average elevation for each class of elevation. The empirical logit is computed as already defined in Equation 3.4, where now the definition of $\mathcal{C}$ is given by a specific decile used to split the distribution of elevation.

```
ggplot(elev_class_data, aes(x = elevation_mean, y = RDT),
                            size = n_obs) +
  geom_point() +
  labs(x="Elevation (meters)",y="Empirical logit")
```

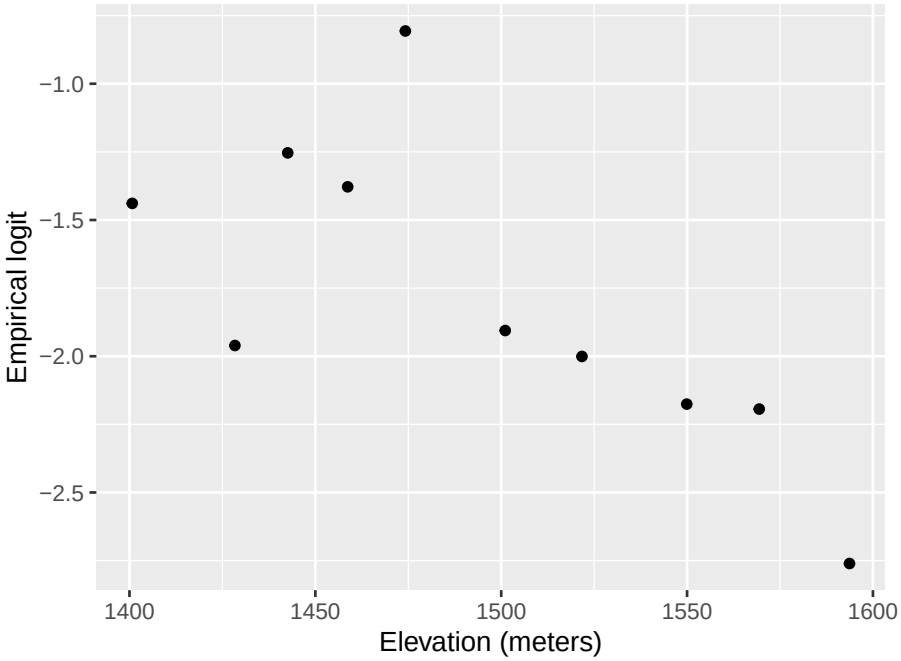



Figure 3.5: Plot of the empirical logit against elevation measured in meters.

The resulting plot in Figure 3.5 shows an approximately linear relationship with decreasing values of the empirical logit for increasing elevation. This is expected because the cooler environment at higher altitudes is less favourable to the development of the overall mosquito life cycle.

An alternative approach to generate a scatter plot for assessing the association between elevation and the empirical logit would be to aggregate the data at household level, rather than using classes of elevation. However, this approach does not work as the one illustrated above when only one individual is sampled for each location. In the case of the `malkenya` data, the great majority of the locations only include one individual making this second approach less useful than the one illustrated.

Other more sophisticated approaches for the exploration of the associations between covariates and binary outcomes are available. For example, the use of the empirical logit could be avoided by using non-parametric regression methods for Binomial outcomes (Bowman 1997), also implemented in `sm` package in R. Our view is that a careful exploratory analysis based on simpler methods, as those illustrated above, can be equally effective to inform the module formulation.

### 3.1.2 Exploring overdispersion in count data

One of the main advantages in the use of covariates is the ability to attribute part of the variation of the outcome to a set of measured variables and, hence, reduce the uncertainty of our inferences. However, it almost always the case that the finite number of covariates at our disposal is not enough to fully explain the variation in the outcome. In other words, the existence of unmeasured covariates that are related to the modelled outcome give rise to the so called residual variation. In a standard linear regression model the extent to which we are able to account for important covariates is directly linked to the size of the variance of the residuals. In the case of count data, instead, this link is less well defined and one of the main consequences of the omission of covariates, which we address in this chapter, is *overdispersion*.

Overdispersion occurs when the variability of the data is larger than that implied by the generalized linear model (GLM) fitted to them. For example, if we consider the Binomial distribution, the presence of overdispersion implies that $V(Y_i) > n_i\mu_i(1-\mu_i)$, where we recall that $n_i$ is the Binomial denominator and $mu_i$ is the probability of "success" for each of the $n_i$ Bernoulli trials; for a Poisson distribution with $E(Y_i) = \mu_i$, instead, overdispersion implies that $V(Y_i) > \mu_i$.

Assessment of the overdispersion for count data can be carried out in different ways depending on the goal of the statistical analysis. Since the focus of this book is to illustrate how to formulate and apply geostatistical models, the most natural approach to assess overdispersion is through the use of general-

ized linear mixed models (GLMMs). The class of GLMMs that we consider in this and the next section are obtained by replacing the spatial Gaussian process $S(x_i)$ in introduced in Equation 1.4 with a set of mutually independent random effects, which we denote as $Z_i$, and thus write

$$g(\mu_i) = d(x_i)^\top \beta + Z_i. \tag{3.7}$$

The model above accounts for the overdispersion in the data through $Z_i$ whose variance can be interpreted as an indicator of the amount of overdispersion. To show this, we carry out a small simulation as follows. For simplicity, we consider the Binomial mixed model with an intercept only, hence

$$\log\left\{\frac{\mu_i}{1-\mu_i}\right\} = \beta_0 + Z_i \tag{3.8}$$

and assume that the $Z_i$ follow a set of mutually independent Gaussian variables with mean 0 and variance $\tau^2$. In our simulation we vary $\beta_0$ over the set $\{-3, -2, -1, 0, 1, 2, 3\}$ and set $\tau^2 = 0.1$ and the binomial denominators to $n_i = 100$. For a given value of $\beta_0$, we then proceed through the following iterative steps.

- Simulate 10,000 values for $Z_i$ from a Gaussian distribution with mean 0 and variance $\tau^2$.
- Compute the probabilities $\mu_i$ based on Equation 3.8.
- Simulate 10,000 values from a Binomial model with probability of success $\mu_i$ and denominator $n_i$.
- Compute the empirical variance of the counts $y_i$ simulated in the previous step.
- Change the value of $\beta_0$ and repeat the previous steps, for all the values of $\beta_0$.

The code below shows the implementation of the above steps in R.

```
# Number of simulations
n_sim <- 10000

# Variance of the Z_i
tau2 <- 0.1

# Binomial denominator
bin_denom <- 100

# Intercept values
beta0 <- c(-3, -2, -1, 0, 1, 2, 3)
```

```r
# Vector where we store the computed variance from
# the simulated counts from the Binomial mixed model
var_data <- rep(NA, length(beta0))


for(j in 1:length(beta0)) {
  # Simulation of the random effects Z_i
  Z_i_sim <- rnorm(n_sim, sd = sqrt(tau2))

  # Linear predictor of the Binomial mixed model
  lp <- beta0[j]  + Z_i_sim

  # Probabilities of the Binomial distribution conditional on
   ↪  Z_i
  prob_sim <- exp(lp)/(1+exp(lp))

  # Simulation of the counts from the Binomial mixed model
  y_i_sim <- rbinom(n_sim, size = bin_denom, prob = prob_sim)

  # Empirical variance from the simulated counts
  var_data[j] <- var(y_i_sim)
}

# Probabilities from the standard Binomial model (Z_i = 0)
probs_binomial <- exp(beta0)/(1+exp(beta0))

# Variance from the standard Binomial model
var_bimomial <- bin_denom*probs_binomial*(1-probs_binomial)
```

```r
matplot(beta0, cbind(var_data, var_bimomial), type = "b", pch =
 ↪  20,
        lty = "solid", ylab = "Variance", xlab =
         ↪  expression(beta[0]))
legend(-3, 80, c("Binomial mixed model", "Standard Binomial
 ↪  model"),
       col=1:2, lty = "solid", cex = 0.75)
```

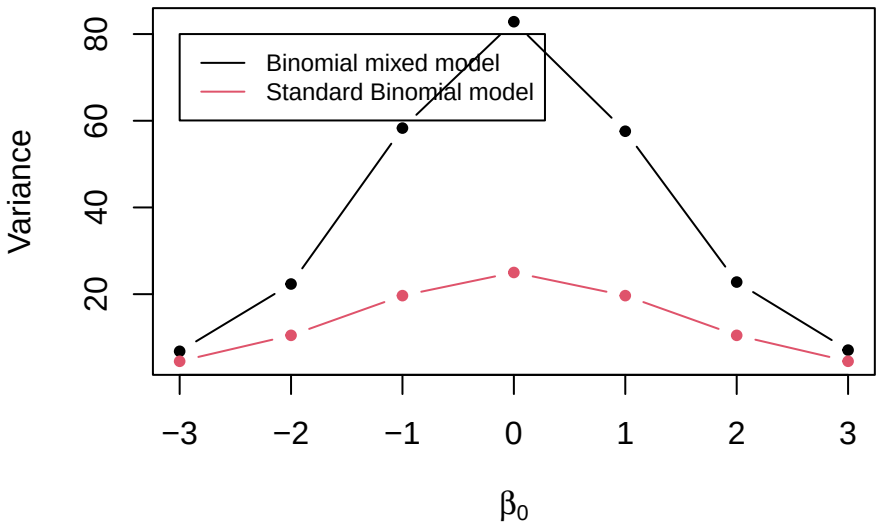



Figure 3.6: Plot of the variances of the standard Binomial model and the Binomial mixed model (see Equation 3.8) against $\beta_0$

Figure 3.6 shows the results of the simulation. In this figure, the red line corresponds to the variance of a standard Binomial model, obtained by setting $Z_i = 0$ and computed as $n_i\mu_i(1-\mu_i)$ with $\mu_i = \exp\{\beta_0\}/(1+\exp\{\beta_0\})$. As expected, this plot shows that the variance of the simulated counts from the mixed model in Equation 3.8 exhibit a larger variance than would be expected under the standard Binomial model. It also indicates that the chosen value for the variance of $Z_i$ of $\tau^2 = 0.1$ corresponds to a significant amount of dispersion. One way to relate $\tau^2$ to the amount of overdispersion is by considering that, following from the properties of a univariate Gaussian distribution, *a priori* the $Z_i$ will take values between $-1.96\sqrt{\tau^2}$ and $+1.96\sqrt{\tau^2}$ with approximately 95% probability. That implies that $\exp\{Z_i\}$, which expresses the effect of the random effects on the odds ratios, will be with 95% probability between $\exp\{-1.96\sqrt{\tau^2}\}$ and $\exp\{+1.96\sqrt{\tau^2}\}$. By replacing $\tau^2$ with the chosen values for the simulation, those two becomes about 0.54 and 1.86, meaning that with the $Z_i$ with 95% probability will have a multiplicative effect on the odds ratios between 0.54 and 1.86.

We encourage you to do Exercise 1 and Exercise 2 at the end of this chapter, to further explore how generalized linear mixed models can be used as a tool to account for overdispersion.

#### 3.1.2.1 Maximum likelihood estimation of generalized linear mixed models for count data

We now illustrate how to fit a generalize linear mixed, using the `anopheles` data-set as an example. We consider two models: an intercept-only model and one that uses elevation as a covariate. Let $\mu(x_i)$ be the number of mosquitoes captured at a location $x_i$; then the linear predictor with elevation as a covariate takes the form

$$\log\{\mu_i\} = \beta_0 + \beta_1 d(x_i) + Z_i \tag{3.9}$$

where $d(x_i)$ indicates the elevation in meters at location $x_i$ and the $Z_i$ are independent and identically distributed Gaussian variables with mean 0 and variance $\tau^2$. The model with an intercept only is simply obtained by setting $\beta_1 = 0$.

We carry out the estimation in R using the `glmer` function from the `lme4` package (see Bates et al. (2015) for a detailed tutorial). The `glmer` function implements the maximum likelihood estimation for generalized linear mixed models. The code below shows how the `glmer` is used to carry out this step for the model in Equation 3.9 and the one withuot covariates.

```
# Create the ID of the location
anopheles$ID_loc <- 1:nrow(anopheles)

# Poisson mixed model with elevation
fit_glmer_elev <- glmer(An.gambiae ~ scale(elevation) +
 ↪  (1|ID_loc), family = poisson,
                         data = anopheles, nAGQ = 25)

# Poisson mixed model with intercept only
fit_glmer_int <- glmer(An.gambiae ~ (1|ID_loc), family =
 ↪  poisson,
                         data = anopheles, nAGQ = 25)
```

To fit the model with `glmer`, we first must create a variable in our data-set that allows us to identify the location associated with each count. In this case, since every row corresponds to a different location, we simply use the row number to identify the locations and save this in the `ID_loc` variable. The random effects $Z_i$ are then included in the model by adding `(1 | ID_loc)` in the formula of the `glmer` function.

When introducing the variable elevation, we standardize the variable so that its mean is 0 and its variance is 1. This is done to aid the convergence of the algorithm used to fit the model and it is generally considered good practice, especially when many variables with different scales are used as covariates. However, we emphasize that standardizing a variable does not affect the fit of the model to the data. This is because the model with the standardized variable is a reparametrization of the model with the unstandardized variable.

In other words, a model that uses standardized covariates only attaches a different interpretation to its regression coefficients while maintaining the same goodness of fit of the model with that uses the covariates on their original scale.

The argument `nAGQ` is used to define the precision of the approximation of the maximum likelihood estimation algorithm. By default `nAGQ = 1`, which corresponds to the Laplace approximation. Values for `nAGQ` larger than 1 are used to define the number of points of the adaptive Gaussian-Hermite quadrature. The general principle is that the larger `nAGQ` the better, but at the expense of an increased computing time. Based on the guidelines and help pages of the `lme4` package, it is stated that a reasonable value for `nAGQ` is 25. For more technical details on this aspect, we refer the reader to Bates et al. (2015).

We can now look at the summary of the fitted models to the mosquitoes data-set.

```
### Summary of the model with elevation
summary(fit_glmer_elev)
## Generalized linear mixed model fit by maximum likelihood
 ↪  (Adaptive
##   Gauss-Hermite Quadrature, nAGQ = 25) [glmerMod]
##  Family: poisson  ( log )
## Formula: An.gambiae ~ scale(elevation) + (1 | ID_loc)
##    Data: anopheles
##
##       AIC       BIC    logLik -2*log(L)  df.resid
##     291.8     300.1    -142.9     285.8       113
##
## Scaled residuals:
##      Min       1Q   Median       3Q      Max
## -0.89574 -0.42469 -0.09483  0.29445  0.53352
##
## Random effects:
##  Groups Name        Variance Std.Dev.
##  ID_loc (Intercept) 0.7146   0.8453
## Number of obs: 116, groups:  ID_loc, 116
##
## Fixed effects:
##                  Estimate Std. Error z value Pr(>|z|)
## (Intercept)       1.53042    0.09365  16.342   <2e-16 ***
## scale(elevation) -0.19794    0.08950  -2.212    0.027 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' '
 ↪  1
##
```

```
## Correlation of Fixed Effects:
##             (Intr)
## scale(lvtn) 0.036

### Summary of the model with the intercept only
summary(fit_glmer_int)
## Generalized linear mixed model fit by maximum likelihood
 ↪  (Adaptive
##   Gauss-Hermite Quadrature, nAGQ = 25) [glmerMod]
##  Family: poisson  ( log )
## Formula: An.gambiae ~ (1 | ID_loc)
##    Data: anopheles
##
##       AIC       BIC    logLik -2*log(L)  df.resid
##     294.6     300.1    -145.3     290.6       114
##
## Scaled residuals:
##      Min       1Q   Median       3Q      Max
## -0.73816 -0.42718 -0.06941  0.26564  0.45022
##
## Random effects:
##  Groups Name        Variance Std.Dev.
##  ID_loc (Intercept) 0.761    0.8724
## Number of obs: 116, groups:  ID_loc, 116
##
## Fixed effects:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)  1.52849    0.09584   15.95   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' '
 ↪  1
```

From the summary of the model that uses elevation, we observe the that the estimated regression coefficient $\beta_1$ is statistically significant different from 0. The interpretation of the estimated regression coefficient is the following: for an increase of about 100 meters in elevation, all other things being equal, the average number of mosquitoes decreases by about $100\% \times [1 - \exp\{-0.19794\}] \approx 18\%$. Note that when using a standardized variable, a unit increase for this corresponds to an increase in the original unstandardized variable equal to its standard deviation, which for the `elevation` variable is about 100 meters.

From the summaries of the two models, under `Random effects:`, we obtain the estimates associates with the random effects introduced in the model. In this case, since we only have introduced $Z_i$, this part of summary provides the maximum likelihood estimate for $\tau^2$, the variance of $Z_i$, which found on the

line where `ID_loc` is printed. We then observe that the estimates for $\tau^2$ for the intercept-only model is `0.761`, whilst for the model with elevation this is `0.7146`. Note that the figures reported under `Std.Dev.` are simply the square root of the value reported under `Variance`. As expected, the introduction of elevation contributes to the explanation of the residual variation captured by $Z_i$, though by a very small amount. The estimated values of $\tau^2$ thus suggest that there is extra-Binomial variation in the data that is not account for by elevation.

In the next section, we will illustrate how to assess the presence of residual correlation for continuous measurements and overdispersed count data.

### 3.1.3 Exploring residual spatial correlation

In its most basic form, the concept of spatial correlation can be succinctly encapsulated by Tobler (1970) first law of geography, which posits that "everything is interconnected, but objects in close proximity exhibit stronger relationships than those situated farther apart." After we have identified the key variables to introduce as covariates in the model (Section 3.1.1) and, in the case of count data, assessed the presence of overdispersion (Section 3.1.2), our final exploratory step consists of assessing whether the residuals of the non spatial model show evidence of spatial correlation. Hence, in geostatistical modelling, the interest is not in the spatial correlation of the data, but rather on understanding whether the variation in the outcome unexplained by the covariates exhibits spatial correlation. We call this *residual spatial correlation*, to emphasize that spatial correlation is a concept relative to the covariates that we have introduced in the model.

In the context of geostatistical analysis, the tool that is generally used to assess the residual spatial correlation is the the so called *empirical variogram*. Before looking at the mathematical definition of the empirical variogram, let us consider a generalized linear mixed model as expressed in Equation 3.7. Our goal is then to question the assumption of independently distributed random effects $Z_i$ by asking whether the $Z_i$ show evidence of spatial correlation. Let $Z_i$ and $Z_j$ be two random effects that are associate with two different locations $x_i$ and $x_j$, respectively, and let us take the squared difference between the two

$$V_{ij} = (Z_i - Z_j)^2. \tag{3.10}$$

How does the spatial correlation affect the value of $V_{ij}$? To answer this question, we can refer to the aforementioned Tobler's law of geography. When $x_i$ and $x_j$ will be closer to each other, then $Z_i$ and $Z_j$ will also tend to be more similar to each other, thus making $V_{ij}$ smaller, on average. On the contrary, when $x_i$ and $x_j$ will be further apart, then $V_{ij}$ will become larger, on average. We can then construct the empirical variogram by considering all possible pairs of locations $x_i$ and $x_j$, for which we then compute $V_{ij}$ and plot this

against the distance between $x_i$ and $x_j$, which we denote as $u_{ij}$. If there is spatial correlation in the random effects $Z_i$, then this will manifest as an average increase in the $V_{ij}$ as $u_{ij}$ increases. However, there are still two issues that we have to address before we can generate and plot the empirical variogram.

The first issue is that we do not observe $Z_i$ as, by definition, this is a latent variable. Hence, we require an estimate for $Z_i$ which we can then feed into $V_{ij}$. To emphasize this point, from now on, we shall replace Equation 3.10 with

$$\hat{V}_{ij} = (\hat{Z}_i - \hat{Z}_j)^2. \tag{3.11}$$

Several options are available for estimating $Z_i$. Our choice is to use the model of the predictive distribution of $Z_i$, that is the distribution of $Z_i$ conditioned to the data $y_i$. This estimator for $Z_i$ is also readily available from the output of the `lmer` and `glmer` functions of the `lme4` package, as we will illustrate later in our example in this section.

The second issue is that if simply plot $\hat{V}_{ij}$ against the distances $u_{ij}$ (also knwon as *cloud variogram*), due to the high noiseness in the $\hat{V}_{ij}$, it may be quite difficult to assess the presence of an increasing trend in the $\hat{V}_{ij}$ and thus detect spatial correlation. Hence, it is general practice to group the distances $u_{ij}$ into classes, say $\mathcal{U}$, and then take average of all the $\hat{V}_{ij}$ that fall within $\mathcal{U}$.

We can now write the formal definition of the empirical variogram as

$$\hat{V}(\mathcal{U}) = \frac{1}{2|\mathcal{U}|} \sum_{(i,j):(u_i,u_j)\in\mathcal{U}} \hat{V}_{ij} \tag{3.12}$$

where $|\mathcal{U}|$ denotes the number of pairs of locations that fall within the distance class $\mathcal{U}$. The rationale behind dividing by 2 in $1/2|\mathcal{U}|$ from the above equation, will be elucidated in Section 3.2, and there is no need for us to delve into this matter at this juncture. When creating the empirical variogram plot, we select the midpoint values of the distance classes $\mathcal{U}$ to represent the x-axis values.

Before we can evaluate residual spatial correlation, there remains one crucial concern: relying solely on a visual inspection of the empirical variogram is susceptible to human subjectivity. Furthermore, it is worth noting that even a seemingly upward trend observed in the empirical variogram might be merely a product of random fluctuations, rather than a reliable indication of actual residual spatial correlation. To address these concerns and enhance the objectivity of the use of the empirical variogram, one approach would involve comparing the observed empirical variogram pattern with those generated in the absence of spatial correlation. Following this principle, we then use a permutation test that allows us to generate empirical variograms under the assumption of absence of spatial correlation through the following iterative steps.

1. Permute the order of the locations in the data-set while keeping everything else fixed.
2. Compute the empirical variogram $\hat{V}(\mathcal{U})$ for the permuted data-set.
3. Repeat 1 and 2 a large number of times, say 10,000.
4. Use the resulting 10,000 empirical varigorams to compute 95% confidence intervals, by taking the 0.025 and 0.975 quantiles of these for each distance class $\hat{V}(\mathcal{U})$.
5. If the observed empirical variogram falls fully within the envelope generated in the previous point, we then conclude that the data do not exhibit residual spatial correlation. If, instead, the observed empirical variogram partly falls outside the envelope we conclude that the data do exhibit residual spatial correlation.

We now show an application of all the concepts introduced in this section to the Liberia data on river-blindness.

#### 3.1.3.1 Example: assessing spatial correlation for the Liberia data

We consider the Binomial mixed model that uses the log-transformed elevation as a covariate to model river blindness prevalence, hence

$$\log\left\{\frac{p(x_i)}{1-p(x_i)}\right\} = \beta_0 + \beta_1 \log\{e(x_i)\} + Z_i \tag{3.13}$$

where $e(x_i)$ is the elevation in meters at location $x_i$ and the $Z_i$ are i.i.d. Gaussian variables with mean 0 and variance $\tau^2$. We first fit the model above using the `glmer` function.

```
# Convert the data-set into an sf object
liberia <- st_as_sf(liberia, coords = c("lat", "long"), crs =
 ↪  4326)


# Create the ID of the location
liberia$ID_loc <- 1:nrow(liberia)

# Binomial mixed model with log-elevation
fit_glmer_lib <- glmer(cbind(npos, ntest) ~ log(elevation) +
 ↪  (1|ID_loc), family = binomial,
                        data = liberia, nAGQ = 25)

summary(fit_glmer_lib)
```

```
Generalized linear mixed model fit by maximum likelihood
(Adaptive
  Gauss-Hermite Quadrature, nAGQ = 25) [glmerMod]
 Family: binomial  ( logit )
Formula: cbind(npos, ntest) ~ log(elevation) + (1 | ID_loc)
   Data: liberia

      AIC       BIC    logLik -2*log(L)  df.resid
    127.9     135.4     -61.0     121.9        87

Scaled residuals:
     Min       1Q   Median       3Q      Max
-2.46033 -0.63341 -0.07633  0.61995  3.12732

Random effects:
 Groups Name        Variance Std.Dev.
 ID_loc (Intercept) 0.003097 0.05565
Number of obs: 90, groups:  ID_loc, 90

Fixed effects:
               Estimate Std. Error z value Pr(>|z|)
(Intercept)    -2.96292    0.21184 -13.987  < 2e-16 ***
log(elevation)  0.26143    0.04071   6.422 1.35e-10 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Correlation of Fixed Effects:
            (Intr)
log(elevtn) -0.981
```

From the output, we observe that the estimate for $\tau^2$ is about 0.003, indicating a moderate level of overdispersion in the data.

```
liberia$Z_hat <- ranef(fit_glmer_lib)$ID_loc[,1]
```

Through the function `ranef` when extract the estimates of the random effects $Z_i$ and save these in the data set. We then use the function `s_variogram` from the `RiskMap` package to compute the empirical varigoram for the estimated $\hat{Z}_i$.

```
liberia_variog <- s_variogram(data = liberia,
                              variable = "Z_hat",
                              bins = c(15, 30, 40, 80, 120,
                                       160, 200, 250, 300, 350),
                              scale_to_km = TRUE,
                              n_permutation = 10000)
```

Through the argument `bins` we can specify the the classes of distance, previously denoted by $\mathcal{U}$; check the help page of `s_variogram` to see how this is defined by default. The value passed to `bins` in the code above correspond to define the following classes of distance $\mathcal{U}$: $[15, 30]$, $(30, 40]$ and so forth, with the last class being $[350, +\infty)$, i.e. all pairs of locations whose distances are above 350km. The argument `n_permutation` allows the user to specify the number of permutations that are performed the generate the envelope for absence of spatial correlation previously described.

```
dist_summaries(data = liberia,
               scale_to_km = TRUE)
## $min
## [1] 3.204561
##
## $max
## [1] 514.9768
##
## $mean
## [1] 199.7156
##
## $median
## [1] 185.7845
```

The `dist_summaries` function within the `RiskMap` package can be used for gauging the extent of the area covered by your dataset, aiding in the selection of appropriate values to be passed to the `bins` argument. In the provided output above, we can observe that for the Liberia dataset, the minimum and maximum distances span approximately 3km and 533km, respectively. While there is not a one-size-fits-all recommendation for setting `bins`, two fundamental principles should inform your decision-making. Firstly, it is advisable to avoid choosing overly large distance intervals, as the uncertainty associated with the empirical variogram tends to increase with distance due to fewer available pairs of observations for estimation. Secondly, especially when spatial correlation is not strong, it is crucial to carefully explore the behavior of the variogram at smaller distances. Consequently, it is generally advisable to experiment with different `bins` configurations and observe how they impact the pattern of the empirical variogram.

```
plot_s_variogram(liberia_variog,
                 plot_envelope = TRUE)
```

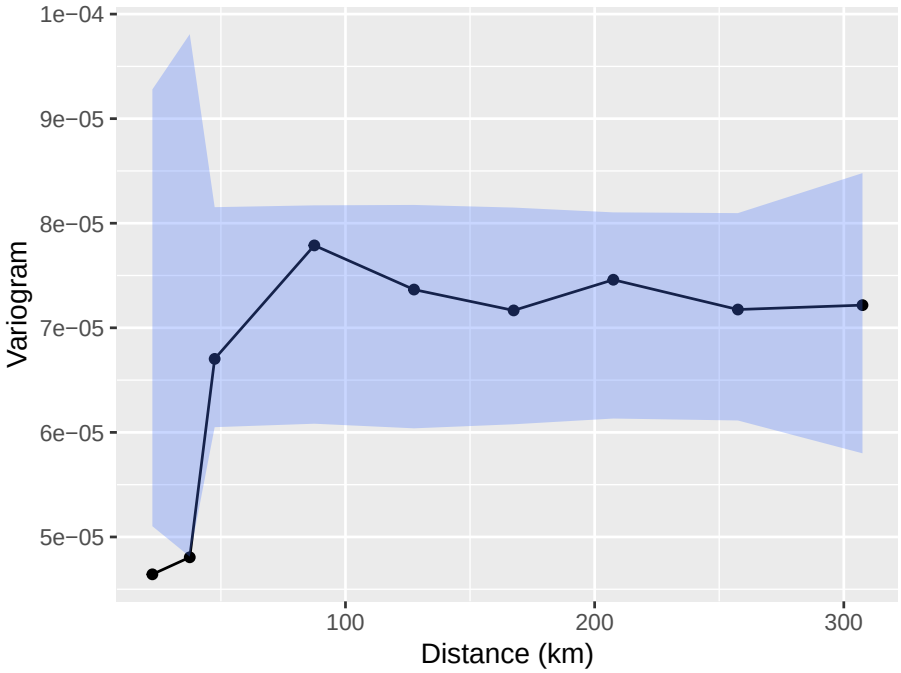



Figure 3.7: Plot of the empirical variogram (solid line) computed using the estimated random effects from the model in Equation 3.13. The blue shaded area is the 95% confidence level envelope generated using the permutation procedure described in Section 3.1.3.

Finally, the `plot_s_variogram` function enables us to visualize the empirical variogram and, through the `plot_envelope` argument, include the envelope generated by the permutation procedure. As illustrated in Figure 3.7, we observe that the empirical variogram falls outside the envelope at relatively short distances, typically below 30km. However, for distances exceeding 30km, the behavior of the empirical variogram does not significantly differ from variograms generated under the assumption of spatial independence. In summary, we interpret the evidence presented in Figure 3.7 as indicative of residual spatial correlation within the data. Nevertheless, it is essential to exercise caution when attempting to ascertain the scale of the spatial correlation using the empirical variogram. As we will emphasize throughout this book, the empirical variogram's sensitivity to the choice of bins values renders it an unreliable tool for drawing statistical inferences. In other words, we advocate employing the empirical variogram primarily to assess the presence of residual correlation.

#### 3.1.3.2 Exploring residual spatial correlation with linear Guassian models

When using a linear model to assess spatial correlation, it is important to distinguish two cases: 1) when the data contain only one location per location; 2) when more than one observation per location is available. We now consider each of these two scenarios separately.

*One observation per location*

To illustrate the use of the variogram under this scenario, we shall use the Gailicia data on lead concentration in moss samples. The simplest possible model for the data is a standard linear model without covariates which assumes independence among the observations, hence

$$Y_i = \beta_0 + U_i \tag{3.14}$$

where the $U_i$ are i.i.d. Gaussian variables with mean zero and variance $\omega^2$. At this stage, our goal is then to assess whether the assumption of independence for the $Z_i$ is supported by the data or whether there is evidence of spatial correlation. However, the measurement error of the device may be present as part of the natural random variation in $Z_i$ which might mask the detection of spatial correlation using the residuals $Z_i$ challenging, especially is the measurement error dominates the spatial variation of the data. One would be tempted to introduce an additional, location-specific random effect, say $Z_i$ with mean zero and variance $\tau^2$ and fit the model

$$Y_i = \beta_0 + Z_i + U_i, \tag{3.15}$$

where we interpret $U_i$ as random variation due to the measurement device and $Z_i$ as a random effect accounting for unmeasured covariates that contribute to the variation between locations in lead concentration. However, the model in Equation 3.15 is not identifiable because we cannot disentangle the separate contributions of $Z_i$ and $U_i$ from the variation of the data, unless: a) we know the precision of the measurement device, $\omega$ (recall that $\omega^2$ is the variance of $U_i$); b) or, if we do not know $\omega$, we can then separate the two sources of variation only if we have multiple observations per location (the scenario which we shall consider in the next section).

For the Galicia data, for we do not know the measurement device precision and we only have one observation per location. However, this does not prevent us from using the variogram based on the residuals from Equation 3.14, while keeping in mind the limitations and uncertainty that are inherent to this exploratory tool as remarked at the end of the last paragraph.

```
# Fitting of the linear model and extraction of the residuals
lm_fit <- lm(log(lead) ~ 1, data = galicia)
galicia$residuals <- lm_fit$residuals

# Convert the galicia data frame into an "sf" object
galicia_sf <- st_as_sf(galicia, coords = c("x", "y"), crs =
 ↪  32629)

# Compute the variogram, using the residuals from the linear
 ↪  model fit,
# and the 95% confidence level envelope for spatial independence
galicia_variog <- s_variogram(galicia_sf, variable =
 ↪  "residuals",
                              scale_to_km = TRUE,
                              bins = seq(10, 140, length = 15),
                              n_permutation = 10000)

# Plotting the results
plot_s_variogram(galicia_variog, plot_envelope = TRUE)
```

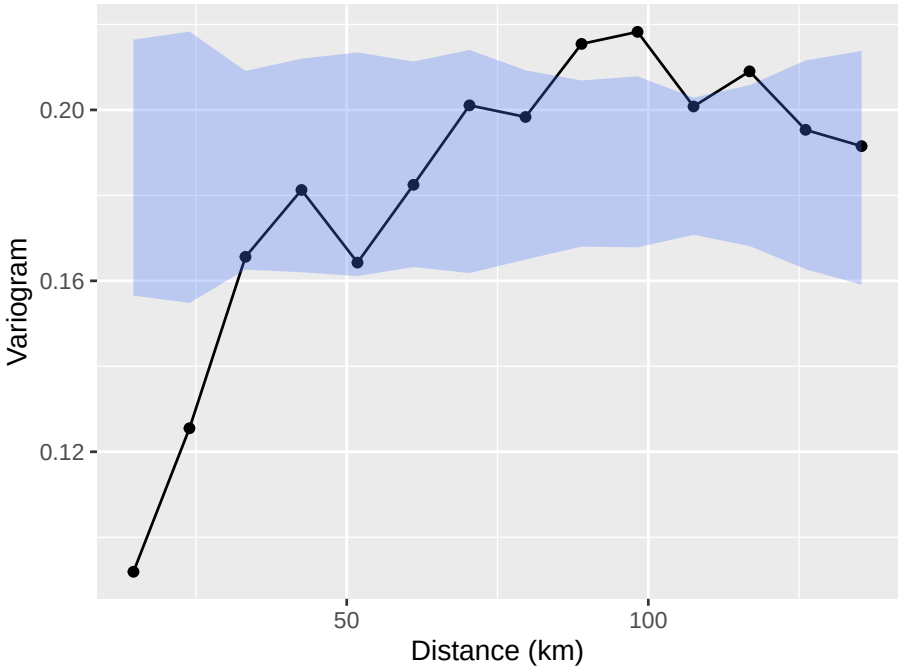



Figure 3.8: Plot of the empirical variogram (solid line) computed using the estimated residuals from the model in Equation 3.14. The blue shaded area is the 95% confidence level envelope generated using the permutation procedure described in Section 3.1.3.

In the code above, we first fit the linear model in Equation 3.14 and then extract the residuals from this. Note that for this simple model, the residuals of the model are obtained by simply centering the outcome to zero by subtracting its mean. However, for more complex linear models that use covariates the computation of the residuals is more involved and can be carried out through a simple extension of the code above by specifying an appropriate `formula` in the `lm` function.

Figure 3.8 shows the empirical variogram and the 95% envelope for spatial independence. This clearly shows that the measurement of lead concentration are spatially correlated.

*More than one observation per location*

For the case of more than one observation per location, we shall consider the `italy_sim` data-set. This data-set contains 10 observations per location, for a total of 200 locations. The variable `ID_loc` is a numeric indicator that can be

used to identify the location each observation belong to. For this data-set, we use the population density, named `pop_dens`, as a log-transformed covariate; we leave you as an exercise to assess that is a reasonable modelling choice.

To specify a non-spatial mixed model for the outcome, we then use two subscripts: $i$ to identify a given location; $j$ to identify the $j$-th observation for a given location $i$. We denote as $Z_i$ the location-specific random effect ans as $U_{ij}$ the random variation due to the measurement error inherent to each observation. Hence, we write

$$Y_{ij} = \beta_0 + \beta_1 \log\{d(x_i)\} + Z_i + U_{ij}, \tag{3.16}$$

where $d(x_i)$ is the population density at location $x_i$; as before, we use $\tau^2$ and $\omega^2$ to denote the variances of $Z_i$ and $U_{ij}$, respectively. To fit this model to the data we use the `lmer` function from the `lme4` package.

```
# Fitting a linear mixed model to the italy_sim data-set
# See main text for model specification
lmer_fit <- lmer(y ~ log(pop_dens) + (1|ID_loc), data =
 ↪  italy_sim)

summary(lmer_fit)
## Linear mixed model fit by REML ['lmerMod']
## Formula: y ~ log(pop_dens) + (1 | ID_loc)
##    Data: italy_sim
##
## REML criterion at convergence: 3390.9
##
## Scaled residuals:
##      Min       1Q   Median       3Q      Max
## -2.85209 -0.64198 -0.01832  0.66014  3.01870
##
## Random effects:
##  Groups   Name        Variance Std.Dev.
##  ID_loc   (Intercept) 2.5006   1.5813
##  Residual             0.1959   0.4426
## Number of obs: 2000, groups:  ID_loc, 200
##
## Fixed effects:
##               Estimate Std. Error t value
## (Intercept)   -0.40448    0.57613  -0.702
## log(pop_dens)  1.33798    0.07643  17.506
##
## Correlation of Fixed Effects:
##             (Intr)
## log(pp_dns) -0.981
```

```
# Incorporating the estimated random effects into the data
italy_sim$rand_eff <-ranef(lmer_fit)$ID_loc[italy_sim$ID_loc,1]

# Converting the italy_sim data frame into an "sf" object
italy_sim_sf <- st_as_sf(italy_sim, coords=c("x1", "x2"), crs =
 ↪  32634)

# Compute the variogram, using the random effects from the
 ↪  linear mixed model fit,
# and the 95% confidence level envelope for spatial independence
italy_sim_variog <- s_variogram(italy_sim_sf, variable =
 ↪  "rand_eff",
                                scale_to_km = TRUE,
                                n_permutation = 200)

# Plotting the results
plot_s_variogram(italy_sim_variog, plot_envelope = TRUE)
```

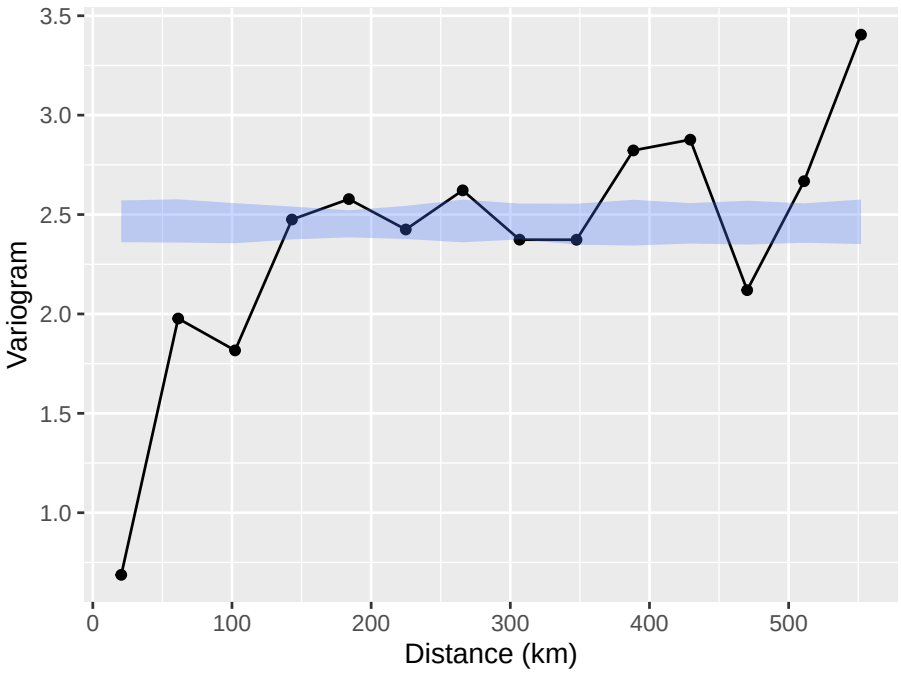



Figure 3.9: Plot of the empirical variogram (solid line) computed using the estimated random effects from the model in Equation 3.16. The blue shaded area is the 95% confidence level envelope generated using the permutation procedure described in Section 3.1.3.

In the code above, the introduction of the $Z_i$ random effect is specified in the `lmer` function though `(1|ID_loc)` in the formula; recall that `ID_loc` is the numerical indicator that identifies each location. From the summary of the model, we obtain both the estimates of the regression coefficients $\beta_0$ and $\beta_1$, as well for the variances $\tau^2$ listed under `Random effects:`. The estimate of $\sigma^2$ is found on the line of printed output starting with `ID_loc`, whilst that for `$ ^2` is next to `Residual`.

The application of the empirical variogram, whose results are shown in Figure 3.9, indicate the presence of residual correlation. This is because we observe that the solid line representing the empirical variogram falls outside of the 95% envelope for spatial independence.

## 3.2 The linear geostatistical model

In this section, we consider spatially referenced outcomes $Y_i$ that are continuous. We first consider the simpler case of a single measurement $Y_i$ per location $x_i$. Recalling the class of generalized linear models introduced in Section 1.5, the linear predictor takes the form

$$\mu_i = d(x_i)^\top \beta + S(x_i). \tag{3.17}$$

Hence, in this case, we interpret $\beta$ as the effect on $\mu_i$ for a unit increase in $d(x_i)$. Let $U_i$ denote i.i.d. random variables representing the measurement error associated with $Y_i$, each having mean zero and variance $\omega^2$. Thanks to the linear properties of Gaussian random variables, we can also express the linear model in a compact expression, as

$$Y_i = \mu_i + U_i = d(x_i)^\top \beta + S(x_i) + U_i. \tag{3.18}$$

To fully specify a geostatistical model for our dataset, we must address two critical aspects.

1. Defining the relationship between each covariate and the mean value $\mu_i$.
2. Selecting an appropriate correlation function for $S(x)$.

As illustrated in the previous sections, the initial step of exploratory analysis allows us to handle the first aspect, where we determine the regression relationship between covariates and the mean value $\mu_i$. However, based on existing methods of exploratory analysis, it is difficult to understand what is a suitable correlation function at this stage. A commonly recommended starting point is the Matern correlation function (see Equation 1.5), which offers considerable flexibility in capturing a wide range of correlation structures, under

the assumption stationarity and isotropy. As we shall illustrate in the next example, even estimating a Matern correlation function is a task that poses many inferential challenges due to the poor identifiability, especially, of its smoothness parameter $\kappa$.

### 3.2.1 Evaluating the inclusion of the measurement error term $U_i$ and the specification of the smoothness parameter $\kappa$

In this section we analyse the Galicia data using a linear geostatistical geostatitsical model for the log-transformed lead concentration, which we denote as $Y_i$. Since we do not use covariates, we then write the model as

$$Y_i = \mu + S(x_i) + U_i \tag{3.19}$$

where were $S(x)$ is a Matern process with variance $\sigma^2$, scale parameter $\phi$ and and smothness parameter $\kappa$; the $U_i$ correspond to the measurement error term and denote with $\omega^2$ their variance.

We carry out the parameter estimation of the model using the `glgpm` function from the `RiskMap` package. This function implements maximum likelihood estimation for generalized linear mixed models using a Matern correlation function, while fixing the smoothness parameter $\kappa$ at prespecified value by the user. The object passed to the argument `data` in `glpm` can either be a `data.frame` object or an `sf` object. Below we illustrate the use of `glgpm` while distinguishing between these two cases.

```
# Parameter estimation when the argument passed to `data` is a
 ↪  data-frame
fit_galicia <-
glgpm(log(lead) ~ gp(x, y, kappa = 1.5), data=galicia, family =
 ↪  "gaussian",
      crs = 32629, scale_to_km = TRUE, messages = FALSE)
```

The code above shows the use of `glgpm` by passing `galicia` as a `data.frame` object to `data`. The specification of the Guassian process $S(x)$ is done through the addition of the term `gp()` in the formula. The function `gp` allows you to specify the columns of the coordinates in the data, in this case `x` and `y`, the smoothness parameter through the argument `kappa`, set to 1.5 in this example. In the help page of `gp`, you can see that by default the nugget term (denoted in this book by the random variable $Z_i$) is excluded from the model by default; to include and estimate the variance parameter of the nugget, you should set `nugget=NULL` in the `gp()` function. However, doing so for a linear model that only has one observation per location will generate error message as this is a non-identifiable model for the same reasons given in Section 3.1.3.2. However, if the measurement error variance is known this can be fixed by the user using the argument `fix_var_me` and the inclusion of the nugget term is then possible (to better understand this point, try Exercise 7 at the end of this chapter).

The argument `crs` is used to specify the coordinate reference system (CRS) of the data. For the Galicia data, as well as for every other data-set used in this book, the CRS is reported in the help page description of the data-set. If `crs` is not specified, the function will assume that the coordinates are in longitude/latitude format and will use these without applying any transformation. Finally, the argument `scale_to_km`, is used to specify whether the distances between locations should be scaled to kilometers or maintained in meter; this argument will not affect the scale, and thus the interpretation of the spatial correlation parameter $\phi$.

```
# Parameter estimation when the argument passed to `data` is an
 ↪  sf object
galicia_sf <- st_as_sf(galicia, coords = c("x", "y"), crs =
 ↪  32629)

fit_galicia_sf <-
glgpm(log(lead) ~ gp(kappa = 1.5), data=galicia_sf, family =
 ↪  "gaussian",
      scale_to_km = TRUE, messages = FALSE)
```

The code above shows the alternative approach to estimate the model, when the argument passed to `data` is an `sf` object. In this case, the data-set `galicia` is converted into an `sf` object before the use of the `glgpm` function using `st_as_sf`. When when then fit the linear geostatistical model with `galicia_sf`, the only differences with the previous chunk of code that used `galicia` instead, is that the coordinates names in `gp()` and the `crs` argument in `glgpm` do not need to be specified as they are both directly obtained from `galicia_sf`.

```
summary(fit_galicia)
## Call:
## glgpm(formula = log(lead) ~ gp(x, y, kappa = 1.5), data =
 ↪  galicia,
##     family = "gaussian", crs = 32629, scale_to_km = TRUE,
 ↪  messages = FALSE)
##
## Linear geostatistical model
## Link: identity
## Inverse link function = x
##
## 'Lower limit' and 'Upper limit' are the limits of the 95%
 ↪  confidence level intervals
##
## Regression coefficients
##             Estimate Lower limit Upper limit   StdErr z.value
 ↪  p.value
```

```
## (Intercept) 0.707418    0.552762    0.862075 0.078908  8.9651
 ↪  < 2.2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' '
 ↪  1
##
##                          Estimate Lower limit Upper limit
## Measuremment error var. 0.0154636   0.0051797      0.0462
##
## Spatial Gaussian process
## Matern covariance parameters (kappa=1.5)
##                       Estimate Lower limit Upper limit
## Spatial process var.  0.17127     0.13303      0.2205
## Spatial corr. scale   9.02085     7.59521     10.7141
## Variance of the nugget effect fixed at 0
##
## Log-likelihood: 69.03029
## AIC: -130.0606
```

We can then inspect the point and interval estimates for the model through the `summary` function of the model as shown in the code chunk above. This outputs is presented in three sections: in the first section, we have the results for the regression coefficients; in the second section, we have the estimate for the variance of the measurement error component, $\omega^2$ whose point estimate is about 0.015; in the final section, we have the estimates for the parameters of the spatial covariance function, $\sigma^2$ and $\phi$ which are about 0.171 and 9.021 (km), respectively. The message `Variance of the nugget effect fixed at 0` indicates that the nugget has not been included in the mode; try Exercise 7 to see how this summary changes in the presence of the nugget term.

At this point, you may be wondering, why we have used 1.5 for the value of $\kappa$ and whether there is a statistical approach to find the most suitable value for this. We show such an approach in the code below. However, before examining the code, we would like to point an important aspect that relates to how the value of $\kappa$ can affect the estimate of the measurement error variance $\omega^2$. As we have shown, in Section 1.5.1, values of $\kappa$ that are closer to zero will give a rougher and less regular surface for $S(x)$. In the case of a single observation, when $\kappa$ is closer to zero, it will thus become increasingly difficult to estimate $\omega^2$ since the likelihood function may attribute most of the noisiness to $S(x)$ and rely less on $U_i$ to explain the unstructured random variation found in the data. As a consequence of this, we can expect that for value of $\kappa$ closer to zero, the estimates of $\omega^2$ will also be smaller and, on the contrary, when $\kappa$ is larger, the estimates of $\omega^2$ will also increase. The results generated in the below clearly illustrate this point.

```
# Number of the values chosen for kappa
n_kappa <- 10

# Set of values for kappa
kappa_values <- seq(0.5, 3.5, length = n_kappa)

# Vector that will store the values of the likelihood function
# evaluated at the maximum likelihood estimate
llik_values <- rep(NA, length = n_kappa)

# Vector that will store the maximum likelihood estimates
# of the variance of the measurement error
sigma2_me_hat <- rep(NA, length = n_kappa)

# List that will contain all the geostatistical models fitted
 ↪  for
# the different values of kappa specified in kappa_values
fit_galicia_list <- list()

for(i in 1:n_kappa) {
  fit_galicia_list[[i]] <- glgpm(log(lead) ~ gp(x, y, kappa =
 ↪  kappa_values[i]),
                                 data=galicia, family =
                                  ↪  "gaussian",
                                 crs = 32629, scale_to_km =
                                  ↪  TRUE, messages = FALSE)
  llik_values[i] <-  fit_galicia_list[[i]]$log.lik
  sigma2_me_hat[i] <-  coef(fit_galicia_list[[i]])["sigma2_me"]
}
```

By examining the results shown in panel (a) of Figure 3.10, we observe that, as expected, smaller values for $\kappa$ leads to smaller point estimates for $\omega^2$ and viceversa. This begs the question, what should be our chosen value for $\kappa$?

To answer this question, a natural approach is to estimate the model for different values of $\kappa$ and see which one give the best fit to the data, according to the likelihood function. In panel (b) of Figure 3.10, we show the results of this approach where we considered 10 values for $\kappa$ within the range 0.5 to 3.5 (see code above). In this plot, we have also added a horizontal line to help us to approximate a range of the most plausible value for $\kappa$. More precisely, the horizontal dashed line is computed by taking the maximum observed values of the likelihoods computed for the different values of $\kappa$, say $\widehat{M}$ and we subtract the quantile 0.95 of a $\chi^2$ distribution with 1 degree of freedom. In R this horizontal line is obtained as

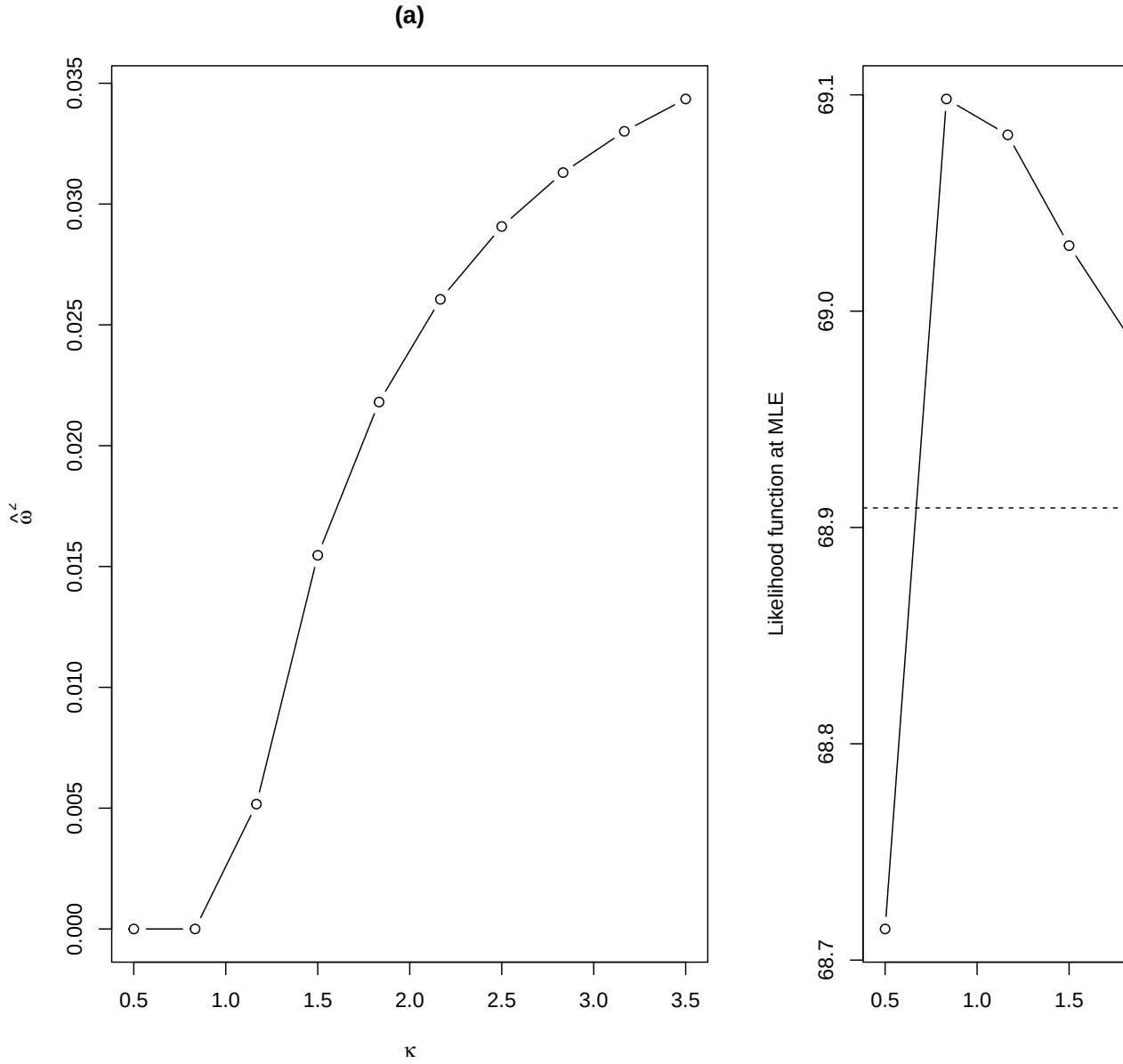



Figure 3.10: (a) plot of the maximum likelihood estimate for the variance of the measurement error $U_i$, denoted in the text as $\omega^2$, against the chosen fixed value for $\kappa$; (b) profile likeihood for $\kappa$ with the horizontal dashed line corresponding to the (approximate) threshold for constructing a 95% confidence interval based on a $\chi^2$ with one degree of freedom.

```
max(llik_values)-pchisq(0.95, df = 2)/2
```

where `llik_values` is as defined in the previous chunk of code. Note that this approach is essentially constructing the profile likelihood for $\kappa$ which one could use to derive a confidence interval for $\kappa$ with a finer segmentation for `kappa_values`. However, our current objective is not to derive the confidence interval for $\kappa$, but rather to gain a broad understanding of the $\kappa$ values supported by the dataset. The values of $\kappa$ that corresponds to likelihood values above the horizontal line are approximately between 0.75 and 2.75. Hence, selecting $\kappa = 1.5$ seems to be a reasonable one in this case. Now, you may be pondering: are there value other than $\kappa = 1.5$ that could fit the data even better? Our answer is that it is not worth the effort to try estimate $\kappa$ more precisely because it is empirically very difficult and, under some scenarios, even impossible. Estimating $\kappa$ poses a well-documented challenge in geostatistics (Zhang (2004)), which justifies our adoption of a pragmatic approach that sets it at a predefined value. This issue is also further exacerbated when analyzing count data, which tend to be less informative about the correlation structure than continuously measured data.

### 3.2.2 Modelling hierarchical geostatistical data using the `re()` function

We now consider the analysis of geostatistical data with a hierarchical structure and show how to formulate and fit a geostatistical model that accounts for the effects of the different layers of the data. For this purpose, we use the `italy_sim` where each of the sampled locations can be grouped according to two administrative subdivisions of Italy, regions (Admin level 2) and provinces (Admin level 3), as shown in Figure 3.11.

```
library(rgeoboundaries)
library(mapview)
italy_regions <- geoboundaries(country = "italy", adm_lvl =
 ↪  "adm2")

italy_provinces <- geoboundaries(country = "italy", adm_lvl =
 ↪  "adm3")

par(mfrow = c(1,2))

# Map of the data with the region boundaries
map_regions <- ggplot() +
geom_sf(data = italy_sim_sf, pch = 4, color = "red") +
geom_sf(data = italy_regions, fill = NA) +
theme_void() +
labs(title = "Regions") +
```

```
theme(plot.title = element_text(hjust = 1/2))

# Map of the data with the province boundaries
map_provinces <- ggplot() +
geom_sf(data = italy_sim_sf, pch = 4, color = "red") +
geom_sf(data = italy_provinces, fill = NA) +
theme_void() +
labs(title = "Provinces") +
theme(plot.title = element_text(hjust = 1/2))

library(gridExtra)
grid.arrange(map_regions, map_provinces, ncol = 2)
```

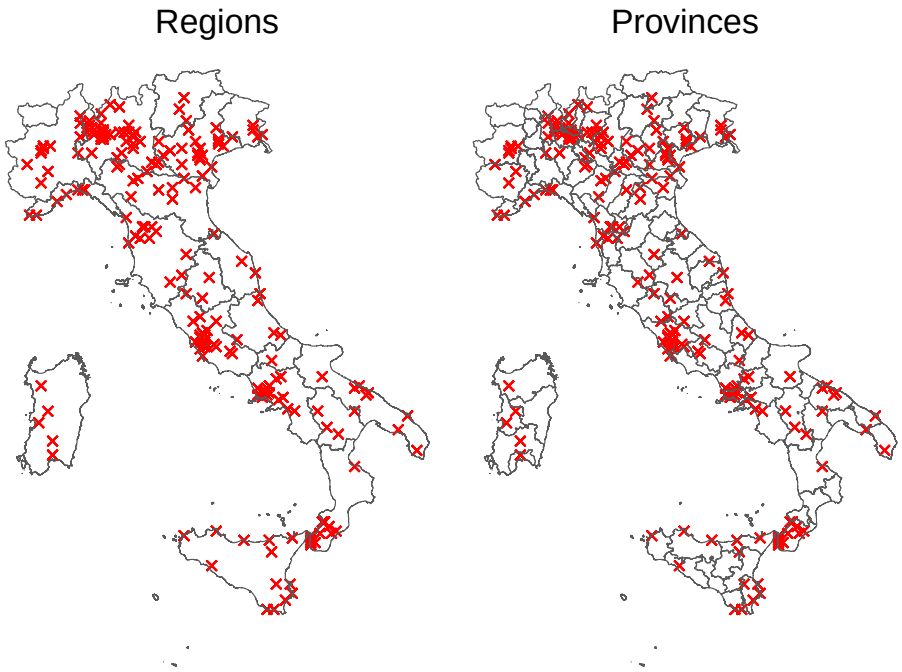



Figure 3.11: The plot shows the location of the data-set `italy_sim` (red crosses), with the boundaries of the regions (left) and provinces (right) of Italy.

Let $V_h$, for $h = 1, \dots, n_V$ and $Z_k$, for $k = 1, \dots, n_Z$, be a set of mutually independent Gaussian variables with zero means and variances $\sigma_V^2$ and $\sigma_Z^2$, respectively. Finally let $\mathcal{A}_h$, for $h = 1, \dots, n_V$ and $\mathcal{B}_k$, for $k = 1, \dots, n_Z$, denote the areas encompassed by the boundaries of the region and provinces, respectively. Since we have 10 repeated observations at each locations, we shall use

$Y_{ij}$ to denote the $j$-th outcome (found in the data under the column `y`) at the $i$-th location $x_i$. The model from which we generated data $y_{ij}$ is

$$Y_{ij} = \beta_0 + \overbrace{\beta_1 d(x_i) + S(x_i)}^{\text{Location-level effect}} + \underbrace{\sum_{h=1}^{n_V} I(x_i \in \mathcal{A}_h) \times V_h}_{\text{Region effect}} + \overbrace{\sum_{k=1}^{n_Z} I(x_i \in \mathcal{B}_k) \times Z_k}^{\text{Province effect}} + \underbrace{U_{ij}}_{\text{Measurement error}}, \tag{3.20}$$

for $j = 1, \ldots 10$ and $i = 1, \ldots, 200$, where, $d(x_i)$ is the log-transformed population density, $I(x \in \mathcal{R})$ is an indicator function that takes value 1 if the location $x$ falls within the boundaries of the area denoted by $\mathcal{R}$ and 0 otherwise. The covariance function chosen in the generation of the data was an exponential correlation function hence $\text{cov}\{S(x), S(x') = \sigma^2 \exp\{-||x - x'||/\phi\}$.

The inclusion of the random effects $V_h$, for the region, and $Z_k$, for the province, can be included by using the `re()` function into the `formula` passed to `glgpm` as shown below.

```
                        # Location-level effect
italy_fit <- glgpm( y ~ log(pop_dens) + gp(kappa = 0.5, nugget
 ↪  = 0) +
                        # Region and province effects
                        re(region, province),
         data = italy_sim_sf, scale_to_km = TRUE,
         family = "gaussian")
##   0:     441.00731: -0.404481  1.33798  0.00000  4.62406
 ↪  0.00000  0.00000  0.00000
##   1:    -130.05497: -0.408825  1.35003 -0.0387110  4.66059
 ↪  -0.910409 -0.00265292 0.0107965
##   2:    -268.13436: -0.865126  1.39525 -0.0305269  5.92133
 ↪  -1.96079 -0.666581 0.487537
##   3:    -329.24560: -1.44091  1.38139 -0.546306  4.98616
 ↪  -1.67936 -1.15202 0.383821
##   4:    -331.20861: -1.38498  1.37256 0.165463  5.61366
 ↪  -1.63033 -2.11435 0.371263
##   5:    -331.28418: -0.874796  1.37322 -0.271990  5.18071
 ↪  -1.62887 -0.970704 0.371407
##   6:    -331.42968: -1.04087  1.37294 -0.119652  5.33634
 ↪  -1.62883 -1.33321 0.375031
##   7:    -331.43375: -1.07312  1.37288 -0.0878211  5.36725
 ↪  -1.62883 -1.42011 0.376251
##   8:    -331.43375: -1.07445  1.37288 -0.0866254  5.36840
 ↪  -1.62883 -1.42489 0.376323
##   9:    -331.43375: -1.07445  1.37288 -0.0866254  5.36840
 ↪  -1.62883 -1.42489 0.376323
```

```
summary(italy_fit)
## Call:
## glgpm(formula = y ~ log(pop_dens) + gp(kappa = 0.5, nugget =
 ↪  0) +
##     re(region, province), data = italy_sim_sf, family =
 ↪  "gaussian",
##     scale_to_km = TRUE)
##
## Linear geostatistical model
## Link: identity
## Inverse link function = x
##
## 'Lower limit' and 'Upper limit' are the limits of the 95%
 ↪  confidence level intervals
##
## Regression coefficients
##                Estimate Lower limit Upper limit    StdErr
 ↪  z.value p.value
## (Intercept)   -1.074451   -2.031690   -0.117212  0.488396
 ↪  -2.2 0.02781 *
## log(pop_dens)  1.372877    1.352018    1.393735  0.010642
 ↪  129.0 < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' '
 ↪  1
##
##                         Estimate Lower limit Upper limit
## Measuremment error var.   1.2167      1.2009      1.2327
##
## Spatial Gaussian process
## Matern covariance parameters (kappa=0.5)
##                       Estimate Lower limit Upper limit
## Spatial process var.   0.91702     0.59470       1.414
## Spatial corr. scale  214.51980   153.60751     299.587
## Variance of the nugget effect fixed at 0
##
## Unstructured random effects
##                             Estimate Lower limit Upper limit
## region (random eff. var.)    0.24053     0.14553      0.3976
## province (random eff. var.)  0.37632     0.35607      0.3977
##
## Log-likelihood: 331.4338
## AIC: -648.8675
```

In the code above, the `factor` variables `region` and `province` found in `italy_sim` are passed to `re()` and these are estimated as unstructured random effects as defined by Equation 3.20. From the summary of model, we then obtain the parameter and interval estimates as reported in Table 3.2.

Table 3.2: Maximum likelihood estimates and, lower and upper limits of the 95% confidence interval for the parameters of the model in Equation 3.16.

| Parameter | Point estimate | Lower limit | Upper limit |
|---|---|---|---|
| $\beta_0$ | -1.074 | -2.032 | -0.117 |
| $\beta_1$ | 1.373 | 1.352 | 1.394 |
| $\sigma^2$ | 0.917 | 0.595 | 1.414 |
| $\phi$ | 214.520 | 153.608 | 299.587 |
| $\sigma^2_V$ | 0.241 | 0.146 | 0.398 |
| $\sigma^2_Z$ | 1.457 | 1.379 | 1.540 |
| $\omega^2$ | 0.196 | 0.194 | 0.199 |

The function `to_table` from the `RiskMap` package can be used to obtain the Latex or HTML code directly from a fit of the model. Here is an example.

```
to_table(italy_fit, digits = 3)
## % latex table generated in R 4.5.1 by xtable 1.8-4 package
## % Fri Aug 15 14:28:10 2025
## \begin{table}[ht]
## \centering
## \begin{tabular}{rrrr}
##   \hline
##  & Estimate & Lower limit & Upper limit \\
##   \hline
## (Intercept) & -1.074 & -2.032 & -0.117 \\
##   log(pop\_dens) & 1.373 & 1.352 & 1.394 \\
##   Spatial process var. & 0.917 & 0.595 & 1.414 \\
##   Spatial corr. scale & 214.520 & 153.608 & 299.587 \\
##   region (random eff. var.) & 0.241 & 0.146 & 0.398 \\
##   province (random eff. var.) & 0.376 & 0.356 & 0.398 \\
##   Measuremment error var. & 1.217 & 1.201 & 1.233 \\
##    \hline
## \end{tabular}
## \end{table}
```

## 3.3 Generalized linear geostatistical models

We now consider the modelling of count outcomes, focusing our attention to the Binomial and Poisson geostatistical models.

The use of Binomial geostatistical model is appropriate whenever the reported counts have a known upper bound. For example, when analyzing aggregated disease counts from cross-sectional survey data, the total number of people sampled at a location defines the largest number of possible reported disease cases. Recalling the model specification of Table 1.3, we specify the Binomial geostatistical model by stating that the reported disease counts $Y_i$ out of $n_i$ (the maximum value that $Y_i$ can take) at location $x_i$, conditionally on the spatial random effects $S(x_i)$, follow a Binomial distribution with probability $p(x_i)$ and logit-linear predictor

$$\log\left\{\frac{p(x_i)}{1-p(x_i)}\right\} = d(x_i)^\top\beta + S(x_i). \tag{3.21}$$

In this model, $p(x_i)$ is commonly understood as representing the prevalence of the disease under consideration, defined as the proportion of individuals with the disease at a specific moment. While $p(x_i)$ is not strictly a proportion, its interpretation as disease prevalence is justified by the fact that if we were to sample a sufficiently large population at location $x_i$, the proportion of individuals with the disease would converge closely to the value of $p(x_i)$.

A Binomial geostatistical model is also used when analyzing data at individual level where the outcome is binary and usually indicates the positive or negative result of a test for a disease under investigation. In this analysis, a mix of individual-level variables and location-level variables could be introduced to model the probability of a positive test. To formally write the model, we now use $Y_{ij}$ to indicate the binary outcome taking value 1, if the test was positive, and 0, if negative, for the $j$-th individual sampled at location $x_i$. The linear predictor now takes the form

$$\log\left\{\frac{p_{ij}}{1-p_{ij}}\right\} = e_{ij}^\top\gamma + d(x_i)^\top\beta + S(x_i). \tag{3.22}$$

In the model above we have distinguished between covariates $e_{ij}$, that express the effect on $p_{ij}$ due to individual traits (such as age and gender), and covariates $d(x_i)$, that instead quantify the effect on $p_{ij}$ resulting from the environmental attributes of location $x_i$ (such as elevation, temperature, and precipitation). In this model $p_{ij}$ – the probability of a positive test – once again can be interpreted as the disease prevalence for a population characterized by the individual traits $e_{ij}$ and residing at location $x_i$.

Finally, when the counts arise from a population whose size is unknown or when the occurrence of a disease is very small relative to the size of the target

population, one should consider the use of a Poisson geostaistical model. We then use $Y_i$ to denote the reported counts at location $x_i$ and assume that this, conditionally on a Gaussian process $S(x_i)$, follows a Poisson distribution with mean $\lambda(x_i)$ and log-linear predictor

$$\log\{\lambda(x_i)\} = d(x_i)^\top\beta + S(x_i). \tag{3.23}$$

If for example, $Y_i$ corresponds to the number of new cases reported at a health facility at location $x_i$ over a certain, the $\lambda(x_i)$ is interpreted as the average number of new cases reported over that period. If the information on the size of the population at risk for the diseases were available, this can be incorporated into the model so as to enable the interpretation of $\lambda(x_i)$ as the disease incidence, defined as the total number of new cases reported over a given period of time divided by the population at risk. If we denote the latter as $m_i$, then we would specify the mean of $Y_i$ as $m_i\lambda(x_i)$ an model $\lambda(x_i)$ as already specified in Equation 3.23. Another prevalent application of the Poisson geostatistical model in epidemiology involves mapping the abundance of disease vectors. Here, $\lambda(x_i)$ corresponds to the mean number of vectors per unit area. Also, note that in this last scenario $m_i$ cannot be defined as it is the goal of the analysis to infer the vector population size at location $x_i$.

In all the three models, namely Equation 3.21, Equation 3.22 and Equation 3.23, one might consider the inclusion of an additional random effect $Z_i$, previously used for the linear geostatistical model, to account for small spatial variation or overdispersion that stems from the omission of covariates that may not necessarily exhibit spatial structure, such as behavioral or genetic traits. We shall further elaborate and provide guidance on this point in the following examples of this section.

### 3.3.1 Example: river-blindness in Liberia

In section Section 3.1.1.1, we carried out the exploratory analysis of the river-blindness prevalence data to assess how the covariate of elevation could be introduced into the model. In Section 3.1.3.1, we then tested for residual spatial correlation, when the log-transformed elevation is used as a covariate, using the empirical variogram. Based on these results, we now formulate a fit a Binomial geostatistical model to data. To achieve this, we extend the model presented in Equation 3.13 through the inclusion of a spatial Gaussian process, $S(x)$, hence we write

$$\log\left\{\frac{p(x_i)}{1-p(x_i)}\right\} = \beta_0 + \beta_1\log\{e(x_i)\} + S(x_i) + Z_i. \tag{3.24}$$

We shall assume that $S(x_i)$ is a stationary isotropic Gaussian process with an exponential correlation function function $\rho(u) = \exp\{-u/\phi\}$, with scale parameter $\phi$.

We apply the Monte Carlo Maximum Likelihood (MCML) method for parameter estimation, which is implemented via the `glgpm` function. In this section, we set aside the technical details of the estimation process to focus exclusively on interpreting the results. A more detailed explanation of the MCML theory can be found in Section 3.5.2, with a concise summary of the practical aspects provided in Section 3.5.2.3.

```
fit_liberia <-
  glgpm(npos ~ log(elevation) + gp(long, lat),
        den = ntest, data = liberia,
        crs = 4326,
        convert_to_crs = 32629,
        family = "binomial")
```

```
  0:     -0.0000000: -2.99847 0.313076  0.00000  4.21726
  1:     -2.1141168: -3.00365 0.291428 -0.0788736  4.26632
  2:     -9.0491723: -3.12903 0.317201 -0.397221  4.49593
  3:     -12.533064: -3.21939 0.335348 -1.14423  3.84502
  4:     -13.483645: -3.15355 0.339255 -1.30011  3.77190
  5:     -13.586683: -3.13751 0.340498 -1.30940  3.77436
  6:     -13.872434: -3.01604 0.330734 -1.36581  3.84158
  7:     -15.117593: -2.86655 0.294037 -1.38728  3.85918
  8:     -15.266129: -2.74575 0.271038 -1.38870  3.87034
  9:     -15.266321: -2.74884 0.271551 -1.38930  3.86796
 10:     -15.266321: -2.74885 0.271552 -1.38930  3.86795
```

```
summary(fit_liberia)
```

```
Call:
glgpm(formula = npos ~ log(elevation) + gp(long, lat), data =
liberia,
    family = "binomial", den = ntest, crs = 4326,
    convert_to_crs = 32629)

Binomial geostatistical linear model
Link: canonical
Inverse link function = stats::plogis(x)

'Lower limit' and 'Upper limit' are the limits of the 95%
confidence level intervals

Regression coefficients
               Estimate Lower limit Upper limit    StdErr
               z.value   p.value
(Intercept)    -2.748848   -3.170559   -2.327136  0.215163
-12.776 < 2.2e-16
```

```
log(elevation)  0.271552    0.230762    0.312341  0.020811
13.048 < 2.2e-16

(Intercept)    ***
log(elevation) ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Spatial Gaussian process
Matern covariance parameters (kappa=0.5)
                    Estimate Lower limit Upper limit
Spatial process var.  0.24925     0.21159      0.2936
Spatial corr. scale  47.84443    40.88098     55.9940
Variance of the nugget effect fixed at 0

Log-likelihood: 15.26632
```

When using the `glgpm` function to fit a Binomial model, we need to specify the Binomial denominators, i.e. the number of tested at a location in this case, through the argument `den`. Other arguments of `glgpm` that are used for carrying out MCML are `control_mcmc`, that defines the setting parameters of the Monte Carlo Markov Chain algorithm, and `par0`, which represents our best guess for the unknown parameters of the Binomial geostatistical model. For more information on how the default values of these two arguments, we refer the reader the to the help page of the `glgpm` function. To understand how to change the default parameters in order to improve the MCML estimation, we advise you to read Section 3.5.2.1. Also, remember that to enable the estimation of the nugget effect, $Z_i$, you should set `nugget=NULL` in the `gp` function, since by default the `glgpm` function excludes $Z_i$ when the out is non-Gaussian.

In the above summary of the model, the interpretation of the regression coefficients is the same as in a standard logistic regression model. Hence, by taking the exponential of the estimate for the regression coefficient $\beta_1$, we could then express the effect for a unit increase in the covariate – in this case the log-transformed elevation – on the odds of disease prevalence. Finally, the interpretation of estimates of the covariance parameters is not affected by the type of distribution used to model the main outcome. From the above summary, we note the larger estimates of $\sigma^2$ than $\tau^2$ which suggest the dominance of spatial variation over the unstructured variation in the unexplained variation of disease prevalence. Finally, the estimate of $\phi$ indicates a relatively large spatial correlation with a *practical range* close to $(58.67 \times 3 \approx)$180km.

### 3.3.2 Example: *Anopheles Gambiae* mosquitoes in Cameroon

In Section 3.1.1.1, we explored the association between elevation and the number of mosquitoes and found that the data showed that on average, the number of "Anopheles Gambiae* (A. Gambiae) decreases for increasing elevation. We now fit a log-linear Poisson geostatistical model, with linear predictor

$$\lambda(x_i) = \beta_0 + \beta_1 d(x_i) + S(x_i) \tag{3.25}$$

where $d(x_i)$ correspond to the elevation in meters at location $x_i$. This time, we first convert the data frame `anopheles` into an `sf` object. This step is optional but recommended as it allows us to simply the code syntax.

```
anopheles <- st_as_sf(anopheles, coords = c("web_x", "web_y"),
                      crs = 3857)
```

We can then use the following script to fit the model in `RiskMap`.

```
set.seed(1)
an_fit <-
glgpm(An.gambiae ~ elevation + gp(),
      data = anopheles,
      family = "poisson", messages = FALSE)

summary(an_fit)
```

```
Call:
glgpm(formula = An.gambiae ~ elevation + gp(), data = anopheles,
    family = "poisson", messages = FALSE)

Poisson geostatistical linear model
Link: canonical
Inverse link function = exp(x)

'Lower limit' and 'Upper limit' are the limits of the 95%
confidence level intervals

Regression coefficients
               Estimate Lower limit Upper limit      StdErr
               z.value   p.value
(Intercept)  3.69384399  3.00784989  4.37983809  0.35000342
10.554 < 2.2e-16
elevation   -0.00283997 -0.00336073 -0.00231922  0.00026569
-10.689 < 2.2e-16

(Intercept) ***
elevation   ***
```

```
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Spatial Gaussian process
Matern covariance parameters (kappa=0.5)
                    Estimate Lower limit Upper limit
Spatial process var.  0.67165     0.59987      0.7520
Spatial corr. scale   4.38649     3.84756      5.0009
Variance of the nugget effect fixed at 0

Log-likelihood: 9.453628
```

The estimate, $\hat{\beta}_1$, for the regression coefficient $\beta_1$ suggests that for a 100 meters drop in elevation, the average number of mosquitoes decreases by about $(100 \times \exp\{100 \times -0.018\} \approx)17\%$. The order of magnitude of the estimates suggest the presence of residual spatial correlation, albeit on a rather small scale.

```
dist_summaries(anopheles, scale_to_km = TRUE)
```

```
$min
[1] 0.3828559

$max
[1] 160.388

$mean
[1] 54.58734

$median
[1] 43.70518
```

Using the `dist_summaries` function, we can see that the average distance is about 55km and the minimum is about 387 meters. The estimate of phi of 4km is thus relatively small compared to the the distances between locations in the data. In Chapter 4, we will further investigate to what extent this small-scale spatial correlation can affect the predictions of the An. Gambiae counts.

### 3.3.3 Using parametric bootstrap for computing confidence intervals

The bootstrap is a widely used resampling technique introduced by Efron (1979) as a general approach to estimating the variability of statistical estimates without relying on strong parametric assumptions. While the classical non-parametric bootstrap resamples directly from the observed data, parametric bootstrap instead relies on simulating new datasets from an assumed

parametric model, using plug-in estimated obtain from the original fit of the model to the data. This simulation-based method is particularly useful for complex models, such as those considered in this book, that yield an intractable likelihood function.

We now describe how to carry out parametric bootstrap for geostatsitical models in `RiskMap`. The use of simulation-based approaches will be more extensively covered in Chapter 4, but, for the purpose of this section, it is first important to understand how data can be simulated from a geostatistical model.

The way we have formulated geostatistical models in Section 1.5 provides a hint as to what step should be followed when simulating from a geostatistical model. Using the same notation of Section 1.5, these are the steps.

1. For a given set of predefined locations $X$ compute the covariance matrix $\Sigma$ with $(i,j)$-th entry $\sigma^2\rho(u_{ij})$.
2. Compute the covariate effects $d(x)^\top\beta$ for all the locations in $X$.
3. Simulate $B$ times from a multivariate Gaussian distribution with mean vector 0 and covariance $\Sigma$ as previously defined. We denote the output from this step as $S_{(j)}(x)$, for $j = 1, \ldots, B$.
4. Compute the linear predictor at each location of $X$, by adding the covariate effects from step 2 to the simulated random effects from step 3, hence $g\{\mu_{(j)}(x)\} = d(x)^\top\beta + S_{(j)}(x)$.

Note that before performing the steps above, the values of the model parameters must be fixed. For example, if simulating from a model fitted to the data, the values of the model parameters should be the maximum likelihood estimates. If the goal is to simulate actual observations for the outcome $Y$ at the locations $X$, then we need to add the following steps.

5A. If the nugget term is included in the model, draw samples $Z_{(j)}$ for each location in $X$ from a Gaussian distribution with mean 0 and variance $\tau^2$. Add the simulated values $Z_{(j)}$ to $g\{\mu_{(j)}(x)\}$.

6A. Compute $\mu_{(j)}(x)$ by applying the inverse link function $g^{-1}(\cdot)$ to the samples of the linear predictor previously computed.

7A. Simulate from the distribution of $Y$ given $\mu_{(j)}(x)$, according to the chosen model, for $j = 1, \ldots, B$.

These steps, from 1 to 7A, have been implemented in the `glgpm_sim` function. To show how to perform parametric bootstrap in `RiskMap`, let us consider the Liberia data on riverblindness and fit the following geostatistical model.

```
fit_liberia_no_nugget <-
  glgpm(npos ~ log(elevation) + gp(long, lat),
        den = ntest, data = liberia,
        crs = 4326,
        convert_to_crs = 32629,
        family = "binomial")
```

Using the fitted Binomial geostatistical above, we can simulate a single dataset as follows:

```
liberia_sim <- glgpm_sim(n_sim = 1, model_fit =
 ↪  fit_liberia_no_nugget)
```

The output `liberia_sim` is a list with as many components as indicated by `n_sim`. In this this case, we can access the simulated data-set as `liberia_sim[[1]]`. For example, we could refit the same model to the simulated data as

```
fit_liberia_no_nugget_sim <-
glgpm(formula = npos ~ log(elevation) + gp(long, lat),
    data = liberia_sim[[1]], family = "binomial", den = ntest,
    crs = 32629)
```

Note that if the data's CRS has been converted, the simulated data-set will inherit the new CRS rather than the original. This means that any transformation applied to the CRS before simulation (e.g., from geographic to projected coordinates) will affect the spatial scale and interpretation of the simulated data.

This idea can be extended naturally to perform parametric bootstrap. The procedure involves simulating multiple data-sets from the fitted model, refitting the model to each of them, and extracting the estimates of interest. For instance, the following code simulates 100 datasets and fits the same model to each, storing the estimated regression coefficients:

```
# Simulate 100 datasets
n_sim <- 100
liberia_boot <- glgpm_sim(n_sim = n_sim, model_fit =
 ↪  fit_liberia_no_nugget)

# List where we will store the estimates from each of the
 ↪  simulated data-sets
par_hat <- list()

for(i in 1:n_sim) {
```

```
  # Extracting the simulated Binomial data into "npos_sim"
  liberia_boot$data_sim$npos_sim <-
    liberia_boot$data_sim[[paste("npos_sim",i,sep="")]]

  # Fitting the geostatistical model
  fit_sim <- glgpm(
    formula = npos_sim ~ log(elevation) + gp(long, lat),
    data = liberia_boot$data_sim,
    family = "binomial",
    par0 = coef(fit_liberia_no_nugget),
    den = ntest,
    crs = 32629
  )

  # Extracting the parameter estimates
  par_hat[[i]] <- coef(fit_sim)

}
```

After obtaining the parameter estimates, we extract them and rearrange them in a matrix format to facilitate the computation of any summary, including the confidence intervals as shown in the code below.

```
# Extract all parameters maintaining beta as a vector
all_params <- sapply(1:n_sim, function(i) {
   c(par_hat[[i]]$beta,
     par_hat[[i]]$sigma2,
     par_hat[[i]]$phi)})


CI <- t(apply(all_params, 1, function(x) quantile(x, c(0.025,
 ↪  0.975))))
p <- 2
rownames(CI)[(p+1):(p+2)] <- c("sigma^2", "phi")

summary(fit_liberia_no_nugget)
## Call:
## glgpm(formula = npos ~ log(elevation) + gp(long, lat), data =
 ↪  liberia,
##     family = "binomial", den = ntest, crs = 4326,
 ↪  convert_to_crs = 32629)
##
## Binomial geostatistical linear model
## Link: canonical (logit)
```

```
## Inverse link function = 1 / (1 + exp(-x))
##
## 'Lower limit' and 'Upper limit' are the limits of the 95%
 ↪  confidence level intervals
##
## Regression coefficients
##                 Estimate Lower limit Upper limit    StdErr
 ↪  z.value   p.value
## (Intercept)    -2.847156   -3.272762   -2.421550  0.217150
 ↪  -13.111 < 2.2e-16
## log(elevation)  0.264436    0.222902    0.305970  0.021191
 ↪  12.479 < 2.2e-16
##
## (Intercept)    ***
## log(elevation) ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' '
 ↪  1
##
## Spatial Gaussian process
## Matern covariance parameters (kappa=0.5)
##                      Estimate Lower limit Upper limit
## Spatial process var.  0.24447     0.21200      0.2819
## Spatial corr. scale  34.04827    29.21324     39.6835
## Variance of the nugget effect fixed at 0
##
## Log-likelihood: 13.75077

CI
##                      2.5%      97.5%
## (Intercept)    -3.6210256 -2.2028113
## log(elevation)  0.1490998  0.4370221
## sigma^2         0.1134865  0.4085383
## phi            14.3625701 80.5595429
```

By comparing the 95% confidence intervals obtained from the `summary` of the fitted model with those derived from the bootstrap approach, we observe that the former are considerably narrower for the covariance parameters, while the intervals are fairly similar for the regression coefficients.

This discrepancy arises because the Wald-type confidence intervals from `summary` rely on a Gaussian approximation of the maximum likelihood estimator for the covariance parameters, which may not be entirely reliable. (To improve the accuracy of this approximation, `RiskMap` applies it to the log-transformed parameters and then transforms the confidence intervals back to

the original scale.) On the other hand, bootstrap-based confidence intervals tend to be more reliable, as they are not dependent on the Gaussian assumption. However, they are also more computationally demanding. The use of only 100 bootstrap samples, as shown above, may indeed be insufficient. It is generally recommended to use at least 1000 iterations, if computationally feasible, to minimize the Monte Carlo error arising from the bootstrap simulations.

## 3.4 The residuals of a geostatistical model: issues to consider

In geostatistical models, the concept of residuals, which are widely used in regression modelling to carry out model diagnostics, becomes more nuanced than in standard regression models (i.e., models that do not include random effects) . Generally, residuals aim to quantify the discrepancy between observed outcomes and what the model predicts. However, in GLGMs, there are multiple valid definitions of residuals, each offering a different perspective on the unexplained variation.

If we define the residuals as the component of the model that expresses the variation in the outcome not explained by the covariates, then we are effectively referring to the random effects component of the linear predictor—i.e. $S(x_i)$, or $S(x_i) + Z_i$ if a nugget effect $Z_i$ is included. We refer to these as *random effects residuals*.

Let $W_i$ denote the sum of all the random effects in the model, whether $W_i = S(x_i)$, $W_i = S(x_i) + Z_i$, or any other random effects structure. In addition to RERs, other types of residuals more common in standard regression modeling can also be used.

One such type is the *Pearson residual*:

$$\text{PR}_i = \frac{y_i - \widehat{\text{E}}\left[Y_i \mid W_i\right]}{\sqrt{\widehat{\text{Var}}\left[Y_i \mid W_i\right]}}$$

where $\widehat{\text{E}}\left[Y_i \mid W_i\right]$ and $\widehat{\text{Var}}\left[Y_i \mid W_i\right]$ are estimates of the conditional expectation and variance of $Y_i$ given $W_i$. See Table 1.3 for the expressions of $\text{E}\left[Y_i \mid W_i\right]\$$ and $\text{Var}\left[Y_i \mid W_i\right]$ for the probability distributions considered in this book.

For example, using Monte Carlo samples from $W_i$, say $W_{i,(j)}$ for $j = 1, \ldots, B$ (see Section 3.5.2.2 for how these samples are generated in `RiskMap`), the expectations can be estimated as:

$$\hat{\mathrm{E}}\left[Y_i \mid W_i\right] = \frac{1}{B}\sum_{j=1}^{B} \hat{\mathrm{E}}\left[Y_i \mid W_{i,(j)}\right], \quad \hat{\mathrm{Var}}\left[Y_i \mid W_i\right] = \frac{1}{B}\sum_{j=1}^{B} \hat{\mathrm{Var}}\left[Y_i \mid W_{i,(j)}\right]$$

An alternative, especially useful when $Y_i$ is binary, is the *deviance residual*:

$$\mathrm{DR}_i = \mathrm{sign}(y_i - \hat{y}_i)\sqrt{2\left(\ell_i^S - \ell_i^F\right)}$$

where $\ell_i^S$ is the log-likelihood of the so-called *saturated model*, and $\ell_i^F$ is the log-likelihood of the fitted model, both conditional on $W_i$. The saturated model is a hypothetical model that fits the data **perfectly**, meaning it has as many parameters as data points. In such a model, the predicted values are exactly equal to the observed values, and therefore the log-likelihood reaches its maximum possible value. This model serves as a benchmark or upper bound against which the fit of any other model, including our fitted model, is compared. Hence, the difference $\ell_i^S - \ell_i^F$ reflects how much worse our model fits the data compared to this idealized scenario.

While the residuals discussed above can provide useful insights into model fit, they are often less informative in the context of count data, especially for binary data and low counts. Moreover, even in the simpler case of linear geostatistical models where all three types of residuals introduced earlier (random effects residuals, Pearson residuals, and deviance residuals) are mathematically equivalent, it is easy to show that the correlation structure of the estimated residuals is related to the spatial correlation matrix of the fitted geostatistical model (see Section 3.5.3). Whether this correlation in the estimated residuals is unimportant or poses a problem, particularly when using residuals to assess whether spatial dependence has been fully accounted for by the model, is still an open question and requires further investigation.

Given these complexities, especially in the context of count data, we have chosen not to delve deeper into residual diagnostics in this edition of the book. Instead, model validation will be addressed more comprehensively in the context of geostatistical prediction in Chapter 4.

Nevertheless, we note that some standard diagnostic tools from regression analysis (such as scatter plots of fitted values versus residuals) can still be informative and are worth exploring as part of routine model assessment (Chapter 9, Weisberg 2014).

## 3.5 Theory

### 3.5.1 Maximum likelihood estimation for non-Guassian mixed models with independent random effects

In this section, we provide more technical details on the derivation of the likelihood function for the mixed model specified in Equation 3.7. While not essential for understanding the application and interpretation of geostatistical models, this section provides useful insights into computationally challenges inherent to the models presented in this book. It is also important to point out that in this section we consider the case when the outcome $Y_i$ is a count.

Let $\theta = (\beta, \tau^2)$ denote the vectore of unkown parameters, where $\beta$ is the vector of regression coefficients and $\tau^2$ is the variance of the random effects $Z = (Z_1, \dots, Z_n)$. The likelihood function for a set of observations $Y = (Y_1, \dots, Y_n)$, for $i = 1, \dots, n$, under the model in Equation 3.7, is obtained by deriving the marginal distribution of $Y_i$, hence

$$L(\theta) = [Y] = \int_{R^n} [Z][Y \mid Z]\, dZ$$

Since the elements $Z_i$ in $Z$ are mutually independent, we can write

$$[Z] = \prod_{i=1}^{n} [Z_i]$$

and, using the conditional independence among the $Y$ given $Z$, we obtain

$$[Y \mid Z] = \prod_{i=1}^{n} [Y_i | Z_i].$$

From these, it then follows that

$$L(\theta) = \int_{R^n} \prod_{i=1}^{n} [Z_i] \prod_{i=1}^{n} [Y_i | Z_i]\, dZ = \prod_{i=1}^{n} \int_{R} [Z_i][Y_i \mid Z_i]\, dZ_i. \tag{3.26}$$

This shows that under the model in Equation 3.7, whilst accounting for overdispersion, this model does not account for any source of correlation between the observations. This is because the likelihood function can be expressed, as shown in the equation above, as the product of $n$ likelihoods associated to each of the observations.

Another important observation is that the integrals that express the likelihood in Equation 3.26, cannot be solved analytically and need to be approximated.

Several solution are available that either based on maximum likelihood estimation or Bayesian methods of inference. The function `glmer` used in Section 3.1.3.2 provides both the Laplace and Gaussian quadrature approximations options. The former consists of finding the *peak* of the integrand - i.e. the function that is being integrate - and approximating that with a Gaussian distributioncentered at that *peak*, using the curvature of the original distribution to determine the width of the Gaussian distribution. The Gaussian quadrature is another numerical integration technique used to approximate the integral of a function. It involves choosing specific points (nodes) and weights within the integration interval to approximate the integral. By strategically selecting these points and weights based on a Gaussian distribution, Gaussian quadrature can provide highly accurate results compared to other numerical integration methods, such as the Laplace approximation. For a comprehensive overview of these and other methods of approximations of intractable integrals, we find that Bishop (2006) is one of the most accessible textbooks on this topic.

### 3.5.2 Maximum likelihood estimation for non-Guassian generalized linear geostatistical models

To derive the likelihood function of a generalized linear geostatistical model (GLGM), we consider the special case of a single observation per location without loss of generality. Let $Y$ and $Z$ denote the same vectors of random variables introduced in the previous section, with components $Y_i$ and $Z_i$ corresponding to the set of locations $X = (x_1, ..., x_n)$. Additionally, we define $S(X) = (S(x_1), ..., S(x_n))$ to represent the spatial Gaussian process at the sampled locations $X$. In this case the vector of unknown parameters is $\theta = (\beta, \sigma^2, \phi, \tau^2)$ and, as done in the previous section, we can obtain the likelihood function though the marinal distribution of the data, hence we write

$$L(\theta) = [Y] = \int_{R^n} \int_{R^n} [Z][S][Y \mid Z, S] \, dZdS \tag{3.27}$$

To simplify the explanation and notation in this section and the following ones, we will first rewrite the likelihood function in terms of the random effect $ W_i = S(x_i) + Z_i $ and will denote this in its vector form as $W = (W_1, ..., W_n)$. Hence we write

$$L(\theta) = [Y] = \int_{R^n} [W][Y \mid W] \, dW \tag{3.28}$$

In this model formulation, we assume that $Y$ conditionally on $W$ is a set of mutually independent outcomes, hence

$$[Y|W] = \prod_{i=1}^{n} [Y_i|W_i].$$

However, unlike in the previous section, the distribution of $[W]$ is a Multivariate Gaussian distribution with correlated components, hence we cannot split

the integral into separate univariate ones as before. More specifically, $W$ follows a Multivariate Gaussian distribution with mean vector 0 and covariance matrix $\Omega$, whose $(i, j)$-th entry is

$$[\Omega]_{ij} = \begin{cases} \sigma^2 \rho(u_{ij}; \phi, \kappa) & \text{if } i \neq j \\ \sigma^2 + \tau^2 & \text{if } i = j \end{cases}$$

where $\rho(u; \phi, \kappa)$ is the Matern correlation function with scale parameter $\phi$ and smoothness parameter $\kappa$; see Equation 1.5.

In the next two sections we provide a brief overview of Monte Carlo maximum likelihood and the use of Markov chain Monte Carlo to approximate Equation 3.28.

#### 3.5.2.1 Monte Carlo maximum likelihood

In maximum likelihood estimation, we aim to maximize the likelihood function to find the parameters that make our data the most likely to have been observed. Monte Carlo Maximum Likelihood (MCML) is a method introduced by Charles J. Geyer (1991), designed to handle situations where the likelihood function is difficult or impossible to compute directly. As shown above, non-Gaussian generalized linear geostatistical models yield an intractable likelihood function that involves a high-dimensional integral, hence we use MCML to approximate this.

MCML solves this problem using Monte Carlo methods, which involve random sampling to approximate the value of these integrals. One way to apply Monte Carlo methods in this context is through *importance sampling*. Importance sampling allows us to approximate the likelihood function by rewriting the likelihood in a form that can be estimated using samples from an easier distribution, known as the *importance sampling distribution*. To this end, we then rewrite Equation 3.28 as

$$\begin{aligned} L(\theta) &= \int [W, Y; \theta] \, dW \\ &= \int \frac{[W, Y; \theta]}{[W, Y; \theta_0]} [W, Y; \theta_0] \, dW \\ &\propto \int \frac{[W, Y; \theta]}{[W, Y; \theta_0]} [W|Y; \theta_0] \, dW \\ &= E\left\{ \frac{[W, Y; \theta]}{[W, Y; \theta_0]} \right\}, \end{aligned} \tag{3.29}$$

where $\theta_0$ represents our best guess of $\theta_0$ and the expectation in the equation above is taken with respect to $[W|Y; \theta_0]$, which represent our *importance sampling distribution*.

Let $W_{(j)}$ represent the $j$-th sample out of $B$ drawn from the distribution $[W \mid Y; \theta_0]$. To simulate samples from $[W \mid Y; \theta_0]$, Markov chain Monte Carlo

(MCMC) algorithms are required and we explain more in detail in the next section.

We approximate the expression in Equation 3.29 as follows:

$$L_{MC}(\theta) = \frac{1}{B}\sum_{j=1}^{B}\frac{[W_{(j)}, Y; \theta]}{[W_{(j)}, Y; \theta_0]}. \tag{3.30}$$

As $B \to \infty$, the Monte Carlo likelihood $L_{MC}$ converges to the exact likelihood function described in Equation 3.29. The maximization of this Monte Carlo likelihood is performed using an optimization algorithm that uses the analytical expressions of the first and second derivatives. This approach is especially useful for exploring the likelihood surface, which can be relatively flat, particularly in the space of covariance parameters. To improve the approximation of the likelihood function in equation Equation 3.30, this process can be repeated iteratively, replacing $\theta_0$ after each step with the estimate obtained from equation Equation 3.30, until convergence is achieved. However, due to Monte Carlo error, the estimates after each iteration may still vary considerably, making convergence difficult to reach if very strict criteria are used.

For spatial prediction, we take a pragmatic approach by running an initial MCML iteration, followed by a second iteration with 10,000 samples, without further iterations. This is because spatial prediction is generally less affected by Monte Carlo error, as will be demonstrated later. On the other hand, when the goal is to draw inferences about model parameters, more iterations may be needed to reduce Monte Carlo error to acceptable levels.

#### 3.5.2.2 Monte Carlo Markov Chain: sampling from the predictive distribution of the spatial Gaussian process

Building on the outline of the MCML method from the previous section, we now address the task of sampling from the distribution $[W \mid Y]$ to generate the samples $W_{(j)}$ used in equation Equation 3.30. To simplify the notation, we will use $[W \mid Y]$\$ in place of $[W \mid Y; \theta_0]$, but it is important to emphasize that the parameters $\theta_0$ are considered fixed when sampling from $[W \mid Y]$. Since direct simulation from $[W \mid Y]$ is not feasible, we must rely on Markov chain Monte Carlo (MCMC) methods, which we briefly introduce next.

The *Metropolis-Hastings algorithm* is a foundational technique in the field of MCMC methods, which are widely used for sampling from complex probability distributions. The primary goal of MCMC methods is to generate a sequence of samples from a target distribution where direct sampling is challenging, such as the distribution of $[W|Y]$. The Metropolis-Hastings algorithm, first introduced by Metropolis et al. (1953) and later generalized by Hastings (1970), achieves this by constructing a Markov chain that has $[W\|Y]$\$ as its station-

ary distribution. In what follows, we shall use $\pi(W)$ to denote the density function of $[W|Y]$ up to a constant of proportionality.

Given a current realization $W_{(k)}$ of the distribution $[W|Y]$ in the Markov chain, the algorithm proceeds as follows.

- Propose a new realization $W_{(k+1)}$ from $q(W_{(k+1)} \mid W_{(k)})$, where $q(\cdot \mid \cdot)$ is the density function of a chosen proposal distribution.
- Compute the acceptance probability:

$$\alpha = \min\left(1, \frac{\pi(W_{(k+1)})q(W_{(k)} \mid W_{(k+1)})}{\pi(W_{(k)})q(W_{(k+1)} \mid W_{(k)})}\right).$$

- Accept the proposed realization $W_{(k+1)}$ with probability $\alpha$; otherwise, retain the current realization $W_{(k)}$.

The algorithm ensures that, under certain conditions, the chain converges to the target distribution $\pi(W)$ as the number of steps increases Roberts and Tweedie (1996).

The functions of the `RiskMap` package for fitting Binomial and Poisson geostatistical models implement the *Metropolis-Adjusted Langevin Algorithm (MALA)*, which is an improvement over traditional Metropolis-Hastings MCMC methods. Introduced by Rossky, Doll, and Friedman (1978) and further developed by Roberts and Tweedie (1996), MALA incorporates gradient information from the target distribution to propose more informed moves, potentially leading to faster convergence and more efficient exploration of the target distribution.

MALA combines the Metropolis-Hastings framework with Langevin dynamics, where proposals $q(\cdot|\cdot)$ are defined as:

$$W_{(k+1)} = W_{(k)} + \frac{\epsilon^2}{2}\nabla \log \pi(W_{(k)}) + \epsilon\,\eta,$$

where $\epsilon$ is a step size parameter, $\nabla \log \pi(W_{(k)})$ is the gradient of the log of the target density with respect to $W_{(k)}$, and $\eta \sim \mathcal{N}(0, I)$ is a Gaussian random variable with mean 0 and independent components with unit variance.

By leveraging the gradient, MALA can propose moves that are more likely to be accepted, especially in high-dimensional spaces or regions of low probability density.

In the context of Laplace sampling for geostatistical models, the *Metropolis-Adjusted Langevin Algorithm (MALA)* is used to construct an efficient proposal distribution. The primary objective is to generate samples from a conditional distribution of latent Gaussian processes, which is challenging due to high dimensionality and complex dependence structures.

The Laplace sampling method, as implemented in the `RiskMap` package, utilizes a specific form of MALA. Here, the proposal distribution is derived from a second-order Taylor expansion around the mode of the target distribution [W | Y]. Let $\widehat{w}$ be the mode of this conditional distribution, and let $\widehat{\Omega}$ be the inverse of the negative Hessian matrix evaluated at $\widehat{w}$, which approximates the local covariance structure of the distribution. We then define the random variable $\tilde{W}$ as

$$\tilde{W} = \widehat{\Omega}^{-1/2}(W - \widehat{w})$$

This linear transformation is applied to decorrelate the components of $W$, making the MCMC updates more efficient (Charles J. Geyer 1994). The function `Laplace_sampling_MCMC` in `RiskMap` uses this approach where by at each iteration the variable $\tilde{W}$ using the MALA algorithm, and samples for $[W|Y]$ are then obtained by inverting the linear transformation from the above equation. The logic behind using this approach is that it allows the proposal distribution to adapt to the local geometry of the posterior, thus improving convergence properties and reducing the correlation between successive samples (see O. F. Christensen (2004)).

Once the MCMC algorithm has completed all iterations, it is important to ensure that it has converged to the target distribution. The diagnostic methods listed below can be used to assess convergence using the obtained MCMC samples.

1. **Trace Plots**: Plotting the sampled values against the iterations can help visually inspect if the chains have reached a stationary distribution. A well-mixed chain should appear as a "random walk" around a stable mean value.
2. **Autocorrelation Plots**: These plots measure the correlation between samples as a function of the lag between them. Low autocorrelation across lags indicates good mixing, suggesting convergence.
3. **Effective Sample Size (ESS)**: The ESS measures the number of independent samples that an MCMC chain is equivalent to, accounting for the autocorrelation in the chain. It is calculated as:

   $$\text{ESS} = \frac{N}{1 + 2\sum_{k=1}^{\infty} \rho_k},$$

   where $N$ is the total number of MCMC samples and $\rho_k$ is the autocorrelation at lag $k$. A high ESS indicates that the chain has good mixing properties and the samples are nearly independent. The summation of autocorrelations up to a sufficiently large lag is used to estimate the reduction in effective sample size due to the dependence between samples. A small ESS suggests high autocorrelation

and that more samples are needed to achieve the same precision as a set of independent samples.

A high ESS (Effective Sample Size) indicates that the chain has good mixing properties, meaning the samples are nearly independent. The summation of autocorrelations up to a sufficiently large lag is used to estimate the reduction in effective sample size due to the dependence between samples. A small ESS suggests high autocorrelation, meaning more samples are needed to achieve the same precision as a set of independent samples. In addition to those, other diagnostics can provide further assurance of convergence. For example, the Gelman-Rubin diagnostic compares the variance between multiple chains to the variance within each chain; a diagnostic value close to 1 indicates that the chains have converged to a common distribution (Gelman and Rubin 1992). Also, the Geweke diagnostic tests for convergence compares the means of the first part and last part of a single MCMC chain. If the means are not significantly different, it suggests that the chain has reached stationarity (Geweke 1992). Using a combination of these diagnostics can be used to assess whether the MCMC algorithm has adequately explored the target distribution.

It is important to point out that in MCMC algorithms, to the quality of samples we need to carefully considers these three control parameters of the MCMC.

- **Number of Simulations**: Increasing the number of simulations allows the algorithm to explore the posterior distribution more thoroughly, reducing the variability in estimates. A higher number of simulations typically leads to more accurate results but also requires more computational time.
- **Burn-in**: The burn-in phase refers to the initial set of iterations where the Markov chain has not yet converged to its stationary distribution. These initial samples are discarded to ensure that the remaining samples represent the target distribution more accurately. Properly setting the burn-in period helps avoid bias due to initial conditions.
- **Thinning**: Thinning reduces autocorrelation by only keeping every $n$-th sample from the chain. While this decreases autocorrelation between retained samples, it also reduces the overall sample size. Thinning is used when chains exhibit high autocorrelation to ensure a more independent set of samples is analyzed. Initially, these values can be set based on prior experience or general rules of thumb (e.g., a burn-in of 10-20% of the total iterations). In the next summary section, we provide more details on specific implementation of these in `RiskMap`.

For a more thorough and complete overview of MCMC algorithms, chapters

10 to 13 of Gelman and Rubin (1992) are a recommended read.

#### 3.5.2.3 Summary with example

In summary, here are the steps required to perform Monte Carlo Maximum Likelihood (MCML):

1. Start by selecting an initial guess for the model parameters, denoted as $\theta_0$.
2. Simulate Samples from $[W \mid Y; \theta_0]$ using MCMC methods. Perform a check on the MCMC samples to assess convergence of the MCMC algorithm.
3. Compute and maximize the approximate likelihood function in Equation 3.30.
4. Set the newly estimated parameter as $\theta_0$ for the next iteration.
5. Repeat 1 to 4 until convergence is achieved.
6. Monitor the estimates for convergence. If strict convergence criteria are set, Monte Carlo error may cause fluctuations, making it harder to achieve convergence.
7. Perform additional iterations, if necessary. For inference on model parameters, continue iterating to reduce Monte Carlo error. For spatial prediction, a pragmatic approach may involve stopping after a second iteration with 10,000 samples, as prediction is less affected by Monte Carlo error.

In `RiskMap`, steps 1 to 3 are automated (with the exception of the check on the convervence of the MCMC algorithm in step 2), while steps 4 to 7 require manual implementation. In Section 3.3.1, estimation is performed using a geostatistical model fitted to prevalence data. The `par0` argument in the `glgpm` function allows you to specify initial guesses for the model parameters $\theta_0$. If no values are provided for `par0` (as in Section 3.3.1), the following defaults are used:

- The initial guess for the regression coefficients $\beta$ is set to the maximum likelihood estimates obtained from a standard generalized linear model (GLM), i.e., a GLM without random effects or overdispersion.
- The initial values for both $\sigma^2$ (the variance of the spatial Gaussian process) and $\tau^2$ (the variance of the nugget term, if included) are set to 1.
- The initial guess for the scale parameter of the spatial correlation function $\phi$ is based on the 0.1 quantile of the distances between all pairs of locations.

In our experience, this set of initial guesses works reasonably well. However, more iterations of the MCMC algorithm, as suggested in the step 7 above, are

generally recommended.

An alternative approach for setting initial guesses for the covariance parameters $\sigma^2$, $\tau^2$, and $\phi$ is to apply an interpolation technique to the variogram, as proposed in the seminal work by N. Cressie (1985). However, in our experience, this approach does not consistently provide better approximations of the likelihood function compared to the method mentioned above. This may be due to the instability in estimating the variogram based on residuals from mixed models, as well as the sensitivity of empirical variogram estimation to the chosen distance bins.

To better illustrate some of the points above, we continue the analysis of the Liberia data from Section 3.3.1. Diagnostic summaries on the convergence of the MCMC algorithm can be obtained with the `check_mcmc` function by passing the fitting model `fit_liberia` to the function.

```
check_mcmc(fit_liberia)
```

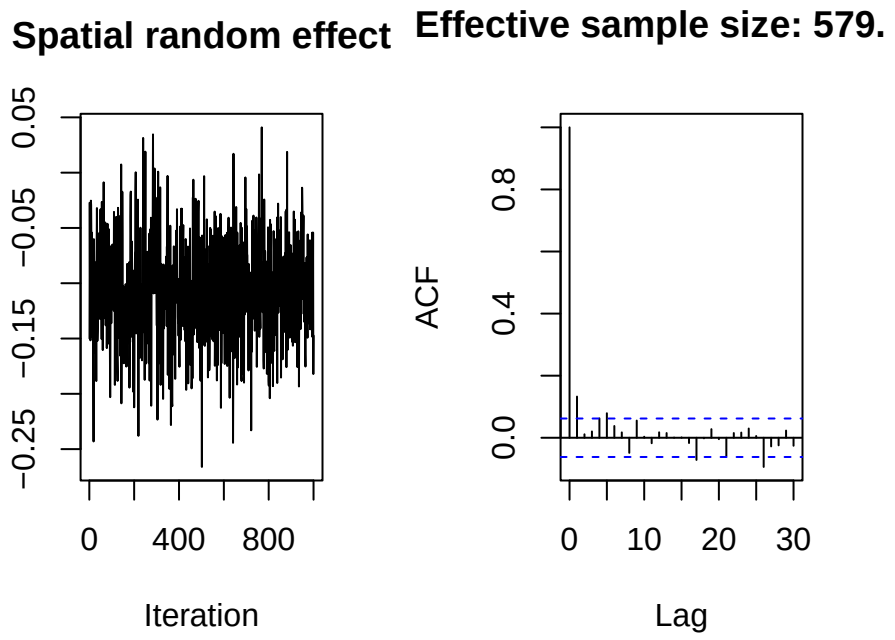



Figure 3.12: Trace plot (left panel) and autocorrelation function plot (right panel) for the averaged spatial random effects from the river-blindness data analysis. The title of the right panel also shows the effective sample size of the 1000 Markov chain Monte Carlo (MCMC) samples. The MCMC samples are obtained by running the MCMC algorithm for 12000 iterations, with a burn-in of 2000 and thinning of 10.

The plot shown in Figure 3.12 are for the MCMC samples of the averaged spatial random effect, that is $\bar{S} = \sum_{i=1}^{n} S(x_i)/n$. Performing the check on $\bar{S}$ is done to avoid the need of checking the convergence for each individual component $S(x_i)$. However, one of the risks of doing this is that if fewer components of $\bar{S}$ have not reached convergence this may not be shown through the inspection of the samples for $\bar{S}$. However, this is an unlikely scenario when using the type of MCMC algorithm implemented in `RiskMap` (see Section 3.5.2.2). Nonetheless, if one wants to check the convergence for individual components this can be done as follows.

```
check_mcmc(fit_liberia, check_mean = FALSE, component = 10)
```

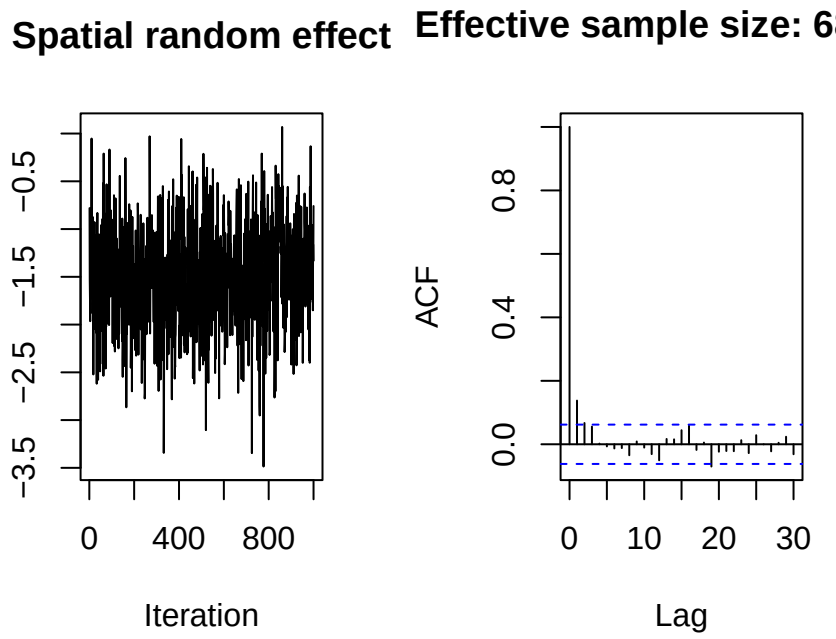



Figure 3.13: Trace plot (left panel) and autocorrelation function plot (right panel) for the tenth component of the spatial random effects from the river-blindness data analysis.

Both Figure 3.12 and Figure 3.13, shows a reasonably good mixing and relatively low correlation in the samples. We point out that the number of MCMC samples in `glgpm` are set through the function `set_control_sim` passed to the argument `control_mcmc`. By default `set_control_sim` uses 12000 iterations, a burn-in of 2000 and thinning of 10, thus giving 1000 MCMC samples. This configuration can be enough for the first run of the Monte Carlo maximum Likelihood (MCML) method, however it is advisable to increase the number of iterations to obtain a larger number of samples with reduced correlation.

```
# Update the guesses for the model parameters
par0_liberia <- coef(fit_liberia)

# Refit the model and increase the MCMC samples
fit_liberia2 <-
glgpm(npos ~ log(elevation) + gp(long, lat, nugget = NULL),
      den = ntest, data = liberia,
      crs = 4326,
      convert_to_crs = 32629,
      par0 = par0_liberia,
      control_mcmc = set_control_sim(n_sim = 110000,
                                     burnin = 10000,
                                     thin = 10),
      family = "binomial", messages = FALSE)
```

```
check_mcmc(fit_liberia2)
```

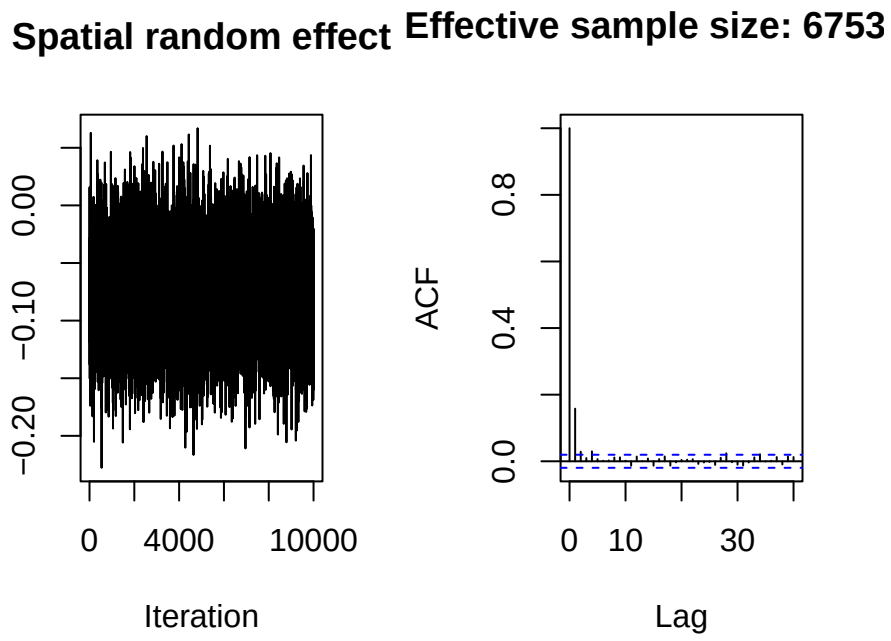



Figure 3.14: Trace plot (left panel) and autocorrelation function plot (right panel) for the tenth component of the spatial random effects from the river-blindness data analysis. The Markov chain Monte Carlo (MCMC) samples are obtained by running the MCMC algorithm for 110000 iterations, with a burn-in of 10000 and thinning of 10.

The code above performs an additional step in the MCML estimation, using 10,000 MCMC samples. As shown in Figure 3.14, these MCMC samples exhibit good mixing and low autocorrelation, which indicates that they adequately approximate the likelihood function of the Binomial geostatistical model. Although additional MCMC iterations could be run to obtain estimates closer to the true maximum likelihood estimate (since we are approximating the likelihood function and not obtaining the exact MLE), we choose not to do so here. This decision is due to both the computational time required and the minimal impact of Monte Carlo error on spatial predictions in this case.

#### 3.5.2.4 Alternative approaches

In the sections above, we focused specifically on the implementation of the Monte Carlo Maximum Likelihood (MCML) method in `RiskMap`. However, other approaches are available, some of which are computationally more efficient than the MCML method we have illustrated. Our preference for this estimation method is based on two main reasons: 1) it delivers robust performance across a wide range of data scenarios, and 2) it does not require the specification of arbitrary prior distributions when prior information on the model parameters is unavailable.

Alternative methods worth mentioning include the expectation-maximization algorithm for maximum likelihood estimation (Zhang 2002), integrated nested Laplace approximation (INLA) (Rue, Martino, and Chopin 2009), and implementations of the Metropolis-Adjusted Langevin Algorithm (MALA) for Bayesian inference (O. Christensen, Roberts, and Sköld 2006). While these methods are valuable, explaining and comparing them is not our focus here, as it would detract from the practical orientation of this book.

### 3.5.3 The Hat matrix in linear geostatistical models

We consider the geostatistical model

$$Y_i = d(x_i)^\top \beta + S(x_i) + Z_i$$

where $ Y_i $ is the observed response at location $ x_i $, $ d(x_i) $ is a $ p $-dimensional vector of covariates, $  $ is a $ p \times 1 $ vector of regression coefficients, $ S(x_i) $ is a spatially structured random effect, and $ Z_i  N(0,  ^2) $ is independent Gaussian noise. The model can be rewritten as

$$Y = D\beta + S + Z$$

where $ Y = (Y_1, ..., Y_n)^ $ is the $ n \times 1 $ vector of observations, $ S = (S(x_1), ..., S(x_n))^ $ is the vector of spatially dependent random effects,

$Z = (Z_1, \dots, Z_n)$ is the noise, and $ D $ is an $ n \times p $ matrix of covariates given by

$$D = (d_1(X), \dots, d_p(X))$$

with each column $ d_j(X) = (d_j(x_1),\dots,d_j(x_n))^\top $ representing the values of the $ j $-th covariate across all locations. This formulation accounts for both fixed effects through $ D $ and spatial correlation through $ S(x_i) $. The covariance matrix of $ Y $ is

$$V = \Sigma + \tau^2 I$$

where $ $ is the spatial covariance matrix of $S$. Assuming the covariance parameters are known, the generalized least squares estimator for $ $ is

$$\hat{\beta} = (D^T V^{-1} D)^{-1} D^T V^{-1} Y.$$

The predictor for $S$ is the the conditional expectation $E[S(x_i) \mid Y_i]$ is

$$\hat{S} = \Sigma V^{-1} (Y - D\hat{\beta}).$$

so the predictors for $Y$ can be written as

$$\hat{Y} = D\hat{\beta} + \hat{S}.$$

The hat matrix, which projects observations onto the fitted space, is

$$H = [D - \Sigma V^{-1} D](D^T V^{-1} D)^{-1} D^T V^{-1} + \Sigma V^{-1}.$$

This matrix is symmetric since it consists of symmetric components, satisfying $H^T = H$. It is also idempotent, meaning $H^2 = H$, which follows from the properties of projection matrices. The diagonal elements $h_{ii}$ represent the leverage of each observation, quantifying how much each $ Y_i $ influences its own fitted value. The rank of $ H $ is at most $ p $, constrained by the rank of $ D $. Unlike in ordinary least squares where the hat matrix is $H = D(D^T D)^{-1} D^T$, here it depends on $V^{-1}$, accounting for spatial dependence. Since $V \neq I$, the hat matrix represents an oblique projection rather than an orthogonal one.

The residuals, i.e. the estimate of $Z$ are given by

$$\hat{Z} = Y - \hat{Y} = (I - H)Y.$$

We not thet the covariance structure of $\hat{Z}$ is

$$\text{Var}(\hat{Z}) = (I - H)V(I - H)^T.$$

Hence, the estimated residuals from a linear geostatistical model do have a correlation structure which relates to the chosen spatial correlation structure in $\Sigma$.

## 3.6 Review questions

- Describe how you would perform exploratory analysis between counts data and covariates, distnsuighing between unbounded counts (e.g. Poisson), bounded counts (e.g. Binomial) and binary outcomes.
- Define the theoretical and empirical variogram, making appropriate use of equations.
- How do we use the empirical variogram to test for spatial correlation? What are the limitations of this approach?
- Define the class of generalized linear geostatistical models and list all the assumptions.
- Explain intuitively what are the computational challenges related to the fit of geostatistical models and what approaches can be used for parameter estimation.

## 3.7 Exercises

1. Consider the Binomial mixed model with linear predictor as defined in Equation 3.7. By editing the code for the simulation shown in Section 3.1.2, generate a graph as in Figure 3.6 under the two following scenarios: *i*) $\tau^2 = 0.2$ and $n_i = 100$; *ii*) $\tau^2 = 0.1$ and $n_i = 1$. How does the variance of $Y_i$ change under *i*) and *ii*) in comparison to Figure 3.6? How do you explain the differences?
2. Similarly to the previous exercise, consider a Poisson mixed model with linear predictor

$$\log\{\mu_i\} = \beta_0 + Z_i,$$

where $Z_i$ are a set of mutually independent Gaussian variables with mean 0 and variance $\tau^2$. Using the code shown in Section 3.1.2, carry out a simulation study to compute the variance of $Y_i$ and generate a graph similar to Figure 3.6 to compare the variance of the Poisson mixed model with that of a standard Poisson model. Generate the graph for different values of $\tau^2$ and summarize your findings. NOTE: In this simulation the offset $n_i$ can be set to 1.

3. Create an R function that computes the *cloud variogram*. As explained in Section 3.1.3, the cloud variogram is obtained by plotting $\widehat{V}_{ij}$ (see Equation 3.11) against the distances $u_{ij}$. The function should take as input a data-set with three columns: the variable for which the cloud variogram is to be computed; and two columns corresponding to the location of the data. Then, use this function to create the cloud variogram for the model for river-blindness in Equation 3.13. How does this compare to empirical variogram that takes the averages within predefined distance classes, as shown in Figure 3.7?

4. Fit a Binomial mixed model to the Liberia data-set on river-blindness without any covariates, i.e.

$$\log\left\{\frac{p(x_i)}{1-p(x_i)}\right\} = \beta_0 + Z_i.$$

Making use of the R code presented in Section 3.1.3.1, use the function `s_variogram` to generate the empirical variogram for this model and compare this to the empirical variogram of Figure 3.7. What differences do you observe?

5. Fit a Poisson mixed model to the anopheles mosquito data using elevation as a covariate, i.e.

$$\log\{\mu(x_i)\} = \beta_0 + \beta_1 d(x_i) + Z_i$$

where $d(x_i)$ is the measured elevation at location $x_i$. How strong is the overdispersion in the data? After fitting the model, extract the estimates of the random effects $Z_i$ and compute the empirical variogram with the 95% envelope for spatial independence. Repeat this for different classes of distances by changing the input passed to `bins` in the `s_variogram` function. How do the different specifications of `bins` affect the results?

6. *Parametric bootstrap.* Using the model fitted to the Anopheles gambiae mosquitoes in Section 3.3.2, simulate 1000 data-sets using the `glgpm_sim` function and estimate the model to each of the simulated data-set. Use the resulting 1000 estimates to compute 95% confidence intervals for each of the model parameters and compare

this with the confidence intervals obtained from the model summary.

# 4

# *Geostatistical prediction*

**List of the main functions used in the chapter**

| Function | R Package | Used for |
|---|---|---|
| `pred_over_grid` | `RiskMap` | Predicting the different component of the linear predictor over a grid: the covariates effects and the random effects |
| `pred_target_grid` | `RiskMap` | Predictions of target defined at pixel-level |
| `pred_target_shp` | `RiskMap` | Predictions of targets defined at areal level |
| `assess_pp` | `RiskMap` | Assessing the predictive performance of geostatistical models using cross-validation |
| `surf_sim` | `RiskMap` | Simulating surfaces and data from geostatistical models |
| `assess_sim` | `RiskMap` | Assessing the predictive performance of geostatistical models using simulations |

## 4.1 Introduction

Geostatistics was originally developed and applied as a predictive tool for improving mining operations and planning. In the seminal paper on model-based geostatistics (MBG) (P. J. Diggle, Tawn, and Moyeed 1998), the authors begin by stating:

*"Conventional geostatistical methodology solves the problem of predicting the realized value of a linear functional of a Gaussian spatial stochastic

process* $S(x)$ based on observations $Y_i = S(x_i) + Z_i$ at sampling locations $x_i$, where the $Z_i$ are mutually independent, zero-mean Gaussian random variables."

It is no coincidence that, although MBG represents a broad class of statistical models applicable to both estimation and prediction problems, the majority of its applications have focused on prediction. This is evident in many scientific fields where MBG has been used, including global public health, which is the primary application focus in this book. In line with the quote above, for most of the examples considered in this book, the analyses we illustrate aim to use a finite set of locations $x_i$ — typically corresponding to households or villages — to make inferences about an observed disease risk surface. Having performed estimation of MBG models in Chapter 3, we can now carry out spatial prediction to answer critical public health questions, such as identifying hot-spots and cold-spots — areas of unusually high or low disease risk, respectively — or determining whether the average prevalence across a region exceeds a predefined policy threshold.

In this chapter, we will explore these issues, beginning with a formal statistical formulation of the spatial prediction problem and a description of the necessary steps for its implementation. We will then demonstrate how spatial prediction is performed when considering predictive targets at both the *pixel-level* and *areal-level*. Finally, we will conclude the chapter by presenting methods for comparing the predictive performance of different models, with some insights into their relative strengths and weaknesses.

## 4.2 Spatial prediction using geostatistical models

Let us consider the class of generalized linear geostatistical models (GLGM) as in Section 1.5. In the formulation of prediction problems, the first step consists of defining our predictive target, which we denote as $T(x)$ where $x$ denotes any locations within the study, which is usually not part of the sampled data. For example in the analysis of the riverblindness in Liberia, our interest lies in identifying areas where disease prevalence exceeds 20%, a threshold that use been used in the past to identify areas in need of mass drug administration due to the potential for significant disease burden and transmission (Amazigo 2008). Hence, in this case, disease prevalence corresponds to our predictive target $T(x)$. To draw inferences on $T(x)$ and allow us to answer the question of where $T(x)$ exceeds a 20% threshold, we first need to obtain the so called *predictive distribution* of $T(x)$. More specifically, the *predictive distribution* of $T(x)$ is the distribution of $T(x)$ conditioned to the data, which we denote by $[T(x) \mid y]$. More importantly, by incorporating the information provided by the data, the predictive distribution of the target enables us to quantify

the uncertainty stemming from the stochastic nature of the process we are modeling. In other words, the predictive distribution reflects the range of possible values that the target $T(x)$ can take and their likelihoods based on the model we have fitted to the data. As we shall see in our examples, we can use the predictive distribution of $T(x)$ both to provide what is our "best guess" for $T(x)$ and a summary of uncertainty which quantifies how much concentrated is the predictive distribution around that guess. However, providing a single "best guess" for $T(x)$ is not always the answer to our research question. In the example on riverblindness mentioned at the start of the this section, a more natural way to use the predictive distribution would be to compute the likelihood that $T(x)$ exceeds 0.2 at any given location $x$.

In summary, the first two steps of spatial prediction are:

- Step 1. define the predictive target $T(x)$;
- Step 2. obtain the predictive distribution of $T(x)$ and use this to compute summaries that helps to answer your original research question.

This begs two practical questions: "How do we obtain the predictive distribution of $T(x)$?" and "How do we use the predictive distribution to compute our summaries?". For a mathematical derivation of the predictive distribution and the form that this takes in the special case of a linear geostatistical model, you can read Section 4.7. In what follows, we shall assume that we can simulate directly from $[T(x) \mid y]$ either using direct simulation or Markov Chains Monte Carlo (see Section 3.5.2.2).

In our exposition so far, we have also made the implicit assumption that prediction is required for our predictive target at a single, unsampled, location $x$ and for this reason we have denote our predictive target simply as $T(x)$. However, this is almost never the case, since in most cases, our predictive target correspond to either of the following.

- **Spatially continuous targets** correspond to an unobserved spatially continuous surface (e.g. disease prevalence), formally denoted by $\mathcal{T} = \{T(x), x \in A\}$, where $A$ is our region of interest.
- **Areal-level targets**, usually defined as transformation of the spatially continuous surface $\mathcal{T}$, defined previously, and which we denote as $\mathcal{T}_A = F(\mathcal{T}, A)$. For example, the average of $T(x)$ over $A$ is an areal-level target, formally defined as $F(\mathcal{T}, A) = \int_A T(x)\, dx/|A|$, where $|A|$ is the area of $A$.

Before we consider these two types of prediction targets more in detail, we first explain the preliminary steps that need to be performed and are common to both. Whether our goal is predict a spatially continuous or areal-level target, the initial task is to draw Monte Carlo samples from the predictive distribution of the spatial process $S(x)$[1]. Next, we define a set of prediction locations,

[1]Here, we are overlooking the fact that when $T(x)$ is a linear transformation of $S(x)$

say $q$ in total, denoted as $\tilde{X} = \tilde{x}_1, ..., \tilde{x}_q$. These locations correspond to a regular grid that spans our region of interest, $A$. the predictive distribution of the spatial process over the grid, represented as $[S(\tilde{X}) \mid y]$. This distribution allows us to generate Monte Carlo samples for $S(\tilde{X})$, which will later be used for computing our target.

At this stage, we need to consider which of the two following types of predictions from $[S(\tilde{X}) \mid y]$ are required.

- **Marginal predictions** are obtained by simulating independently from the $q$ marginal predictive distributions of $[S(\tilde{X})|y]$. In other words, we consider a prediction location $\tilde{x}_j$, simulate from $[S(\tilde{x}_j) \mid y]$ say $B$ samples, and repeat this for $j = 1, ..., q$
- **Joint predictions** are obtained by simulating from the joint distribution of $[S(\tilde{X}) \mid y]$. Unlike marginal predictions, joint predictions take into account the correlation between the different components of $[S(\tilde{X}) \mid y]$.

To better understand the technical differences between marginal and joint predictions, we refer you to Section 4.7.1. However, in practice, the two following observations are important.

O1. Joint predictions are computationally more intensive than marginal predictions.

O2. Joint predictions are required when the predictive target is at areal-level.

O3. For spatially continuous targets, both marginal and joint predictions can be used but, in light of O1, we might prefer using the former.

### 4.2.1 Generating samples from the predictive distribution of $S(\tilde{X})$ (continue from Section 3.3.1)

We now show how the ideas and concepts discussed above are put into practice in an geostatistical analysis, using the Liberia data example. Using the fitted model in Section 3.3.1, we use the `predict_over_grid` function in `RiskMap`, to sample from the predictive distribution of the spatial Gaussian process $S(x)$ based on a regular grid spanning the country of Liberia.

The `pred_over_grid` function enables us to predict the the separate effects of the covariates, given by $d(x)^\top \beta$, and the spatial random effects, $S(x)$, at any desired location $x$. Hence, before we can use the `pred_over_grid` function, we need to do 1) obtain a shape file that defines the boundaries of our study

and the the model fitted to the data is a linear geostatistical model, then we can derive any summary of the predictive distribution analytically, without the need of carrying out any simulation. Although this may displease some of our fellow statisticians who care about computational efficiency, for pedagogical reasons, here we have chosen to explain only the Monte Carlo based approach, which works in all scenarios.

area, 2) use that to create a regular grid and 3) extract the covariates values at those locations. The R script below performs all these steps.

```
# 1) Obtaining Liberia boundaries
library(rgeoboundaries)
liberia_adm0 <- geoboundaries("liberia", adm_lvl = "adm0")
liberia_adm0 <- st_transform(liberia_adm0, crs = 32629)

# 2) Create the grid at 5 km resolution
liberia_grid <- create_grid(liberia_adm0, spat_res = 5)

# Download elevation data
library(elevatr)
liberia_elev <- get_elev_raster(locations = liberia_adm0,
                                z = 5, clip = "locations")

# 3) Extracting elevation at the grid locations
lb_predictors <- data.frame(elevation =
                         terra::extract(liberia_elev,
                                st_coordinates(liberia_grid)))
```

In the code above, we generated a prediction grid — which, we recall, that we denote as $\tilde{X}$ — with a spatial resolution of 5 km. Generally, a higher spatial resolution (note that a "higher spatial resolution" corresponds to a lower value for `spat_res`) yields more detailed prediction maps but also increases the computational cost. In this case, the choice of a 5 km grid strikes a balance between sufficient visualization clarity and manageable computation time.

At this point, we have all the necessary inputs to execute the `pred_over_grid` function for both marginal and joint predictions, as demonstrated below.

```
# Predicting the spatial process S(x) over the grid, as well as
# covariates effects

# Marginal predictions
lb_pred_S_m <- pred_over_grid(fit_liberia,
                              grid_pred = liberia_grid,
                predictors = lb_predictors, messages = FALSE,
                type = "marginal")

# Joint predictions
lb_pred_S_j <- pred_over_grid(fit_liberia,
                              grid_pred = liberia_grid,
                predictors = lb_predictors, messages = FALSE,
                type = "joint")
```

Both `lb_pred_S_m` and `lb_pred_S_j` contain lists with the same arguments, with one key difference: the samples in the list element named `S_samples` for the former are drawn from the marginal predictive distributions of the components of $S(\tilde{X})$, while for the latter, they come from the joint predictive distribution of $S(\tilde{X})$.

An important note is that in the code above we have used the default settings for the Markov Chain Monte Carlo (MCMC). To change these setting use the function `set_control_sim` whose output should be passed to the argument `contol_sim` of `pred_over_grid`. The diagnostic on the convergence of the MCMC can also be applying the `check_mcmc` function to either `lb_pred_S_m` or `lb_pred_S_j`; see Section 3.5.2.3 for more details.

We can inspect the predictive distribution of any component $S(\tilde{X})$ through the Monte Carlo samples stored in `S_samples`. For example, the script below generates the histogram for the predictive distribution of the first component of $S(\tilde{X})$ (Figure 4.1).

```
hist(lb_pred_S_m$S_samples[,1], main = "Predictive
 ↪  distribution",
     xlab = expression(S(tilde(x)[1])))
```

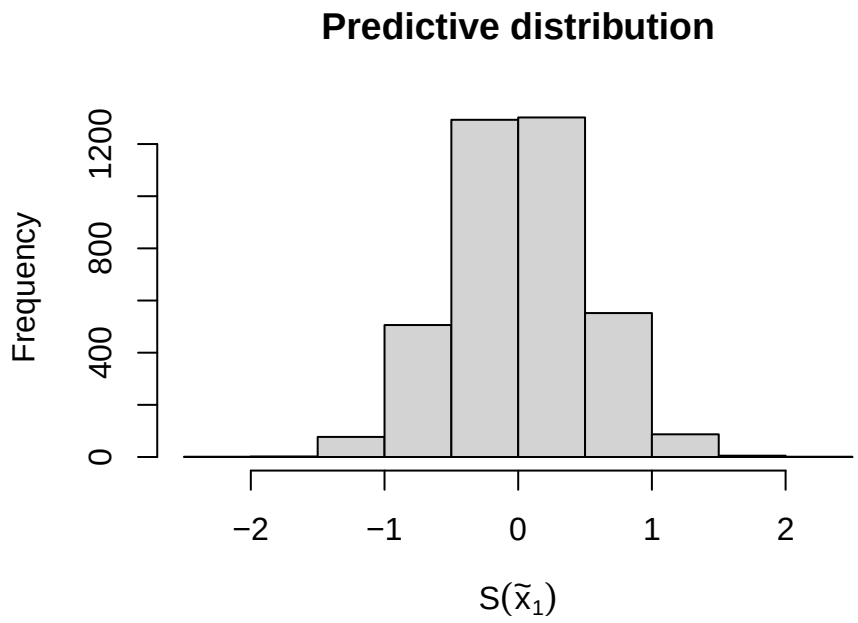



Figure 4.1: Histrogram of the predictive distribution of the first component, $S(\tilde{x}_1)$ of $S(\tilde{X})$ for the Liberia data analysis.

## 4.3 Spatially continuous targets

In the previous section, we explained and demonstrated how to obtain Monte Carlo samples from the predictive distribution of $S(\tilde{X})$, which we denoted by $[S(\tilde{X}) \mid y]$. Let $S_{(j)}(\tilde{X})$ and $[S(\tilde{X}) \mid y]$ represent the $j$-th Monte Carlo samples for the entire grid and the specific location $x_k$, respectively, for $j = 1, \ldots, B$ and $k = 1, \ldots, q$.

We define our predictive target, $T(x)$, at any given location $x$, as a transformation of $S(x)$. Thus, we express it as $T(x) = f\{S(x)\}$. Note that $f(\cdot)$ can represent any type of non-linear transformation of $S(x)$ and may include covariate effects and other non-structured random effects. A list of common predictive targets frequently used in geostatistical analysis is provided in Table 4.2.

Table 4.2: List of some of the most common spatially continuous predictive targets in a geostatistical analysis. For each target, we also indicate the generalized linear model (GLM) family for which these can be defined. Note that this list not exhaustive and different predictive other than those listed in the table could also be considered.

| Predictive target $T(x)$ | Name of the predictive target | Generalized linear model (GLM) family |
|---|---|---|
| $d(x)^\top\beta + S(x)$ | Linear predictor | Any GLM |
| $\frac{\exp\{d(x)^\top\beta+S(x)\}}{1+\exp\{d(x)^\top\beta+S(x)\}}$ | Prevalence | Binomial |
| $\exp\{d(x)^\top\beta + S(x)\}$ | Mean number of cases | Poisson |
| $S(x)$ | Spatial random effects | Any GLM |
| $d(x)^\top\beta$ | Covariates effects | Any GLM |

An important feature of the predictive targets listed in Table 4.2 is that none of them include the nugget effect, denoted by $Z_i$ in our linear predictor definition for generalized linear geostatistical models. The reason for excluding $Z_i$ from the predictive targets is that, in most cases, it lacks a clear, unambiguous scientific interpretation. However, $Z_i$ might be included, for example, in environmental studies focusing on highly localized phenomena where measurement error or small-scale variability is of direct scientific interest; or in epidemiological studies where inferences at the individual level, rather than the population level, are required, and where $Z_i$ is introduced to account for unexplained individual-level variation. It is also worth noticing, the primary impact of including $Z_i$ in $T(x)$ would be an increase in the uncertainty of

predictive inferences for $T(x)$, with minimal effect (or none at all, if $T(x)$ corresponds to the linear predictor) on the point predictions.

The predictive targets in Table 4.2 corresponding to $d(x)^\top\beta$ or the spatial random effects $S(x)$ might be considered when the goal is to highlight the different contribution to the overall predictive inferences from the measured covariates.

### 4.3.1 Example: mapping riverblidness prevalence in Liberia (continuing from Section 4.2.1)

Let us continue the geostatistical of the riverblindness data in Liberia. The predictive target we consider is prevalence, hence

$$T(x) = \frac{\exp\{\beta_0 + \beta_1 \log\{e(x)\} + S(x)\}}{1 + \exp\{\beta_0 + \beta_1 \log\{e(x)\} + S(x)\}}, \tag{4.1}$$

where $e(x)$ is the elevation in meters at location $x$.

Through the `pred_target_grid` we can use the output in `lb_pred_S_m` (or `lb_pred_S_j` as well, in this case) to generate predictions of prevalence over the regular grid at 5 km we have created earlier.

```
# Prediction of riverblindness prevalence
lb_prev <-
pred_target_grid(
  lb_pred_S_m,
  f_target = list(prev = function(lp) exp(lp)/(1+exp(lp))),
  pd_summary = list(mean = function(Tx) mean(Tx),
                    cv = function(Tx) sd(Tx)/ mean(Tx),
                    exceed20 = function(Tx) mean(Tx > 0.2))
)
```

In the function above the argument `f_target` can take a list of functions, as multiple predictive targets can be defined. Here, we only specified prevalence (`prev`). Note that in defining the predictive target in `f_target`, we need to express that as a function of $d(x)^\top\beta+S(x)$. The argument `include_covariates` also allows us to define predictive target that do not use covariates from the model. By default `include_covariates = TRUE`, hence in the code above covariates are part of what is defined as `lp`. To predict the spatial random effects $S(x)$ only, for example, you can set `include_covariates = FALSE` and `f_target = list(Sx = function(lp) lp)`. In `pd_summary`, we define the summaries that we want to visualize from the predictive distribution. Here, we have specified the mean, coefficient of variation and the probability of exceeding a 20% prevalence threshold.

We can then visualize the resulting maps as follows.

```
# Displying the results
par(mfrow = c(1,3))
plot(lb_prev, which_target = "prev",
     which_summary = "mean", main = "Mean")
plot(lb_prev, which_target = "prev",
     which_summary = "cv", main = "Coefficient of variation")
plot(lb_prev, which_target = "prev",
     which_summary = "exceed20", main = "Exceedance
      ↪  probability")
par(mfrow=c(1,1))
```

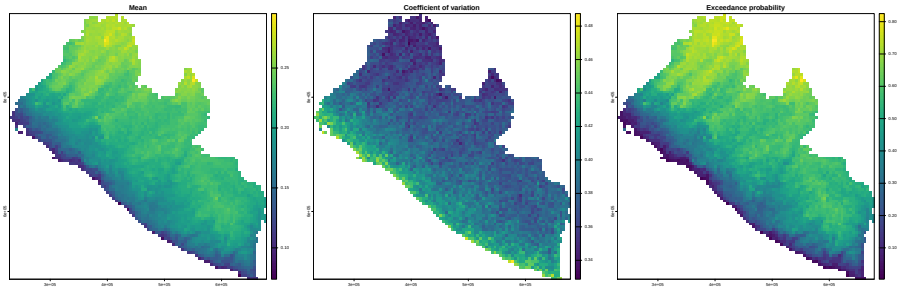



Figure 4.2: Mapping riverblindness in Liberia. Plots of the mean (left panel), coefficient of variation (central panel) and exceedance probability (right panel) for a 20% prevalence threshold.

The maps in Figure 4.2 indicate a higher prevalence as we move away from the coast and at higher altitudes. A very similar spatial pattern is observed in the likelihood of exceeding 20% prevalence. Also, based on the map of the coefficient of variation, the uncertainty around the point predictions of prevalence is higher along the coast and lower in the inland areas of Liberia.

### 4.3.2 Example: mapping malaria using age and elevation as predictors

We now consider an example on malaria mapping, where we use two different types of covariates: a spatially referenced covariate corresponding to elevation; the individual age. We use the `malkenya_comm` subset of the `malkenya` data-set (see Section 3.1.1.2) and, to alleviate the computational burden, we further reduce this data-set by taking the first 1000 rows. Hence, we create our new data-set, `malkenya_comm1000`, using the followind code.

```
malkenya_comm1000 <- malkenya_comm[1:1000,]

malkenya_comm1000 <- st_as_sf(malkenya_comm1000, coords =
 ↪  c("Long", "Lat"),
```

```
                                  crs = 4326)
malkenya_comm1000 <- st_transform(malkenya_comm1000, crs =
 ↪  32736)
```

Based on the exploratory analysis shown in Section 3.1.1.2, we consider a geostatistical model fitted to the individual binary outcomes, $Y_{ij}$, which correspond the rapid diagnostic test (RDT) results for the $j$-th individual in the $i$-th community, and take value $Y_{ij} = 1$ if the RDT is positive and $Y_{ij} = 0$ if the RDT is negative. Using a Binomial geostatistical model, we model the individual probability, $p_{ij}$ of a positive RDT using the following linear predictor.

$$\log\left\{\frac{p_{ij}}{1-p_{ij}}\right\} = \beta_0 + \beta_1 a_{ij} + \beta_2 \times \max\{a_{ij} - 15, 0\} + \beta_3 \max\{a_{ij} - 40, 0\} + \beta_4 e(x_i) + S(x_i), \tag{4.2}$$

where $e(x_i)$ and $a_{ij}$ are the elevation in meter and the age in years for the $j$-th individual residing at the $i$-th household, respectively.

When using the model in Equation 4.2 to predict RDT prevalence, a key question arises: how should we handle the age variable, which, unlike elevation, is not available as a geo-referenced covariate at all locations in the study area? The answer depends on the research question being addressed.

For instance, if the goal is to infer disease risk across different age groups, we can generate different maps for each group of interest. This can be easily achieved by fixing the value of $a_{ij}$ to a specific age for all prediction locations. Alternatively, if the objective is to generate a predictive map for the general population, rather than for a specific age group, we must employ a probabilistic model for age and integrate out its effect.

Developing a credible probabilistic model for age is beyond the scope of this section. Instead, we demonstrate a simple yet reasonable solution which uses the empirical age distribution from the data. By random sampling from this distribution, we can then assign age values to prediction locations. The steps are as follows.

1. Obtain the empirical distribution of age from the data.
2. Sample with replacement from such distribution (the `sample` function in R can be used for this purpose) as many times as the number of prediction locations, so that each grid location $\tilde{x}$ has a value of age assigned.
3. Generate a sample from the predictive distribution of the target $T(\tilde{x})$ at each of the grid locations $\tilde{x}$.
4. Repeat 1 to 3, for as many times as the samples generates from $S(\tilde{X})$.

This approach relies on two key assumptions. First, that the age distribution

is spatially neutral, i.e., it does not vary across space. Second, that the data are representative of the age distribution in the target population. In the data used for this example, we believe these assumptions are reasonable, given the relatively small study area, where age is unlikely to exhibit significant spatial variation, and the fact that the data are a random sample from the community.

In the scripts presented in the remainder of this section, we show how to generate a predictive map of prevalence for children age 15 years, and another map that instead uses the approach earlier described to remove the effect of age and generate a map of prevalence for the general population.

First, we fit the model. Note that the effect of age is not linear and we use linear splines to account for this, using the results of the exploratory analysis shown in @Section 3.1.1.2.

```
fit_malkenya <- glgpm(RDT ~ Age +  pmax(Age-15, 0) +
                        pmax(Age-40, 0) + elevation +
                        gp(),
                      data = malkenya_comm1000,
                      family = "binomial")
```

After fitting the model, we create a prediction grid at a spatial resolution of 500 meters and extract the values of elevation. In this analysis, due to the small size of the study area, we do not use administrative boundaries as they are too large. Instead, we use the convex hull[2] of the sample locations to define the boundaries of our study area.

```
# Create grid from convex hull
shp_ch <- convex_hull_sf(malkenya_comm1000)
ken_grid <- create_grid(shp_ch, spat_res = 0.5)

# Get elevation raster
ken_elev <-
get_elev_raster(locations = shp_ch,
                z = 9, clip = "locations")
```

We then create the data frame of predictors where age is set to 15.

```
# Create the data fr
ken_predictors <- data.frame(elevation =
                           extract(ken_elev,
                            ↪  st_coordinates(ken_grid)),
```

[2]The convex hull of a set of spatial locations is the smallest possible shape (usually a polygon) that completely encloses all the points, such that if you take any two points inside the shape, the straight line connecting them is also entirely inside the shape. In simple terms, it can be thought of as stretching a rubber band around the outermost points in a set, and the shape the rubber band forms is the convex hull.

```
                                Age = 15)
```

We now have all the ingredients to carry out prediction.

```
pred_ken_S <- pred_over_grid(fit_malkenya, grid_pred = ken_grid,
                             predictors = ken_predictors)

pred_age15 <-
pred_target_grid(pred_ken_S,
                 f_target = list(prev = function(lp)
                  ↪  exp(lp)/(1+exp(lp))),
                 pd_summary = list(mean = function(Tx)
                  ↪  mean(Tx)))
```

The code below shows how to perform the 4 steps described above to generate a predictive map for the general population. Below we use the function `update_predictors` to update the covariates effects that are stored in `mu_pred`, a list element generated as the output of `pred_over_grid`.

```
# Number of Monte Carlo samples
n_sim <- ncol(pred_age15$lp_samples)

# Number of prediction locations
n_pred <- nrow(predictors)

# The create object `pred_ken_S_i` with the goal of changing the
 ↪
# coviariates effects stored in `mu_pred` according the sample
 ↪  values of age
pred_ken_S_i <- pred_ken_S
pred_ken_S_i$mu_pred <- matrix(NA, nrow = n_pred, ncol = n_sim)

for(i in 1:n_sim) {
  # Generate n_pred samples from the empirical distribution of
   ↪  age
  ken_predictors$Age <- sample(malkenya_comm1000$Age, n_pred,
 ↪  replace = TRUE)
  # Generate the new covariates effects with `update_predictors`
   ↪  and store them in
  # `mu_pred`
  pred_ken_S_i$mu_pred[,i] <- update_predictors(pred_ken_S,
 ↪  ken_predictors)$mu_pred
}

# Prediction of prevalence
```

```
pred_aver_pop <- pred_target_grid(pred_ken_S_i,
                f_target = list(prev = function(lp)
                 ↪  exp(lp)/(1+exp(lp))))
```

We can now plot the two maps and compare them.

```
par(mfrow = c(1,2))
plot(pred_age15, which_target = "prev", which_summary = "mean",
 ↪  main = "Prevalence (15 years)", range = c(0,0.6))
plot(pred_aver_pop, which_target = "prev", which_summary =
 ↪  "mean",  main = "Prevalence (General population)", range =
 ↪  c(0,0.6))
par(mfrow = c(1,1))
```

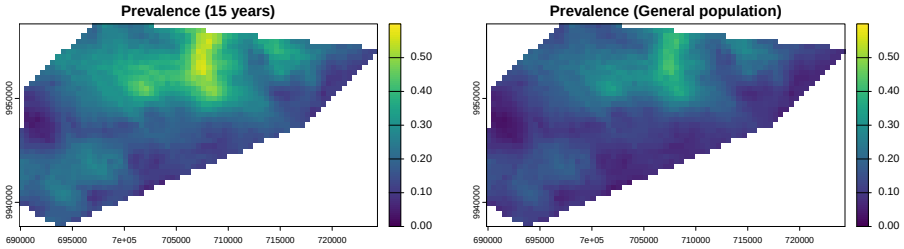



Figure 4.3: Maps of the predictive mean of the rapid diagnostic test (RDT) prevalence for children of the age of 15 (left panel) and the general population (right panel).

The exploratory analysis of Section 3.1.1.2 had shown that the prevalence around the age of 15 is higher than at other ages. This is also reflected in Figure 4.3, when the left panel, corresponding to the RDT prevalence for children at the age of 15 years, shows a higher level of prevalence than the right panel, which is instead for the general population.

## 4.4 Areal-level targets

When determining areal-level targets, we must first address the following questions:

1. What spatially continuous target, $T(x)$, are we seeking to aggregate?
2. What are the spatial units, denoted as $A_i$ for $i = 1, \ldots, N$, over which the aggregation is required?

3. What aggregating function should be applied?
4. Should spatial weights be incorporated in the aggregation, and if so, what weights are appropriate?

The first question was discussed in earlier sections, where examples of predictive targets that define $T(x)$ can be found in Table 4.2. The answers to the remaining questions are context-dependent and closely linked to the primary research objectives. In public health settings, the areas $A_i$ often correspond to administrative units or regions where decisions are made and resources are allocated.

A common aggregating function for $T(x)$ is the mean, which is formally defined as:

$$\mathcal{M}_i = \frac{1}{|A_i|} \int_{A_i} T(x)\, dx. \tag{4.3}$$

In addition to the mean, several other aggregating functions can be applied depending on the context of the analysis. We also note that computing the integral in Equation 4.3 requires some approximations. Our approach utilizes the regular grid $\tilde{X}$, previously defined when dealing with spatially continuous targets. This grid covers the area $A_i$, allowing us to approximate $M_i$ as:

$$\mathcal{M}_i \approx \frac{1}{\#\{j : \tilde{x}_j \in A_i\}} \sum_{\tilde{x}_j \in A_i} T(\tilde{x}_j), \tag{4.4}$$

where $\#\{j : \tilde{x}_j \in A_i\}$ denotes the number of grid locations $\tilde{x}_j$ that fall within $A_i$. The same approximation will be applied to other areal-level targets discussed in this section. However, for simplicity, we will omit the detailed explanation of this step and instead express the predictive target as an integral.

For instance, in the study of Anopheles mosquitoes in Cameroon, we may be interested in estimating the total number of mosquitoes trapped within the study area ($A_i$ represents a single region in this case). If $T(x)$ denotes the number of mosquitoes trapped at location $x$, the areal-level target can be expressed as:

$$\mathcal{S}_i = \int_{A_i} T(x)\, dx.$$

Additionally, to capture the heterogeneity of $T(x)$ within an area, variance-based measures can be used. The variance of $T(x)$ in $A_i$ is given by:

$$\mathcal{V}_i = \frac{1}{|A_i|} \int_{A_i} \left(T(x) - \mathcal{M}_i\right)^2 dx,$$

where $\mathcal{M}_i$ is the mean of $T(x)$ in $A_i$.

The formulation of the areal-level targets given so far assumes equal weighting for all locations within the area. Alternatively, if the we consider for example the areal level target in Equation 4.3, one could use spatial weights, $w(x) > 0$, and redefine $\mathcal{M}_i$ as:

$$\mathcal{M}_i = \frac{\int_{A_i} w(x)T(x)\,dx}{\int_{A_i} w(x)\,dx}.$$

Selecting appropriate spatial weights is crucial, as it reflects the significance of different locations within the area. We can define three types of weighting: population-density, risk-based, and distance-based. We point out that this distinction is not always clear cut (population could indeed be a risk factor in our analysis) but this classification is made only for the sake of making the explanation clearer.

For example, if the goal is to prioritize areas with higher populations, weights $w(x)$ could be proportional to population density at location $x$. This approach gives greater importance to locations where more people reside, which can be particularly relevant when aggregating disease prevalence data for resource allocation.

In cases where certain sub-regions within $A_i$ are at higher risk for disease (e.g., proximity to environmental hazards or areas with lower access to healthcare), risk-based weights could be applied. Here, $w(x)$ would be higher in regions identified as higher risk, providing a more targeted aggregation of disease prevalence. For example, populations in rural areas may face higher exposure to infectious diseases than those in urban areas, making it important to assign greater weight to these higher-risk regions in the aggregation process.

If the objective is to account for proximity effects (such as the spread of an infectious disease or contamination from a known source), distance-based weights could be used. Locations closer to a known disease outbreak area or source of exposure could be given more weight to reflect the spatial dynamics of disease transmission.

Using spatial weights in these ways ensures that the aggregation of disease prevalence $T(x)$ reflects not only the distribution of the disease but also the underlying population, risk, or spatial characteristics relevant to the public health problem under investigation.

### 4.4.1 Example: predicting the average riverblindness prevalence at admin level 1 in Liberia (continuing from Section 4.3.1)

We continue our analysis of the Liberia data and consider areal-level targets. More specifically, we aim to predict the average prevalence over the regions

which give the administrative level 1 of Liberia. By denoting with $A_i$ the i-th out of the 15 regions in Liberia. By denoting with $T(x)$ reiverblindness prevalence as defined in Equation 4.1, we define our predictive target as

$$T_i = \frac{1}{|A_i|}\int_{A_i} T(x)\, dx.$$

The R code below shows how predictions for $T_i$ can be performed using the `pred_target_shp` function.

```
# Admin level 1 boundaries
library(rgeoboundaries)
lb_adm1 <- geoboundaries(country = "Liberia", adm_lvl = "adm1")

# Prediction of prevalence
pred_shp <- pred_target_shp(lb_pred_S_j, shp = lb_adm1,
                            shp_target = function(Tx) mean(Tx),
                            f_target = list(prev =
                                       function(lp)
                                        ↪  exp(lp)/(1+exp(lp))),
                            pd_summary = list(mean = mean),
                            col_names = "shapeName")
```

The argument `shp_target` is used define the type of aggregation function over the region $A_i$, in this case the mean of the target $T(x)$. Also note that, like for spatially continuous targets, we can define any summary of the predictive distribution of $\mathcal{M}_i$ through the argument `pd_summary`. Here, we chose to compute only the mean of the predictive distribution but other summaries, such as exceedance probabilities, standard error, quantiles and so on, can be specified if needed.

Before we plot the results on a map, let us also consider the population-density weighted target, defined as

$$T_i^* = \frac{\int_{A_i} w(x)T(x)\, dx}{\int_{A_i} w(x)\, dx}.$$

To aid the explanation of the R code script, we first rewrite the above areal-level target as

$$T_i^* = \int_{A_i} \tilde{w}_i(x)T(x)\, dx.$$

where $\tilde{w}_i(x)$ are the standardized weights that integrate to 1 in $A_i$ (i.e. $\int_{A_i} \tilde{w}_i(x)\, dx = 1$).

We first retrieve the population density data from the World Pop database (see Section 2.3.2) and use these to derive the un-standardized weights $w(x)$.

```
# Obtaining population density
library(wpgpDownloadR)
lbr_url <- wpgpGetCountryDataset(ISO3 = "LBR", covariate =
 ↪  "ppp_2014")
library(terra)
lbr_pop <- rast(lbr_url)
lbr_pop <- project(lbr_pop, "EPSG:32629")

# Extra pop density weights at the prediction grid
weights_pred <- extract(lbr_pop,
 ↪  st_coordinates(liberia_grid))$lbr_ppp_2014
```

In the code above the population density weights are extracted at the locations of the prediction grid. In the `pred_target_shp` function, in order to use the weights for our predictive target, we need to specify two additional arguments: `weights` to which we pass the population density values (in our case these are stored in `weights_pred`); `standardize_weights`, which is a logical argument taking value `TRUE` if the weights to be standardized so that the sum over the grid covering a region $A_i$ adds up to one.

```
pred_shp_w <- pred_target_shp(lb_pred_S_j, shp = lb_adm1,
                              shp_target = function(Tx) sum(Tx),
                              f_target = list(prev =
                                         function(lp)
                                          ↪  exp(lp)/(1+exp(lp))),
                              pd_summary = list(mean = mean),
                              weights = weights_pred,
                              standardize_weights = TRUE,
                              col_names = "shapeName")
```

We can then finally plot the results from the prediction at admin level 1, without using any weights and by weighing according to population density.

```
plot_1 <-
plot(pred_shp, which_target = "prev", which_summary = "mean",
     palette = "RdYlGn",
    limits = c(0.1, 0.30),
    breaks = seq(0.1,0.30, by = 0.05)) +
  guides(fill=guide_legend(title="Prevalence")) +
  ggtitle("Average prevalence \n (no weights)") +
   theme(plot.title = element_text(size = 15))

plot_2  <-
  plot(pred_shp_w, which_target = "prev", which_summary =
   ↪  "mean",
```

```
     palette = "RdYlGn",
    limits = c(0.1, 0.30),
    breaks = seq(0.1,0.30, by = 0.05)) +
  guides(fill=guide_legend(title="Prevalence")) +
  ggtitle("Average prevalence \n (population weighted)") +
  theme(plot.title = element_text(size = 15))

library(gridExtra)
grid.arrange(plot_1, plot_2, ncol = 2)
```

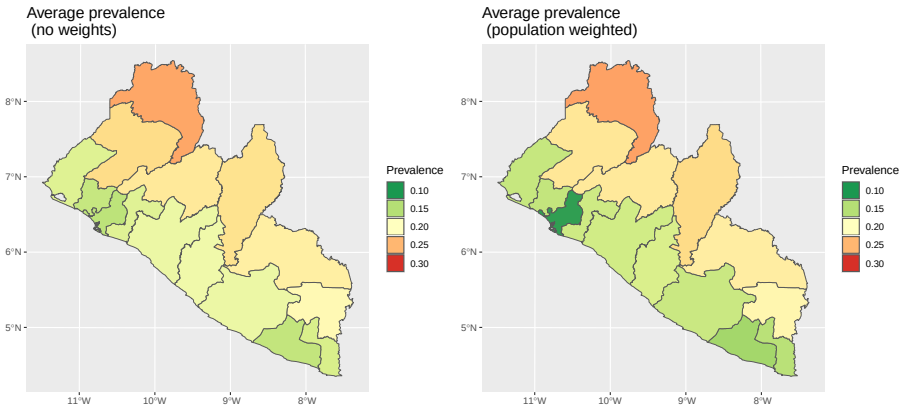



Figure 4.4: Maps of the predictive mean for the admin level 1 average prevalence in Liberia. The left panel uses a standard average of locations within an admin level 1 region, whilst in the right panel each location is weighted according to population density.

In Figure 4.4, we observe that the differences between the two predictive targets, with and without weights, are not substantial. However, we observe that the predicted average prevalence shows slightly greater variation in coastal regions, particularly in Montserrado County, where the majority of the population residing in the capital of Monrovia. This concentration of population likely accounts for the more noticeable differences in this area.

### 4.4.2 Example: predicting the total number of Anopheles gambiae mosquitoes (continuing from Section 3.3.2)

For the analysis on Anopheles gambiae mosquitoes, we consider the prediction of the total number of mosquitoes within the study area. In this case, because there is not a natural definition of the study area borders as in the previous analysis, we consider the convex hull as representing those borders. To pursue this let us first, visualize the predictive map of the number of mosquitoes

on a regular grid cover the study area. In other words, we first consider the spatially continuous target

$$T(x) = \exp\{\beta_0 + \beta_1 e(x) + S(x)\},$$

where, we recall, $e(x)$ is the elevation in meters at location $x$. In the code below in addition to generate prediction for $T(x)$, we also create a binary indicator defined as

$$w(x) = \begin{cases} 1 & \text{if } 390 < e(x) < 837 \\ 0 & \text{otherwise.} \end{cases}$$

The values of 390 and 837 meters correspond to the minimum and maximum values of elevation observed in the data.

```
# Create grid from convex hull
shp_ch <- convex_hull_sf(an_fit$data_sf)
an_grid <- create_grid(shp_ch, spat_res = 2)


an_elev <- get_elev_raster(locations = shp_ch,
                                z = 9, clip = "locations")

predictors <- data.frame(elevation= terra::extract(an_elev,
                                                   ⌋
                                                    ↪  st_coordinates⌋
                                                    ↪  (an_grid)))

pred_an_S <- pred_over_grid(an_fit, grid = an_grid,
               predictors = predictors,
               type = "joint")

pred_n_mosq_grid <-
pred_target_grid(pred_an_S,
                 f_target = list(n_mosq = function(lp) exp(lp)),
                 pd_summary = list(mean = function(Tx)
                  ↪  mean(Tx)))

an_weights <- 1*(predictors$elevation > 390 &
 ↪  predictors$elevation < 837)
```

We now define our predictive target corresponding to the total number of mosquitoes within the study area $A$, given by the convex hull of the observed locations, as

$$T = \int_A w(x)T(x)\,dx \tag{4.5}$$

The rationale is to estimate the total number of mosquitoes in the study area while restricting predictions to locations within the observed range of elevation.

This approach helps avoid the risk of predicting in ecological areas that may not be suitable for mosquito presence. It is also important to note that in Equation 4.5, we do not standardize the weights in this case.

```
par(mfrow = c(1,2))
plot(pred_n_mosq_grid, which_target = "n_mosq", which_summary =
 ↪  "mean",
     main = "Number of mosquitoes")
plot(an_grid, pch = 20, cex = 1, col = an_weights+1, main =
 ↪  "Weights")
par(mfrow = c(1,1))
```

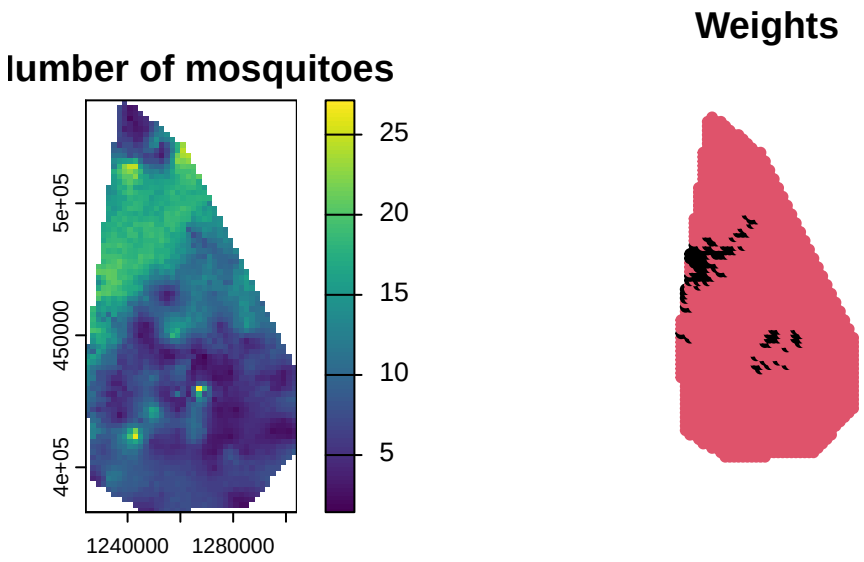



Figure 4.5: Maps of the predicted averae number of mosqutoes (left panel). The right panel shows prediction locations in red if they have an elevation between 390 and 837 meters, or black otherwise.

Figure 4.5 shows the prediction map for the number of mosquitoes at each prediction location, indicating a relatively high spatial heterogeneity as was also indicated by the low value for the estimate of the scale of the spatial correction (parameter $\phi$). The right panel, we have an image of the weights $w(x)$, with locations that will not contribute to the prediction of the total number of mosquitoes denoted in black.

```
pred_n_mosq_shp <-
pred_target_shp(pred_an_S, shp = shp_ch,
                weights = an_weights,
                shp_target = sum,
```

```
                  f_target = list(n_mosq = function(lp) exp(lp)),
                  pd_summary = list(mean = function(Tx) mean(Tx),
                                    q025 = function(Tx)
                                     ↪  quantile(Tx, 0.025),
                                    q075 = function(Tx)
                                     ↪  quantile(Tx, 0.975)))

pred_n_mosq_shp$target$reg1$n_mosq
## $mean
## [1] 19479.92
##
## $q025
##     2.5%
## 16667.93
##
## $q075
##    97.5%
## 22904.26
```

Since we are not standardizing the weights, there is no need to specify `standardize_weights`, as it is set to `FALSE` by default. In the code above, we printed the mean of the predictive distribution of $T$ along with the 95% prediction interval. The point estimate of approximately 17,719 mosquitoes is likely an underestimate of the total number of Anopheles gambiae in the area, as the trap used to count the mosquitoes may not capture all of them, particularly those outside the immediate trapping zone or those active at different times.

## 4.5 Assessing the predictive performance of geostatistical models with cross-validation

In this section, we address the problem of identifying the geostatistical model that offers the best predictive performance among a set of candidates. However, we first need a clear definition of predictive performance which can be used to select suitable statistical tools that can be to used to evaluate it. Broadly speaking, predictive performance is defined by how well a model's predictive distribution aligns with observed data. Evaluating predictive performance requires examining two key characteristics of the predictive distribution: *sharpness* and *calibration*.

### 4.5.1 How to split geostatistical data for model performance comparisons

To carry out the assessment and comparison of the predictive performance of geostatistical models using the methods illustrated in the following sections, the first crucial step to decide which approach to use to use split the data-set into a *training set* and *test set*. In the context of geostatistical analysis, the *training set* is the subset of the original data-set that is used to estimate the model parameters and then predict at the locations of the *test set*, which is used to assess how well the model can predict unseen data.

When splitting geostatistical data-sets for model performance evaluation, it is essential to consider the objective of the spatial prediction. Here, we consider two main prediction objectives: 1) predicting in areas disjoint from the study area where there are no data; 2) inferring the spatial surface of disease risk within a given study area. Let us give an example to better explain the difference between these two objectives. Under objective 1), if survey data exist for a specific country, say Kenya, but not for a neighboring country, say Somalia, a model trained on Kenyan data might then be used to predict disease prevalence in Somalia. In objective 2), instead, high-resolution risk maps of a health outcome might be required within a single country. The splitting of geostatistical data for predictive performance assessment should align with the intended application of the model and reflect whether the goal is to predict in entirely disjoint areas with no data (objective 1) ) or to infer the spatial surface of interest within the same study area (objective 2) ). Below, we elaborate on suitable methods for each of these two.

In the first scenario (objective 1) ), splitting the data using k-fold cross-validation with spatially coherent folds is more appropriate. These folds can be created using clustering methods such as k-means or hierarchical clustering (see Yin et al. (2024) for review of the main clustering methods). These methods group spatial locations based on their coordinates or covariates, ensuring each fold represents a distinct spatial region. This approach mimics the challenge of predicting in disjoint areas by withholding entire spatial clusters from the training data. For instance, k-means minimizes within-cluster variance to produce compact, non-overlapping groups, while hierarchical clustering organizes data into nested clusters based on a linkage criterion like geographic distance. In the code below we show an example of the use of these methods for data splitting, using the `spatial_clusterin_cv` function from the `spatialsample` package (Mahoney et al. 2023).

```
library(spatialsample)
set.seed(123)
liberia_sf <- liberia %>%
              st_as_sf(., coords = c("long", "lat"),
                       crs = 4326) %>%
              st_transform(.,crs = 32629)
```

```
# K-means
kmeans_liberia <- spatial_clustering_cv(liberia_sf,
                      v = 10, cluster_function = "kmeans")
kmeans_plot <- autoplot(kmeans_liberia) + ggtitle("K-means")

# Hierarchical clustering
hclust_liberia <- spatial_clustering_cv(liberia_sf,
                      v = 10, cluster_function = "hclust")


hclust_plot <- autoplot(hclust_liberia) +
  ggtitle("Hierarchical clustering")

library(gridExtra)
grid.arrange(kmeans_plot, hclust_plot, ncol = 2)
```

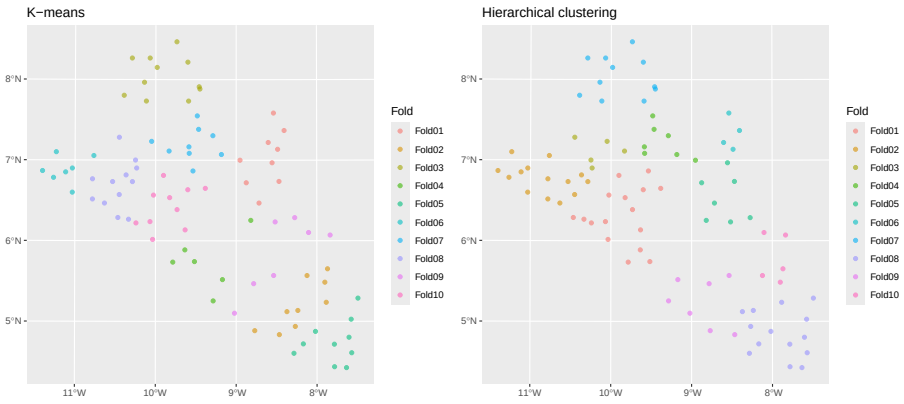



Figure 4.6: Examples un the use of spatial clustering methods for the splitting of the Liberia data on riverblindness.

The results of the splitting are shown in Figure 4.6, where we have generated 10 folds using the k-means and hierarchical clustering methods with the default options. We can observe that the test sets are very similar, with fold 3 from the k-means and fold 7 from hierarchical clustering being identical. If using only the distance between locations to cluster the data into different folds, the choice of clustering method should have much impact on the model assessment as long as we repeat the split and assessment of the methods multiple times as we will show later in the next sections.

Before proceeding further, it is important to highlight that when the objective

is predicting an outcome in entirely disjoint areas, the risks and limitations of this assessment should be carefully considered. In such cases, predictions are predominantly driven by covariates because the spatial Gaussian process $S(x)$ cannot extrapolate spatial patterns across regions with no connecting data. When the disjoint region requiring predictions is far from the sampled data, the Gaussian process contributes primarily to inflating prediction uncertainty and has no tangible impact on the point predictions, reflecting the lack of ground data in the area. Consequently, this validation approach primarily assesses the predictive power of the covariates rather than the geostatistical model as a whole. This limitation may lead to overestimating the model's generalization capabilities, especially if the model is being used to predict areas that are far away from the study area. Hence, the results of this validation should be interpreted cautiously, emphasizing the critical role of covariates in driving predictions under such conditions.

We now consider the second prediction objective of inferring the spatial surface within the study area. In this case, a random sampling approach can be employed. However, to preserve a good spatial coverage of the study area, a minimum distance can be imposed between selected locations. For example, methods such as spatial thinning or stratified sampling can be adapted to enforce a minimum spatial separation. This can be done in R using the `subsample.distance` function from the `spatialEco` package (Evans and Murphy 2021).

```
library(spatialEco)
set.seed(123)

# Regulirized spatial sampling: 30km
dist30_liberia <- subsample.distance(liberia_sf,
                      size = 20, d = 30000)
dist30_plot <- ggplot(dist30_liberia) + geom_sf() +
               theme_minimal() +
               ggtitle("Minimum distance: 30km")

# Regulirized spatial sampling: 40km
dist40_liberia <- subsample.distance(liberia_sf,
                      size = 20, d = 40000)
dist40_plot <- ggplot(dist40_liberia) + geom_sf() +
               theme_minimal() +
               ggtitle("Minimum distance: 40km")

grid.arrange(dist30_plot, dist40_plot, ncol = 2)
```

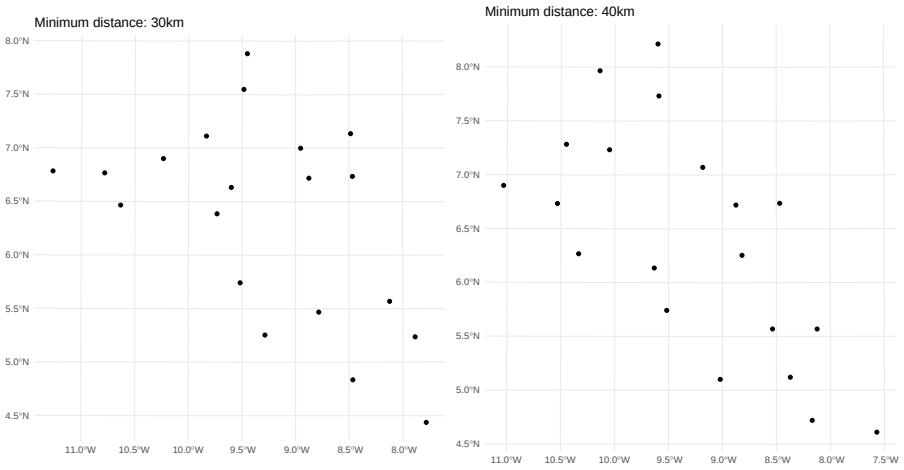



Figure 4.7: Examples un the use of spatialky regularized thinning for the splitting of the Liberia data on riverblindness into training and test set. The plots show the resulting test sets by randomly selecting 20 locations from the original data-set and imposing a minimum distance of 30km (left panel) and 40km (right panel).

In the code above, we randomly selected 20 locations from the original data-set, enforcing minimum distances of 30 km and 40 km between points. The results are presented in Figure 4.7. The choice of the minimum distance for the test sample should ideally consider the spatial correlation scale (previously denoted by $\phi$) of the spatial Gaussian process $S(X)$. To minimize correlation within the test set, the minimum distance could be set to a value greater than $\phi$. However, increasing the minimum distance makes it more challenging to obtain a test set of the desired sample size. Reducing correlation within the test set is important for ensuring that the assessment of predictive performance is as robust as possible.

A key challenge in both approaches is the potential lack of independence between the training and test data-sets due to spatial correlation. This can lead to an overly optimistic evaluation of model performance, as the training set may already contain information that is correlated with the test set. However, if we assume that all models benefit equally from this correlation during performance evaluation, the methods we illustrate next can still provide a reliable comparison of model performance in relative terms, even if the considered performance metrics may be somewhat biased.

### 4.5.2 Assessing calibration using the nonrandomized probability integral transform

A model is considered *well-calibrated* if its predictions adequately reflect the true uncertainty of the data. Assessment of a model's calibration can be carried out in several ways. We can examine the agreement between the point predictions, say $\hat{y}_i$, and the observed outcomes $y_i$, commonly referred to as *accuracy*. The mean squared error (MSE), defined as $\text{MSE} = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2$, is an example of a commonly used metric to evaluate the accuracy of a model. To characterize calibration more fully, it is also essential to quantify the spread of the predictive distributions, which also defines the *precision* of predictions. For example, prediction intervals generated from the predictive distribution for $y_i$ can help assess the precision of a geostatistical model. In summary, a well-calibrated model is both accurate and precise.

Assessment of the calibration of the model should precede the assessment of sharpness, since, as we shall see in the next section, this relies on the model being well-calibrated. The probability integral transform (PIT) was originally proposed by Dawid (1984) as a way to assess the calibration of a model. The PIT is based on the simple observation that if we consider a variable $Y$ and apply the transformation $Y^* = F_Y(Y)$, where $F_Y(\cdot)$ is the cumulative density function (or cumulative distribution if $Y$ is discrete) of $Y$, it then follows that $Y^*$ follows a uniform distribution in the unit interval. The fundamental problem and the reason why statistical modelling exists is that we do not $F_Y$ but we would like to propose a model $M$ that we believe adequately approximates $F_Y$ with $F_M$. The main advantage of using the PIT is that assessment of whether $M$ is well calibrated reduces to assessing whether the transform set of data $F_M(y_1), \ldots, F_M(y_n)$ follows a uniform distribution. To illustrate this, let us a consider a simple simulated example.

We first create a function that can compute the PIT.

```
# Define a function for the PIT
# F_M is the cumulative density function generated by the
 ↪  adopted model
pit_transform <- function(data, F_M) {
  # Apply the model CDF to the data
  pit_values <- F_M(data)
  return(pit_values)
}
```

We then apply this function to show how the assessment of calibration through the PIT is carried out under two scenarios: 1) $F_Y$ is a Gamma distribution with shape 2 and scale parameter 1; 2) $F_Y$ is Student's T distribution with 3 degrees of freedom. In each of the two scenarios, we show how the PIT behaves when our model coincides with the the true model (i.e. $F_M = F_Y$) and when instead our model $M$ is a Gaussian distribution, with mean and

variance estimated from the data.

```
# Generate data from a skewed (gamma) distribution
n <- 1000
shape <- 2
rate <- 1
data_gamma <- rgamma(n, shape = shape, rate = rate)

# Define correct and incorrect model CDFs for gamma data
# Correct CDF
model_cdf_correct_gamma <- function(x) pgamma(x, shape = shape,
 ↪  rate = rate)
# Incorrect CDF: Assume data is normal (wrong model)
model_cdf_incorrect_gamma <- function(x) pnorm(x, mean =
 ↪  mean(data_gamma), sd = sd(data_gamma))

# Compute PIT values for gamma data under both models
pit_correct_gamma <- pit_transform(data_gamma,
 ↪  model_cdf_correct_gamma)
pit_incorrect_gamma <- pit_transform(data_gamma,
 ↪  model_cdf_incorrect_gamma)

# Generate data from a heavy-tailed (t) distribution
df <- 3
data_t <- rt(n, df = df)

# Define correct and incorrect model CDFs for t data
# Correct CDF
model_cdf_correct_t <- function(x) pt(x, df = df)
# Incorrect CDF: Assume data is normal (wrong model)
model_cdf_incorrect_t <- function(x) pnorm(x, mean =
 ↪  mean(data_t), sd = sd(data_t))

# Compute PIT values for t data under both models
pit_correct_t <- pit_transform(data_t, model_cdf_correct_t)
pit_incorrect_t <- pit_transform(data_t, model_cdf_incorrect_t)

# Combine results into a data frame for plotting
df_plot <- data.frame(
  PIT = c(pit_correct_gamma, pit_incorrect_gamma, pit_correct_t,
   ↪  pit_incorrect_t),
  Model = rep(c("Correct Gamma Model", "Incorrect Normal Model
   ↪  for Gamma",
                "Correct t Model", "Incorrect Normal Model for
                 ↪  t"), each = n)
```

```
)

# Plot PIT distributions
hist_plot <- ggplot(df_plot, aes(x = PIT)) +
  geom_histogram(aes(y = ..density..), bins = 20, fill =
   ↪  "skyblue", color = "black", alpha = 0.7) +
  geom_hline(yintercept = 1, linetype = "dashed", color = "red")
   ↪  +
  facet_wrap(~ Model, scales = "free") +
  labs(title = "Probability Integral Transform (PIT)
   ↪  Distributions",
       x = "PIT Values",
       y = "Density") +
  theme_minimal()

# Display the plot
print(hist_plot)
```

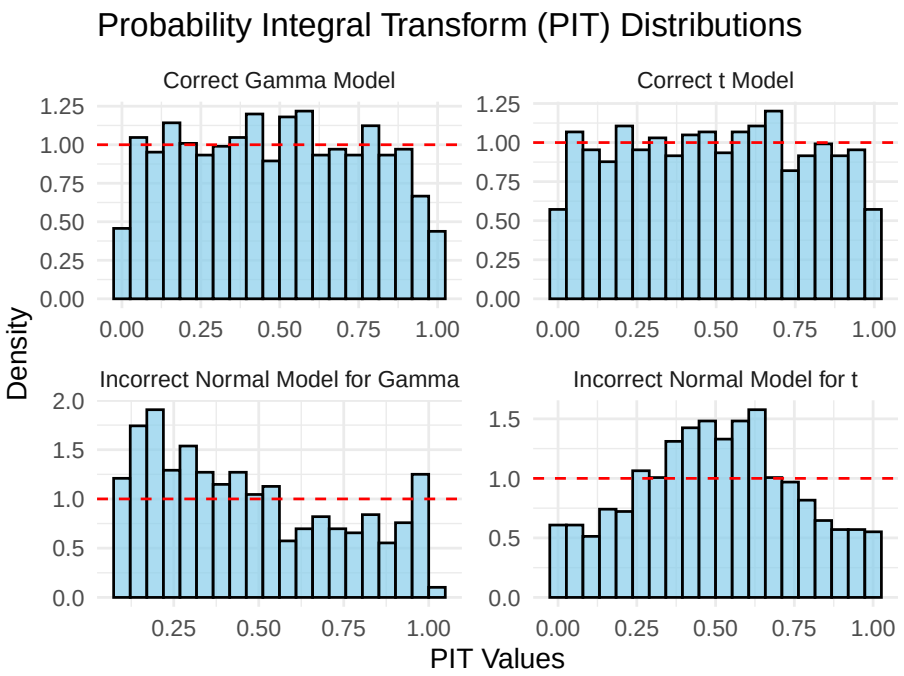



Figure 4.8: Histograms of the probability integral transform applied to simulated data. The top panels show the histograma of the PIT when the fitted model coincides with the true data generating model. The lower panels show how the PIT histrograms deviate from a uniform distribution. For more details, we refer to main text.

In Figure 4.8, we assess whether the PIT transformed data follow a uniform distribution if the bars of the histogram are all approximately at the same height as indicated by the dashed line. Our preference to this diagnostic plot, is to use a qq-plot for a uniform distribution as shown below.

```
# QQ plot to check uniformity of PIT values
qq_plot <- ggplot(df_plot, aes(sample = PIT)) +
  stat_qq(distribution = qunif) +
  stat_qq_line(distribution = qunif, color = "red") +
  facet_wrap(~ Model, scales = "free") +
  labs(title = "QQ Plot of PIT Values against Uniform
   ↪  Distribution",
       x = "Theoretical Quantiles (Uniform)",
       y = "Sample Quantiles (PIT Values)") +
  theme_minimal()

# Display both plot
```

```
print(qq_plot)
```

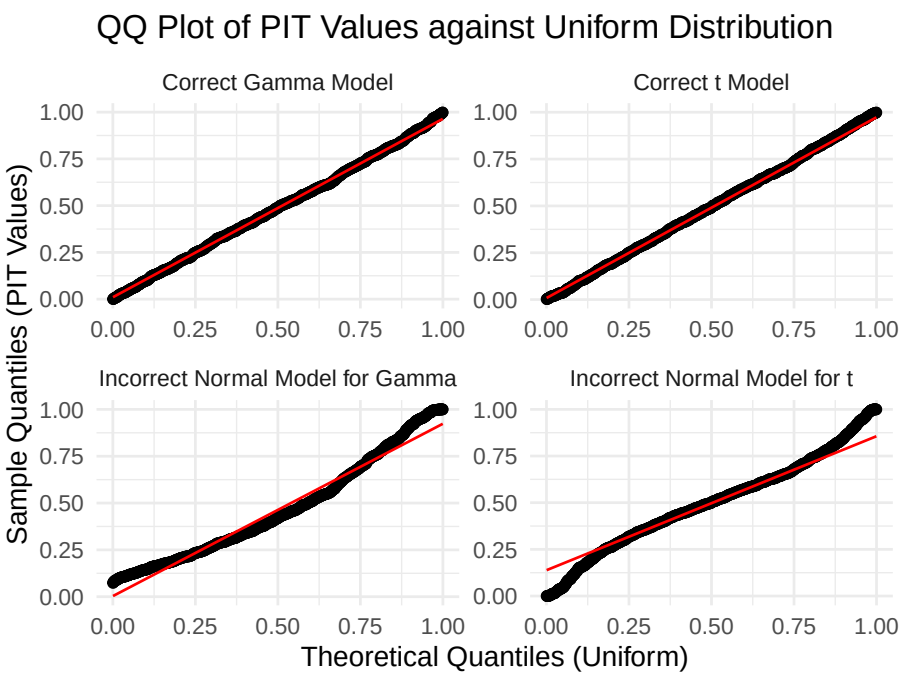



Figure 4.9: QQ-plots of the probability integral transform applied to simulated data. The top panels show the histograma of the PIT when the fitted model coincides with the true data generating model. The lower panels show how the PIT qq-plots deviate from a uniform distribution. The red line is the identity line For more details, we refer to main text.

In the results of Figure 4.9, we consider a model well-calibrated if the qq-plot is as close as possible to the identity line. It is clear that in the first case the use of Gaussian distribution is violated by the skewness of the data generated from a Gamma distribution, and in the second scenario by the heavier tail of the Student's T distribution.

In the simulated example, the PIT might seem unnecessary because we can directly assess if the data follow the assumed distribution (Gamma, Gaussian, or Student's T) by comparing empirical and theoretical distributions. In these simple models, including linear Gaussian geostatistical models, the marginal distribution of the data is analytically available, so assessing goodness-of-fit is straightforward.

However, in the case of Binomial and Poisson geostatistical models, the marginal distribution of the data $Y$ – unconditioned on the spatial random ef-

fects $S(x)$ – cannot be obtained analytically. Specifically, geostatistical models for count data assume that $Y$, conditioned on $S(x)$, follows either a Binomial or Poisson distribution. But once we integrate out $S(x)$ to obtain the marginal (i.e., unconditioned) distribution of $Y$, this distribution is no longer Poisson or Binomial. Instead, it becomes an overdispersed count distribution that lacks a closed-form expression.

In such cases, the PIT becomes valuable for evaluating model adequacy, as it allows for assessing the uniformity of PIT values against the assumed model distribution, even when the marginal distribution is intractable. However, direct application of the PIT raises the issue that the the transformation $F_Y(Y)$ does not follow a uniform distribution, because of the discrete nature of the random variable $Y$. Hence, for a given discrete-outcome model $M$, we use a modified version of the PIT, referred to as nonrandomized PIT (henceforth, nPIT), originally proposed by Czado, Gneiting, and Held (2009), taking the form

$$\text{nPIT}(u, y) = \begin{cases} 0, & u \leq F_M(y-1) \\ \frac{u - F_M(y-1)}{F_M(y) - F_M(y-1)}, & F_M(y-1) \leq u \leq F_M(y) \\ 1, & u \geq F_M(y) \end{cases}. \tag{4.6}$$

We then take the average nPIT over the observed outcomes $y_1, \ldots, y_n$, hence

$$\text{AnPIT}(u) = \frac{1}{n} \sum_{i=1}^{n} \text{nPIT}(u, y_i).$$

Assessment of calibration is then carried out by checking wether the AnPIT is as close as possible to the identity function, i.e. $\text{AnPIT}(u) = u$. Hence, by plotting $\text{AnPIT}(u)$ against $u$, we can consider any deviations from the identity line as evidence that the model is not well-calibrated.

We now illustrate the use of the AnPIT through a simulated example. We simulate data from a negative Binomial distribution, with mean $\lambda = 5$ and dispersion parameter $\alpha = 1/2$, hence $E[Y] = \lambda$ and $\text{Var}[Y] = \lambda \times (1 + \alpha\lambda)$.

```
# Parameters for the Negative Binomial distribution
set.seed(123)
n <- 200                # number of simulations
lambda <- 5             # mean
dispersion <- 0.5       # dispersion parameter
size <- 1 / dispersion  # size parameter for negative binomial

# Simulate data from the Negative Binomial distribution
sim_data_nb <- rnbinom(n, size = size, mu = lambda)

# Define the nonrandomized PIT function
compute_npit <- function(y, u, cdf_func) {
```

```
  # Calculate cumulative probabilities F_M(y-1) and F_M(y) using
   ↪  the provided CDF function
  f_y_minus_1 <- cdf_func(y - 1)
  f_y <- cdf_func(y)

  # Apply the piecewise formula for nPIT
  if (u <= f_y_minus_1) {
    return(0)
  } else if (u <= f_y) {
    return((u - f_y_minus_1) / (f_y - f_y_minus_1))
  } else {
    return(1)
  }
}

# Generate a sequence of u values from 0 to 1
u_values <- seq(0, 1, length.out = 100)

# Calculate the average nPIT for each u value for both models
AnPIT_nb <- sapply(u_values, function(u) {
  mean(sapply(sim_data_nb, compute_npit, u = u, cdf_func =
   ↪  function(k) pnbinom(k, size = size, mu = lambda)))
})

AnPIT_poisson <- sapply(u_values, function(u) {
  mean(sapply(sim_data_nb, compute_npit, u = u, cdf_func =
   ↪  function(k) ppois(k, lambda = lambda)))
})

# Combine the results into a data frame for plotting
AnPIT_data <- data.frame(
  u = rep(u_values, times = 2),
  AnPIT = c(AnPIT_nb, AnPIT_poisson),
  Model = rep(c("Negative Binomial (correct model)", "Poisson
   ↪  (wrong model)"), each = length(u_values))
)

# Plotting the average nPIT as a function of u for both models
ggplot(AnPIT_data, aes(x = u, y = AnPIT, color = Model)) +
  geom_line() +
  geom_abline(slope = 1, intercept = 0, linetype = "dashed",
   ↪  color = "black") +  # Identity line
  facet_wrap(~ Model) +
  coord_cartesian(xlim = c(0, 1), ylim = c(0, 1)) +
```

```
  xlab("u") +
  ylab("AnPIT") +
  theme_minimal() +
  theme(legend.position = "none")
```

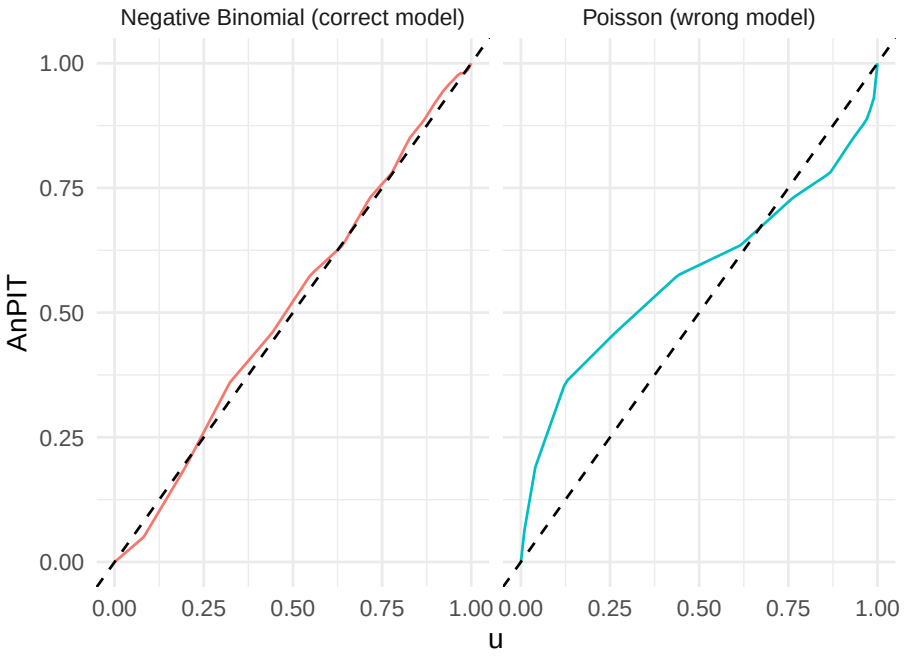



Figure 4.10: Average nonrandomized Probability Integral Transform (AnPIT) applied to simulated data from a Negative Binomial distribution. The left plot shows the AnPIT when the model is correctly specified. The right panel shows the AnPIT when a Poisson model is instead used. For more details we refer to the main text.

Figure 4.10 shows clear evidence that the Poisson model is not adequately accounting for the overdispersion of the simulated data from the Negative Binomial distribution.

How can we adapt this approach to generalized linear geostatistical models (GLGMs)? The concept is straightforward, though the implementation is less so (but fortunately, we have already done this in `RiskMap`). Essentially, in all relevant equations above, we need to replace $M$ with the predictive distribution of the fitted geostatistical model.

To explain further, suppose we have fitted our model using data $Y^{(1)} = \left(y_1^{(1)}, \ldots, y_n^{(1)}\right)$ (the training data-set), and we want to use another set of data

$Y^{(2)} = \left(y_1^{(2)}, \ldots, y_m^{(2)}\right)$ (the test data-set) to assess calibration; the selection of $Y^{(1)}$ and $Y^{(2)}$ can be done using the methods described in Section 4.5.1. Our model $M$ in the equations above corresponds to the distribution of $\left[Y^{(2)} \mid Y^{(1)}\right]$ – the distribution of $Y^{(2)}$ conditioned to $Y^{(1)}$ derived from our geostatistical model. Since its derivation cannot be done analytically, we use Monte Carlo methods to approximate $\left[Y^{(1)} \mid Y^{(2)}\right]$. We discuss this in more detail in Section 4.7. However, the key point here is to understand why we perform this assessment and how to interpret the results. We next show the application of the AnPIT diagnostic to the example on riverblindess mapping in Liberia.

#### 4.5.2.1 Example: assessing the calibration of two geostatistical models for riverblindness mapping

In our example on riverblindness mapping, we now consider two models. A mode, which we call $M_0$, which is an intercept only geostatistical model, with linear predictor

$$M_0 \; : \; \log\left\{\frac{p(x_i)}{1-p(x_i)}\right\} = \beta_0 + S(x_i),$$

and a geostatistical model, which we denote as $M_1$, which uses a linear spline with a not in 150 meters, and has linear predictor

$$M_1 \; : \; \log\left\{\frac{p(x_i)}{1-p(x_i)}\right\} = \beta_0 + \beta_1 e(x_i) + \beta_2 \max\{e(x_i) - 150, 0\} + S(x_i).$$

The reason for using this type of linear spline in $M_1$ is explained in Section 3.1.1.

Hence, we first fit the two models.

```
# Fitting M0
set.seed(123)
M0_fit <-
glgpm(npos ~  gp(long, lat, nugget = NULL),
      den = ntest, data = liberia,
      crs = 4326,
      convert_to_crs = 32629,
      family = "binomial", messages = FALSE)

# Fitting M1
M1_fit <-
glgpm(npos ~  elevation + pmax(elevation - 150, 0) +
      gp(long, lat, nugget = NULL),
      den = ntest, data = liberia,
      crs = 4326,
      convert_to_crs = 32629,
      family = "binomial", messages = FALSE)
```

Since the main goal of this analysis is to draw predictive inference on riverblindness prevalence within Liberia, it would be more appropriate to use a regularized subsampling scheme to split the data. To generate the AnPIT diagnostic plot in R, we can use the `assess_pp` function in `RiskMap`

```
regularized <-
assess_pp(list(M0 = M0_fit, M1 = M1_fit),
          which_metric = "AnPIT",
          method = "regularized", min_dist = 20,
          n_size = 9, iter = 10)
```

```
p1 <- plot_AnPIT(regularized, combine_panels = TRUE) +
      theme(plot.margin = unit(c(0.5, 0.5, 0.5, 0.5), "lines"))

p2 <- plot_AnPIT(regularized, mode = "all") +
      theme(plot.margin = unit(c(0.5, 0.5, 0.5, 0.5), "lines"))

grid.arrange(p1, p2, nrow = 2, heights = c(1, 1))
```

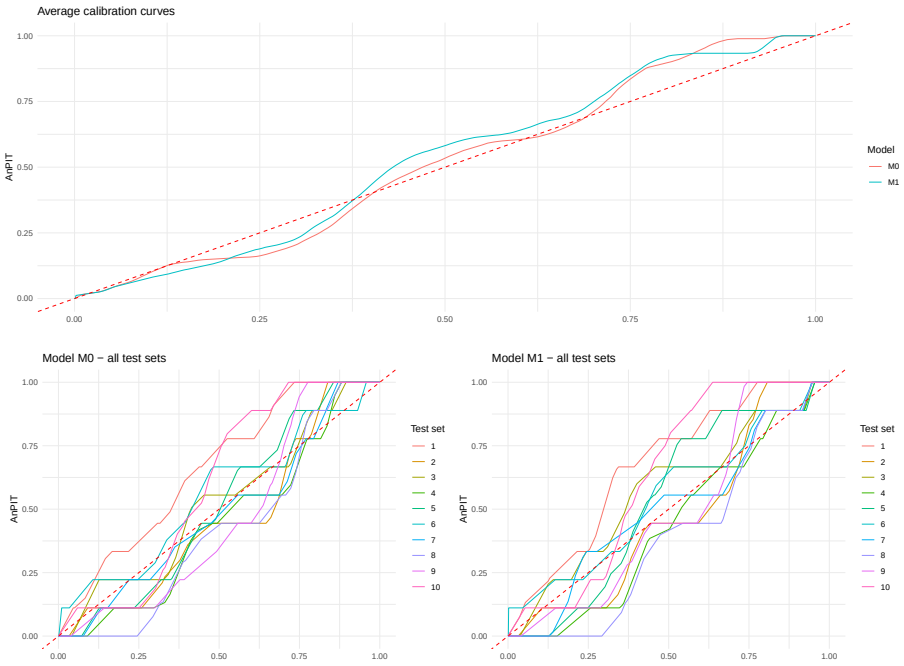



Figure 4.11: The average nonrandomized probability integral transform (AnPIT) averaged across 10 test sets of size 9 (upper panel) and for each of the 10 tests sets (lower panel). The test sets are generated using a spatially regularized sampling scheme.

In the above function, we have used a spatially regularized sampling scheme (`method = "clusterized"`), where the locations are at least 20 kilometers (`min_dist = 20`) apart from each other to create 10 test sets (`iter = 10`) each of size 9 (`n_size = 9`) which corresponds to 10% of the original data-set. The results reported in Figure 4.11, show both the AnPIT for each of the 10 test sets and the averaged AnPIT across the 10 test sets, for both $M_0$ and $M_1$. The AnPIT plots show curves that are not too far from the identity line, which leads us to conclude that both $M_0$ and $M_1$ can be considered to be approximately well calibrated models.

Let us how if these results would change if we were to split the data using a clustering algorithm approach. We then use the `assess_pp`, to create 10 folds and set `method = "cluster"`.

```
cluster <-
assess_pp(list(M0 = M0_fit, M1 = M1_fit),
          which_metric = "AnPIT",
          method = "cluster", fold = 10)
```

```
p1 <- plot_AnPIT(cluster, combine_panels = TRUE) +
      theme(plot.margin = unit(c(0.5, 0.5, 0.5, 0.5), "lines"))
       ↪ # Tighter margins

p2 <- plot_AnPIT(cluster, mode = "all") +
      theme(plot.margin = unit(c(0.5, 0.5, 0.5, 0.5), "lines"))
       ↪ # Tighter margins

grid.arrange(p1, p2, nrow = 2, heights = c(1, 1)) # Equal
 ↪ heights
```

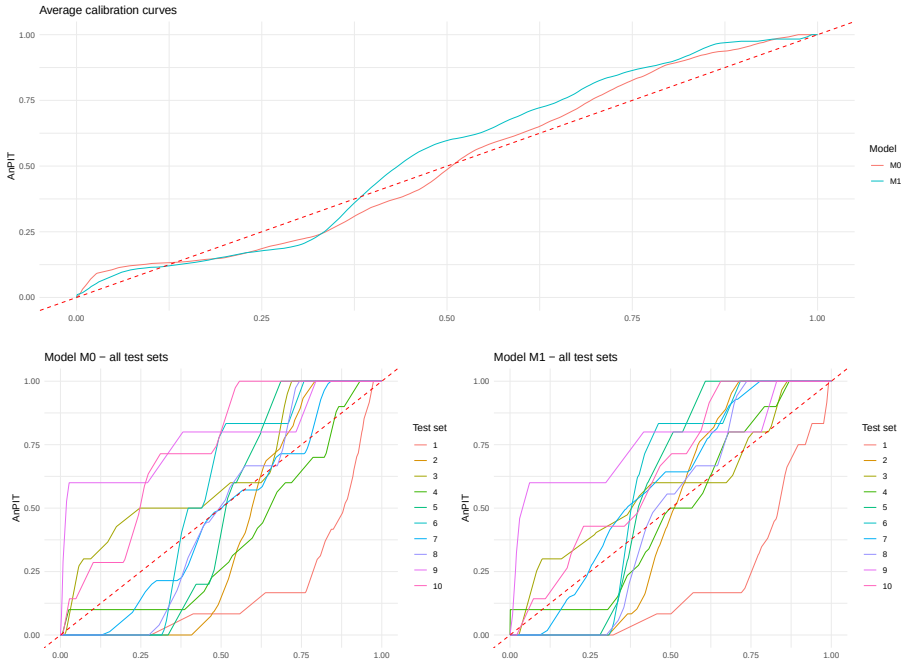



Figure 4.12: The average nonrandomized probability integral transform (AnPIT) averaged across 10 test sets of size 9 (upper panel) and for each of the 10 test sets (lower panel). The test sets are generated using a clustering algorithm based on the locations of the data.

The AnPIT from Figure 4.12 show very similar results to those of Figure 4.11, hence similar conclusions are drawn.

### 4.5.3 Assessing calibration and sharpness using continuous ranked probability scores

We now consider another important aspect that helps to describe the predictive performance of a geostatistical model, namely sharpness. As the name suggests, sharpness refers to the spread of the predictive distribution. A sharp predictive distribution has a narrow spread, indicating that the model is highly confident in its predictions. Importantly, sharpness does not depend on how close predictions are to the true values (i.e., it is independent of accuracy). A model with high sharpness will produce predictions with low variance, but this sharpness is only meaningful if the model is well-calibrated, ensuring that the high confidence aligns with observed values. For example, a geostatistical model for disease prevalence mapping that predicts a very tight range of possible prevalence values for a location is producing sharp predictions. However, if the actual prevalence often falls outside this range, then the predictions

may be sharp but poorly calibrated, thus limiting the value of quantifying sharpness alone.

To assess predictive performance while considering both *sharpness* and *calibration*, we introduce the *continuous ranked probability score (CRPS)* (Gneiting, Balabdaoui, and Raftery 2007) and its scaled variant, the *scaled CRPS (SCRPS)* (Bolin and Wallin 2023). The CRPS evaluates the agreement between the predicted cumulative distribution function of the model $M$, denoted as in the previous section by $F_M$, and the observed value $y_i$. It is formally defined as

$$\text{CRPS}(F_M, y_i) = \int_{-\infty}^{\infty} \left[F_M(y) - \mathbf{1}(y \geq y_i)\right]^2 dy,$$

penalizing deviations of the predictive distribution from the observed value. The CRPS balances sharpness and calibration, with smaller values indicating better predictive performance.

The *scaled CRPS (SCRPS)*, proposed by Bolin and Wallin (2023), refines the CRPS by addressing its dependence on the scale of the outcome variable, which can vary across observations. The SCRPS is defined as

$$\text{SCRPS}(F_M, y_i) = -\frac{1}{2}\left[1 + \frac{\text{CRPS}(F_M, y_i)}{\text{CRPS}(F_M, F_M)} + \log\left\{2\,\text{CRPS}(F_M, F_M)\right\}\right], \tag{4.7}$$

where $\text{CRPS}(F_M, F_M)$ denotes the expected CRPS when the observation is drawn from the predictive distribution itself, i.e., $Y \sim F_M$.

In settings where the predictive distribution is discrete, such as the Binomial and Poisson geostatistical models, a discrete analogue to CRPS is used. Let $F_M(k)$ be the cumulative distribution function over discrete support $\{0, 1, 2, \dots\}$. The discrete CRPS is given by

$$\text{CRPS}(F_M, y_i) = \sum_{k \in \mathcal{S}}^{\infty} \left[F_M(k) - \mathbf{1}(k \geq y_i)\right]^2,$$

where $\mathcal{S}$ represents the support of the outcome $Y_i$, corresponding to $\{0, 1, \dots, m_i\}$ for the Binomial distribution, and the set of non-negative integers as in a Poisson distribution. This version remains proper and interpretable as a measure of calibration and sharpness for integer-valued outcomes. Analogously, a scaled version of the discrete CRPS can be constructed using the same transformation as in the continuous case, by replcing the continuous CRPS with its discrete counterpart in Equation 4.7.

providing a scale-invariant and proper scoring rule for discrete predictive models. These discrete versions are particularly useful in count data models or models for ordinal responses.

The expressions for the CRPS and SCRPS may initially appear complex, but their appropriate application and interpretation lie in understanding their differences and the contexts in which each is more suitable. These differences become more intuitive when considering the more familiar concept of the standard error. The standard error of a predictive distribution represents its uncertainty, but it is not inherently standardized. In many cases, there exists a mean-variance relationship, where the variance increases with the mean. For instance, in Poisson-distributed data, higher mean values correspond to larger variances. This lack of standardization means that directly comparing predictive uncertainties across different scales can be misleading. Similarly, the CRPS is sensitive to the scale of the predictive distribution, as it penalizes errors in absolute terms, often giving disproportionate weight to observations with larger predictive uncertainty.

The SCRPS addresses this issue by standardizing predictive errors relative to the expected spread of the predictive distribution, ensuring *local scale invariance*. This means that the penalty for an incorrect prediction is adjusted proportionally to the scale of the predictive distribution. As a result, the SCRPS enables meaningful comparisons across observations with varying levels of uncertainty. To draw an analogy, the SCRPS standardizes the CRPS in much the same way that the coefficient of variation (CV) standardizes the standard error. By dividing the standard error by the mean of the predictive distribution, the CV expresses variability relative to the mean. This property is particularly valuable in geostatistical applications, where uncertainty often varies spatially due to factors such as non-stationarity of the predictive distribution or irregularly spaced observations.

However, while the SCRPS is more robust in settings where variability differs significantly across observations, there are contexts where the CRPS may be preferred. Specifically, in cases where greater emphasis should be placed on accurately predicting observations with larger uncertainty, the CRPS is more appropriate because it assigns greater weight to predictions with higher variance. For example, as highlighted in Bolin and Wallin (2023), when forecasting rare but impactful events (e.g., extreme weather conditions), the larger uncertainties associated with these events may naturally demand greater attention. In such scenarios, the CRPS's sensitivity to scale can be an advantage, as it prioritizes minimizing errors for these high-variance predictions.

When evaluating the predictive performance of a model $M$, we proceed as follows. Let $\left(x_1^{(i)}, \dots, x_{m_i}^{(i)}\right)$ represent the set of locations in the $i$-th test set, for $i = 1, \dots, T$. For each location, we compute the score—either CRPS or SCRPS—and denote it as $r_M\left(x_j^{(i)}\right)$.

To assess the overall performance of model $M$, we calculate a weighted average of the scores across all test sets, where the weights are proportional to the sample size of each test set. The final score assigned to model $M$ is given by:

$$\frac{1}{\sum_{i=1}^{T} m_i} \sum_{i=1}^{T} \sum_{j=1}^{m_i} r_M\left(x_j^{(i)}\right). \tag{4.8}$$

Models assumed to be well-calibrated and achieving a lower score based on the above expression are considered to provide sharper predictions and, consequently, better predictive performance.

#### 4.5.3.1 Example: mapping riverblindness in Liberia (continuing from Section 4.5.2.1)

In Section 4.5.2.1, we have assessed the calibration of models $M_0$ and $M_0$ and found these to be reasonably well calibrated models. We now compare the predictive performance of the two, using both the CRPS and SCRPS metrics introduced previously.

The `assess_pp` function also allows the user to compute the CRPS and SCRPS through the argument `which_metric` as shown below.

```
regularized_scores <-
assess_pp(list(M0 = M0_fit, M1 = M1_fit),
          which_metric = c("CRSP", "SCRPS"),
          method = "regularized", min_dist = 20,
          n_size = 9, iter = 10)

cluster_scores <-
assess_pp(list(M0 = M0_fit, M1 = M1_fit),
          which_metric = c("CRSP", "SCRPS"),
          method = "cluster", fold = 10)
```

Note that that `assess_pp` can be run once in order to obtain both the AnPIT diagnostic, introduced in Section 4.5.2; by default `which_metric = c("AnPIT", "CRSP", "SCRPS")` which computes all three diagnostic using the same test sets. The reason we have separated the AnPIT and the scores computations is only for pedagogical purposes.

By doing a `summary` of the output in `regularized_scores` and `cluster_scores`, we obtain the average score across all test sets as defined in Equation 4.8.

```
summary(regularized_scores, view_all = FALSE)
## Summary of Cross-Validation Scores
## ---------------------------------
## Model: M0
```

```
##   Overall average across test sets:
##     CRPS: -0.0384
##     SCRPS: 0.2488
##
## Model: M1
##   Overall average across test sets:
##     CRPS: -0.0415
##     SCRPS: 0.2091

summary(cluster_scores, view_all = FALSE)
## Summary of Cross-Validation Scores
## ---------------------------------
## Model: M0
##   Overall average across test sets:
##     CRPS: -0.0464
##     SCRPS: 0.1635
##
## Model: M1
##   Overall average across test sets:
##     CRPS: -0.0446
##     SCRPS: 0.1683
```

In the code above, we have set `view_all = FALSE` to display only the overall average. If `view_all = TRUE`, the scores are instead reported for each test set, in addition to the overall average. The results from the `summary` suggest that the two models have a similar predictive performance in terms of both CRPS and SCRPS, regardless of the sampling scheme used to create the test sets. To examine this further we can plot a point map of the CRPS and SCRPS using the locations of the test sets.

```
# Define ranges for the color scales

crps_range <-
 ↪  range(c(unlist(cluster_scores$model$M0$score$CRPS),
                     unlist(cluster_scores$model$M1$score$
                      ↪  CRPS)))
                      ↪
scrps_range <-
 ↪  range(c(unlist(cluster_scores$model$M0$score$SCRPS),
                     unlist(cluster_scores$model$M1$score$
                      ↪  SCRPS)))
                      ↪

# Generate the plots with consistent color scales
p1 <- plot_score(cluster_scores,
```

```
                  which_score = "CRPS", which_model = "M0") +
      scale_color_viridis_c(option = "C",
      limits = crps_range, direction = 1)
p2 <- plot_score(cluster_scores,
                  which_score = "CRPS", which_model = "M1") +
      scale_color_viridis_c(option = "C",
      limits = crps_range, direction = 1)
p3 <- plot_score(cluster_scores,
                  which_score = "SCRPS", which_model = "M0") +
      scale_color_viridis_c(option = "C",
      limits = scrps_range, direction = 1)
p4 <- plot_score(cluster_scores,
                  which_score = "SCRPS", which_model = "M1") +
      scale_color_viridis_c(option = "C",
      limits = scrps_range, direction = 1)
grid.arrange(p1, p2, p3, p4, nrow = 2, ncol = 2)
```

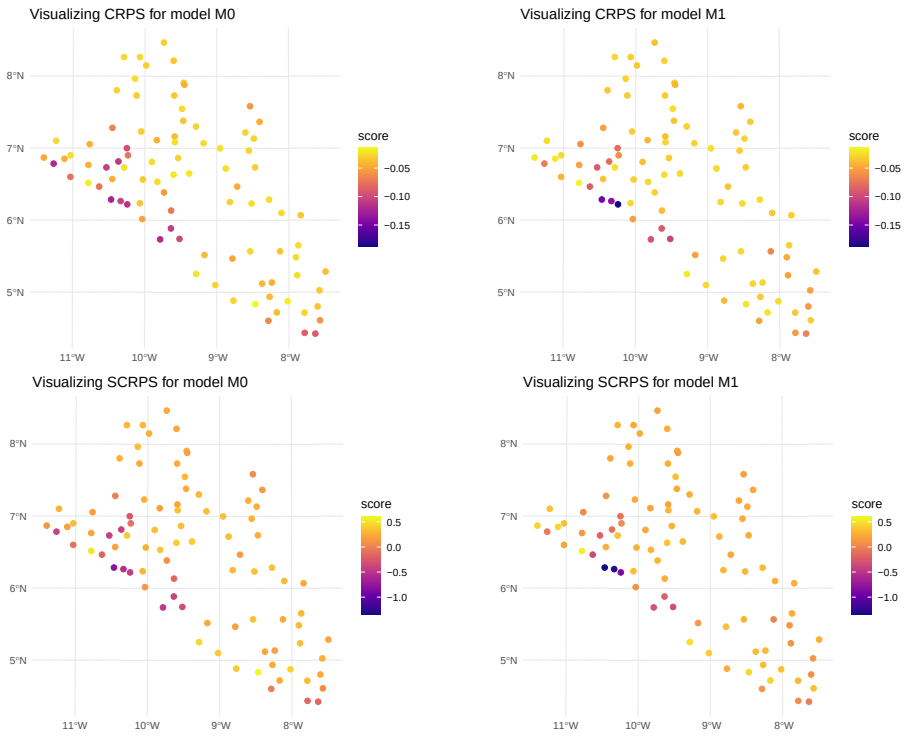



Figure 4.13: Point maps of the scores for model $M_0$ and $M_1$ across the test sets generated using a clusering approach for splitting the data. The upper panels are the map for the continuous ranked probability score (CRPS) and the lower panels for the scale CRPS (SCRPS).

In the code above, we evaluated the cross-validation results generated using a clustering algorithm to split the original data-set. The map reveals that the scores across most locations are generally similar, with only three locations near the coast of Liberia showing slightly lower CRPS and SCRPS values for $M_1$. This suggests a consistent predictive performance across the dataset, with some localized variations.

## 4.6 Simulation-based assessment of predictive performance

In Section 4.5, we illustrated how to evaluate predictive performance in relative terms, specifically by comparing candidate models under consideration. In this section, we demonstrate how simulation studies can be used to assess the predictive performance of a geostatistical model in predicting a spatially continuous target within a region of interest. Since a spatially continuous target cannot be directly observed, simulations provide a way to generate such targets under specified assumptions. This approach allows us to evaluate how effectively the model can recover the underlying spatial surface using suitable predictive performance metrics.

### 4.6.1 Step 1: Define the aim of the simulation study

As previously stated, one of the main reasons for carrying out a simulation study is the ability to generate spatial surfaces, which are typically unobservable in the real world. The purpose of the simulation is generally to evaluate how well a given geostatistical model can recover that surface and/or draw inferences on some of its properties.

To fully understand the how to carry out a simulation study, it is essential to define its two main components: the *true model* and the *candidate models* (which can indeed include multiple options).

The **true model** represents the underlying stochastic process that generates the data. In a simulation study, the true model is assumed to be known because it is explicitly specified by us. It serves as the benchmark against which other models are assessed.

The **candidate models** are the geostatistical models that are tested or evaluated during the simulation study. These models are proposed alternatives to approximate or recover the true spatial surface or its properties. Candidate models may differ in terms of their assumptions and model complexity.

It is important to highlight that the true model not only serves as a benchmark but can also be included among the candidate models. However, the true model

does not always necessarily outperform other candidate models. In some cases, certain candidate models may provide better approximations of the spatial surface or its properties, particularly when the true model's complexity is too high relative to the available sample size. Here, *model complexity* refers to the number of parameters in a model. For instance, a true geostatistical model with high complexity may include numerous covariates or a highly parameterised covariance function. When the sample size is insufficient to reliably estimate these parameters, simpler candidate models can sometimes yield better performances.

Consider an extreme example where the true model is a spatial Gaussian process $S(x)$ with a scale parameter of $\phi = 1$. If only a very small fraction of sampled locations are at distances below 1, estimating $\phi$ accurately becomes highly challenging, especially with a small sample size. In particular the sparsity of close pairs reduces the information available to estimate the spatial dependence structure, making the estimation of $\phi$ imprecise or even unreliable. In this case, a candidate model that does not attempt to explicitly estimate the spatial dependence (e.g., a model that relies solely on spatial covariates to interpolate the surface) might perform better in recovering the spatial structure. This example suggests that a simpler or alternative model can sometimes outperform the true model under specific sampling and data constraints.

At this point, a natural question arises: how should we choose the true model for a simulation study? The answer depends on the specific objectives of the study and what we aim to demonstrate. We now provide an example to demonstrate this point.

#### 4.6.1.1 Example

In our analysis of the river blindness dataset, we now aim to evaluate the effectiveness of using elevation as a spatial predictor. To achieve this, we conduct a simulation study to compare the performance of two geostatistical models: one that incorporates elevation as a predictor and another that excludes it. This example will be revisited throughout the steps outlined in this section on simulation. We can chose the following true model (henceforth $M_T$) for the simulation

$$M_T : \log\left\{\frac{p(x_i)}{1-p(x_i)}\right\} = \beta_0 + \beta_1 e(x_i) + \beta_2 \max\{e(x_i) - 150, 0\} + S(x_i). \tag{4.9}$$

The true model, described by the equation above, assumes a nonlinear relationship with elevation based on a linear spline with a single knot at 150 meters. To simulate data from the true model, which constitutes the next step in our simulation study, we must first define the model parameter values. A reasonable approach is to use the maximum likelihood estimates obtained by fitting the binomial geostatistical model specified in Equation 4.9. These

estimates can be derived as shown below.

```
set.seed(123)
true_model <- glgpm(npos ~ elevation + pmax(elevation - 150, 0)
 ↪  +
                           gp(long, lat),
                     den = ntest,
                     crs = 4326,
                     convert_to_crs = 32629,
                     family = "binomial",
                     data = liberia)
```

```
# The parameters of the true model
coef(true_model)
```

```
$beta
             (Intercept)                elevation
             pmax(elevation - 150, 0)
            -2.002985655              0.003160402
            -0.001706721

$sigma2
[1] 0.3181267

$phi
[1] 41.02334
```

A natural candidate model to evaluate the usefulness of elevation as a covariate is an intercept-only model (referred to here as $M_C$). This model is defined as:

$$M_C : \log\left\{\frac{p(x_i)}{1-p(x_i)}\right\} = \beta_0 + S(x_i).$$

where no covariates are included. This approach makes $M_C$ a suitable baseline model, commonly selected when the objective is to assess the contribution of covariates to spatial prediction.

### 4.6.2 Step 2: Simulate the spatial surface and the data from the true model

In Section 3.3.3, we have shown how to simulate from a geostatistical model, when the goal is simulate data at the observed locations. When we want to simulate a spatial surface, a similar approach can be used. First of all, we need to generate a regular grid, say $\tilde{X}$, covering the study region of interest, say $A$. Based on the notation used so far, we use $X$ to denote the set of observed locations which are used to simulate the outcome data $Y$. Finally, we use $X_+ = (X, \tilde{X})$ to denote the full set of locations both on the grid

and at the data points. We can then proceed through the following steps to simultaneously simulate values of the surface of interest over $\tilde{X}$ and $X$.

1. Compute the covariance matrix $\Sigma_+$ using the combined set of locations $X_+$.
2. If covariates are used, compute the covariates effects $d(x)^\top\beta$ for all the locations in $X_+$.
3. Simulate $B$-times from a multivariate Gaussian distribution with mean vector 0 and covariance $\Sigma_+$ as previously defined. We denote the output form this step as $S_{(j)}(\tilde{x})$, for simulations generate on $\tilde{X}$, and with $S_{(j)}(x)$ for simulations on $X$, respectively, for $j = 1, \ldots, B$.
4. Compute the linear predictor at each locations of $X_+$, by adding the covariates effects in 3 to the simulated random effects in 4, hence $g\{\mu_{(j)}(\tilde{x})\} = d(\tilde{x})^\top + S_{(j)}(\tilde{x})$ for $\tilde{X}$, and $g\{\mu_{(j)}(x)\} = d(x)^\top + S_{(j)}(x)$ for $X$.
5. Only for locations of the data $X$, perform the steps 5A to 7A as explained in Section 3.3.3, to simulate the data $Y$.

The function `surf_sim` in `RiskMap` can be used to carry out all these steps automatically as shown next.

#### 4.6.2.1 Example

Following the steps previously outlined, we first create a grid within the boundaries of Liberia and extract the values of elevation on the grid from a raster.

```
library(rgeoboundaries)
shp <- geoboundaries(country = "liberia", adm_lvl = "adm0")
shp <- st_transform(shp, crs= 32629)

pred_grid <- create_grid(shp, spat_res = 5)
pred_grid <- st_as_sf(pred_grid)
liberia_elev <- get_elev_raster(locations = shp,
                                z = 5, clip = "locations")

pred_grid$elevation <- terra::extract(liberia_elev,
 ↪  st_coordinates(pred_grid))
```

The function `surf_sim` also requires a sampling functions that generates the locations $X$ from which data $Y$ are simulated. In this example, we shall consider a sampling function that simply returns the locations of the original data.

```
sampling_f_lib <- function() {
  coords_sf <- st_as_sf(liberia[,c("ntest", "long", "lat")],
                        coords = c("long", "lat"), crs = 4326)
  coords_sf <- st_transform(coords_sf, 32629)
  coords_sf$units_m <- coords_sf$ntest
  return(coords_sf)
}
```

The output of the sampling function must be an `sf` object that must contain the number of tested people and its corresponding column must be named `units_m`. Also, the returned `sf` object must have a projection that correspond to the one used when fitting the model.

We are now ready to run the `surf_sim` function.

```
lib_surf_sim <- surf_sim(n_sim = 200,
                         pred_grid = pred_grid,
                         formula = ~ elevation + pmax(elevation
                          ↪  - 150,0) +
                                     gp(kappa=0.5),
                         sampling_f = sampling_f_lib,
                         family = "binomial",
                         par0 = coef(true_model))
```

In the above function, we have simulated $B = 200$ simulations from the true model as specified in `formula`. The argument `par0` is used to pass the values of the parameters of the true model.

We can now inspect the individual simulated surfaces using the `plot_sim_surf` function, as demonstrated below, which produces Figure 4.14. This step is useful verifying that the simulations are consistent with expectations and free from any warning signs or anomalies.

```
plot_sim_surf(lib_surf_sim, sim = 1)
```

### 4.6.3 Step 3: Fit the candidate models to the simulated data and predict the target

To summarize, the results obtained from Step 1 and Step 2 are as follows:

1. Simulated values for the linear predictor of the true model ($M_T$) over a grid covering the study area.
2. Simulated values for the linear predictor of $M_T$ at the data locations.
3. Simulated realizations of the data at the data locations.

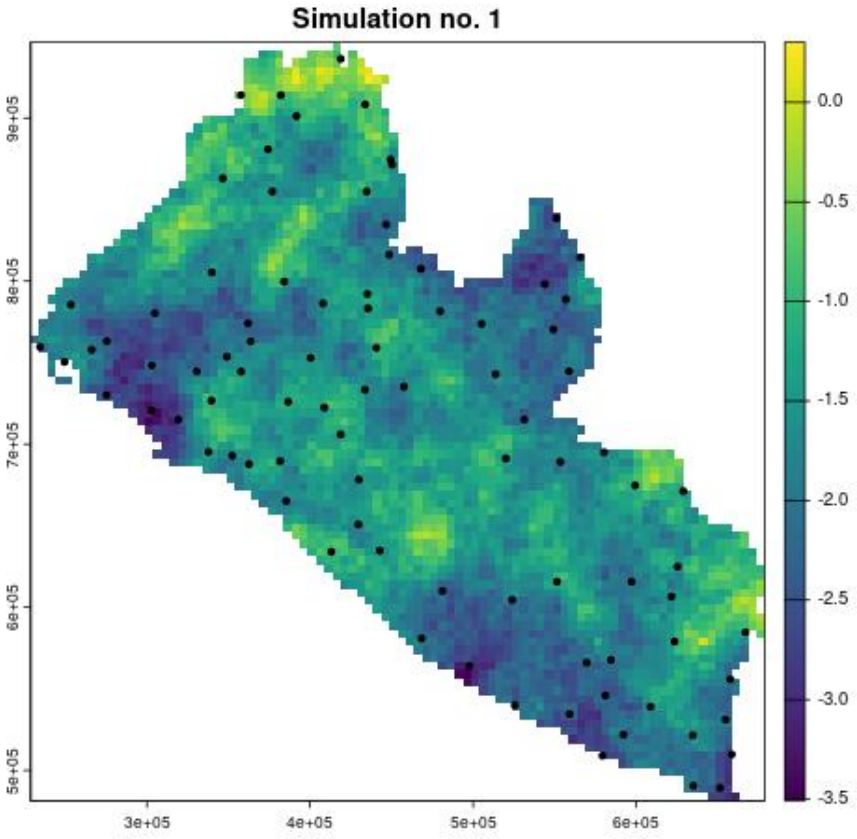



Figure 4.14: Simulated surface for the linear predictor, with locations of the simulated data noted by solid black points.

Next, we use the third output listed above to fit the candidate models ($M_C$) and make predictions over the grid. Note that each of the outputs above has been generated $B$ times, which denotes the number of simulations.

Using the notation introduced in Section 4.3, let $T(x)$ denote the spatially continuous target. When the predictive target is at the areal level, the necessary adjustments to Step 3 and Step 4 will be detailed in Section 4.6.5.

The outputs generated in these steps include predictive samples of the target over the grid, represented by $T^{(h)}_{(j)}(\tilde{x})$, where: $j$ is the index of the $j$-th simulated data-set, for $j = 1, ..., B$; $h$ is the index of the Monte Carlo samples, for $h = 1, ..., N$, generated by the Monte Carlo Markov Chain (MCMC) algorithm for a given data-set $j$.

An example in R for Step 3 will be provided in the next section, where we shall use the function `assess_sim`, from the `RiskMap` package, which combines Step 3 and Step 4 into a single step.

### 4.6.4 Step 4: Summarize the results using a pre-defined objective function

In this final step, we compare the predictions of the target $T(x)$ against its true values, originally simulated in Step 1. To do this, we need a metric to summarize the results across all $B$ simulations for each candidate model $M_C$.

One approach is to use the predictive mean from $M_C$ in each simulation and evaluate how it differs from $T(x)$. A natural choice for this comparison is the mean squared error (MSE), defined as:

$$\frac{1}{B}\sum_{j=1}^{B}\sum_{i=1}^{q}\left(T(\tilde{x}_i) - \hat{T}_C(\tilde{x}_i)\right)^2 \tag{4.10}$$

where $\tilde{x}_i$ is the $i$-th location on the grid $\tilde{X}$, $T(\tilde{x}_i)$ represents the true value of the target, and $\hat{T}_C(\tilde{x}_i)$ is the predictive mean from $M_C$ at location $\tilde{x}_i$.

Other options for summarizing the results include using the bias or the absolute median deviation. The bias measures the systematic error between the predicted and true values and is defined as:

$$\frac{1}{B}\sum_{j=1}^{B}\sum_{i=1}^{q}\left(T(\tilde{x}_i) - \hat{T}_C(\tilde{x}_i)\right)$$

The median absolute deviation (MAD) focuses on the central tendency of the errors, providing a robust measure of variability. It is defined as:

$$\text{median}\left(|T(\tilde{x}_i) - T_C^*(\tilde{x}_i)|\right),$$

where $T_C^*(\tilde{x}_i)$ is the median across all $N$ the Monte Carlo samples $T_{(j)}^{(h)}(\tilde{x})$.

#### 4.6.4.1 Example

Using the output from the `surf_sim` function, named `lib_surf_sim`, we can use as the input to the function `assess_sim` which performs Step 3 and Step 4, previously outlined.

```
res_sim_grid <- assess_sim(lib_surf_sim,
                      models = list(M_T = ~  elevation +
                                             pmax(elevation -
                                              ↪  150, 0) +
                                             gp(long, lat),
                                    M_C = ~ gp(long, lat)),
                      f_grid_target = function(x)
                       ↪  exp(x)/(1+exp(x)),
                      pred_objective = "mse",
                      spatial_scale = "grid")
```

In this function, we specify the models to be assessed using the `models` argument, which is a list of `formula` objects. The formula `M_T` represents the true model $M_T$, while `M_C` corresponds to the intercept-only model. Note that the coordinate names used in the `gp()` function within the formulas are converted according to the CRS specified in `lib_suf_sim`.

The argument `f_grid_target` defines our target function $T(x)$ over the grid as a transformation of the linear predictor. Specifically, `exp(x) / (1 + exp(x))` expresses disease prevalence as our predictive target over the grid. The `pred_objective` argument specifies the objective function from Step 4, with `pred_objective = "mse"` setting it to the mean squared error from Equation 4.10. Finally, `spatial_scle = "grid"` indicates that the objective function applies to a pixel-level target according to the grid stored in `lib_surf_sim`.

We can now look at the summary of `res_sim_grid`.

```
summary(res_sim_grid)
```

```
Summary of Simulation Results

Mean Squared Error (MSE):
    Model    MSE_mean       MSE_sd
M_T   M_T 0.004426811 0.0009040507
M_C   M_C 0.004390553 0.0008564500
```

The results reported here present the average and standard deviation across 200 simulations for the models $M_C$ and $M_T$. The differences between the two models are negligible, indicating that the point predictions generated by $M_C$ and $M_T$ perform similarly in recovering the true prevalence surface for river blindness in Liberia.

This result should not be surprising, as estimating regression relationships with prevalence data can be challenging, potentially limiting their predictive advantage. However, incorporating covariates enhances model interpretability, allowing us to understand why the model makes certain predictions in specific areas, which an intercept-only model cannot provide. Thus, even if the model with covariates does not achieve the best predictive performance, we may still choose to retain the covariates, in this case elevation, for their interpretative value.

### 4.6.5 Assessessment of threshold-based classification of spatial units using geostatistical models

In this section, we address the specific problem of evaluating how effectively a geostatistical model classifies areal units into predefined categories of a disease indicator, such as prevalence. This issue is particularly relevant when dealing with areal-level targets (Section 4.4) as opposed to spatially continuous targets (Section 4.3). This prediction problem arises when policy decisions regarding interventions are often based on the classification of the areal-level target into predefined categories at the level of administrative regions; see, for example, Johnson et al. (2021) and Puranik et al. (2024).

To formalize this problem mathematically, let $\mathcal{M}_i$ represent the true value of the areal-level target (e.g., the average prevalence over a district), and let $\widehat{\mathcal{M}}_{C,i}$ denote the corresponding prediction produced by the candidate model $M_C$. Note that $\mathcal{M}_i$ can be easily calculated in Step 1 during simulations from the true model (Section 4.6.1).

Now, consider that the target $\mathcal{M}_i$ is categorized into predefined classes, which are typically thresholds or ranges of the disease indicator (such as values that define low, medium, and high levels). Denote the true membership class of areal unit $i$ by $\mathcal{D}_i$, which is determined based on $\mathcal{M}_i$, and the predicted membership class by $\widehat{\mathcal{D}}_{C,i}$, determined based on $\widehat{\mathcal{M}}_{C,i}$. The goal is to assess the overall quality of the classification, which involves evaluating how well the predicted classes $\widehat{\mathcal{D}}_{C,i}$ agree with the true classes $\mathcal{C}_i$ across all areal units and simulations. This assessment provides insight into the model's ability to accurately classify regions according to the chosen disease indicator and thus guide effective policy decisions.

The required changes to Step 3 (Section 4.6.3) consist of simply carrying the predictions over the areal units as explained in Section 4.4.

For Step 4, which concerns how we summarize the agreement between the classification generated by $M_C$ and the true classes $\mathcal{D}_i$, we can proceed as follows. First, we distinguish between indicators that can be computed for a given region and those that summarize performance across all regions.

Assessing classification performance can be approached similarly to evaluating a diagnostic test. Here, true positives, true negatives, false positives, and false negatives are defined at the level of individual spatial units that subdivide the study area. Hence, key indicators include:

1. *Sensitivity*: The proportion of areal units correctly classified as belonging to the true membership class when conditioned on that true class:

$$\text{Sensitivity} = \frac{\text{True Positives (TP)}}{\text{True Positives (TP)} + \text{False Negatives (FN)}}$$

2. *Specificity* (True Negative Rate): The proportion of areal units not belonging to a given membership class that are correctly classified as not being in that class:

$$\text{Specificity} = \frac{\text{True Negatives (TN)}}{\text{True Negatives (TN)} + \text{False Positives (FP)}}$$

3. *Positive Predictive Value (PPV)*: The proportion of areal units predicted to belong to a membership class that are correctly classified, conditioned on the predicted membership class:

$$\text{PPV} = \frac{\text{True Positives (TP)}}{\text{True Positives (TP)} + \text{False Positives (FP)}}$$

4. *Negative Predictive Value (NPV)*: The proportion of areal units predicted not to belong to a membership class that are correctly classified, conditioned on the predicted membership class:

$$\text{NPV} = \frac{\text{True Negatives (TN)}}{\text{True Negatives (TN)} + \text{False Negatives (FN)}}$$

5. *Correct Classification Rate*: The overall proportion of areal units whose predicted membership class matches the true membership class:

$$\text{Correct Classification Rate} = \frac{\text{True Positives (TP)} + \text{True Negatives (TN)}}{\text{Total Cases (TP + TN + FP + FN)}}$$

Note that for multiple classes, the indicators above are computed by comparing each class in turn against all other classes combined. This allows us to create a 2 by 2 table, enabling the calculation of the indicators above.

The choice between sensitivity, specificity, PPV, and NPV depends on the specific public health context and the priorities of the decision-making process. Sensitivity and specificity are conditioned on the *true membership class*, making them useful for assessing the model's performance in terms of correctly identifying or excluding true positives or negatives. PPV and NPV are conditioned on the *predicted membership class*, making them more informative about the trustworthiness of predictions, especially when prevalence varies significantly across regions.

We can thus summarize the difference of each indicator as follows.

- Sensitivity is most appropriate when the priority is to ensure that true positive cases (e.g., high-prevalence regions) are not missed. Sensitivity is useful in contexts where failing to identify true cases could have severe consequences, such as delaying critical interventions or underestimating resource needs. For example, during an outbreak, ensuring all high-risk areas are flagged is crucial, even if this increases the number of false positives.
- Specificity is more suitable when the focus is on minimizing false positives, which is critical in contexts where over-classification could lead to unnecessary interventions or misallocation of resources. Specificity may be particularly important in low-prevalence scenarios, where the cost of acting on false alarms could outweigh the benefits of capturing all true positives.
- PPV is appropriate when the emphasis is on the reliability of positive predictions, i.e., how confident we can be that regions classified as high-prevalence are truly high-prevalence. PPV is heavily influenced by the prevalence of the condition being classified, making it especially relevant in scenarios where actionable decisions depend on the certainty of a positive classification.
- NPV is most relevant when it is critical to trust negative predictions, i.e., ensuring that regions predicted as low-prevalence are indeed low-prevalence. NPV is particularly important in low-prevalence settings, where a high proportion of predicted negatives are true negatives.

Ultimately, the selection of indicators should result from a dialogue between statisticians, epidemiologists, and policy makers, ensuring alignment with the specific priorities and policy decisions being informed by the model's classifications.

#### 4.6.5.1 Example

Using the same models, $M_T$ and $M_C$, from the river blindness example in Liberia, we now compare their performance in classifying admin level 1 regions into two prevalence categories, $(0, 0.2)$ and $(0.2,\ 1)$. The assessment is conducted using `assess_sim`, where the prevalence surface is first aggregated at the areal level before classification.

```
# Admin level 1 shape files
shp_adm <- geoboundaries(country = "liberia", adm_lvl = "adm1")
shp_adm <- st_transform(shp_adm, crs= 32629)

res_sim_area <- assess_sim(lib_surf_sim,
                      models = list(M_T = ~  elevation +
                                      pmax(elevation - 150, 0) +
                                      gp(long, lat),
                                    M_C = ~ gp(long, lat)),
                      f_grid_target = function(x)
                       ↪  exp(x)/(1+exp(x)),
                      f_area_target = mean,
                      pred_objective = "classify", shp =
                       ↪  shp_adm,
                      categories = c(0,0.2,1),
                      spatial_scale = "area")
```

The implementation first loads the admin level 1 shapefiles for Liberia, transforming them to the appropriate coordinate reference system. The arugment `f_area_target` is set to `mean`, ensuring that the target prevalence for each region is computed as the average over the area, corresponding to

$$\frac{1}{|A|}\int_A T(x)\,dx$$

The classification is performed by specifying `pred_objective = "classify"`, meaning the models are evaluated based on their ability to assign regions to the correct prevalence class. The `categories` argument defines the classification thresholds, requiring at least three values to distinguish between two classes. The `spatial_scale` is set to `"area"`, meaning predictions are made at the level of administrative regions which are specified by the shapefile `shp_adm`, passed to the argument `shp`.

Finally, we can examine the results of the simulation through the summary function.

```
summary(res_sim_area)
```

```
Summary of Simulation Results

Classification Results:

Model: M_T

Averages across simulations by Category:
  Sensitivity Specificity       PPV       NPV        CC   Class
```

```
1   0.9020851   0.8151707 0.8923583 0.8654312 0.9020851 (0,0.2]
2   0.8151707   0.9020851 0.8654312 0.8923583 0.8151707 (0.2,1]

Proportion of Correct Classification (CC) across categories:
Mean: 0.871, 95% CI: [0.665, 1.000]

Model: M_C

Averages across simulations by Category:
  Sensitivity Specificity       PPV       NPV        CC   Class
1   0.9048404   0.7974989 0.8890838 0.8581346 0.9048404 (0,0.2]
2   0.7974989   0.9048404 0.8581346 0.8890838 0.7974989 (0.2,1]

Proportion of Correct Classification (CC) across categories:
Mean: 0.868, 95% CI: [0.600, 1.000]
```

In this example, the performance of models $M_T$ and $M_C$ is comparable across all considered indicators. This similarity arises for the same reasons previously discussed examining the quality of prevalence surface predictions.

## 4.7 Theory

### 4.7.1 Expression of the predictive distribution for a linear geostatistical model

In this section, we consider the simple case of a linear geostatistical model with a single observation per location.

Let $Y = (Y_1, \dots, Y_n)^\top$ be the vector of data associated with spatial locations $x_1, \dots, x_n$. The model assumes that:

$$Y_i = d(x_i)^\top \beta + S(x_i) + Z_i, \quad i = 1, \dots, n, \tag{4.11}$$

where $d(x_i)$ is a vector of covariates at location $x_i$, $\beta$ is a vector of regression coefficients, $S(x)$ is a stationary Gaussian process with mean zero and Matern correlation function $\rho(u)$ (where $u$ is the distance between locations), and $Z_i \sim \mathcal{N}(0, \tau^2)$ represents independent measurement errors (nugget effect).

We first consider the simple case of a linear target, hence $T(x) = d(x)^\top \beta + S(x)$. The goal is to predict the unobserved process $T^* = (T_{n+1}, \dots, T_{n+q})^\top$ at new locations $x_{n+1}, \dots, x_{n+q}$. The predictive distribution of $T^*$, conditional on the observed data $Y$, follows a multivariate Gaussian distribution:

$$T^*|Y \sim \mathcal{N}(\mu^*, \Sigma^*), \tag{4.12}$$

where the conditional mean and covariance are given by:

$$\mu^* = D^*\beta + C\Sigma^{-1}(Y - D\beta), \tag{4.13}$$
$$\Sigma^* = V - C\Sigma^{-1}C^\top. \tag{4.14}$$

Here, $D^*$ is the design matrix for the prediction locations, $C$ is the cross-covariance matrix between $T^*$ and $Y$, $\Sigma$ is the covariance matrix of $Y$, and $V$ is the covariance matrix of $T^*$. Using a multivariate Gaussian distribution with mean $\mu^*$ and covariance matrix $\Sigma^*$, we are now able to predict $T^*$ and obtain summaries of uncertainties.

For non-linear prediction targets, i.e. targets that involve non-linear transformations of the spatial process $S(x)$, say $W(T^*)$, Monte Carlo methods can be employed. More specifically, we generate samples $T^*_{(j)}$, for $j = 1, \ldots, B$ from the predictive distribution $T^*|Y$ using the multivariate Gaussian distribution with mean $\mu^*$ and covariance $\Sigma^*$. For each sample $T^*_{(j)}$, the non-linear transformation $W(T^*_{(j)} = W^*_{(j)}$ of interest is then computed. The predictive summary statistics, such as the mean, variance, or quantiles of $W(T^*)$, are then estimated by averaging over the simulated samples, e.g.

$$E[W(T^*)|Y] \approx \frac{1}{B}\sum_{J=1}^{B} W^*_{(j)}), \tag{4.15}$$

$$\mathrm{Var}[W(T^*)|Y] \approx \frac{1}{B-1}\sum_{j=1}^{B}\left(W^*_{(j)} - E[W(T^*)]\right)^2. \tag{4.16}$$

For more detailed explanations on this we refer you to Chapter 6 of P. Diggle and Ribeiro (2007).

### 4.7.2 A brief overview of scoring rules

We now provide an overview of the theoretical framework of *scoring rules*, of which the continuous ranked probability score (CRPS) and its scaled variant (SCRPS), previously presented, are an example. Scoring rules are a tool used in the evaluation of probabilistic forecasts. As shown for CRPS and SCRPS, scoring rules are designed to measure how well a predictive distribution aligns with observed outcomes, balancing both *calibration* and *sharpness*.

In general, a scoring rule is a function $G(F, y)$ that assigns a numerical score to a predictive distribution $F$ and an observed outcome $y$. Lower scores typically

indicate better predictive performance, as they reflect a closer match between the predicted distribution and the observed value.

A central concept in the theory of scoring rules is that of *proper scoring rules*. A scoring rule is said to be *proper* if it favours honest predictions or, in other words, if the expected score is minimized when the predictive distribution $F$ matches the true data-generating distribution $F^*$. More formally, a scoring rule $G$ is proper if, for all distributions $F$ and $F^*$, we have:

$$E_{Y\sim F^*}[G(F^*, Y)] \leq E_{Y\sim F^*}[G(F, Y)],$$

with equality if and only if $F = F^*$ when the rule is *strictly proper*. This property is crucial because it ensures that forecasters cannot improve their scores by manipulating their predictions to deviate from their true beliefs.

Proper scoring rules encourage the selection of models that yield predictions that are both well-calibrated (reflecting the true distribution of outcomes) and sharp (concentrated around likely values). However, this does not imply that predictions from the (unknown) true model $F^*$ will always be favoured in practice, when dealing with finite samples. In other words, even if two scoring rules are proper, this does not guarantee that the true data-generating model will always achieve a lower score compared to a misspecified model in finite samples. *Sample variability* can in fact lead to the selection of a misspecified model by chance, especially when the sample size is small or the data are noisy. *Model complexity* is another important factor. If the true model $F^*$ is highly parameterised and the sample size is insufficient to estimate its parameters accurately, it may perform poorly in predictive tasks due to overfitting or parameter estimation error. In contrast, a simpler misspecified model, despite not capturing the full complexity of the data-generating process, might generalize better in such scenarios. This is related to the bias-variance trade-off, where a model with higher bias (due to misspecification) can still achieve lower overall predictive error if it has substantially lower variance.

The CRPS and SCRPS are examples of a strictly proper scoring rule for continuous outcomes. Other scoring rules are the following:

- *Logarithmic Score (or Log Score):* For a predictive density $f(y)$, the log score is defined as:
$$G_{\log}(f, y) = -\log f(y).$$
This rule heavily penalizes low-probability predictions for observed outcomes, making it sensitive to the tails of the predictive distribution.

- *Brier Score:* For binary outcomes, the Brier score measures the mean squared difference between the predicted probability $p$ of an event and the actual outcome $y \in \{0, 1\}$:
$$G_{\text{Brier}}(p, y) = (p - y)^2.$$
The Brier score is both proper and easy to interpret, with connections to mean squared error in classical regression.

- *Energy Score:* A generalization of the CRPS to multivariate outcomes, the energy score for a predictive distribution $F$ and observation $y$ is given by:

$$G_{\text{energy}}(F, y) = E_{X,X' \sim F}[\|X - X'\|] - 2E_{X \sim F}[\|X - y\|],$$

  where $\|\cdot\|$ denotes a norm, typically the Euclidean norm. The energy score is particularly useful for evaluating spatial or multivariate forecasts.

Many proper scoring rules, including the CRPS, can be decomposed into components that separately capture calibration and sharpness. This decomposition helps to understand the different aspects of predictive performance. Specifically, the expected score can often be expressed as:

$$E[G(F, Y)] = \underbrace{\text{Uncertainty}}_{\text{inherent in data}} - \underbrace{\text{Resolution}}_{\text{sharpness of predictions}} + \underbrace{\text{Reliability}}_{\text{calibration error}},$$

where: - *Uncertainty* represents the inherent variability in the data, independent of the model. - *Resolution* quantifies the model's ability to produce sharp, confident forecasts. - *Reliability* (or calibration error) measures the degree to which the forecast probabilities align with observed frequencies.

This decomposition highlights an important trade-off: improving sharpness without compromising calibration is the hallmark of a good probabilistic model. A model that produces overly sharp forecasts may suffer from poor calibration, while a model that is perfectly calibrated but overly diffuse lacks resolution.

As we have shown in this chapter, scoring rules are particularly appropriate in cross-validation frameworks, where the goal is to evaluate how well a model predicts future data realizations. When instead conducting simulation studies, we generate data from a known simulated true surface which becomes our benchmark. Hence, in simulation studies, we do not perform cross-validation but, for a given simulation, we compare the model predictions against the true surface. In such cases, scoring rules are not used because they are designed to assess predictive performance based on observed data, where both calibration and sharpness are critical considerations. Hence, in simulation studies, prediction performance metrics, such as bias, mean squared error (MSE), or other distance-based metrics that directly quantify the deviation from the known truth can be used to quantify the predictive performance of a model.

For a more detailed explanation of scoring rules, Gneiting, Balabdaoui, and Raftery (2007) is a recommended read.

## 4.8 Summary

In this chapter, we have demonstrated how geostatistical models can be used to draw predictive inference for targets that are spatially continuous (e.g., a prevalence surface within a study region) and aggregated at the areal level (e.g., the average disease prevalence in a district). The fundamental concept underpinning geostatistical prediction is that of the **predictive distribution**, which we define as the distribution of the target $T(x)$ conditioned on the data $y$. This distribution is the key element that enables the derivation and computation of point predictions, predictive intervals and the probability that the target falls within a pre-specified interval or above/below a given threshold.

To assess the predictive performance of a geostatistical model, we have considered two complementary approaches: **cross-validation** and **simulation studies**.

In the cross-validation framework, the concepts of **calibration** and **sharpness** are introduced to evaluate the quality of spatial predictions. Calibration refers to the agreement between the predicted probability of an event occurring and its actual observed frequency. Sharpness, on the other hand, quantifies the concentration of the predictive distribution around its point estimate. However, sharper predictions are not necessarily better; they must be well-calibrated to be meaningful. Thus, it is appropriate to discuss sharpness only in the context of well calibrated models.

A simulation study, unlike cross-validation, allows us to assess the ability of our model to predict an entire spatial surface within a study area or a predictive target defined over spatial units (e.g., districts) that make up the study area. This is possible because, in a simulation study, we can generate data from a **true model** and then evaluate the quality of inferences from **candidate models** on the simulated surface. The way we summarize predictive performance in a simulation study depends on the objective. If the goal is to assess how closely our point predictions align with the true surface, we can use metrics such as the mean squared error (Equation 4.10), averaged over the grid used to approximate the surface. Conversely, if the objective is to evaluate classification performance by categorizing spatial units into predefined health outcome categories, we can use sensitivity, specificity, negative predictive value, and positive predictive value. The choice of metric depends on whether the priority is minimizing errors in detecting true positives or true negatives (sensitivity and specificity), or ensuring that the spatial units classified as positive or negative are truly so (positive predictive value and negative predictive value). A summary of the metrics considered in this chapter is given in Table 4.3.

Table 4.3: Summary of the different diagnostic tools and metrics, that are used to assess the predictive performance of a geostatistical model using both cross-validation and simulatoin-based approaches.

| Diagnostic/Metric | Performance characteristic | Recommended use in |
|---|---|---|
| Bias | Accuracy | Simulation |
| Continuous ranked probability score (CRPS) | Calibration and sharpness | Cross-validation |
| Coverage probability | Calibration | Cross-validation and simulation |
| Mean square error | Accuracy | Simulation |
| Nonrandomized probability integral transform (PIT) | Calibration | Cross-validation |
| PIT | Calibration | Cross-validation |
| Scaled CRPS | Calibration and sharpness | Cross-validation |

## 4.9 FAQs

- *What should be the spatial resolution of the grid used for prediction?*

The choice of spatial resolution for the prediction grid involves a trade-off between computational efficiency and the ability to capture spatial correlation. Higher spatial resolution provides more detailed predictions but increases computational demands. A reasonable approach is to ensure that the distance between adjacent grid pixels is small enough to maintain a high spatial correlation. For instance, selecting a resolution where the correlation between neighboring pixels is at least 0.95 can be a useful guideline. However, this threshold is somewhat arbitrary, similar to the convention of using 0.95 in confidence intervals. Ultimately, the resolution should be chosen based on the spatial scale of variation in the data, while avoiding excessive computational burden.

- *I have covariates with different spatial resolutions. Which spatial resolution should I use?*

Two main approaches can be considered for handling covariates at different spatial resolutions in geostatistical analysis. The first involves aggregating all covariates to the coarsest resolution, ensuring consistency and avoiding artificial detail. This prevents high-resolution predictions based on low-resolution data but may sacrifice fine-scale spatial variation. The second approach disaggregates covariates to the finest resolution, enabling higher-detail predictions

but risking false precision and artifacts if the data do not support such resolution. To assess the impact on inferences, one might apply both approaches and evaluate how they influence the final results.

- *My prediction map looks like noise. What could be the error?*

The prediction map appearing noisy could be due to several factors. First, one should verify that the spatial grid was created correctly by plotting it alongside the original data to ensure alignment and consistency. Second, the spatial correlation in the data might be very small relative to the resolution of the grid, causing the model to struggle with capturing meaningful patterns. To address this, increasing the spatial resolution of the grid may help, as finer resolution can better capture the underlying spatial structure.

- *The main goal of my analysis is to carry out predictions of the main outcome. However, can I also use the model to draw inferences on the association between the outcome and the covariates that I have included in the model?*

The model developed for prediction, in theory, should not be primarily used to draw inferences on the regression relationships between the outcome and covariates. Understanding regression relationships and making predictions are distinct goals that often require different modeling approaches. While it is possible to interpret the estimated regression coefficients, caution is necessary, as these associations are not causative. Predictive models are optimized for accuracy in predicting th outcome rather than for identifying or explaining underlying relationships, so any inferences drawn should be treated as exploratory rather than definitive. An recommended reading on this topic is Shmueli (2010).

- *In my model I have some covariates that are attached to the individual (e.g. age, gender, occupation, etc.). How do I specify them when doing prediction?*

The answer to this question is given in Section 4.3.2.

## 4.10 Review questions

- What are the steps that needs to taken when performing spatial prediction?
- Define the predictive distribution of a predictive target and explain how this is used to answer a research question.
- Distinguish between spatially continuous and areal-level targets, giving at least two examples for each.
- Explain the difference between marginal and joint predictions, and for which targets each can be used.
- Define predictive performance in terms of calibration and sharpness.

- Explain how calibration can be assessed using graphical tools, by distinguishing between continuous and count outcomes.
- Give examples of metrics that can be used to evaluate calibration and sharpness.
- Explain how simulation studies can be carried to assess the predictive performance of geostatistical models and how this differ from cross-validation.

## 4.11 Exercises

1. Using the code from Section 4.5, redraw the plot of Figure 4.10 by choosing different values for the dispersion parameter of the Negative Binomial distribution. What happens when $\alpha$ is close to 0 and why?

2. Consider the example in Section 4.5.2.1. Fit a third model, that extends $M1$ by adding a nugget effect, hence

$$M_2 : \log\left\{\frac{p(x_i)}{1-p(x_i)}\right\} = \beta_0+\beta_1 e(x_i)+\beta_2 \max\{e(x_i)-150, 0\}+S(x_i)+Z_i.$$

   Generate the AnPIT plot for $M_2$ and compare it to those of $M_0$ and $M_1$. Make sure to apply the function `assess_pp` to all three models so that the same test sets are used. Is $M_2$ showing a better calibration? Why?

3. In the analysis of the *Anopheles gambiae* moqsuitoes data, consider two models. One that uses elavation, with linear predictor

$$M_1 \ : \ \log\{\lambda(x_i)\} = \beta_0 + \beta_1 e(x_i) + S(x_i)$$

   where $e(x)$ is the elevation in meters at location $x$ and $S(x_i)$ is a stationary and isotropic Gaussian process with variance $\sigma^2$ and exponential correlation function $\rho(u) = \exp\{-u/\phi\}, \phi > 0$. Consider also an intercept only model

$$M_0 \ : \ \log\{\lambda(x_i)\} = \beta_0 + S(x_i).$$

   After fitting both models to the data, assess if they are well calibrated models using the average nonrandomized probability integral transform (AnPIT). Assuming both models are well-calibrated, compare the predictive performance using continuous ranked probability score (CRPS) and the scaled CRPS. In doing this, use different sampling schemes to split the data, as illustrated in Section 4.5.1, and assess if and how the adopted approach to split the data affects the results.

4. Consider the Liberia data-set on riverblindess. Carry out a simulation study where the objective is to assess the impact of mispecifying non-linear relationships between elevation and prevalence. Consider Equation 4.9 as the true model and a model that specifies the relationship as linear on the logit scale as the candidate model, hence

$$M_C : \log\left\{\frac{p(x_i)}{1-p(x_i)}\right\} = \beta_0 + \beta_1 e(x_i) + S(x_i).$$

Carry out 200 iterations in the simulation and define a suitable metric of predictive performance to compare the two models. Provide a clear answer on the impact of assuming a linear relationship with elevation when this is not valid.

# 5

# *Case studies*

## 5.1 Mapping stunting and underwieght risk in Ghana

Malnutrition remains a critical public health issue in many low- and middle-income countries, particularly affecting children under five years of age. It can have long-term consequences on physical growth, cognitive development, and susceptibility to disease. Monitoring childhood nutritional status is essential for evaluating the effectiveness of health interventions and informing policy decisions.

Two widely used anthropometric indicators derived from child growth measurements are:

- Height-for-Age Z-score (HAZ): A measure of stunting, which reflects chronic malnutrition. Children with a HAZ below -2 standard deviations from the WHO growth reference median are considered stunted, indicating long-term nutritional deprivation or repeated infections.
- Weight-for-Age Z-score (WAZ): A composite indicator of underweight, capturing both acute and chronic malnutrition. Children with WAZ below -2 are considered underweight. This measure is commonly used in population-based surveys because it is relatively easy to collect and interpret.

These Z-scores are calculated by comparing individual anthropometric measurements to the WHO Child Growth Standards, which were developed based on data from healthy children under optimal environmental and nutritional conditions (WHO 2006).

Geostatistical models applied to HAZ and WAZ, or to the prevalence of stunting and underweight derived from these scores, can help answer critical questions such as: *Where is the burden of malnutrition highest? How does it relate to socioeconomic or environmental factors? And which areas should be prioritized for targeted nutritional interventions?*

In this case study, we analyse HAZ and WAZ as continuous outcomes to assess the spatial variation in child malnutrition across Ghana and its association with socioeconomic and demographic covariates.

### 5.1.1 Exploratory analysis

We begin by loading the necessary R packages and the `malnutrition` dataset. This geostatistical dataset is derived from the 2014 Ghana Demographic and Health Survey (DHS) (Ghana Statistical Service, Ghana Health Service, and ICF International 2015), which provides nationally representative data on child health and nutrition. The dataset includes anthropometric measurements, household characteristics, and georeferenced sampling cluster coordinates, making it suitable for spatial analysis of child malnutrition.

```
# Load packages
library(RiskMap)
library(ggplot2)
library(dplyr)
library(patchwork)

# Load the data
data(malnutrition)
```

The explanatory variables considered in this analysis are `age` and `wealth`. Figure Figure 5.1 displays scatterplots of the two nutritional outcomes (`HAZ` and `WAZ`) against age, alongside boxplots stratified by the three levels of the `wealth` variable.

To explore the relationship between age and nutritional status, the code below applies locally estimated scatterplot smoothing (LOESS) to highlight potential non-linear trends. In addition, we fit linear splines using the function `pmax()` to model a change in slope at specified knots—2 months for HAZ and 1 month for WAZ. These change points were selected heuristically based on visual inspection of the LOESS curves.

As shown in Figure Figure 5.1, the linear splines (green dashed lines) closely follow the LOESS fits (solid red lines), indicating that the piecewise linear models provide an adequate approximation of the age-nutrition relationship.

```
# Filter complete cases for relevant variables
mal <- malnutrition %>%
  filter(!is.na(HAZ), !is.na(WAZ), !is.na(age), !is.na(wealth))

# Fit spline model for HAZ with knot at 2 months
mod_haz <- lm(HAZ ~ age + pmax(age - 2, 0), data = mal)

# Fit spline model for WAZ with knot at 1 month
mod_waz <- lm(WAZ ~ age + pmax(age - 1, 0), data = mal)

# Prediction grid for HAZ
newdat_haz <- data.frame(age = seq(min(mal$age), max(mal$age),
 ↪  length.out = 200))
```

```
newdat_haz$fit <- predict(mod_haz, newdata = newdat_haz)

# Prediction grid for WAZ
newdat_waz <- data.frame(age = seq(min(mal$age), max(mal$age),
 ↪  length.out = 200))
newdat_waz$fit <- predict(mod_waz, newdata = newdat_waz)

# Plot HAZ vs Age
p1 <- ggplot(mal, aes(x = age, y = HAZ)) +
  geom_point(alpha = 0.3) +
  geom_smooth(method = "loess", colour = "red", se = FALSE) +
  geom_line(data = newdat_haz, aes(x = age, y = fit),
            colour = "green", linetype = "dashed", linewidth =
             ↪  1.2) +
  labs(title = "HAZ vs Age", x = "Age (months)", y = "HAZ") +
  theme_minimal()

# Plot WAZ vs Age
p2 <- ggplot(mal, aes(x = age, y = WAZ)) +
  geom_point(alpha = 0.3) +
  geom_smooth(method = "loess", colour = "red", se = FALSE) +
  geom_line(data = newdat_waz, aes(x = age, y = fit),
            colour = "green", linetype = "dashed", linewidth =
             ↪  1.2) +
  labs(title = "WAZ vs Age", x = "Age (months)", y = "WAZ") +
  theme_minimal()

# Boxplots for HAZ and WAZ by wealth
p3 <- ggplot(mal, aes(x = factor(wealth), y = HAZ)) +
  geom_boxplot(fill = "lightblue") +
  labs(title = "HAZ by Wealth Group", x = "Wealth (1 = Poor, 3 =
   ↪  Rich)", y = "HAZ") +
  theme_minimal()

p4 <- ggplot(mal, aes(x = factor(wealth), y = WAZ)) +
  geom_boxplot(fill = "lightgreen") +
  labs(title = "WAZ by Wealth Group", x = "Wealth (1 = Poor, 3 =
   ↪  Rich)", y = "WAZ") +
  theme_minimal()

# Combine plots
(p1 | p2) / (p3 | p4)
```

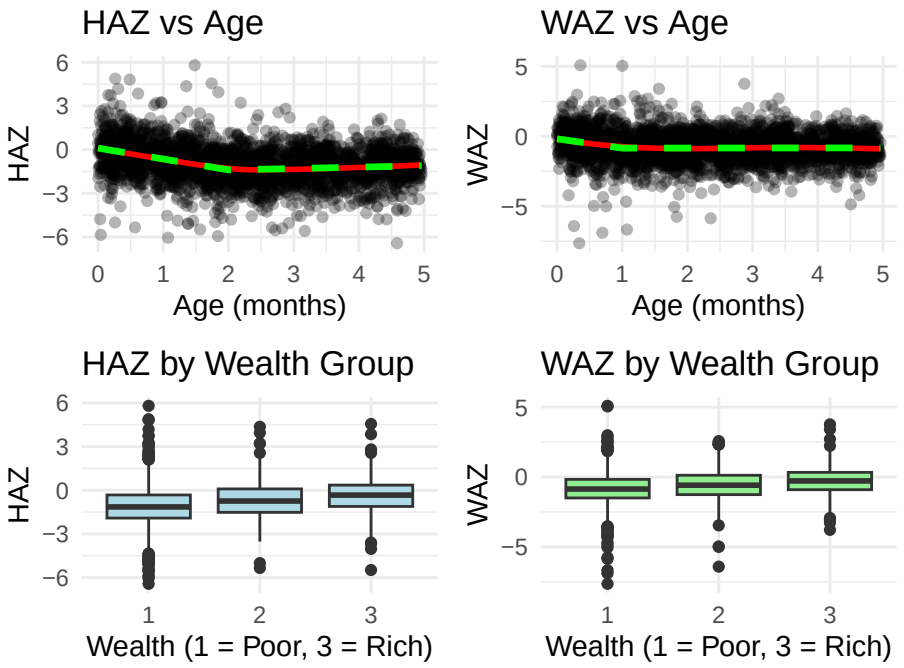



Figure 5.1: Exploratory analysis of HAZ and WAZ by age and wealth. The red lines correspond to LOESS smoothed curves and the green lines to linear splines with change points at 2 months (HAZ) and 1 month (WAZ) of age. Boxplots show variation by household wealth group.

We begin by fitting a non-spatial model to assess whether there is residual spatial correlation that remains after accounting for the effects of age and wealth. Let $Y_{ij}$ represent either the HAZ or WAZ score for the $j$-th child in the $i$-th cluster. The model takes the following form:

$$Y_{ij} = \beta_0 + \beta_1 a_{ij} + \beta_2 \max\{a_{ij} - c, 0\} + \beta_3 w_i + Z_i + U_{ij}, \tag{5.1}$$

where $a_{ij}$ denotes the age in months of child $j$ in cluster $i$, and $w_i$ is the wealth score associated with cluster $i$; as previously stated, the change point $c$ of the linear spline is set to 2 months for HAZ and 1 month for WAZ. The term $Z_i$ is a Gaussian random effect with mean zero and variance $\tau^2$, capturing between-cluster variation not explained by the covariates. The error term $U_{ij}$ represents child-specific residual variation and is assumed to be Gaussian with mean zero and variance $\omega^2$.

To fit this model, we create a cluster ID variable based on the sampling coordinates and include it as a random intercept in the `lmer` function to incorporate the term $Z_i$.

```
# Create ID for the location
malnutrition <- malnutrition %>%
  group_by(lng, lat) %>%
  mutate(loc_id = cur_group_id()) %>%
  ungroup()
```

We then fit the models using the `lmer` function.

```
library(lme4)
haz_lmer <-
lmer(HAZ ~ age + pmax(age - 2, 0) + wealth + (1 | loc_id), data
 ↪  = malnutrition)

summary(haz_lmer)
```

```
Linear mixed model fit by REML ['lmerMod']
Formula: HAZ ~ age + pmax(age - 2, 0) + wealth + (1 | loc_id)
   Data: malnutrition

REML criterion at convergence: 8593.6

Scaled residuals:
    Min      1Q  Median      3Q     Max
-4.6407 -0.5992  0.0154  0.6098  5.6443

Random effects:
 Groups   Name        Variance Std.Dev.
 loc_id   (Intercept) 0.06939  0.2634
 Residual             1.39362  1.1805
Number of obs: 2671, groups:  loc_id, 410

Fixed effects:
                 Estimate Std. Error t value
(Intercept)      -0.39571    0.08170  -4.844
age              -0.77185    0.04471 -17.265
pmax(age - 2, 0)  0.88486    0.06698  13.211
wealth            0.38630    0.03611  10.699

Correlation of Fixed Effects:
            (Intr) age    p(-2,0
age         -0.654
pmax(g-2,0)  0.521 -0.932
wealth      -0.616 -0.034  0.038
```

```
waz_lmer <-
  lmer(WAZ ~ age + pmax(age - 1, 0) + wealth + (1 | loc_id),
   ↪  data = malnutrition)
summary(waz_lmer)
```

```
Linear mixed model fit by REML ['lmerMod']
Formula: WAZ ~ age + pmax(age - 1, 0) + wealth + (1 | loc_id)
   Data: malnutrition

REML criterion at convergence: 7997.3

Scaled residuals:
    Min      1Q  Median      3Q     Max
-6.5967 -0.5816  0.0098  0.6312  5.7860

Random effects:
 Groups   Name        Variance Std.Dev.
 loc_id   (Intercept) 0.05625  0.2372
 Residual             1.11823  1.0575
Number of obs: 2668, groups:  loc_id, 410

Fixed effects:
                 Estimate Std. Error t value
(Intercept)      -0.50444    0.08951  -5.636
age              -0.72230    0.09615  -7.512
pmax(age - 1, 0)  0.73510    0.10754   6.835
wealth            0.27359    0.03238   8.450

Correlation of Fixed Effects:
            (Intr) age    p(-1,0
age         -0.765
pmax(g-1,0)  0.716 -0.989
wealth      -0.494 -0.036  0.037
```

Of particular interest in the summary of the fitted models above are the estimates of the variances associated with the random effects: $\tau^2$ for the cluster-level terms $Z_i$, and $\omega^2$ for the child-specific residuals $U_{ij}$. The estimates suggest that $\omega^2$ is more than twenty times larger than $\tau^2$, indicating that the majority of the unexplained variation occurs at the individual child level rather than between clusters. This has important implications for interpretation: any subsequent mapping of the prevalence of stunting or underweight should acknowledge the high degree of within-cluster variability as a limitation. In particular, predictions at unsampled locations may be subject to substantial uncertainty driven by this fine-scale heterogeneity.

### 5.1.2 Assessing residual spatial correlation and model fitting

We next explore the residual spatial correlation through the empirical variogram. Hence, we extract the estimate of the random effects from Equation 5.1 and input these into the `s_variogram` function that generates the empirical variogram.

```
# Extract random effects for HAZ and WAZ and combine them
re <- data.frame(
  loc_id = as.numeric(rownames(ranef(haz_lmer)$loc_id)),
  Z_hat_haz = ranef(haz_lmer)$loc_id[,1],
  Z_hat_waz = ranef(waz_lmer)$loc_id[,1]
)

# Extract one row per location with coordinates
loc_coords <- malnutrition %>%
  select(loc_id, lng, lat) %>%
  distinct()

# Merge coordinates into random effects
re <- left_join(re, loc_coords, by = "loc_id")

# Convert to sf and transform into UTM
library(sf)
re <- st_as_sf(re, coords = c("lng","lat"), crs = 4326)
re <- st_transform(re, crs = propose_utm(re))

# Variogram for HAZ
var_haz <- s_variogram(re, "Z_hat_haz", scale_to_km = TRUE,
                       n_permutation = 1000)
var_waz <- s_variogram(re, "Z_hat_waz", scale_to_km = TRUE,
                       n_permutation = 1000)
library(patchwork)

# Create ggplot objects
p_haz <- plot_s_variogram(var_haz, plot_envelope = TRUE) +
  ggtitle("HAZ")

p_waz <- plot_s_variogram(var_waz, plot_envelope = TRUE) +
  ggtitle("WAZ")

p_haz + p_waz
```

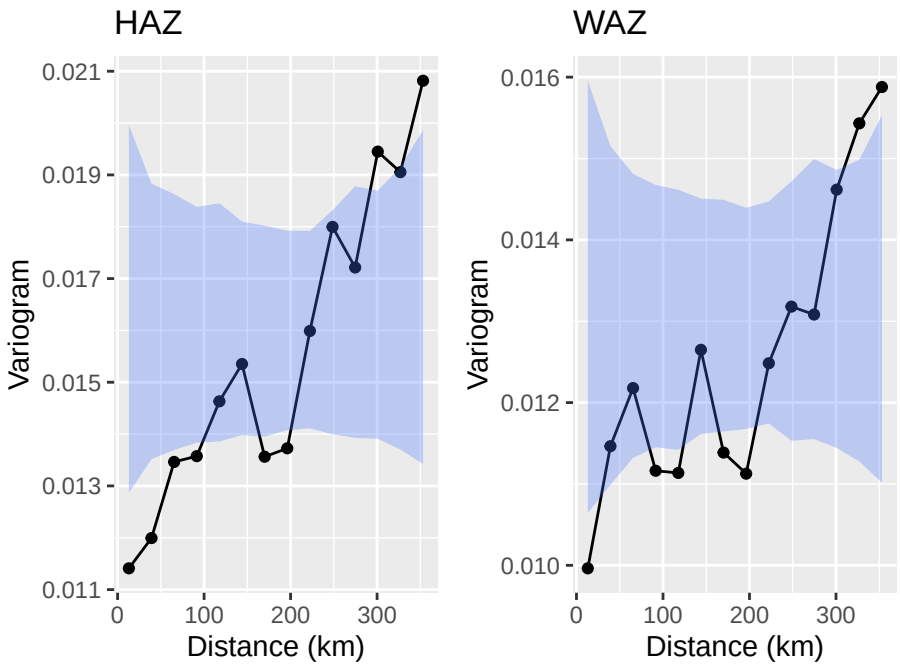



Figure 5.2: Empirical variograms for HAZ and WAZ based on the estimated random effects from Equation 5.1.

Figure 5.2 shows the variogram for both HAZ and WAZ. In both cases, we observe that is residual spatial correlation in the data that is not explained by either age or the wealth index. We next extend the model in Equation 5.1 to address this issue.

For our geostatistical analysis of HAZ and WAZ, we consider the following model:

$$Y_{ij} = \beta_0 + \beta_1 a_{ij} + \beta_2 \max\{a_{ij} - c, 0\} + \beta_3 w_i + S(x_i) + U_{ij},$$

where $S(x)$ is a zero-mean, stationary, and isotropic Gaussian process with variance $\sigma^2$ and an exponential correlation function with scale parameter $\phi$. In this geostatistical extension, we replace the cluster-level random effect $Z_i$ from Equation 5.1 with the spatial process $S(x_i)$. The random effect $Z_i$ was previously used in the Binomial mixed model to account for unexplained variation between clusters. It is also possible to include both $S(x_i)$ and $Z_i$ in the model, so that the total unexplained variation between clusters is decomposed into a spatially structured component, $S(x_i)$, and an unstructured component, $Z_i$. However, this approach is not recommended in the initial model specification, as it may lead to identifiability issues. It can be considered later in the

analysis if justified by model diagnostics.

```
# Remove missing data from the WAZ outcome
malnutrition <-
 ↪  malnutrition[complete.cases(malnutrition[,"WAZ"]),]

# Converting the data-frame into an sf object
malnutrition_sf <- st_as_sf(malnutrition, coords = c("lng",
 ↪  "lat"), crs = 4326)
malnutrition_sf <- st_transform(malnutrition_sf, crs =
 ↪  propose_utm(malnutrition_sf))

# Maximum likelihood estimation for the HAZ and WAZ outcomes
haz_fit <-
glgpm(HAZ ~ age + pmax(age - 1, 0) + wealth + gp(),
      data=malnutrition_sf, family = "gaussian")

waz_fit <-
  glgpm(HAZ ~ age + pmax(age - 1, 0) + wealth + gp(),
        data=malnutrition_sf, family = "gaussian")
```

We then summarize the fit of models.

```
summary(haz_fit)
```

```
Call:
glgpm(formula = HAZ ~ age + pmax(age - 1, 0) + wealth + gp(),
    data = malnutrition_sf, family = "gaussian")

Linear geostatistical model
Link: identity
Inverse link function = x

'Lower limit' and 'Upper limit' are the limits of the 95%
confidence level intervals

Regression coefficients
                  Estimate Lower limit Upper limit     StdErr
                  z.value p.value
(Intercept)     -0.2158950  -0.4338747   0.0020848  0.1112162
-1.9412 0.05223
age             -1.3073560  -1.4135715  -1.2011404  0.0541926
-24.1243 < 2e-16
pmax(age - 1, 0)  1.2288215   1.1496224   1.3080207  0.0404085
30.4100 < 2e-16
```

```
wealth            0.3520544   0.3319696   0.3721393  0.0102476
34.3550 < 2e-16

(Intercept)      .
age              ***
pmax(age - 1, 0) ***
wealth           ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

                       Estimate Lower limit Upper limit
Measuremment error var.   1.4371      1.4254      1.4489

Spatial Gaussian process
Matern covariance parameters (kappa=0.5)
                     Estimate Lower limit Upper limit
Spatial process var.  0.069453    0.061423      0.0785
Spatial corr. scale  36.529380   30.187727     44.2032
Variance of the nugget effect fixed at 0

Log-likelihood: -1859.246
AIC: 3732.491
```

```
summary(waz_fit)
```

```
Call:
glgpm(formula = WAZ ~ age + pmax(age - 1, 0) + wealth + gp(),
    data = malnutrition_sf, family = "gaussian")

Linear geostatistical model
Link: identity
Inverse link function = x

'Lower limit' and 'Upper limit' are the limits of the 95%
confidence level intervals

Regression coefficients
                  Estimate Lower limit Upper limit     StdErr
                  z.value
(Intercept)      -0.4580088  -0.6499668  -0.2660509  0.0979395
-4.6764
age              -0.7295504  -0.8235412  -0.6355596  0.0479554
-15.2131
pmax(age - 1, 0)  0.7421647   0.6720850   0.8122443  0.0357556
20.7566
```

```
wealth            0.2254477   0.2076797   0.2432157  0.0090655
24.8689
                   p.value
(Intercept)      2.919e-06 ***
age              < 2.2e-16 ***
pmax(age - 1, 0) < 2.2e-16 ***
wealth           < 2.2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

                       Estimate Lower limit Upper limit
Measuremment error var.   1.1253      1.1161      1.1346

Spatial Gaussian process
Matern covariance parameters (kappa=0.5)
                     Estimate Lower limit Upper limit
Spatial process var.  0.049912    0.043750      0.0569
Spatial corr. scale  35.836303   28.815346     44.5679
Variance of the nugget effect fixed at 0

Log-likelihood: -1530.593
AIC: 3075.187
```

The spatial component of the models reveals a moderate degree of spatial structure in the HAZ and WAZ outcomes. For HAZ, the estimated spatial variance is approximately 0.069, while the individual-level radom effect's variance is substantially larger at around 1.437. For WAZ, the spatial variance is estimated at 0.050, compared to the variance of the individual-level radom effect's variance of 1.125. In both models, as already observed when fitting Binomial mixed models, this indicates that the majority of the residual variation is due to individual-level noise or unstructured heterogeneity rather than spatially structured effects. The relative contribution of the spatial process to the total unexplained variation is therefore relatively small. Nevertheless, geostatistical analysis remains valuable also in this context. Even when spatial effects explain only a modest proportion of the variation, the model still enables spatial prediction of the average spatial pattern for the outcome of interest across the study area. This is useful for identifying areas where child malnutrition is systematically higher or lower, guiding targeted interventions and informing resource allocation. We explore these implications further in the next section on geostatistical prediction for HAZ and WAZ.

### 5.1.3 Prediction and assessment of model calibration

Stunting and underweight are standard indicators used to assess child malnutrition. A child is defined as *stunted* if their height-for-age $z$-score (HAZ) is

below −2 standard deviations from the median of the WHO Child Growth Standards, i.e., HAZ < -2. Similarly, a child is considered *underweight* if their weight-for-age $z$-score (WAZ) falls below −2, i.e., WAZ < -2 (United Nations Statistics Division, World Health Organization, and UNICEF 2025). These thresholds are based on the World Health Organization (WHO) growth reference standards, which provide a normative basis for assessing nutritional status in children under five years of age (Organization 2024).

We now define the predictive targets for stunting and underweight prevalence in terms of the adopted geostatistical model. Both indicators correspond to the probability that a child's z-score falls below −2, that is, $Y(x) < -2$, where $Y(x)$ denotes either the HAZ or WAZ score at location $x$. Under the model specification, the prevalence is therefore defined as

$$T(x) = P\left\{Y(x) < -2 \,|\, S(x)\right\} = \Phi\left(-\frac{\mu + S(x) + 2}{\omega}\right), \tag{5.2}$$

where $\Phi(\cdot)$ is the cumulative distribution function of the standard normal distribution, $\mu$ is the linear predictor including the effects of covariates at individual- and cluster-level (as defined by the coefficients $\beta_1$ to $\beta_3$ of Equation 5.1) as well as the intercept, and $\omega^2$ is the variance of the individual-level residual term. In Equation 5.2, we condition on $S(x)$ because this allows us to define both stunting an underweight prevalence as a spatially varying predictive target.

To implement this in `RiskMap`, we begin by generating a predictive grid over the study region. In this case, we use a regular grid with 10 km by 10 km resolution. We then draw predictive samples of the spatial effect $S(x)$ at each grid location, conditional on the observed data. For the `age` variable, we fix it at 2 months for HAZ and 1 month for WAZ, respectively. The `wealth` variable is set to its lowest value of 1. These choices are made to visualize the predictive target for the subgroup of children most at risk of stunting and underweight, as identified through exploratory analysis and the estimated regression coefficients. The code below illustrates this procedure.

```
# Boundaries of Ghana
library(rgeoboundaries)
ghana <- geoboundaries(country = "Ghana", adm_lvl = "adm0")
ghana <- st_transform(ghana, st_crs(malnutrition_sf))

ghana_grid <- create_grid(ghana, spat_res = 10)

n_pred <- nrow(st_coordinates(ghana_grid))

# Prediction of S(x) and covariates effects over the grid for
```

```
# HAZ
haz_pred_grid <- pred_over_grid(haz_fit,
                                grid_pred = ghana_grid,
                                predictors = data.frame(
                                  # Setting age to 2 months
                                  age = rep(2, n_pred),
                                  # Setting wealth to 1 (least
                                   ↪  wealthy)
                                  wealth = rep(1, n_pred)
                                ))
# WAZ
waz_pred_grid <- pred_over_grid(waz_fit,
                                grid_pred = ghana_grid,
                                predictors = data.frame(
                                  # Setting age to 1 month
                                  age = rep(1, n_pred),
                                  # Setting wealth to 1 (least
                                   ↪  wealthy)
                                  wealth = rep(1, n_pred)
                                ))
```

We then use the `pred_target_grid` function to generate predictive summaries of stunting and underweight prevalence, as defined in Equation 5.2. To quantify uncertainty in the predictions, we report the coefficient of variation, which provides a standardized measure of relative uncertainty across the prediction grid.

```
# Prediction of stunting prevalence
sd_ind_haz <- sqrt(coef(haz_fit)$sigma2_me)
stunting_prev <-
pred_target_grid(haz_pred_grid,
                 f_target = list(prev = function(x)
                  ↪  pnorm((-2-x)/sd_ind_haz)),
                 pd_summary = list(mean = mean,
                                   cv = function(x)
                                    ↪  sd(x)/mean(x)))

# Prediction of underweight prevalence
sd_ind_waz <- sqrt(coef(waz_fit)$sigma2_me)
underw_prev <-
  pred_target_grid(waz_pred_grid,
                   f_target = list(prev = function(x)
                    ↪  pnorm((-2-x)/sd_ind_waz)),
                   pd_summary = list(mean = mean,
                                     cv = function(x)
                                      ↪  sd(x)/mean(x)))
```

Finally, we visualize the results in the map shown in Figure 5.3. The maps reveal a clear hotspot for both stunting and underweight, with predicted prevalence values reaching approximately 45% and 24%, respectively. Notably, this same region also exhibits lower predictive uncertaint, measured by the coefficient of variation, compared to other areas in Ghana.

```
# Set up a 2x2 layout
layout(matrix(1:4, nrow = 2, byrow = TRUE))
par(mar = c(4, 4, 4, 5))  # Adjust margins

# Top row: stunting
plot(stunting_prev, which_target = "prev", which_summary =
 ↪  "mean",
     main = "Stunting: Predictive Mean")
plot(stunting_prev, which_target = "prev", which_summary = "cv",
     main = "Stunting: Coefficient of Variation")

# Bottom row: underweight
plot(underw_prev, which_target = "prev", which_summary = "mean",
     main = "Underweight: Predictive Mean")
plot(underw_prev, which_target = "prev", which_summary = "cv",
     main = "Underweight: Coefficient of Variation")
```

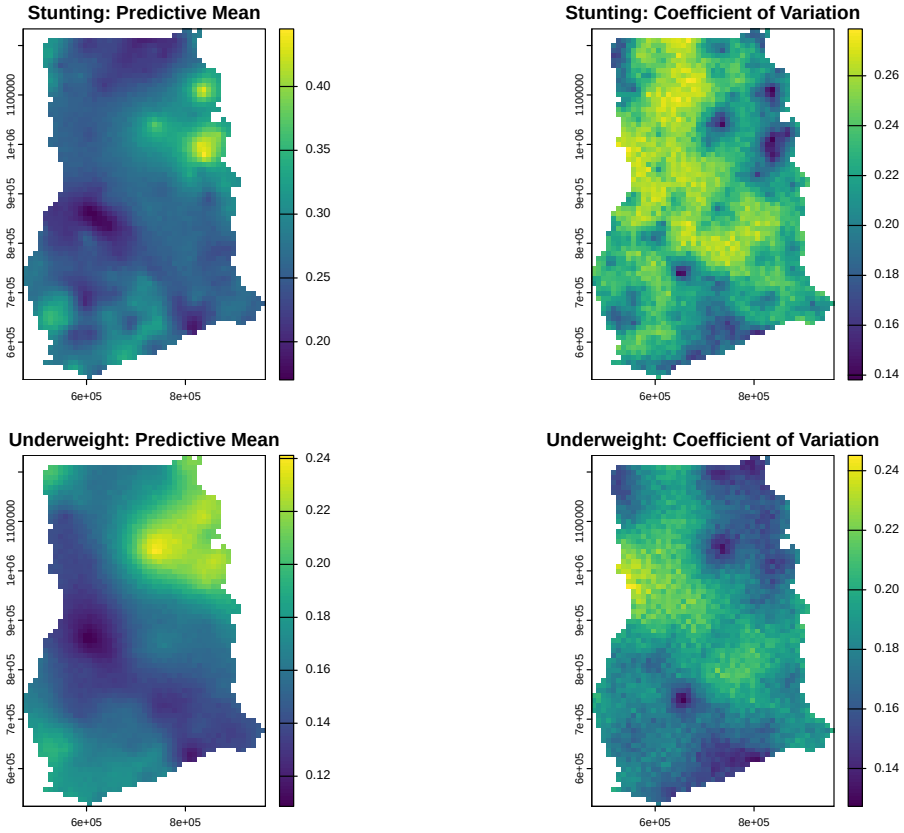



Figure 5.3: Predictive mean and coefficient of variation for stunting and underweight prevalence estimates. The top row shows predictions for stunting, while the bottom row corresponds to underweight. For each indicator, the left panel shows the predictive mean and the right panel the coefficient of variation.

To assess the predictive performance and calibration of our geostatistical models for HAZ and WAZ, we perform a cross-validation exercise using the `assess_pp` function. Cross-validation is conducted by repeatedly splitting the data into training and testing sets, ensuring that prediction locations are at least 5 km away from the nearest training point. Specifically, we set `n_size = 100` prediction locations for the test set, impose a minimum distance constraint of 5 km (`min_dist = 5`), and repeat the process over 10 test sets randomly drawn under those conditions (`iter = 10`) using `method="regularized"` as the sampling method.

```
# HAZ
assess_haz <-
```

```
assess_pp(list(HAZ = haz_fit),
          n_size = 100,
          min_dist = 5,
          iter = 10,
          method = "regularized")

# WAZ
assess_waz <-
  assess_pp(list(WAZ = waz_fit),
            n_size = 100,
            min_dist = 5,
            iter = 10,
            method = "regularized")
```

We note that for a Gaussian linear geostatistical model, the `assess_pp` function does not compute the average non-randomized probability integral transform (AnPIT, introduced in Section 4.5.2), as this measure is specifically designed for count outcomes. Instead, it employs the standard probability integral transform (PIT), defined as $\text{PIT}(y) = F_M(y)$, where $F_M(y)$ is the cumulative distribution function of the predictive distribution evaluated at the observed value $y$. The PIT transforms the observed outcomes in the test set such that, if the model is well-calibrated, these transformed values should follow a uniform distribution. Deviations from uniformity indicate potential issues with calibration and predictive accuracy.

Figure 5.4 shows the results of the cross-validation exercise visualized through PIT curves for HAZ and WAZ. Similarly to the way we have interpreted the AnPIT curves, the PIT curves indicate good calibration when they lie close to the identity line, meaning the predictive distributions accurately reflect the observed variability in the test sets. The results shown in Figure 5.4 demonstrate that, for all ten test sets and for both HAZ and WAZ, the PIT curves closely align with the identity line, confirming that the models are well-calibrated and provide reliable uncertainty quantification.

```
# Generate PIT plots
p_haz <- plot_AnPIT(assess_haz, mode = "all")
p_waz <- plot_AnPIT(assess_waz, mode = "all")

# Arrange side-by-side
library(gridExtra)
grid.arrange(p_haz, p_waz, ncol = 2)
```

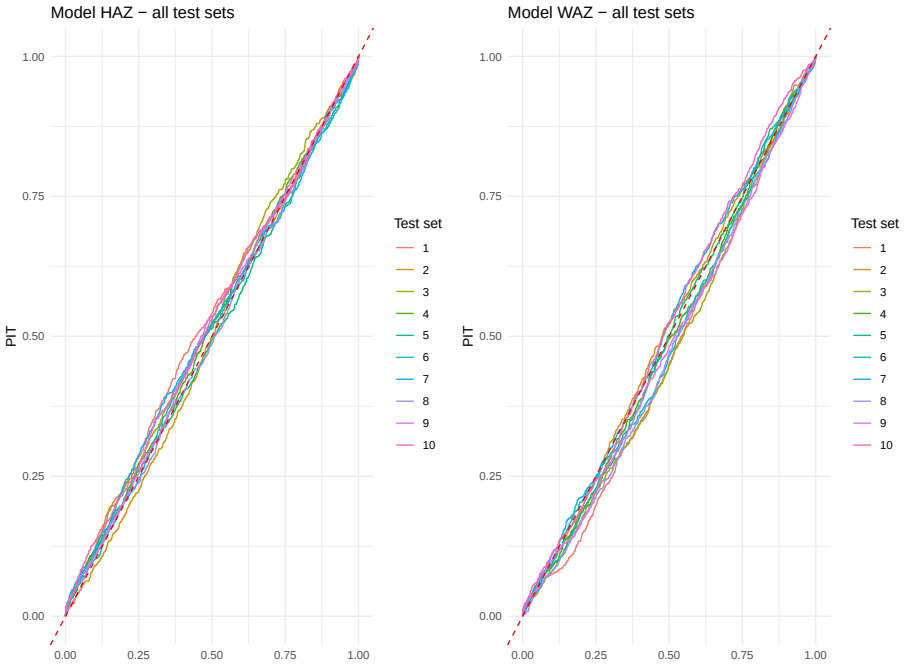



Figure 5.4: Assessment of the models calibration. The left panel shows the probability integral transform (PIT) curve for HAZ, and the right panel for WAZ. In both panels, the red dahsed line corresponds to the identity line.

### 5.1.4 Summary and conclusions

This case study demonstrated the utility of geostatistical modeling for mapping child malnutrition in Ghana using continuous anthropometric measurements, namely HAZ and WAZ Z-scores. By modeling these outcomes directly, we were able to define stunting and underweight prevalence as predictive probabilities based on threshold exceedance (e.g., HAZ $<$ -2), rather than by dichotomizing the data prior to analysis. This approach allowed us to retain the full information content of the measurements and to produce spatially resolved predictions of prevalence along with associated uncertainty estimates. Readers interested in a more in-depth treatment of this modeling strategy are referred to Kyomuhangi et al. (2021), which discusses the loss of information and degradation in predictive performance that can result from dichotomizing continuous outcomes in geostatistical settings.

More broadly, this case study illustrated how geostatistical models can be used to analyze individual-level outcomes while accounting for both child-level and household-level covariates. Aggregation of data across individuals, such as using average HAZ or WAZ scores per location, is not required and can in fact introduce several complications. In particular, the number of children per clus-

ter varies, which implies that an analysis based on cluster-level averages would require a heteroscedastic specification for the residual term to appropriately reflect varying precision across locations. Ignoring this could lead to unreliable predictive inferences.

In conclusion, this analysis showed that neither dichotomization nor aggregation is necessary in geostaistical analysis. Instead, geostatistical models should be fitted to individual-level continuous outcomes, and suitable predictive targets and summaries of uncertainty can then be derived from the fitted model.

## 5.2 Mapping malaria in Malawi

In this section, we demonstrate how to conduct a geostatistical analysis using Malaria Indicator Survey (MIS) data accessed through the `malariaAtlas` R package (Lucas et al. 2020). This package provides streamlined access to a curated repository of georeferenced malaria data, including prevalence surveys and associated metadata, making it a valuable resource for spatial epidemiological studies. The primary aim here is to illustrate how relevant environmental covariates can be obtained via Google Earth Engine and incorporated into a geostatistical model for prevalence mapping. Although the analysis is not guided by a specific research question, its value lies in showcasing the full workflow and technical aspects introduced in previous chapters, including data preparation, covariate extraction, exploratory analysis, model fitting, and spatial prediction of disease prevalence.

### 5.2.1 Downloading prevalence and raster data

This section illustrates how to access and visualize Malaria Indicator Survey data using the `malariaAtlas` R package, focusing on *Plasmodium falciparum* prevalence in Malawi for the year 2006. We walk through the steps to extract survey data, obtain country boundaries, and produce a basic map of raw prevalence data.

We begin by loading the necessary packages and initializing the Earth Engine API via the `rgee` package.

```
# Load libraries
library(malariaAtlas)
library(dplyr)
library(tidyr)
library(sf)
library(rgee)
library(terra)
```

```
library(rgeoboundaries)
library(grid)

# Initialize Earth Engine
ee_Initialize()
```

We then use `rgeoboundaries` to obtain Malawi's administrative boundary and extract P. falciparum prevalence survey data from the `malariaAtlas` package for surveys conducted in or overlapping the year 2006.

```
# Get Malawi boundary
mlw_admin0_sf <- geoboundaries(country = "Malawi", adm_lvl =
 ↪  "adm0")
mlw_admin0_ee <- sf_as_ee(mlw_admin0_sf)

# Download PfPR survey data
mlw_pfpr <- getPR(country = "Malawi", species = "Pf") %>%
  filter(year_start <= 2006 & year_end >= 2006) %>%
  filter(!is.na(longitude) & !is.na(latitude))

# Convert to sf object
mlw_sf <- st_as_sf(mlw_pfpr, coords = c("longitude",
 ↪  "latitude"), crs = 4326)

# Convert to UTM
mlw_crs <- propose_utm(mlw_sf)
mlw_sf <- st_transform(mlw_sf, crs = mlw_crs)

# Remove columns with all NAs
mlw_sf <- mlw_sf[, colSums(!is.na(mlw_sf)) > 0]
```

The following map displays the raw prevalence data at each survey location. The size and color of each point reflect the estimated malaria prevalence (pr) at that location.

```
# Compute prevalence
# Compute observed prevalence
mlw_sf$pr <- mlw_sf$positive / mlw_sf$examined

# Plot raw PfPR data with uniform point size and no axis labels
ggplot() +
  geom_sf(data = mlw_admin0_sf, fill = "white", color = "black")
   ↪  +
  geom_sf(data = mlw_sf, aes(color = pr), size = 1, alpha = 0.8)
   ↪  +
```

```
  scale_color_viridis_c(name = "PfPR") +
  theme_minimal() +
  ggtitle("Malaria Prevalence in Malawi (2006)") +
  theme(
    legend.position = "right",
    axis.title = element_blank(),
    axis.text = element_blank(),
    axis.ticks = element_blank()
  )
```

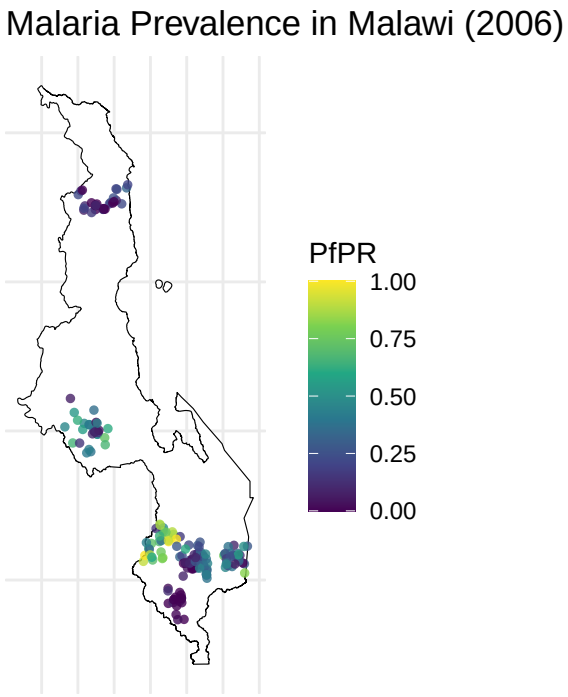



Figure 5.5: Raw PfPR survey data for Malawi in 2006. Each point represents a georeferenced survey location, with color and size indicating estimated prevalence.

In the next step we retrieve a suite of environmental covariates that are mechanistically linked to malaria transmission and therefore commonly included in spatial risk-mapping studies. All layers are clipped to the Malawi national boundary and combined into a single multi-band image ready for extraction at the survey points.

These covariates are summarised in Table Table 5.1, which also explains their epidemiological relevance in the context of malaria transmission.

Table 5.1: Environmental covariates extracted from Google Earth Engine and their relevance for malaria risk modelling.

| Covariate | Source | Processing | Relevance |
|---|---|---|---|
| Precipitation | CHIRPS Daily | Annual mean (mm · day $^{1}$) | Determines suitability of breeding sites for *Anopheles* mosquitoes. |
| Land-surface temperature | MODIS MOD11A2 | Kelvin converted to °C; annual mean | Affects mosquito survival and parasite development rates. |
| NDVI | MODIS MOD13A2 | Scaled to 0–1; annual mean | Proxy for habitat moisture and vegetation cover. |
| Elevation | SRTM DEM | 90 m; clipped to country boundary | Integrates climatic variation; highlands tend to have lower transmission. |
| Relative humidity | ERA5-Land | Derived from temperature and dew-point; annual mean (%) | High humidity increases mosquito survival. |
| Urbanicity (built-up areas) | MODIS MCD12Q1 | Binary indicator (class 13 = urban) | Urban areas generally have lower risk due to infrastructure and fewer breeding sites. |

We point out that in order to compute relative humidity, we first extract hourly estimates of air temperature and dew-point temperature at 2 meters above ground level from the ERA5-Land reanalysis product. These two quantities are then combined using a standard empirical formula derived from the Clausius–Clapeyron relation:

$$\mathrm{RH} = 100 \times \left( \frac{e^{\frac{17.625 \cdot T_d}{243.04 + T_d}}}{e^{\frac{17.625 \cdot T}{243.04 + T}}} \right)$$

where $T_d$ is the dew-point temperature and $T$ is the air temperature, both in degrees Celsius. This yields the mean annual relative humidity over the target

region, expressed as a percentage.

Table 5.1 does not provide an exhaustive list of covariates, and other factors such as soil moisture, land surface water, nighttime temperature, or socio-economic indicators like night-time light intensity and land surface emissivity, which may serve as proxies for human activity, infrastructure, or economic development, could also influence malaria risk. However, to maintain this illustrative analysis simpler, we restrict our attention to a core set of covariates that capture key climatic, ecological, and anthropogenic drivers.

In the code below, all layers are clipped to the boundary of Malawi and combined into a single multi-band image, ready to be extracted at survey point locations for subsequent modeling.

```
# ------------------------------
# STEP 2: Covariates from Earth Engine
# ------------------------------
scale_m <- 1000

# Precipitation
precip <- ee$ImageCollection("UCSB-CHG/CHIRPS/DAILY")$
  filterDate("2006-01-01", "2006-12-31")$
  mean()$rename("precip")$clip(mlw_admin0_ee)$toFloat()

# Temperature
lst <- ee$ImageCollection("MODIS/061/MOD11A2")$
  filterDate("2006-01-01", "2006-12-31")$
  select("LST_Day_1km")$mean()$
  multiply(0.02)$subtract(273.15)$rename("lst_c")$
  clip(mlw_admin0_ee)$toFloat()

# NDVI
ndvi <- ee$ImageCollection("MODIS/061/MOD13A2")$
  filterDate("2006-01-01", "2006-12-31")$
  select("NDVI")$mean()$
  multiply(0.0001)$rename("ndvi")$
  clip(mlw_admin0_ee)$toFloat()

# Elevation
elev <- ee$Image("USGS/SRTMGL1_003")$
  rename("elev")$clip(mlw_admin0_ee)$toFloat()

# Humidity from ERA5 (computed from dewpoint and temperature)
era5 <- ee$ImageCollection("ECMWF/ERA5_LAND/HOURLY")$
  filterDate("2006-01-01", "2006-12-31")$
  select(c("dewpoint_temperature_2m", "temperature_2m"))$
```

```
  mean()$clip(mlw_admin0_ee)

td <- era5$select("dewpoint_temperature_2m")$subtract(273.15)
t <- era5$select("temperature_2m")$subtract(273.15)

rh <- td$expression(
  "100 * (exp((17.625 * TD)/(243.04 + TD)) / exp((17.625 *
   ↪  T)/(243.04 + T)))",
  list(TD = td, T = t)
)$rename("humidity")$toFloat()

# Use MODIS Land Cover, version 061
urban_raw <- ee$ImageCollection("MODIS/061/MCD12Q1")$
  filterDate("2006-01-01", "2006-12-31")$
  first()$select("LC_Type1")$clip(mlw_admin0_ee)

# Urban areas are class 13 in IGBP classification
built_up <- urban_raw$eq(13)$rename("built_up")$toFloat()



# Combine all covariates
covariates <- precip$
  addBands(lst)$
  addBands(ndvi)$
  addBands(elev)$
  addBands(rh)$
  addBands(built_up)
```

After defining the multi-band raster image `covariates` in Google Earth Engine, we download it as a GeoTIFF file using the `ee_as_rast()` function from the `rgee` package. The file is saved temporarily and clipped to the bounding box of Malawi. We then read the raster into R using the `terra`.

```
# ------------------------------
# STEP 3: Download Earth Engine raster
# ------------------------------
out_file <- file.path(tempdir(), "malawi_covariates_2006.tif")
region <- mlw_admin0_ee$geometry()$bounds()

rgee::ee_as_rast(
  image = covariates,
  region = region,
  scale = scale_m,
  dsn = out_file,
```

```
  via = "drive"
)

# Load raster
r_covs <- rast(out_file)

# --------------------------------
# STEP 4: Extract covariates at survey locations
# --------------------------------
vals <- terra::extract(r_covs, vect(mlw_sf))
mlw_sf <- bind_cols(mlw_sf, vals[, -1])  # remove ID column
```

The following code generates six raster maps, one for each covariate, using ggplot2. We first convert the raster stack to a data frame and then create a separate ggplot object for each variable, ensuring a consistent and clean visual style across all maps. The plots in Figure 5.6 are arranged in a $2 \times 3$ grid using the base grid package.

```
# 1. Prepare a data frame

r_df <- as.data.frame(r_covs, xy = TRUE, na.rm = TRUE)

# Rename layers for clarity
names(r_df) <- c("x", "y",
                 "Precipitation",
                 "Temperature",
                 "NDVI",
                 "Elevation",
                 "Humidity",
                 "Urbanicity")

# Urbanicity as a factor for a discrete palette
r_df$Urbanicity <- factor(r_df$Urbanicity,
                          levels = c(0, 1),
                          labels = c("Rural", "Urban"))

# 2. Build one map per covariate, each with its own colour scale
↪
p_precip <- ggplot(r_df, aes(x, y, fill = Precipitation)) +
  geom_raster() +
  scale_fill_gradient(name = "mm", low = "lightblue", high =
  ↪  "darkblue") +
  coord_equal() + theme_void() + ggtitle("Precipitation")

p_temp <- ggplot(r_df, aes(x, y, fill = Temperature)) +
```

```
  geom_raster() +
  scale_fill_gradient(name = "°C", low = "lemonchiffon", high =
   ↪  "red3") +
  coord_equal() + theme_void() + ggtitle("LST")

p_ndvi <- ggplot(r_df, aes(x, y, fill = NDVI)) +
  geom_raster() +
  scale_fill_gradient(name = " ", low = "cornsilk", high =
   ↪  "darkgreen") +
  coord_equal() + theme_void() + ggtitle("NDVI")

p_elev <- ggplot(r_df, aes(x, y, fill = Elevation)) +
  geom_raster() +
  scale_fill_gradient(name = "m", low = "grey90", high =
   ↪  "sienna4") +
  coord_equal() + theme_void() + ggtitle("Elevation")

p_humid <- ggplot(r_df, aes(x, y, fill = Humidity)) +
  geom_raster() +
  scale_fill_gradient(name = "%", low = "white", high =
   ↪  "darkcyan") +
  coord_equal() + theme_void() + ggtitle("Humidity")

p_urban <- ggplot(r_df, aes(x, y, fill = Urbanicity)) +
  geom_raster() +
  scale_fill_manual(values = c("lightgrey", "black"),
                    name = " ") +
  coord_equal() + theme_void() + ggtitle("Urban vs Rural")

# 3. Arrange the six plots in a 2 × 3 grid using base 'grid'
grid.newpage()
pushViewport(viewport(layout = grid.layout(2, 3)))
vplayout <- function(row, col)
  viewport(layout.pos.row = row, layout.pos.col = col)

print(p_precip, vp = vplayout(1, 1))
print(p_temp,   vp = vplayout(1, 2))
print(p_ndvi,   vp = vplayout(1, 3))
print(p_elev,   vp = vplayout(2, 1))
print(p_humid,  vp = vplayout(2, 2))
print(p_urban,  vp = vplayout(2, 3))
```

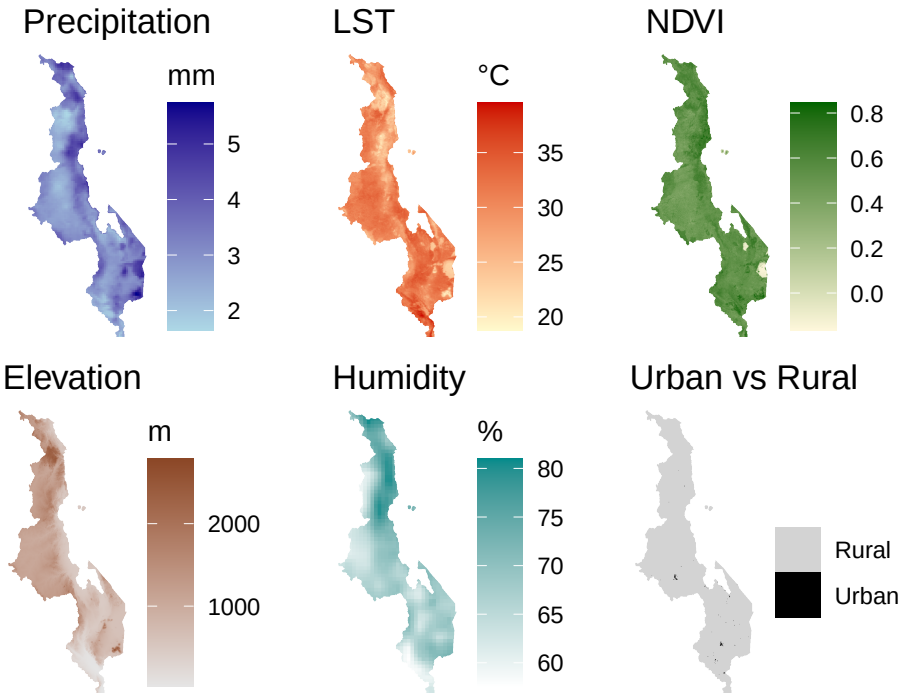



Figure 5.6: Raster maps of environmental covariates extracted from Google Earth Engine for Malawi in 2006. Each panel shows one covariate: precipitation, land surface temperature (LST), vegetation cover (NDVI), elevation, relative humidity, and urbanicity. These layers are used as predictors in the geostatistical modelling of malaria prevalence.

### 5.2.2 Exploratory analysis

To investigate how *Plasmodium falciparum* prevalence varies with environmental conditions, we begin by computing the empirical logit transformation of the observed prevalence at each survey location. Each covariate is then explored in relation to the empirical logit using scatterplots. For continuous covariates, we produce scatter plots with two fitted curves: a LOESS smoother shown in red, and a linear spline model with a single change point shown in blue. The LOESS curve serves as a flexible, nonparametric reference for assessing the general trend in the data. The spline model provides a simple parametric approximation to this trend, with the change point selected based on visual inspection of where the LOESS curve departs from linearity.

```
# 1. Prepare data
mlw_sf <- mlw_sf %>%
  mutate(
```

```
    elogit = log((positive + 0.5) / (examined - positive +
     ↪  0.5)),
    `Precipitation` = precip,
    `Temperature (°C)` = lst_c,
    `NDVI` = ndvi,
    `Elevation (m)` = elev,
    `Humidity (%)` = humidity,
    `Urbanicity` = factor(built_up, levels = c(0, 1), labels =
     ↪  c("Rural", "Urban"))
  )

# 2. Drop geometry and reshape
plot_data_cont <- mlw_sf %>%
  select(elogit, `Precipitation`, `Temperature (°C)`, `NDVI`,
   ↪  `Elevation (m)`, `Humidity (%)`) %>%
  st_drop_geometry() %>%
  pivot_longer(cols = -elogit, names_to = "Covariate", values_to
   ↪  = "Value")

plot_data_cat <- mlw_sf %>%
  select(elogit, Urbanicity) %>%
  st_drop_geometry()

# 3. Define change points
knots <- c(
  "Precipitation" = Inf,
  "Temperature (°C)" = 33,
  "NDVI" = Inf,
  "Elevation (m)" = 400,
  "Humidity (%)" = 65
)

# 4. Create individual continuous plots
plots <- lapply(split(plot_data_cont, plot_data_cont$Covariate),
 ↪  function(df) {
  var <- unique(df$Covariate)
  knot <- knots[[var]]

  df <- df %>%
    mutate(
      linear_part = Value,
      spline_part = pmax(Value - knot, 0)
    )
```

```
  ggplot(df, aes(x = Value, y = elogit)) +
    geom_point(alpha = 0.5, size = 1) +
    geom_smooth(method = "loess", se = FALSE, color = "red",
     ↪  linewidth = 1) +
    geom_smooth(method = "lm",
                formula = y ~ x + pmax(x - knot, 0),
                se = FALSE, color = "blue", linewidth = 1) +
    labs(x = var, y = NULL) +
    theme_minimal()
})

# 5. Add Urbanicity boxplot
p_urban <- ggplot(plot_data_cat, aes(x = Urbanicity, y =
 ↪  elogit)) +
  geom_boxplot(outlier.size = 0.5) +
  labs(x = "Urbanicity", y = NULL) +
  theme_minimal()

# 6. Assemble 2 x 3 plot grid
final_plot <- (plots[[1]] | plots[[2]] | plots[[3]]) /
  (plots[[4]] | plots[[5]] | p_urban) +
  plot_annotation(title = "Empirical logit of prevalence vs
   ↪  environmental covariates")

final_plot
```

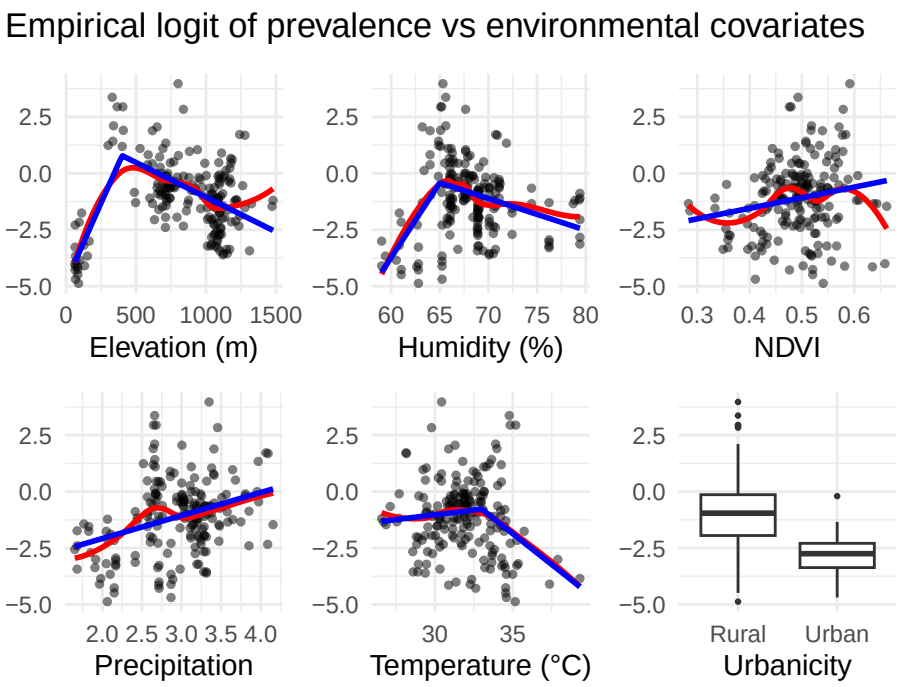



Figure 5.7: Exploratory plots showing the relationship between the empirical logit of *P. falciparum* prevalence and the chosen environmental covariates. Scatter plots show continuous covariates with LOESS (red) and linear spline (blue) fits. A boxplot is used for the binary urbanicity variable. Change points for splines are based on graphical inspection of the LOESS fitted relationship.

Figure 5.7 shows the results of this exploratory approach for the relationship between prevalence and covariates. For some variables, such as temperature, elevation, and humidity, there is a clear nonlinear association that is reasonably captured by a single change point. For others, including precipitation and NDVI, the relationship appears approximately linear, and for this reason no change point is used in the spline fit.

Urbanicity is treated separately as a binary covariate and visualized with a boxplot comparing the empirical logit of prevalence between rural and urban settings. The boxplot indicates that, as we expect, the level of prevalence in urban areas is considerable lower than in rural areas.

To assess the degree of correlation between covariates, we compute the pairwise Pearson correlation coefficients among the continuous environmental variables.

```
library(ggcorrplot)

# Select continuous covariates only
```

```r
cont_vars <- mlw_sf %>%
  st_drop_geometry() %>%
  select(Precipitation = precip,
         Temperature = lst_c,
         NDVI = ndvi,
         Elevation = elev,
         Humidity = humidity)

# Compute correlation matrix
corr_mat <- cor(cont_vars, use = "complete.obs")

# Plot correlation matrix
ggcorrplot(corr_mat,
           method = "circle",
           type = "lower",
           lab = TRUE,
           lab_size = 3,
           colors = c("blue", "white", "red"),
           title = "Correlation between covariates",
           ggtheme = theme_minimal())
```

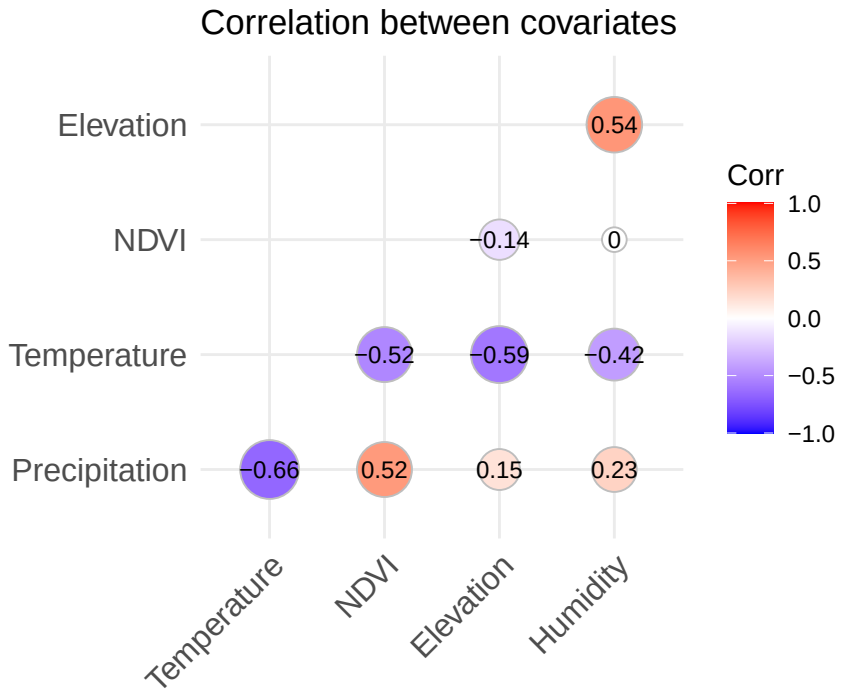



Figure 5.8: Pairwise Pearson correlation matrix between continuous environmental covariates.

The correlations, shown in Figure Figure 5.8, are generally moderate in magnitude, ranging from –0.66 between temperature and precipitation to 0.54 between elevation and relative humidity. These values indicate that collinearity is not a major concern in our model development. Each covariate is likely to provide complementary information about malaria risk, and the absence of strong dependencies also supports the reliability of the scatterplots used to explore their individual relationships with prevalence.

We also perform a principal component analysis (PCA) on the standardized continuous environmental covariates to assess the potential for reducing dimensionality in our model. The code below shows the implementation of the PCA.

```
library(factoextra)

# Prepare only continuous covariates and standardize
pca_data <- mlw_sf %>%
  st_drop_geometry() %>%
  select(Precipitation = precip,
         Temperature = lst_c,
```

```
        NDVI = ndvi,
        Elevation = elev,
        Humidity = humidity) %>%
  scale()

# Perform PCA
pca_result <- prcomp(pca_data)

# Scree plot
fviz_eig(pca_result, addlabels = TRUE, barfill = "steelblue",
 ↪  barcolor = "grey30") +
  theme_minimal()
```

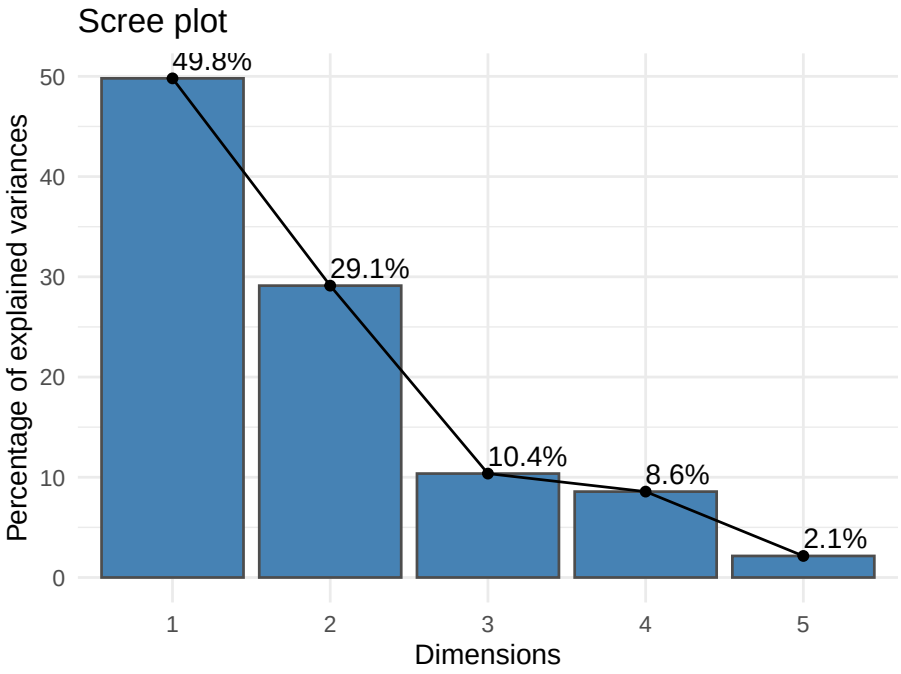



Figure 5.9: Proportion of variance explained by the principal components computed from standardized continuous environmental covariates.

The scree plot in Figure Figure 5.9 shows the proportion of variance explained by each principal component. The first principal component (PC1) explains approximately 50% of the total variation, suggesting it captures a substantial environmental gradient across the study region. Although additional components may carry useful information, PC1 stands out as a potential candidate for use in model building as a composite environmental index and spatial

predictor of malaria prevalence.

Table 5.2: Loadings of the first two principal components. Each value indicates the contribution of a standardized covariate to the corresponding component.

| Covariate | PC1 | PC2 |
|---|---|---|
| Precipitation | -0.480 | -0.339 |
| Temperature | 0.596 | 0.043 |
| NDVI | -0.338 | -0.605 |
| Elevation | -0.391 | 0.560 |
| Humidity | -0.385 | 0.451 |

Table Table 5.2 displays the loadings of the first two principal components. The sign and magnitude of each loading reflect how much each standardized covariate contributes to the corresponding component. PC1 shows strong positive loading for temperature and strong negative loadings for precipitation, humidity, and elevation. This suggests that PC1 represents a gradient from cool, humid, high-elevation areas with more rainfall to hotter, drier lowland regions. NDVI also contributes negatively, implying more vegetation cover in cooler, wetter zones. We can thus interpret PC1 as a general heat-aridity gradient, which well aligns with known ecological drivers of malaria risk.

```
# Create raster for PC1
r_stack_std <- scale(r_covs[[c("precip", "lst_c", "ndvi",
 ↪  "elev", "humidity")]])
pc1_weights <- c(-0.480, 0.596, -0.338, -0.391, -0.385)
pc1_rast <- sum(r_stack_std * pc1_weights)
names(pc1_rast) <- "PC1"

# Extract PC1 values
pc1_vals <- terra::extract(pc1_rast, vect(mlw_sf))
mlw_sf$PC1 <- pc1_vals[, 2]

# Convert raster to df for ggplot
pc1_df <- as.data.frame(pc1_rast, xy = TRUE, na.rm = TRUE)
names(pc1_df) <- c("x", "y", "PC1")

# --- Create PC1 map plot ---
p_map <- ggplot(pc1_df, aes(x = x, y = y, fill = PC1)) +
  geom_raster() +
  scale_fill_viridis_c(option = "C") +
  coord_equal() +
  theme_void() +
  ggtitle("PC1: Environmental Gradient")
```

```r
# --- Create scatterplot with LOESS and spline ---
knot_pc1 <- 0.75
scatter_df <- mlw_sf %>%
  mutate(spline_part = pmax(PC1 - knot_pc1, 0))

p_scatter <- ggplot(scatter_df, aes(x = PC1, y = elogit)) +
  geom_point(alpha = 0.5, size = 1) +
  geom_smooth(method = "loess", se = FALSE, color = "red",
   ↪  linewidth = 1) +
  geom_smooth(method = "lm", formula = y ~ x + pmax(x -
   ↪  knot_pc1, 0),
              se = FALSE, color = "blue", linewidth = 1) +
  theme_minimal() +
  labs(x = "PC1", y = "Empirical logit") +
  ggtitle("Empirical logit vs PC1")

# --- Combine using patchwork ---
# Also add some margin space
p_map <- p_map + theme(plot.margin = margin(r = 20))
p_scatter <- p_scatter + theme(plot.margin = margin(l = 20))
p_map + p_scatter
```

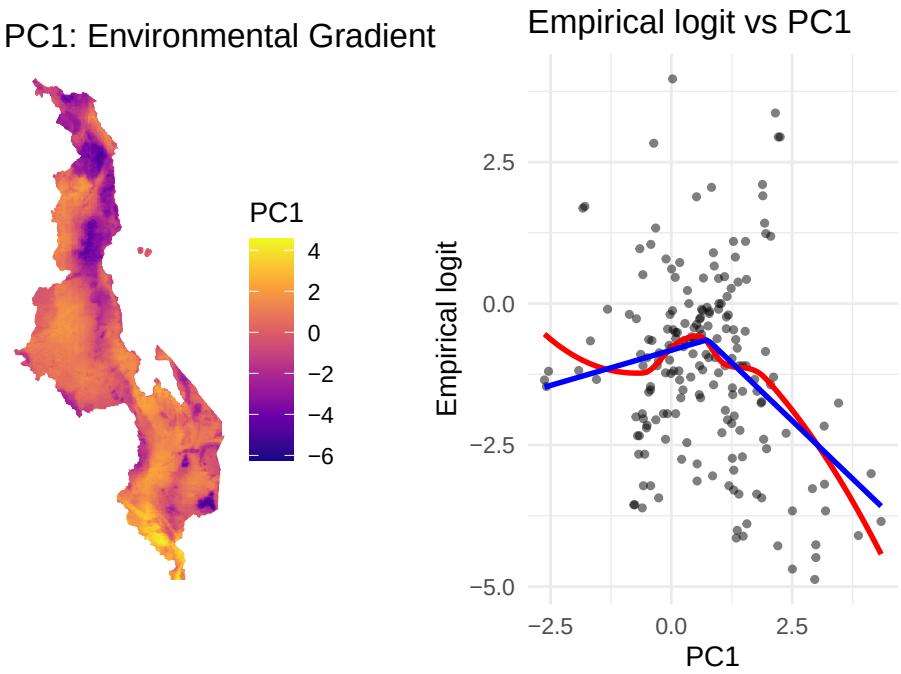



Figure 5.10: Left: Spatial distribution of the first principal component (PC1). Right: Empirical logit of malaria prevalence plotted against PC1 with LOESS (red) and linear spline (blue) fits.

Figure 5.10 displays the map of PC1 alongside its relationship with malaria prevalence. As done previously, to aid interpretation, we fit a LOESS curve (red) to visualise the general trend, and a linear spline (blue) with a change point at 0.75 to capture a simplified parametric relationship. The spline fit suggests a nonlinear association: malaria risk increases with PC1 up to around 0.75, after which it begins to decline. This pattern implies that malaria prevalence is highest in areas with intermediate PC1 values, i.e. environments that are neither too cold and wet nor too hot and arid. In other words, there appears to be an optimal range of environmental conditions, as summarised by PC1, that are most conducive to malaria transmission. Beyond this range, particularly in more heat-arid regions, the risk may decline due to ecological constraints on mosquito survival or parasite development.

In the next section, we compare two modelling approaches. The first model includes the original environmental covariates using the linear spline specifications identified during our exploratory analysis. The second model replaces these covariates with PC1 alone, treating it as a composite spatial predictor that summarises the dominant environmental gradient in the study area. In both models, we include urbanicity as a separate fixed effect. Urbanicity is

excluded from the PCA because it is a binary variable and highly unbalanced, with most survey locations classified as rural. Modelling it separately ensures that its distinct contribution to malaria risk, more linked to infrastructure, housing conditions, and land use, is properly accounted for.

### 5.2.3 Model fitting and spatial prediction

In this section, we fit two geostatistical models to malaria prevalence data from Malawi, comparing the effects of using separate environmental covariates versus summarizing them into a principal component.

The first model (`mod_all_cov`) includes all environmental covariates as separate predictors. Namely these are precipitation $d_{\text{prec}}(x)$, NDVI $d_{\text{ndvi}}(x)$, temperature $d_{\text{temp}}(x)$, elevation $d_{\text{elev}}(x)$, humidity $d_{\text{hum}}(x)$, and urbanicity $d_{\text{urb}}(x)$. Following from the results of the exploratory analysis, nonlinear effects for temperature, elevation, and humidity are modeled using linear splines with thresholds at 33°C, 400 m, and 65%, respectively. The logit-linear model is:

$$\begin{aligned}\log\left\{\frac{p(x)}{1-p(x)}\right\} = \beta_0 &+ \beta_1 d_{\text{prec}}(x) + \beta_2 d_{\text{ndvi}}(x) + \beta_3 d_{\text{temp}}(x) + \beta_4 \max\{d_{\text{temp}}(x) - 33,\ 0\} \\ &+ \beta_5 d_{\text{elev}}(x) + \beta_6 \max\{d_{\text{elev}}(x) - 400,\ 0\} \\ &+ \beta_7 d_{\text{hum}}(x) + \beta_8 \max\{d_{\text{hum}}(x) - 65,\ 0\} \\ &+ \beta_9 d_{\text{urb}}(x) + S(x)\end{aligned}$$

.

The second model (`mod_pca`) replaces the continuous covariates with their first principal component $\text{PC}_1(x)$, allowing for a spline above 0.75. The model becomes:

$$\log\left\{\frac{p(x)}{1-p(x)}\right\} = \beta_0 + \beta_1 \text{PC}_1(x) + \beta_2 \max\{\text{PC}_1(x) - 0.75, 0\} + \beta_3 d_{\text{urb}}(x) + S(x)$$

This formulation allows us to assess whether the dimensionality reduction via PCA retains the essential variation in environmental risk while simplifying the model structure.

```
# Model with all the covariates as separate predictors
mod_all_cov <-
```

```
glgpm(positive ~
    precip +
    ndvi +
    lst_c + pmax(lst_c - 33, 0) +
    elev + pmax(elev - 400, 0) +
    humidity + pmax(humidity - 65, 0) +
    built_up + gp(),
    den = examined,
    data = mlw_sf,
    family = "binomial")

# Model with all the covariates (except `built_up`)
# combined into PC1 which is then used as predictor
mod_pca <-
  glgpm(positive ~
          PC1 + pmax(PC1 - 0.75, 0) +
          built_up + gp(),
        den = examined,
        data = mlw_sf,
        family = "binomial")
```

After fitting the two models, when can then compare the summaries of the fits.

```
summary(mod_all_cov)
```

```
Call:
glgpm(formula = positive ~ precip + ndvi + lst_c + pmax(lst_c -
    33, 0) + elev + pmax(elev - 400, 0) + humidity +
    pmax(humidity -
    65, 0) + built_up + gp(), data = mlw_sf, family =
    "binomial",
    den = examined)

Binomial geostatistical linear model
Link: canonical (logit)
Inverse link function = 1 / (1 + exp(-x))

'Lower limit' and 'Upper limit' are the limits of the 95%
confidence level intervals

Regression coefficients
                          Estimate Lower limit Upper limit
                          StdErr  z.value
```

```
(Intercept)           -4.1484e+01 -5.4098e+01 -2.8871e+01
6.4356e+00  -6.4461
precip                 3.6000e-01  2.0822e-01  5.1178e-01
7.7441e-02   4.6487
ndvi                   5.1991e+00  4.3228e+00  6.0754e+00
4.4710e-01  11.6285
lst_c                  9.9929e-02  6.9301e-02  1.3056e-01
1.5626e-02   6.3948
pmax(lst_c - 33, 0)    3.0237e-01  2.5342e-01  3.5132e-01
2.4973e-02  12.1076
elev                   1.0984e-02  1.0581e-02  1.1386e-02
2.0527e-04  53.5074
pmax(elev - 400, 0)   -1.1358e-02 -1.1732e-02 -1.0984e-02
1.9088e-04 -59.5030
humidity               4.7098e-01  4.4719e-01  4.9476e-01
1.2135e-02  38.8129
pmax(humidity - 65, 0) -6.8110e-01 -7.0309e-01 -6.5912e-01
1.1217e-02 -60.7232
built_up              -6.8574e-01 -7.4216e-01 -6.2931e-01
2.8788e-02 -23.8200
                        p.value
(Intercept)           1.148e-10 ***
precip                3.340e-06 ***
ndvi                  < 2.2e-16 ***
lst_c                 1.607e-10 ***
pmax(lst_c - 33, 0)   < 2.2e-16 ***
elev                  < 2.2e-16 ***
pmax(elev - 400, 0)   < 2.2e-16 ***
humidity              < 2.2e-16 ***
pmax(humidity - 65, 0) < 2.2e-16 ***
built_up              < 2.2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Spatial Gaussian process
Matern covariance parameters (kappa=0.5)
                    Estimate Lower limit Upper limit
Spatial process var.  0.95284     0.91937      0.9875
Spatial corr. scale  15.60594    15.02071     16.2140
Variance of the nugget effect fixed at 0

Log-likelihood: 13.833
```

```
summary(mod_pca)
```

```
Call:
glgpm(formula = positive ~ PC1 + pmax(PC1 - 0.75, 0) + built_up
+
    gp(), data = mlw_sf, family = "binomial", den = examined)

Binomial geostatistical linear model
Link: canonical (logit)
Inverse link function = 1 / (1 + exp(-x))

'Lower limit' and 'Upper limit' are the limits of the 95%
confidence level intervals

Regression coefficients
                    Estimate Lower limit Upper limit    StdErr
                     z.value
(Intercept)         -0.496575   -0.595285   -0.397866  0.050363
 -9.8600
PC1                  0.258180    0.194448    0.321912  0.032517
  7.9399
pmax(PC1 - 0.75, 0) -1.076688   -1.170656   -0.982719  0.047944
-22.4572
built_up            -0.542019   -0.654257   -0.429781  0.057265
 -9.4650
                      p.value
(Intercept)         < 2.2e-16 ***
PC1                 2.024e-15 ***
pmax(PC1 - 0.75, 0) < 2.2e-16 ***
built_up            < 2.2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Spatial Gaussian process
Matern covariance parameters (kappa=0.5)
                    Estimate Lower limit Upper limit
Spatial process var.   2.0008      1.7939      2.2316
Spatial corr. scale   30.6995     27.8239     33.8724
Variance of the nugget effect fixed at 0

Log-likelihood: 15.90143
```

In the output above, we see that, compared to the model using all covariates as separate predictors, the model including only PC1 exhibits both a larger estimated spatial correlation scale and higher spatial variance. This indicates that replacing the original covariates with PC1 captures a smaller proportion of the spatial variation in prevalence, leaving the spatial Gaussian process to

account for more of the structured residual variability.

We then compare the predicted prevalence from each model using a regular 5 by 5 km grid covering the whole of Malawi, and display the results in Figure 5.11. The maps show broadly similar spatial patterns, with both models identifying areas of high and low predicted prevalence in the southern region of the country. However, these visual comparisons do not allow us to determine which model performs better. We address this question in the next stage of the analysis by formally assessing the calibration and sharpness of the predictions generated by each model.

```
# Create a 5 by 5 regular grid
grid_mlw <- create_grid(mlw_admin0_sf, spat_res = 5)

# Extract the covariates over the grid
r_covs <- terra::project(r_covs, paste0("epsg:",mlw_crs))
pc1_rast <- terra::project(pc1_rast, paste0("epsg:",mlw_crs))

predictors <- cbind(terra::extract(r_covs,
 ↪  st_coordinates(grid_mlw)),
                     terra::extract(pc1_rast,
 ↪  st_coordinates(grid_mlw)))

# Prevalence prediction over the grid for the two fitted models
pred_all_cov <- pred_over_grid(mod_all_cov,
                               grid_pred = grid_mlw,
                               predictors = predictors)

pred_pca <- pred_over_grid(mod_pca,
                           grid_pred = grid_mlw,
                           predictors = predictors)

pred_all_cov_prev <- pred_target_grid(pred_all_cov,
                                      f_target = list(prev =
                                       ↪  function(x)
                                       ↪  exp(x)/(1+exp(x))))

pred_pca_prev <- pred_target_grid(pred_pca,
                              f_target = list(prev = function(x)
                               ↪  exp(x)/(1+exp(x))))
```

```
par(mfrow = c(1,2))
plot(pred_all_cov_prev, which_target = "prev", which_summary =
 ↪  "mean",
     main = "Model: 'all_cov'", range = c(0,1))
plot(pred_pca_prev, which_target = "prev", which_summary =
 ↪  "mean",
```

```
    main = "Model: 'pca'", range = c(0,1))
```

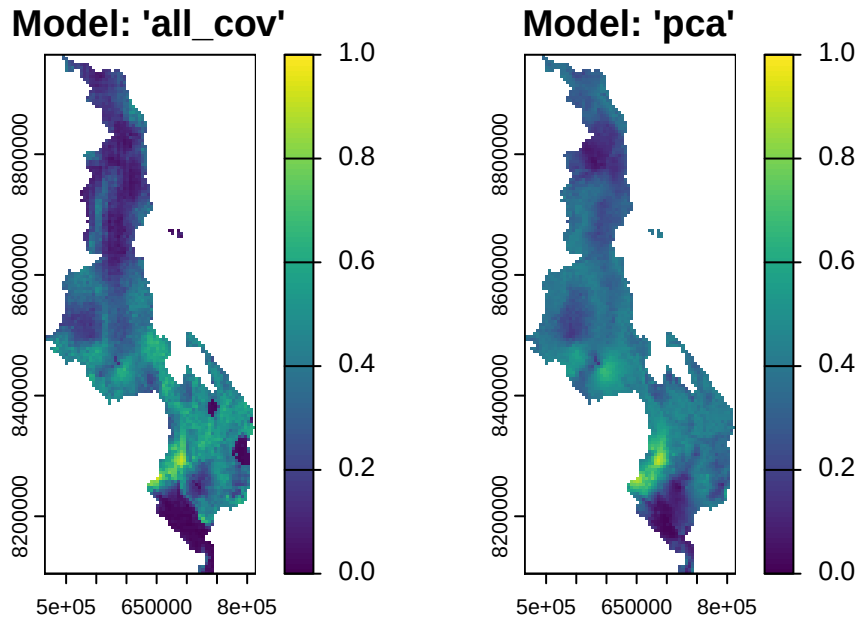



Figure 5.11: Predictive mean of prevalence obtained from geostatistical models using all covariates ('all_cov') versus the first principal component ('pca') of the standardized continuous environmental variables.

### 5.2.4 Comparison of the predictive performance between models

To evaluate predictive performance, we use three spatially distinct hold-out test sets corresponding to the Northern, Central, and Southern regions of Malawi, as illustrated in Figure 5.12. These were constructed using the `assess_pp` function with `method = "cluster"` and `fold = 3`, ensuring geographically stratified cross-validation.

```
assess_pred_mlw <-
assess_pp(list(all_cov = mod_all_cov,
               pca = mod_pca),
          method = "cluster",
          which_metric = c("AnPIT", "CRPS"),
          iter = 1,
          fold = 3)
```

The CRPS score shows that the model including all environmental covariates achieves better predictive accuracy across all test sets. Specifically, it yields

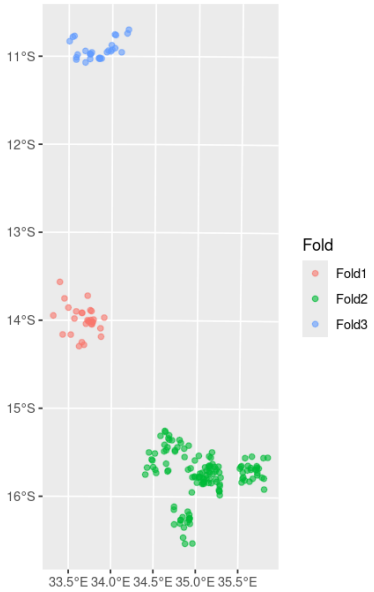



Figure 5.12: Geographic distribution of sampled locations in Malawi, coloured by their assignment to one of three cross-validation test sets.

a lower average CRPS. This improvement is consistent across the three hold-out data, with particularly notable gains in Test Set 1. Additionally, the non-randomized probability integral transform (AnPIT) plot, shown in Figure 5.13, demonstrates that the full covariate model is better calibrated: its cumulative distribution curves lie closer to the identity line, suggesting that the predicted probabilities align more closely with observed prevalence.

```
plot_AnPIT(assess_pred_mlw, mode = "all")

summary(assess_pred_mlw)
```

```
Summary of Cross-Validation Scores
----------------------------------
Model: all_cov
  Test Set 1:
    CRPS: 1.9452
  Test Set 2:
    CRPS: 2.1383
  Test Set 3:
    CRPS: 1.5148
  Overall average across test sets:
    CRPS: 1.9067
```

```
Model: pca
  Test Set 1:
    CRPS: 2.8064
  Test Set 2:
    CRPS: 2.5029
  Test Set 3:
    CRPS: 1.9321
  Overall average across test sets:
    CRPS: 2.6220
```

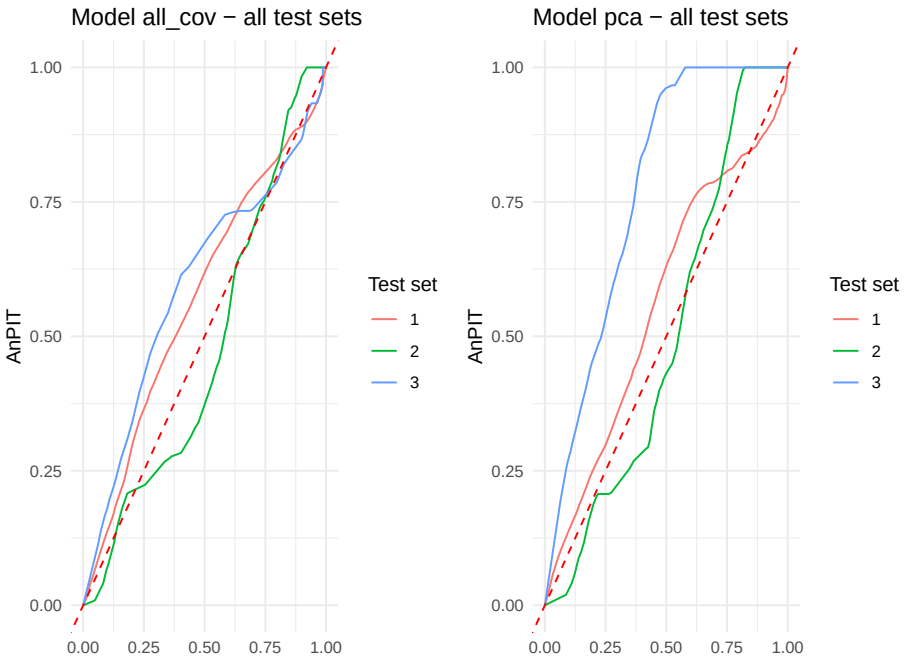



Figure 5.13: Plot of the average non-randomized probability integral transform (AnPIT) function to assess the calibration of the two geostatistical models ('all_cov' and 'pca') for the malaria prevalence data from Malawi.

### 5.2.5 Summary and conclusions

In this case study, we have walked through the full pipeline of conducting a geostatistical analysis of malaria prevalence using Malaria Indicator Survey data from Malawi. We began by downloading geo-referenced survey data from the `malariaAtlas` R package and retrieving relevant environmental covariates from Google Earth Engine. These covariates, selected for their mechanistic relevance to malaria transmission, were processed, clipped to the national boundary, and extracted at survey locations to be used as predictors in subsequent

modelling.

A critical step in the analysis involved the exploration of relationships between environmental conditions and malaria prevalence. Through scatterplots and spline fits, we identified key nonlinearities in variables such as temperature, elevation, and humidity, which informed the functional form of covariate effects in the regression model. This exploratory work is essential not only for model formulation but also for understanding the ecological underpinnings of disease risk.

To address potential collinearity and reduce the complexity of the model, we also illustrated the use of PCA as a method for dimensionality reduction. The first principal component (PC1) captured a dominant environmental gradient, summarizing variation across temperature, humidity, elevation, and vegetation, which was then used as a composite predictor in a reduced model. A simpler alternative is to let the data guide you through stepwise selection based on p-values—for example, adding or removing covariates until all retained terms meet a preset significance threshold. When the pool of candidate predictors is large relative to the sample size, however, stepwise methods can become unstable and over-fit; in such cases penalised approaches (e.g. ridge, lasso) or resampling-based strategies are usually preferred. A thorough discussion of these issues, with practical recommendations, is given in Section 4.7 of Harrell (2015).

We compared two geostatistical models: one using all environmental covariates as separate predictors (with spline terms for nonlinear effects), and another using PC1 as a summary predictor. Cross-validation based on geographically stratified folds revealed that the full model was better calibrated and achieved sharper predictions, as measured by lower CRPS and AnPIT curves closer to the identity line.

We encourage the reader to explore Exercise 2 at the end of this chapter, where you will investigate whether including a second principal component (PC2) alongside PC1 leads to improved predictive performance. This offers an opportunity to assess whether PCA-based dimensionality reduction can match or exceed the predictive accuracy of models that include all covariates, while potentially offering greater parsimony.

## 5.3 Mapping the vector index for West Nile Virus in the Sacramento Metropolitan Area, United States

In this section, we analyse entomological surveillance data on *Culex pipiens* in the Sacramento Metropolitan Area (SMA), California, from 2015 to 2021. *Cx. pipiens* is one of the primary vectors of West Nile virus (WNV) in North

America (Petersen, Brault, and Nasci 2013; Kramer, Li, and Shi 2007). WNV is a mosquito borne flavivirus sustained in an enzootic cycle, that is, persistent circulation of the virus within non human animal populations, primarily birds, in a defined geographic area. Transmission is maintained between *Culex* mosquitoes and birds, while humans and other mammals are incidental dead end hosts that do not contribute to further transmission. Most human infections are asymptomatic, a minority present with febrile illness, and a small proportion develop neuroinvasive disease (Petersen, Brault, and Nasci 2013). Surveillance of both vector abundance and WNV infection prevalence is therefore important for anticipating human risk and guiding public health interventions.

In this analysis, we quantify local entomological risk using the vector index (VI), defined as

$$\text{VI} \;=\; \text{abundance} \times \text{infection prevalence}. \tag{5.3}$$

Here, *abundance* is the mean number of female mosquitoes collected per trap night, and *infection prevalence* is the estimated proportion of mosquitoes infected with WNV in the vector population. In practice, prevalence is inferred from pooled PCR testing using maximum likelihood estimators for pooled samples (Centers for Disease Control and Prevention 2024). Many operational analyses also report the minimum infection rate and apply either the maximum likelihood estimate or the minimum infection rate when computing the vector index (e.g., Bolling et al. 2009; Jones et al. 2011). In this section, we extend these approaches using model-based geostatistics.

In what follows, we model abundance and infection prevalence for *Cx. pipiens* as two independent processes and combine the resulting estimates to obtain VI. We return to this assumption and other limitations of the analysis in Section 5.3.3.

### 5.3.1 Modelling the abundance of *Culex pipiens*

We begin by loading the necessary libraries and the data derived from the `vectorsurvR` package in R. The original datasets, `sample_pools` and `sample_collections`, contain mosquito surveillance data from California, including species identification, collection date, location, pathogen testing results, trap type, and trapping effort. For this case study, these have been processed to produce `abund_sma` in the `RiskMap` package, which summarises the abundance of female *Culex pipiens* within the Sacramento Metropolitan Area (SMA).

```
rm(list = ls())

library(ggplot2)
library(patchwork)
```

```
data(abund_sma)
str(abund_sma)
## 'data.frame':    1551 obs. of  6 variables:
##  $ lon          : num  -121 -121 -121 -121 -121 ...
##  $ lat          : num  38.9 38.8 38.9 38.8 38.9 ...
##  $ total_females: int  1 36 6 31 1 1 1 1 1 4 ...
##  $ date         : Date, format: "2015-01-02" "2015-01-02" ...
##  $ trap_nights  : int  15 15 15 8 8 8 8 6 6 6 ...
##  $ trap_type    : chr  "NJLT" "NJLT" "NJLT" "GRVD" ...
```

The outcome variable for this analysis is `total_females`, which records the total number of *Cx. pipiens* captured in a single trap. The variable `trap_nights` indicates the number of nights the trap was deployed and is used to account for variation in sampling effort across traps. The dataset also records the type of trap used (`trap_type`), which can influence capture rates. Trap types include New Jersey Light Traps (`NJLT`), Gravid Traps (`GRVD`), Mosquito Magnet Traps (`MMT`), BG-Sentinel Traps (`BGSENT`), $CO_2$-baited traps (`CO2`), Backpack aspirators (`BACKPACK`), Lockyer traps (`LCKR`), and Ovitraps (`OVI`). These traps differ in their mode of attraction: for example, light traps attract host-seeking females using light, gravid traps target egg-laying females with an infusion lure, $CO_2$-baited and BG-Sentinel traps mimic host cues through carbon dioxide and human scent, and backpack or hand-collection methods are opportunistic. Based on average catch per trap-night, we group these into low-yield traps (Backpack, Lockyer, New Jersey Light Trap, Ovitrap), moderate-yield traps (Gravid Trap, Mosquito Magnet), and high-yield traps ($CO_2$-baited and BG-Sentinel), providing a more interpretable classification for analysing trap efficiency in subsequent modelling. We thus create the variable `trap_group` for later use in the model fitting as follows.

```
abund_sma <- abund_sma %>%
  mutate(trap_group = case_when(
    trap_type %in% c("BACKPACK", "LCKR", "NJLT", "OVI") ~ "low
↪   yeild",
    trap_type %in% c("GRVD", "MMT") ~ "moderate yeild",
    trap_type %in% c("CO2", "BGSENT") ~ "high yield"
  ))
```

To provide spatial context, we use the `rgeoboundaries` package to download second-level administrative boundaries (ADM2) for the United States. We extract the boundaries for Placer, El Dorado and Sacrament counties which make up the SMA.

```
library(rgeoboundaries)
```

```
# Get California ADM2 units and filter to relevant counties
ca_counties <- geoboundaries("United States of America", adm_lvl
 ↪  = "ADM2")
sma_names <- c("Sacramento", "Placer", "El Dorado")
sma_boundaries <- ca_counties[ca_counties$shapeName %in%
 ↪  sma_names, ]
```

Given the time span of the data, it is important to consider the potential for temporal variation in mosquito abundance and infection prevalence. Changes in environmental conditions, climate, or vector control efforts may lead to substantial year-to-year fluctuations in mosquito populations. Additionally, if different geographic areas were sampled in different years, this could introduce spatial confounding, making it difficult to disentangle spatial from temporal effects.

The top panel of Figure 5.14 reveals that while some spatial variation in sampling locations is present across years, there remains sufficient geographic overlap to enable estimation of temporal trends in mosquito counts. Overall, *Cx. pipiens* abundance declines over time, although the trend is more pronounced in high-yield traps, while moderate- and low-yield traps exhibit a more variable pattern. The decline in *Cx. pipiens* likely reflects two key drivers: reduced availability of aquatic breeding sites due to drought, and intensified vector control efforts. Bhattachan et al. (2023) reported that *Cx. pipiens* populations in southern California dropped by about 40 during the 2012–2016 drought, linked to water-use restrictions that limited larval habitat in storm drains and catch basins. At the same time, vector control districts expanded targeted interventions. For example, the Sacramento–Yolo district treated over 160,000 drains in 2018 alone (Sacramento-Yolo Mosquito and Vector Control District 2018), and the statewide response plan formalised such practices as part of integrated mosquito management (California Department of Public Health 2025). For these reasons, our model shall include a covariate for year, with a log-linear effect on the mean number of *Cx. pipiens*, under the expectation that abundance will show a decreasing trend over time.

```
# Creation of the variable year
abund_sma$year <- as.numeric(substr(abund_sma$date, 1, 4))

# bbox for coord_sf limits
bb <- st_bbox(sma_boundaries)
abund_sma <- st_as_sf(abund_sma, coords = c("lon","lat"), crs =
 ↪  4326, remove = FALSE)


# Panel 1: faceted map, colored by trap_group (scales fixed!)
p_locs <- ggplot() +
```

```
  geom_sf(data = sma_boundaries, fill = NA, color = "black",
   ↪  linewidth = 0.3) +
  geom_sf(data = abund_sma, color = "black", alpha = 0.85, size
   ↪  = 0.9) +
  coord_sf(xlim = c(bb["xmin"], bb["xmax"]), ylim =
   ↪  c(bb["ymin"], bb["ymax"])) +
  facet_wrap(~ year, ncol = 4) +
  theme_minimal() +
  labs(title = "Trap locations by year") +
  theme(
    strip.text = element_text(size = 13, face = "bold"),
    plot.title = element_text(size = 16, face = "bold", hjust =
     ↪  0.5),
    axis.text = element_blank(),
    axis.ticks = element_blank()
  )

# Panel 2: average mosquitoes per trap by year & trap_group
annual_summary <- abund_sma %>%
  group_by(year, trap_group) %>%
  summarise(
    total_mosquitoes = sum(total_females, na.rm = TRUE),
    total_traps = n(),
    mosq_per_trap = ifelse(total_traps > 0, total_mosquitoes /
     ↪  total_traps, NA_real_),
    .groups = "drop"
  )

p_pertrap <- ggplot(annual_summary,
                    aes(x = factor(year), y = mosq_per_trap,
                     ↪  fill = trap_group)) +
  geom_col(position = position_dodge(width = 0.7), width = 0.7)
   ↪  +
  theme_minimal() +
  labs(title = "Average mosquitoes per trap by year and trap
   ↪  group",
       x = "Year", y = "Mosquitoes per trap", fill = "Trap
        ↪  group") +
  theme(
    plot.title = element_text(size = 14, face = "bold", hjust =
     ↪  0.5),
    axis.title = element_text(size = 12),
    axis.text = element_text(size = 11)
  )
```

```
# Combine
(p_locs / p_pertrap) +
  plot_layout(heights = c(3, 1))
```

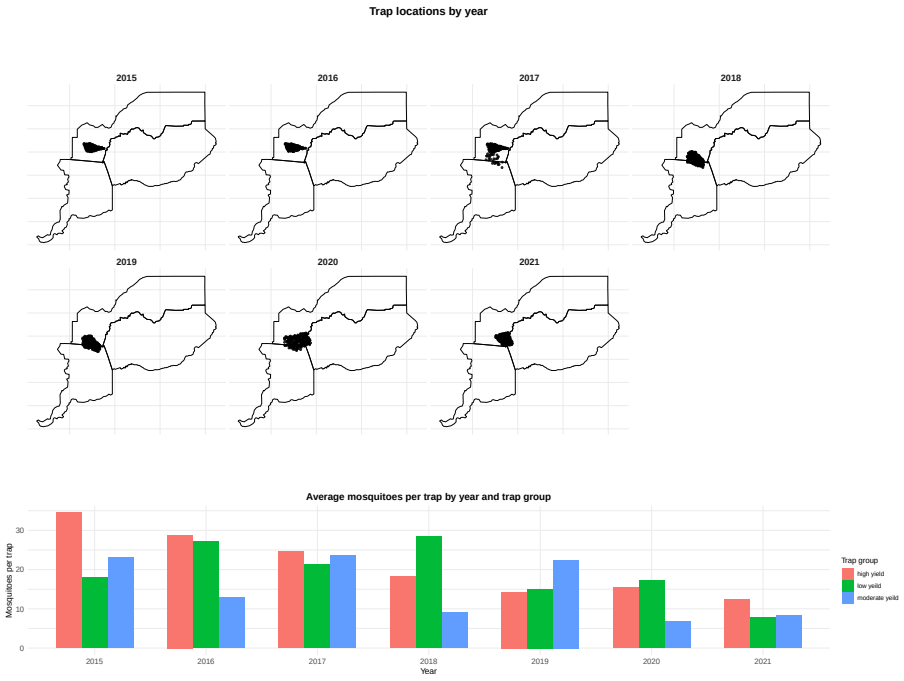



Figure 5.14: Spatial and temporal overview of *Cx. pipiens* surveillance in the Sacramento Metropolitan Area from 2015 to 2021. The top panel shows the spatial distribution of trap locations for each year. The bottom panel displays the annual average number of mosquitoes captured per trap by trap group, illustrating variation in abundance over time.

In our analysis, we first assess whether there is residual spatial correlation in mosquito counts by fitting the following Poisson mixed-effects model. Let $Y_i$ denote the number of trapped female *Cx. pipiens* at location $x_i$ and year $t_i$. Conditionally on independent and identically distributed zero-mean Gaussian random effects $Z_i \sim \mathcal{N}(0, \sigma^2)$, we assume:

$$Y_i \mid Z_i \sim \text{Poisson}(m_i \lambda_i),$$

where $m_i$ is the number of nights during which the trap has been deployed, and $\lambda_i$ is the average number of $Cx.pipiens$ per trap-night, which we model

as:

$$\log\{\lambda_i\} = \sum_{j=1}^{3} d_j(x_i, t_i)\beta_j + \beta_4 t_i + Z_i \\ = \mu_i + Z_i, \tag{5.4}$$

where the $d_j(x_i, t_i)$ are binary indicators for the type of trap ($j = 1$ corresponding to "high-yield", $j = 2$ to "moderate-yield" and $j = 3$ to "low-yield") and $\mu_i = \sum_{j=1}^{3} d_j(x_i, t_i)\beta_j + \beta_4 t_i$. We then fit this model using the `glmer` function from the `lme4` package as follows.

```
abund_sma$loc <- 1:nrow(abund_sma)

# In the fit, we scale the variable year to help convergence
glmer_wvn <- glmer(total_females ~ -1 + trap_group + scale(year)
 ↪  + offset(log(trap_nights)) + (1 | loc),
                   data = abund_sma, family = poisson,
                   nAGQ = 100)
summary(glmer_wvn)
```

```
Generalized linear mixed model fit by maximum likelihood
(Adaptive
  Gauss-Hermite Quadrature, nAGQ = 100) [glmerMod]
 Family: poisson  ( log )
Formula:
total_females ~ -1 + trap_group + scale(year) +
offset(log(trap_nights)) +
    (1 | loc)
   Data: abund_sma

      AIC       BIC    logLik -2*log(L)  df.resid
   5715.5    5742.2   -2852.7    5705.5      1546

Scaled residuals:
     Min       1Q   Median       3Q      Max
-0.81365 -0.27049 -0.02063  0.10058  0.40034

Random effects:
 Groups Name        Variance Std.Dev.
 loc    (Intercept) 2.356    1.535
Number of obs: 1551, groups:  loc, 1551

Fixed effects:
                        Estimate Std. Error z value Pr(>|z|)
```

```
trap_grouphigh yield      1.68272    0.06643  25.331  < 2e-16
***
trap_grouplow yeild      -0.41146    0.08736  -4.710 2.48e-06
***
trap_groupmoderate yeild  0.29882    0.06762   4.419 9.92e-06
***
scale(year)              -0.20370    0.04206  -4.843 1.28e-06
***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Correlation of Fixed Effects:
            trp_grphy trp_grply trp_grpmy
trp_grplwyl  0.006
trp_grpmdry -0.013     0.002
scale(year) -0.119    -0.014     0.150
```

The model summary indicates that, as expected, average mosquito counts decline over time. The effects associated with the different levels of `trap_group` also follow expectations, with high-yield traps having the highest average counts, followed by moderate- and low-yield traps. The relatively large estimate of the variance $\sigma^2$ of the random effect $Z_i$ suggests substantial extra-Poisson variation between traps that is not explained by the yearly decline and the type of trap.

We then use the empirical variogram computed on the estimated $Z_i$ to assess the presence of residual spatial correlation.

```
# Extracting the estimates of the random effects
wnv_summary$Z_hat <- ranef(glmer_wvn)$loc[,1]

# Computing the empirical variogram
variogram_wnv <-
s_variogram(data = wnv_summary,
            variable = "Z_hat",
            n_permutation = 1000,
            scale_to_km = TRUE,
            bins = seq(0,10, length = 15))
```

We plot the empirical variogram and add the envelope generated under the assumption of absence of spatial correlation.

```
plot_s_variogram(variogram_wnv, plot_envelope = TRUE)
```

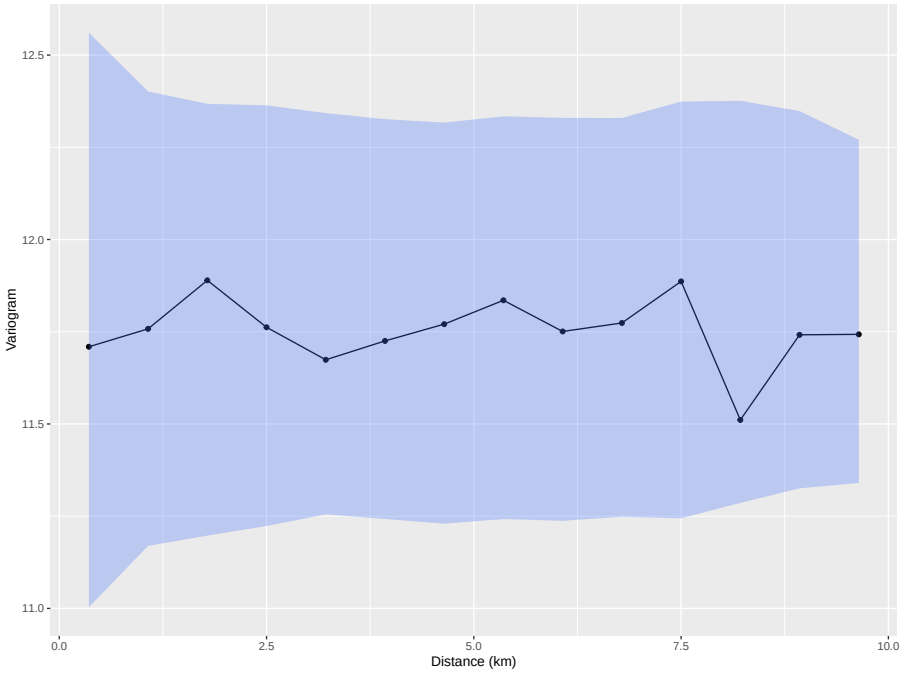



Figure 5.15: Empirical variogram (black line) for based on the random effects fitted from the model in Equation 5.4. The shaded blue area correspond to the envelope of spatial independence at 95% confidence level generated using a permuation test.

The empirical variogram shown in Figure 5.15 lies entirely within the simulation envelope, indicating no detectable spatial correlation in the residuals. This lack of evidence may be due to the fact that mosquito abundance is influenced by very local environmental conditions such as vegetation, drainage, or breeding site availability, which vary at spatial scales smaller than 1 km. Since only a few trap locations in the dataset are spaced closely enough to capture such fine-scale variation, the analysis may lack the resolution needed to detect spatial dependence. Therefore, although a geostatistical model is not justified in this case, we can still pursue our objective of estimating the average number of mosquitoes at the sampled locations using the model in Equation 5.4.

The principles used in this case to derive the predictive distribution of $\lambda(x_i)$, the expected number of mosquitoes at a sampled location $x_i$, are similar to those applied in geostatistical models. However, they take a simplified form here because we do not need to account for spatial correlation. The predictive

distribution is given by:

$$[\lambda(x_i) \mid Y_i = y_i] = \int [Z_i]\,[Y_i \mid Z_i]\,dZ_i \tag{5.5}$$

In this expression, $[Z_i]$ denotes the density of a Gaussian distribution with mean zero and variance $\sigma^2$, and $[Y_i \mid Z_i]$ is the likelihood from a Poisson model with mean $\lambda(x_i)$. The integral in Equation 5.5 can be computed numerically in R with relative efficiency. This approach also allows for direct simulation from the predictive distribution, enabling us to compute any desired summary statistics.

For this analysis, we use a custom function called `simulate_random_effects`, which can be copied from Section 5.4.1 and pasted into an R script, as it is not included in the `RiskMap` package. Further details of its implementation are provided in Section 5.4.1; here, we focus solely on the analysis.

Hence, we first simulate 1000 samples from the predictive distribution of $Z_i$ and denote those by $Z_i^{(j)}$, for $i = 1, \ldots, 598$, and $j = 1 \ldots, 1000$.

```
n_samples <- 1000
samples_z <- simulate_random_effects(glmer_wvn, n_sim =
 ↪  n_samples)
```

We then obtain predictive samples for $\lambda(x_i)$ by applying the exponential transformation to the linear predictor $\hat{\mu}_i + Z_i^{(j)}$, where in $\hat{\mu}_i$ we have plugged in the maximum likelihood estimates of the regression coefficients.

The predictions shown in Figure 5.16 indicate that the pattern of the mean number of mosquitoes across sampled locations is broadly comparable across time.

```
# Create dataframe
mosq_df <- data.frame(
  id = 1:nrow(abund_sma),
  mean = mosq_mean,
  lower = mosq_lower,
  upper = mosq_upper
)

# Adding trap_group and year
mosq_df$trap_group <- abund_sma$trap_group[match(mosq_df$id,
 ↪  abund_sma$loc)]
mosq_df$year <- abund_sma$year[match(mosq_df$id, abund_sma$loc)]

# Rearranging the data-frame in increasing order based
# on the mean number of mosquitoes
# by year and trap group
```

```
mosq_df <- mosq_df %>%
  group_by(year, trap_group) %>%
  arrange(mean, .by_group = TRUE) %>%
  mutate(x_order = row_number()) %>%
  ungroup()

library(scales)  # Used for reporting the y-axis on the
 ↪  log-scale

ggplot(mosq_df, aes(x = x_order, y = mean,
                    color = trap_group, fill = trap_group, group
                     ↪  = trap_group)) +
  geom_ribbon(aes(ymin = lower, ymax = upper), alpha = 0.20,
   ↪  linewidth = 0) +
  geom_line(linewidth = 0.9) +
  scale_y_log10(
    breaks = c(1, 10, 100, 1000),
    labels = label_number()
  ) +
  facet_wrap(~year, scales = "free_x") +
  labs(x = "Locations (ordered within trap group)",
       y = "Predicted no. mosquitoes (per trap-night)",
       color = "Trap group", fill = "Trap group") +
  theme_minimal()
```

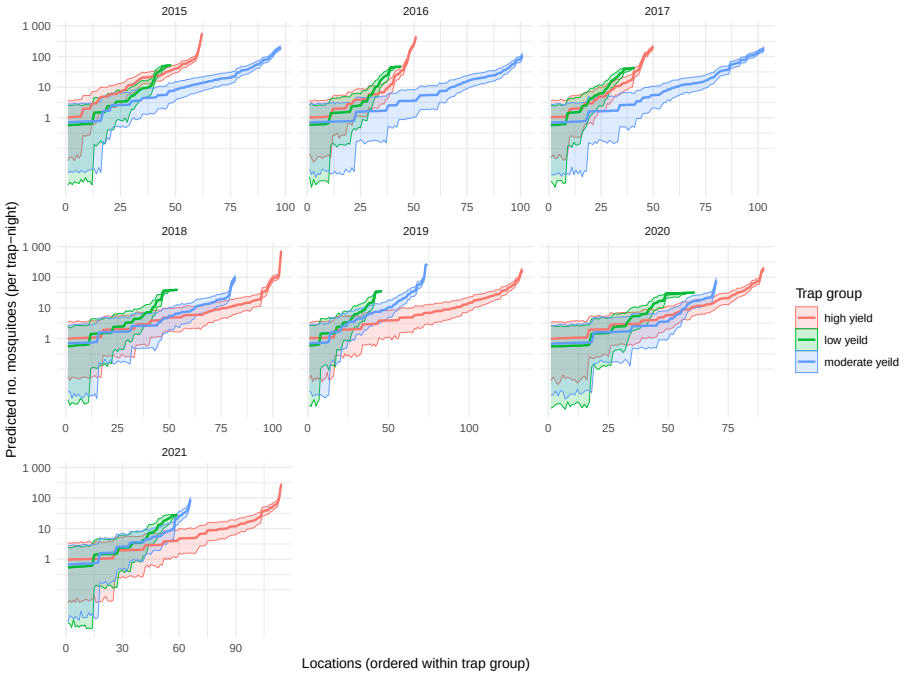



Figure 5.16: Predictive mean number of mosquitoes per trap-night for each location with 95% prediction intervals, stratified by year and trap group. Within each panel, locations are ordered by increasing predicted mean abundance. Shaded areas represent 95% prediction intervals for the number of mosquitoes at sampled locations. The y-axis is on a log scale, with tick labels shown on the original scale.

To understand whether we can trust the inferences shown in Figure 5.16, we use the AnPIT graphical check as used in the previous case studies. Since we are not using a geostatistical model in this case, we will use a simplified implementation of the AnPIT for the Poisson mixed model fitted in this section. The function that we use to compute the AnPIT is called `anpit_wnv` and has been implemented specifically for this case study. For more details on its implementation, read Section 5.4.2.

```
wnv_anpit_res <- anpit_wnv(glmer_wvn, test_prop = 0.30, nsim =
 ↪  10000)
```

In the code above the AnPIT is computed using a test-set corresponding to 30% of the original data-set. The number of simulation used to approximate the AnPIT is set to 1000. Finally, we plot the results.

```
plot_anpit(wnv_anpit_res)
```

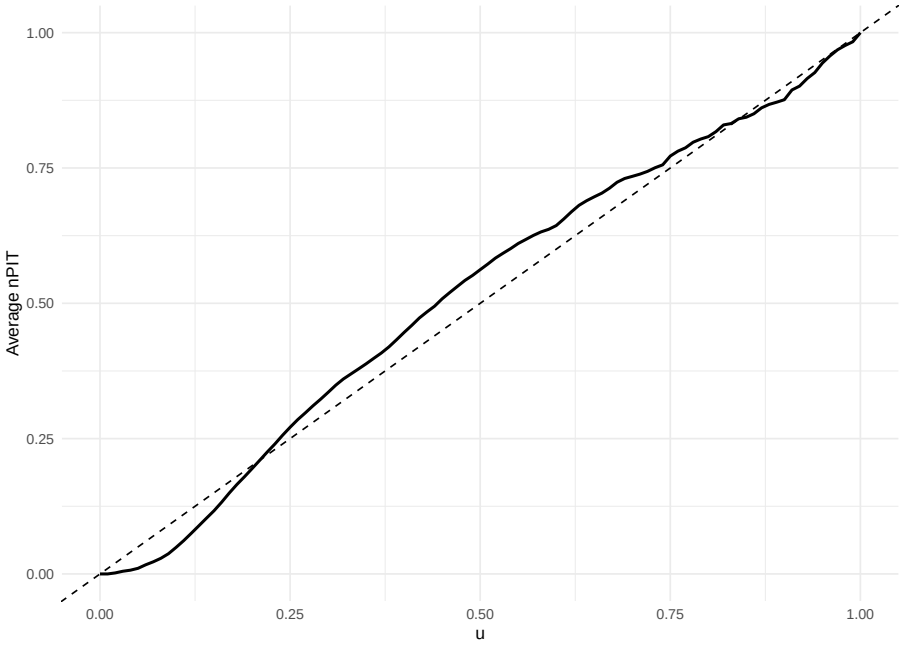



Figure 5.17: Average non-randomized probability integral transform (AnPIT) for the Poisson mixed model specified in Equation 5.4. The dahsed line corresponds to the identity line.

The results shown in Figure 5.17 an AnPIT curve from the fitted Poisson mixed model in Figure 5.17 that closely follows the identity line. Hence, we can conclude that we find no evidence against the compatibility of the model in Figure 5.17 with the data.

### 5.3.2 Modelling West Nile virus infection with pooled mosquito testing

Pooled testing is widely used in mosquito surveillance. Rather than testing each mosquito individually for a pathogen, multiple mosquitoes collected from the same location and week are combined into a single pool and assayed by PCR. The test returns positive if at least one mosquito in the pool is infected and negative otherwise. This approach greatly reduces laboratory time and costs, but statistical analysis must account for variation in pool size, as larger pools have a higher probability of testing positive even if the underlying infection risk per mosquito is low.

We illustrate this using West Nile virus (WNV) PCR results from pools of female *Culex pipiens* collected in the Sacramento Metropolitan Area, available in the `infect_sma` data set of the `RiskMap` package.

Let $k_i$ be the number of mosquitoes in pool $i$. Define the binary outcome $Y_i = 1$ if the PCR test for pool $i$ is positive and $Y_i = 0$ otherwise. Conditional on a stationary and isotropic spatial Gaussian process $S(x_i)$, the probability that pool $i$ is positive is

$$P(Y_i = 1 \mid S(x_i)) = 1 - [1 - p(x_i)]^{k_i}, \tag{5.6}$$

where $p(x_i)$ is the probability that an individual mosquito at location $x_i$ is infected with WNV. We model $p(x_i)$ using a logit-linear form

$$\log\left\{\frac{p(x_i)}{1 - p(x_i)}\right\} = \beta + S(x_i), \tag{5.7}$$

which explicitly incorporates the dependence of the pool positivity probability on pool size $k_i$.

We must emphasize that in the `infect_sma` data set the pool sizes $k_i$ are not taken directly from laboratory records but are estimated. For each pool, mosquito collections from nearby locations in the same epidemiological week and within a given spatial radius, for example 2 km, are identified. The total number of female mosquitoes from these nearby collections, denoted $T_{\text{near}}$, is divided by the number of pools formed from the same nearby collections in that week, denoted $m_{\text{near}}$, to obtain an estimate of pool size. The estimated pool size is therefore

$$\hat{k}_i = \text{round}\left(\max\{1, T_{\text{near}}/m_{\text{near}}\}\right),$$

and a default conservative value, such as 25, is used if no nearby collections are available.

The data comprise 596 responses, of which only 18 pools test positive. This very low prevalence constrains our modelling choices, for example by limiting the feasibility of incorporating temporal dynamics. As a result, the model in Equation 5.6 represents the simplest specification that can still capture spatial correlation, under the simplifying assumption that WNV risk remains relatively stable over the study period. Furthermore, because there is only a single binary outcome per location, overdispersion cannot occur (see box "Why Bernoulli extra-variation does not exist" in Section 5.1.1 of P. J. Diggle and Giorgi (2019)), and the empirical variogram provides limited value for assessing spatial correlation due to its high uncertainty in this context. We therefore proceed directly to fitting a geostatistical model and assess the estimate of the spatial correlation parameter, which quantifies the strength and range of spatial dependence in the data.

We then proceed in R as follows. We first load the data and convert them into an `sf` object with the appropriate UTM projection.

```
data(infect_sma)
```

```
infect_sma <- st_as_sf(infect_sma, coords = c("lon", "lat"), crs
 ↪  = 4326)
wnv_crs <- propose_utm(infect_sma)
infect_sma <- st_transform(infect_sma, wnv_crs)
```

We define the inverse link function based on Equation 5.6 and Equation 5.7.

```
invlink_wnv <- function(x)
 ↪  1-(1-exp(x)/(1+exp(x)))^infect_sma$est_pool_n
```

We then fit the model by passing the inverse link function to the argument `invlink` within the `glgpm` function.

```
inf_fit <-
  glgpm(wnv_pos ~ gp(lon,lat),
        crs = wnv_crs,
        family = "binomial",
        invlink = invlink_wnv,
        data=infect_sma)
```

Finally, we examine the the summary of the fitted model.

```
summary(inf_fit)
```

```
Call:
glgpm(formula = wnv_pos ~ gp(lon, lat), data = infect_sma,
family = "binomial",
    invlink = invlink_wnv, crs = wnv_crs, par0 = coef(inf_fit),
    start_pars = coef(inf_fit))

Binomial geostatistical linear model
Link: custom
Inverse link function = 1 - (1 - exp(x)/(1 +
exp(x)))^infect_sma$est_pool_n

'Lower limit' and 'Upper limit' are the limits of the 95%
confidence level intervals

Regression coefficients
            Estimate Lower limit Upper limit  StdErr z.value
            p.value
(Intercept)  -4.9648     -7.0190     -2.9105  1.0481 -4.7369
2.17e-06 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
Spatial Gaussian process
Matern covariance parameters (kappa=0.5)
                    Estimate Lower limit Upper limit
Spatial process var.   5.9851      1.5402      23.259
Spatial corr. scale   24.5370      9.1354      65.905
Variance of the nugget effect fixed at 0

Log-likelihood: 0.4061681
```

The fitted model indicates a substantial spatial residual component, with the estimated scale parameter corresponding to a practical range of approximately 70 km. This represents a long-range correlation relative to the spatial extent of the study area.

We first carry out predictions over a regular grid covering the area of the data collection efforts from 2015 to 2021. To achieve this, we create a regular grid 250 meters within the convex hull boundaries obtained by the full set of locations sampled from 2015 to 2021 (see Figure 5.18).

```
# Compute convex hull
geom_union <- st_union(inf_fit$data_sf)
chull_sf <- st_convex_hull(geom_union)
chull_sf <- st_as_sf(data.frame(geometry = st_sfc(chull_sf)),
                     crs = st_crs(inf_fit$data_sf))

# Plot points and convex hull
ggplot() +
  geom_sf(data = inf_fit$data_sf, color = "blue", size = 1) +
  geom_sf(data = chull_sf, fill = NA, color = "red", linewidth
   ↪  = 1) +
  theme_minimal()
```

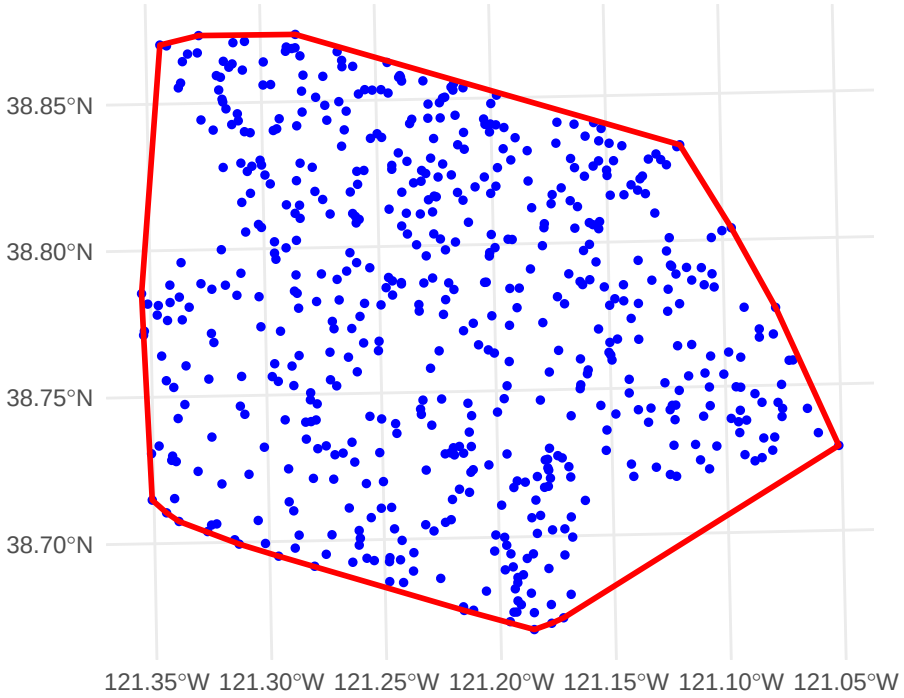



Figure 5.18: Sampling locations of pooled mosquito collections testing for West Nile virus in the Sacramento Metropolitan Area (blue points) and their convex hull (red outline) defining the minimal polygon enclosing all sites.

```
grid_pred_sac <- create_grid(chull_sf, spat_res = 0.25)

pred_S_inf <- pred_over_grid(inf_fit,
                             grid_pred = grid_pred_sac)

pred_inf_grid <- pred_target_grid(pred_S_inf,
                                  f_target = list(prev =
                                    ↪ function(x)
                                    ↪ exp(x)/(1+exp(x))))
```

Figure 5.19 displays the predictive mean of $p(x)$ over the regular grid. Overall, the predicted prevalence of WNV is very low across the study area, with a small zone in the south-west showing a relatively higher value of approximately 0.40%.

```
plot(pred_inf_grid,"prev")
```

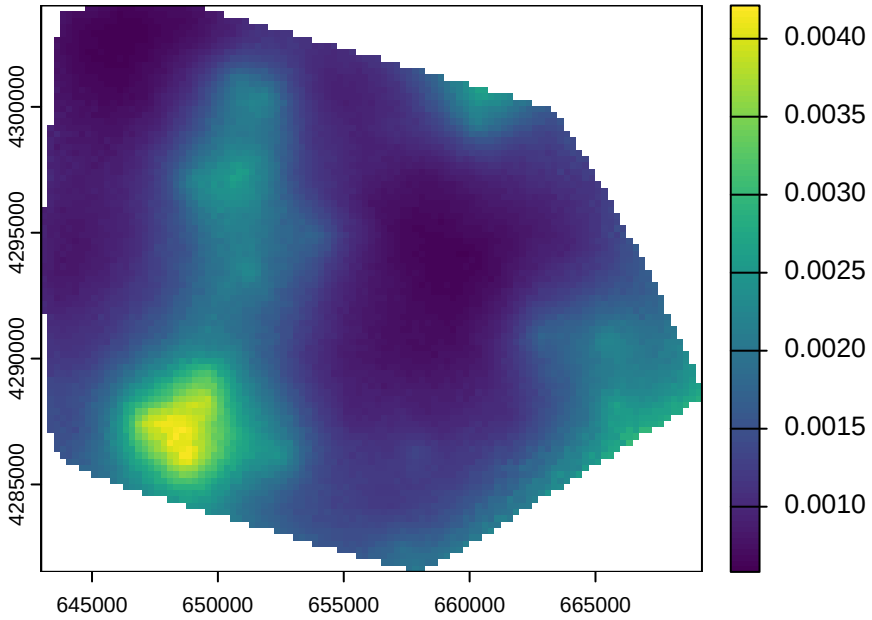



Figure 5.19: Map of the predictive mean for the prevalence of West Nile Virus infection among female *Culex pipiens* mosquitoes in an area of the Sacramento Metropolitan Area.

Finally, we predict the vector index as defined in Equation 5.3. Because *Cx. pipiens* abundance and WNV infection prevalence have been modelled as independent processes, the predictions from the two models can be combined directly. Specifically, for each location in the `abund_sma` data set, we take predictive samples of WNV prevalence and multiply them by the corresponding predictive samples of the mean number of female *Cx. pipiens* per trap-night. This yields predictive samples of the vector index, which can then be summarised or mapped as required. In R, this is implemented as follows.

```
# Converting the abund_sma into an sf object
loc_pred <- st_as_sfc(st_transform(st_as_sf(abund_sma, coords =
 ↪  c("lon","lat"), crs = 4326), crs = wnv_crs))

# Prediction of the sptial process S(x) over the locations of
 ↪  the
# abund_sma data-set
pred_S_loc <- pred_over_grid(inf_fit, grid_pred = loc_pred)

# Computation of the samples for West Nile Virus prevalence
beta_hat <- coef(inf_fit)$beta
prev_inf_samples <- 1/(1+exp(-(beta_hat +
```

```
                                  pred_S_loc$S_samples)))

# Predictive samples for the vector index (VI)
vi_samples <- mean_nmosq * prev_inf_samples

# Calculate predictive summaries for VI
vi_mean  <- apply(vi_samples, 1, mean, na.rm = TRUE)
vi_lower <- apply(vi_samples, 1, quantile, probs = 0.025, na.rm
 ↪  = TRUE)
vi_upper <- apply(vi_samples, 1, quantile, probs = 0.975, na.rm
 ↪  = TRUE)

# Build dataframe directly from original abund_sma order
vi_df <- data.frame(
  id    = 1:nrow(abund_sma),
  mean  = vi_mean,
  lower = vi_lower,
  upper = vi_upper,
  trap_group = abund_sma$trap_group,
  year       = abund_sma$year
)

# Order within (year, trap_group) by increasing mean
vi_df <- vi_df %>%
  group_by(year, trap_group) %>%
  arrange(mean, .by_group = TRUE) %>%
  mutate(x_order = dplyr::row_number()) %>%
  ungroup()
```

Several U.S. programs consider a vector index (VI) of 0.75 or higher as indicative of likely human transmission and use this threshold to trigger intensified control measures (City of Boulder 2025; City of Fort Collins 2025, 2014). In Figure 5.20, the dashed red line marks this operational threshold for elevated WNV risk. We observe that only in few instances before the year 2020 the predicted VI exceeded this threshold. We thus conclude that between 2015 and 2021 the risk for WNV based as measured by the VI was relatively low in the SMA.

```
ggplot(vi_df,
       aes(x = x_order, y = mean,
           color = trap_group, fill = trap_group, group =
            ↪  trap_group)) +
  geom_hline(yintercept = 0.75, linetype = "dashed", color =
   ↪  "red", linewidth = 0.8) +
  geom_ribbon(aes(ymin = lower, ymax = upper), alpha = 0.20,
   ↪  linewidth = 0) +
```

```
  geom_line(linewidth = 0.9) +
  scale_y_log10(
    breaks = c(0.01, 0.1, 1),
    labels = scales::label_number()
  ) +
  facet_wrap(~ year, scales = "free_x") +
  labs(
    x = "Locations (ordered within trap group)",
    y = "Vector index (infected mosquitoes per trap night)",
    title = "West Nile Virus",
    color = "Trap group", fill = "Trap group"
  ) +
  theme_minimal()
```

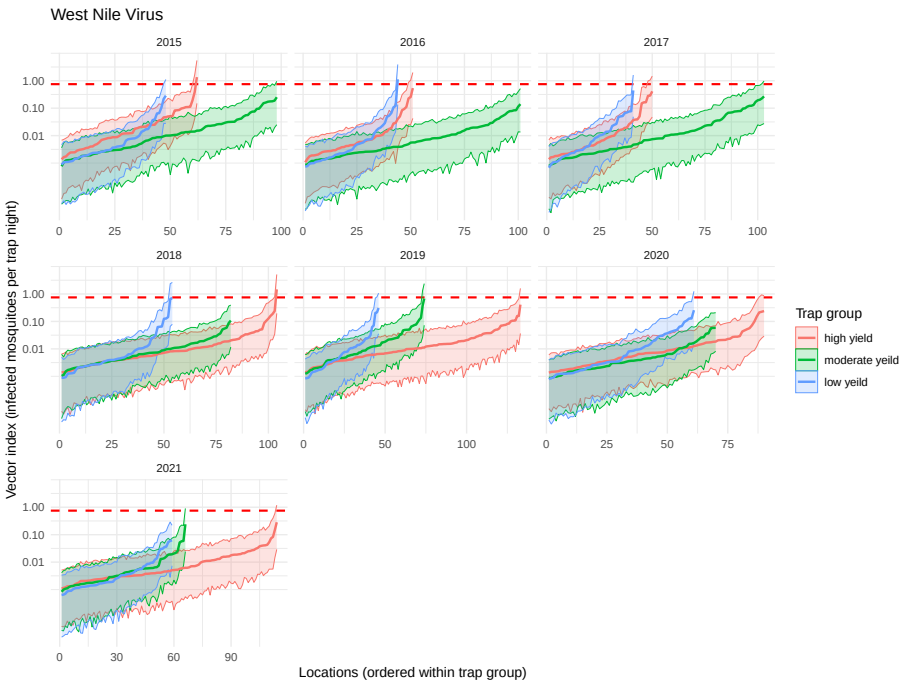



Figure 5.20: Predicted vector index (infected mosquitoes per trap night) by location, stratified by year and trap group. Locations within each panel are ordered by increasing predicted VI. Shaded bands represent 95% prediction intervals. The dashed red line marks the 0.75 threshold commonly used in U.S. West Nile virus surveillance to indicate elevated transmission risk. The y-axis is on a log scale with labels shown on the original scale.

### 5.3.3 Summary and conclusions

In this analysis, our predictive target was the vector index (VI) in Equation 5.3. Our proposed modelling approach was based on the assumption that abundance and infection prevalence are two independent processes. Treating prevalence as independent of vector density is a practical simplification. It can be reasonable when the fraction of infected mosquitoes at a place and time is driven mainly by host infection dynamics, temperature dependent incubation, and mosquito age structure, whereas trap counts reflect local production, as well as catchability. An interesting result of our analysis is that abundance showed no detectable residual spatial correlation, while infection prevalence did. The empirical variogram of random effects from the abundance model lay within the envelope in Figure 5.15, suggesting that remaining variation after accounting for year and trap group is dominated by micro scale heterogeneity and trap specific noise at distances smaller than those represented by the sampling design. In contrast, the infection model estimated a spatial scale of correlation that is large relative to the study area, and may be explained in terms of processes that operate over broader extents, such as movement and aggregation of avian reservoir hosts, shared urban landscape features, and temperature fields. However, as a result of the disparity in the scales of spatial dependence for the two processes and the inability of the model for abundance to interpolate at unobserved locations, we could only predict the VI at the observed locations of the abundance data.

All West Nile virus data analysed in this section originate from the `vectorsurvR` workflow and are simulated rather than observed in the field. The `vectorsurvR` package provides a framework for generating realistic mosquito surveillance data by mimicking the structure, spatial layout, and sampling frequency of actual surveillance programmes. Hence, the results presented here should not be used to draw scientific conclusions about West Nile virus in the Sacramento Metropolitan Area. In addition to this, pool size is a key input when analysing pooled infection outcomes (see Exercise 6 below) and strongly affects the predictive inferences in prevalence (see Exercise 6 below). In this case study we used an estimated pool size (denoted in the text above by $k_i$) rather than laboratory recorded counts, which adds additional uncertainty that we did not propagate. The results illustrate modelling steps, diagnostics, and how to combine components to obtain the vector index, which remains useful for pedagogical purposes.

## 5.4 Theory

In the following two sections we explain the coding of the functions used in Section 5.3.1. These functions are not part of any existing R package. They are

written for clarity rather than computational efficiency, and they are tailored to the current modelling example without attempts at generalisation. The aim is to present code that is correct and easy to follow, so that by understanding each component you can adapt and extend it to fit your own modelling requirements and learn more efficient implementations your own.

### 5.4.1 Simulating from the predictive of non-spatial mixed models

In this section, we provide a detailed explanation of the simulate_random_effects function used in this case study.

Simulating from the conditional distribution of $Z_i$ given the observed counts $y_i$ under a Poisson mixed model can be done in several ways in R, often more efficiently than the approach illustrated here. However, our goal is not to present the fastest implementation, but to offer a transparent, step-by-step example. This level of detail is useful for readers who wish to understand the underlying computations, or who may want to adapt and generalize the function for other analyses. From a pedagogical perspective, dissecting such a function provides a hands-on introduction to computational methods for generalized linear mixed models and illustrates ideas that are extended in more advanced Bayesian and likelihood-based approaches.

The function is reproduced below:

```
simulate_random_effects <- function(model, nsim = 1000,
 ↪  grid_range = c(-10, 10), grid_length = 5000) {
  # Extract fixed effects and RE variance
  beta_hat <- fixef(model)
  sigma2_hat <- as.numeric(VarCorr(model)[[1]][1])

  # Extract model frame and grouping
  mf <- model@frame
  group_var <- names(ranef(model))[1]
  group_ids <- mf[[group_var]]
  unique_groups <- unique(group_ids)
  ngroups <- length(unique_groups)

  # Design matrix for fixed effects
  D <- model.matrix(model)

  # Create result matrix
  u_samples <- matrix(NA, nrow = ngroups, ncol = nsim)
  rownames(u_samples) <- unique_groups

  # Log-predictive density function for one group
```

```
  log_pd_u <- function(u, y_j, D_j, beta_hat, sigma2_hat) {
    eta <- as.vector(D_j %*% beta_hat) + u
    loglik <- sum(dpois(y_j, lambda = exp(eta), log = TRUE))
    logprior <- dnorm(u, mean = 0, sd = sqrt(sigma2_hat), log =
↪   TRUE)
    return(loglik + logprior)
  }

  # For each group, compute predictive samples of u_j
  for (j in seq_along(unique_groups)) {
    gid <- unique_groups[j]
    idx <- which(group_ids == gid)
    y_j <- model@resp$y[idx]
    D_j <- D[idx, , drop = FALSE]

    # Create grid and compute log predictive density
    u_grid <- seq(grid_range[1], grid_range[2], length.out =
↪   grid_length)
    logdens <- sapply(u_grid, log_pd_u, y_j = y_j, D_j = D_j,
                      beta_hat = beta_hat, sigma2_hat =
                       ↪  sigma2_hat)
    logdens <- logdens - max(logdens)
    dens <- exp(logdens)
    pdf_u <- dens / sum(dens)
    cdf_u <- cumsum(pdf_u)
    cdf_u <- cdf_u / max(cdf_u)

    # Drop duplicated CDF values (necessary for approx)
    keep <- !duplicated(cdf_u)
    u_samples[j, ] <- approx(cdf_u[keep], u_grid[keep], xout =
↪   runif(nsim), rule = 2)$y

  }

  return(u_samples)
}
```

Below, we break down the function step by step.

First, we extract the fixed-effect estimates and the random effect variance from the fitted model:

```
beta_hat <- fixef(model)                          # Fixed effect
 ↪  coefficients (MLEs)
sigma2_hat <- as.numeric(VarCorr(model)[[1]][1])# Estimated
 ↪  variance of random intercepts
```

These parameters will be treated as known when simulating the predictive of the random effects. Next, we identify the grouping variable for the random effects and compute the fixed-effects design matrix:

```
mf <- model@frame                          # Model frame containing
 ↪  the data
group_var <- names(ranef(model))[1]        # Name of the grouping
 ↪  variable
group_ids <- mf[[group_var]]               # Group ID for each
 ↪  observation
unique_groups <- unique(group_ids)         # Unique group levels
ngroups <- length(unique_groups)           # Number of groups

D <- model.matrix(model)                   # Fixed-effects design
 ↪  matrix
```

Here, `group_ids` associates each observation with its random effect, and `D` is the design matrix used to compute the linear predictor for the fixed effects.

We initialize a results matrix to store the predictive samples for each group:

```
u_samples <- matrix(NA, nrow = ngroups, ncol = nsim)
rownames(u_samples) <- unique_groups
```

Each row corresponds to a group and each column to a predictive sample of $u_j$.

We then define a helper function to compute the unnormalized log-predictive density of a single random effect $u_j$ given the data for that group:

```
log_pd_u <- function(u, y_j, D_j, beta_hat, sigma2_hat) {
  eta <- as.vector(D_j %*% beta_hat) + u      # Linear
 ↪  predictor
  loglik <- sum(dpois(y_j, lambda = exp(eta), log = TRUE))   #
 ↪  Poisson likelihood
  logprior <- dnorm(u, mean = 0, sd = sqrt(sigma2_hat), log =
 ↪  TRUE) # Normal prior
  return(loglik + logprior)                  # Unnormalized
   ↪  log-posterior
}
```

This combines the Poisson likelihood for the observations in group $j$ with the Gaussian prior on the random effect.

For each group, we first compute the predictive on a grid of $u$ values, then normalize it to form a cumulative distribution function (CDF), and finally

draw samples using inverse CDF sampling.

```
for (j in seq_along(unique_groups)) {
  gid <- unique_groups[j]
  idx <- which(group_ids == gid)
  y_j <- model@resp$y[idx]
  D_j <- D[idx, , drop = FALSE]

  # Step 1: Compute predictive density on a grid
  u_grid <- seq(grid_range[1], grid_range[2], length.out =
↪  grid_length)
  logdens <- sapply(u_grid, log_pd_u, y_j = y_j, D_j = D_j,
                    beta_hat = beta_hat, sigma2_hat =
                     ↪  sigma2_hat)
  logdens <- logdens - max(logdens)    # Stabilize
↪  exponentiation
  dens <- exp(logdens)
  pdf_u <- dens / sum(dens)

  # Step 2: Compute cumulative distribution
  cdf_u <- cumsum(pdf_u) / sum(pdf_u)

  # Step 3: Sample from predictive using inverse CDF
  keep <- !duplicated(cdf_u)  # Remove duplicates to avoid
↪  interpolation warnings
  u_samples[j, ] <- approx(cdf_u[keep], u_grid[keep], xout =
↪  runif(nsim), rule = 2)$y
}
```

The function uses a grid-based approximation, evaluating the predictive on a fine grid to avoid running MCMC for these one-dimensional distributions. The computation `logdens - max(logdens)` provides numerical stabilization, preventing underflow when exponentiating very small log-likelihood values. Finally, the function `approx()` performs inverse transform sampling. This works by first computing the CDF of the random effect on a fine grid, which maps each candidate value of the random effect to its cumulative probability. We then draw random numbers from a uniform distribution on $[0, 1]$ and use `approx()` to interpolate the CDF to find the corresponding values of the random effect. In other words, we are transforming uniform draws into samples from the desired predictive distribution by inverting the CDF numerically, a method commonly known as the inverse transform method; for a thorough explanation of this approach, Robert and Casella (2004) is a highly recommended reading.

As a self-directed learning activity for readers interested in the computational aspects of model fitting, we invite them to work through Exercise 4 at the end

of this chapter.

### 5.4.2 The non-randomized probability integral transform for non-spatial models

The function for generating and plotting the average non-randomized probability transform (AnPIT) used in Section 5.3.1 are given below.

```
anpit_wnv <- function(model, test_prop = 0.25, nsim = 2000,
                      u_grid = seq(0, 1, length.out = 101), seed
                       ↪  = NULL) {
  stopifnot(inherits(model, "glmerMod"))
  if (!is.null(seed)) set.seed(seed)
  if (test_prop <= 0 || test_prop >= 1) stop("test_prop must be
   ↪  in (0,1)")

  # Extract pieces from the fitted model
  D   <- model.matrix(model)                  # fixed-effects
↪  design as used at fit
  off <- model@frame$`offset(log(trap_nights))`
  y   <- model@resp$y
  beta <- lme4::fixef(model)
  sigma <- sqrt(as.numeric(lme4::VarCorr(model)[[1]][1]))

  n <- nrow(D)
  test_idx  <- sort(sample(seq_len(n), size = ceiling(n *
↪  test_prop)))
  train_idx <- setdiff(seq_len(n), test_idx)

  # Linear predictor without REs
  eta0 <- as.numeric(D %*% beta) + off

  # For these data loc is unique per row; for test rows we
   ↪  integrate over prior u ~ N(0, sigma^2)
  # Draw nsim random-effect values once and reuse across test
   ↪  rows (common random numbers)
  u_draws <- rnorm(nsim, mean = 0, sd = sigma)

  y_test <- y[test_idx]
  eta0_test <- eta0[test_idx]
  n_test <- length(test_idx)

  # Posterior predictive CDFs at y and y-1 via Monte Carlo over
   ↪  u
  Fi_y      <- numeric(n_test)
```

```
  Fi_yminus <- numeric(n_test)
  y_minus <- pmax(y_test - 1L, -1L)

  for (i in seq_len(n_test)) {
    lam <- exp(eta0_test[i] + u_draws)
    Fi_y[i] <- mean(ppois(y_test[i], lambda = lam))
    Fi_yminus[i] <- if (y_minus[i] < 0) 0 else
↪   mean(ppois(y_minus[i], lambda = lam))
    if (Fi_y[i] < Fi_yminus[i]) { tmp <- Fi_y[i]; Fi_y[i] <-
     ↪  Fi_yminus[i]; Fi_yminus[i] <- tmp }
  }

  # Nonrandomized PIT for each u in u_grid and average across
   ↪  test observations
  npit_avg <- function(u, Fy, Fym1) {
    denom <- pmax(Fy - Fym1, .Machine$double.eps)
    lt <- u <= Fym1; gt <- u >= Fy; mid <- !(lt | gt)
    out <- numeric(length(Fy))
    out[lt] <- 0; out[gt] <- 1; out[mid] <- (u - Fym1[mid]) /
↪   denom[mid]
    mean(out)
  }
  anpit <- vapply(u_grid, npit_avg, FUN.VALUE = 0.0, Fy = Fi_y,
↪   Fym1 = Fi_yminus)

  list(u = u_grid,
       anpit = as.numeric(anpit),
       Fi_y = Fi_y,
       Fi_yminus = Fi_yminus,
       test_index = test_idx,
       train_index = train_idx)
}

plot_anpit <- function(res) {
  if (!requireNamespace("ggplot2", quietly = TRUE)) stop("Need
   ↪  ggplot2 for plotting.")
  df <- data.frame(u = res$u, anpit = res$anpit)
  ggplot(df, aes(u, anpit)) +
    geom_line(linewidth = 0.9) +
    geom_abline(slope = 1, intercept = 0, linetype = 2) +
    labs(x = "u", y = "Average nPIT", title = "") +
    theme_minimal()
}
```

We start by defining the function and adding input checks.

```
anpit_wnv <- function(model, test_prop = 0.25, nsim = 2000,
                      u_grid = seq(0, 1, length.out = 101), seed
                       ↪  = NULL) {
  stopifnot(inherits(model, "glmerMod"))
  if (!is.null(seed)) set.seed(seed)
  if (test_prop <= 0 || test_prop >= 1) stop("test_prop must be
   ↪  in (0,1)")
```

The function accepts a fitted Poisson mixed model, a proportion for the test set, number of Monte Carlo draws, a grid of u values (used as input of the AnPIT), and an optional seed for reproducibility. It checks that the model is of class `glmerMod` and that the test proportion is valid.

```
  D    <- model.matrix(model)
  off  <- model@offset; if (is.null(off)) off <- rep(0, nrow(D))
  y    <- model@resp$y
  beta <- lme4::fixef(model)
  sigma <- sqrt(as.numeric(lme4::VarCorr(model)[[1]][1]))
```

This block extracts the design matrix, the offset (log of trap nights in our case), the response counts, the fixed effects, and the standard deviation of the random intercept.

```
  n <- nrow(D)
  test_idx  <- sort(sample(seq_len(n), size = ceiling(n *
 ↪  test_prop)))
  train_idx <- setdiff(seq_len(n), test_idx)
```

Here we randomly split the data into a test and training set according to `test_prop`.

```
  eta0 <- as.numeric(D %*% beta) + off
  u_draws <- rnorm(nsim, mean = 0, sd = sigma)
```

We compute the linear predictor without random effects and generate random intercept samples from the marginal distribution of the random effects $Z_i$, namely a Gaussian distribution with mean 0 and standard deviation `sigma`. Since each location is unique, we integrate over the prior for the test rows.

```
  y_test    <- y[test_idx]
  eta0_test <- eta0[test_idx]
  n_test    <- length(test_idx)

  Fi_y      <- numeric(n_test)
  Fi_yminus <- numeric(n_test)
```

```
  y_minus   <- pmax(y_test - 1L, -1L)
```

The response, linear predictor, and number of test observations are stored. We also prepare storage for the predictive CDFs.

```
  for (i in seq_len(n_test)) {
    lam <- exp(eta0_test[i] + u_draws)
    Fi_y[i]      <- mean(ppois(y_test[i],    lambda = lam))
    Fi_yminus[i] <- if (y_minus[i] < 0) 0 else
↪   mean(ppois(y_minus[i], lambda = lam))
    if (Fi_y[i] < Fi_yminus[i]) { tmp <- Fi_y[i]; Fi_y[i] <-
     ↪  Fi_yminus[i]; Fi_yminus[i] <- tmp }
  }
```

For each test observation, we compute the predictive CDF at $y$ and $y-1$ via Monte Carlo integration over the u draws. These quantities are then used to compute the AnPIT below.

```
  npit_avg <- function(u, Fy, Fym1) {
    denom <- pmax(Fy - Fym1, .Machine$double.eps)
    lt <- u <= Fym1; gt <- u >= Fy; mid <- !(lt | gt)
    out <- numeric(length(Fy))
    out[lt] <- 0; out[gt] <- 1; out[mid] <- (u - Fym1[mid]) /
↪   denom[mid]
    mean(out)
  }
  anpit <- vapply(u_grid, npit_avg, FUN.VALUE = 0.0, Fy = Fi_y,
↪   Fym1 = Fi_yminus)
```

We define a helper to compute the AnPIT for a given u and average it over the test set (see Equation 4.6 for the definition of the AnPIT). This is repeated for each u in the grid.

```
  list(u = u_grid,
       anpit = as.numeric(anpit),
       Fi_y = Fi_y,
       Fi_yminus = Fi_yminus,
       test_index = test_idx,
       train_index = train_idx)
}
```

The function returns a list with the u grid, the averaged nPIT values, the predictive CDFs, and the indices of the test and training sets.

```
plot_anpit <- function(res) {
  if (!requireNamespace("ggplot2", quietly = TRUE)) stop("Need
   ↪  ggplot2 for plotting.")
```

```
  df <- data.frame(u = res$u, anpit = res$anpit)
  ggplot(df, aes(u, anpit)) +
    geom_line(linewidth = 0.9) +
    geom_abline(slope = 1, intercept = 0, linetype = 2) +
    labs(x = "u", y = "Average nPIT", title = "") +
    theme_minimal()
}
```

This helper function takes the output and produces a plot of the averaged nPIT curve against the identity line.

## 5.5 Exercises

1. Re-analyse the Ghana malnutrition data set (see Section 5.1) after dichotomising the height-for-age Z-score (HAZ) and weight-for-age Z-score (WAZ). Define two binary outcomes:

   - `HAZ_bin` $= 1$ if HAZ $< -2$, and 0 otherwise
   - `WAZ_bin` $= 1$ if WAZ $< -2$, and 0 otherwise

   Fit binomial geostatistical models to the individual-level data and compare the point estimates and spatial predictions to those obtained from the geostatistical models described in Section 5.1. Discuss which model provides a better fit and why.

2. In the Malawi prevalence mapping analysis, we used the first principal component (PC1) of the environmental covariates to summarize covariate variation and reduce dimensionality. In this exercise, you will investigate whether including the second principal component (PC2) improves the model.

   - Compute the second principal component (PC2) from the same set of standardized covariates used to derive PC1. What is the interpretation of PC2? What kind of relationship does PC2 show with the empirical logit?
   - Fit a geostatistical model that includes both PC1 and PC2 as covariates. How do the covariance parameters of the spatial process change after including PC2 compared to the model with PC1 only?
   - Compare this model to:

     –the model using only PC1, and

–the model using all covariates as separate predictors.

Use the AnPIT plot and CRPS/SCRPS scores to assess predictive performance.

- Discuss whether adding PC2 captures additional meaningful variation or introduces multicollinearity or overfitting. What implications does this have for model interpretability and parsimony?

3. In this exercise, you will repeat the full analysis pipeline developed for Malawi using data from a different country.

   - Choose a different country available in the `malariaAtlas` package (e.g., Zambia or Nigeria), and download parasite prevalence survey data for a specific year.
   - Obtain the same set of environmental covariates used in the Malawi analysis, and prepare them for analysis (e.g., resample, standardize, and extract values at survey locations).
   - Fit and compare two geostatistical models:
     - –A model using the environmental covariates as separate fixed effects.
     - –A model using the first principal component (PC1) as a synthetic covariate summarizing the covariate space.
   - Evaluate model performance using the AnPIT plot and CRPS/SCRPS scores.
   - Reflect on which model performs better in this new setting. Do your conclusions align with what was observed in the Malawi case study? What might explain any differences in performance across countries?

4. The function `simulate_random_effects()` in Section 5.4.1 samples random effects using a simple grid-based inverse CDF approach.
   While easy to understand, this method can be slow when the number of groups or simulations is large. Rewrite the function using a more efficient sampling strategy such as slice sampling (Neal 2003) or a simple Metropolis–Hastings sampler (Gelman et al. 2013). After implementing your new function, compare the sampled random effect distributions to those obtained from `simulate_random_effects` and discuss any differences in accuracy and computational speed.

5. The function `anpit_wnv()` computes the average non-randomized probability integral transform for the specific model in Equation 5.4.

Full details of the implementation are given in Section 5.4.2. You may wish to enhance this function in one of two ways: first, by generalising it so that it can be applied to any generalized linear mixed model with the same random effects structure; second, by replacing the Monte Carlo step that samples from a Gaussian distribution with a numerical integration approach. Once you have implemented this function, you can apply it to the analysed data-sets of the previous chapter and compare the generated AnPIT against those of the fitted geostatistical models. Does the AnPIT show an unsatisfactory diagnostic in some cases?

6. Repeat the analysis in Section 5.3 from the beginning to predict the vector index, but this time assume alternative fixed values for the PCR pool size $k_i$ in Section 5.3.2, setting all $k_i$ to 10, 25, and 50 in turn. Compare the resulting inferences on West Nile virus infection prevalence and the vector index across these three scenarios, and discuss how they differ from each other as well as from the results presented earlier in this chapter.

# References


Amazigo, U. 2008. "The African Programme for Onchocerciasis Control (APOC)." *Ann Trop Med Parasitol* 102 (Suppl 1): 19–22. https://doi.org/10.1179/136485908X337436.

Bates, Douglas, Martin Mächler, Ben Bolker, and Steve Walker. 2015. "Fitting Linear Mixed-Effects Models Using lme4." *Journal of Statistical Software* 67 (1): 1–48. https://doi.org/10.18637/jss.v067.i01.

Bhattachan, Abhinav, Rachel Barrios, Laura Harrington, Valerie A. Paz-Soldan, and A. Marm Kilpatrick. 2023. "Drought-Associated Reductions in Urban Mosquito Abundance During California's Historic Drought." *Science of The Total Environment* 891: 164519. https://doi.org/10.1016/j.scitotenv.2023.164519.

Bishop, Christopher M. 2006. *Pattern Recognition and Machine Learning*. Springer.

Bolin, David, and Jonas Wallin. 2023. "Local scale invariance and robustness of proper scoring rules." *Statistical Science* 38 (1): 140–59. https://doi.org/10.1214/22-STS864.

Bolling, Bethany G., Christopher M. Barker, Chester G. Moore, W. John Pape, and Lars Eisen. 2009. "Seasonal Patterns for Entomological Measures of Risk for Exposure to *Culex* Vectors and West Nile Virus in Relation to Human Disease Cases in Northeastern Colorado." *Journal of Medical Entomology* 46 (6): 1519–31. https://doi.org/10.1603/033.046.0641.

Bowman, A. W. 1997. *Applied Smoothing Techniques for Data Analysis : The Kernel Approach with s-Plus Illustrations*. Oxford Statistical Science Series ; 18. Oxford : New York: Clarendon Press ; Oxford University Press.

Breslow, N. E., and D. G. Clayton. 1993. "Approximate Inference in Generalized Linear Mixed Models." *Journal of the American Statistical Association* 88: 9–25.

Brooks, Steve, Andrew Gelman, Galin L. Jones, and Xiao-Li Meng. 2011. *Handbook of Markov Chain Monte Carlo*. CRC Press.

California Department of Public Health. 2025. "California Mosquito-Borne Virus Surveillance and Response Plan." Online.

Centers for Disease Control and Prevention. 2024. "West Nile Virus Surveillance and Control Guidelines." CDC guidance. https://www.cdc.gov/west-nile-virus/php/surveillance-and-control-guidelines/index.html.

Chilès, J-P, and P. Delfiner. 2016. *Geostatistics (Second Edition)*. Hoboken: Wiley.

Christensen, OF, GO Roberts, and M Sköld. 2006. "Robust Markov Chain Monte Carlo Methods for Spatial Generalized Linear Mixed Models." *Journal of Computational and Graphical Statistics* 15 (1): 1–17.

Christensen, Ole F. 2004. "Monte Carlo Maximum Likelihood in Model-Based Geostatistics." *Journal of Computational and Graphical Statistics* 13 (3): 702–18.

City of Boulder. 2025. "West Nile Virus — Estimating Risk to People: The Vector Index." https://bouldercolorado.gov/west-nile-virus.

City of Fort Collins. 2014. "West Nile Virus Program Manual." City of Fort Collins. https://www.fcgov.com/westnile/pdf/wnv_program_manual.pdf.

———. 2025. "Local Data — West Nile Virus." https://www.fcgov.com/westnile/local-data.

Cowles, Mary Kathryn, and Bradley P. Carlin. 1996. "Markov Chain Monte Carlo Convergence Diagnostics: A Comparative Review." *Journal of the American Statistical Association* 91 (434): 883–904.

Cressie, N. A. C. 1991. *Statistics for Spatial Data*. New York: Wiley.

Cressie, Noel. 1985. "Fitting Variogram Models by Weighted Least Squares." *Mathematical Geology* 17 (5): 563–86.

Czado, Claudia, Tilmann Gneiting, and Leonhard Held. 2009. "Predictive Model Assessment for Count Data." *Biometrics* 65 (4): 1254–61. https://doi.org/10.1111/j.1541-0420.2009.01191.x.

Dawid, A. P. 1984. "Statistical Theory: The Prequential Approach." *Journal of the Royal Statistical Society: Series A (General)* 147 (2): 278–92.

Diggle, P J, and E Giorgi. 2019. *Model-Based Geostatistics for Global Public Health : Methods and Applications.* Chapman and Hall/CRC Interdisciplinary Statistics Ser. Milton: Chapman; Hall/CRC.

Diggle, P. J., J. A. Tawn, and R. A. Moyeed. 1998. "Model-Based Geostatistics." *Journal of the Royal Statistical Society: Series C (Applied Statistics)* 47 (3): 299–350. https://doi.org/10.1111/1467-9876.00113.

Diggle, Peter, and Paulo Justiniano Ribeiro. 2007. *Model-Based Geostatistics*. Springer Series in Statistics. Springer.

Dobson, A. J., and A. Barnett. 2008. *An Introduction to Generalized Linear Models*. Third. Chapman; Hall/CRC.

Efron, Bradley. 1979. "Bootstrap Methods: Another Look at the Jackknife." *The Annals of Statistics* 7 (1): 1–26.

Evans, Jeffrey S., and Melanie A. Murphy. 2021. *spatialEco*. https://github.com/jeffreyevans/spatialEco.

Fernández, J. A, A Rey, and A Carballeira. 2000. "An Extended Study of Heavy Metal Deposition in Galicia (NW Spain) Based on Moss Analysis." *Science of The Total Environment* 254 (1): 31–44. https://doi.org/10.1016/S0048-9697(00)00431-9.

Gelman, Andrew, John B. Carlin, Hal S. Stern, David B. Dunson, Aki Vehtari, and Donald B. Rubin. 2013. *Bayesian Data Analysis*. 3rd ed. CRC Press.

Gelman, Andrew, and Donald B Rubin. 1992. "Inference from Iterative Sim-

ulation Using Multiple Sequences." *Statistical Science* 7 (4): 457–72.

Geweke, John. 1992. "Evaluating the Accuracy of Sampling-based Approaches to the Calculation of Posterior Moments." Edited by Jose M. Bernardo, James O. Berger, A. Philip Dawid, and Adrian F. M. Smith, 169–93.

Geyer, Charles J. 1991. "Markov Chain Monte Carlo Maximum Likelihood." *Journal of Computational and Graphical Statistics* 1 (4): 39–55.

Geyer, Charles J. 1994. "Likelihood and Exponential Families." *Department of Statistics, University of Minnesota*.

———. 1996. "Markov Chain Monte Carlo Maximum Likelihood." *Department of Statistics, University of Minnesota*.

———. 2019. "Monte Carlo Methods in MCMC." In *Handbook of MCMC*, edited by Steve Brooks, Andrew Gelman, Galin L. Jones, and Xiao-Li Meng, 3–48. CRC Press.

Ghana Statistical Service, Ghana Health Service, and ICF International. 2015. *Ghana Demographic and Health Survey 2014*. Rockville, Maryland, USA: Ghana Statistical Service, Ghana Health Service,; ICF International. https://dhsprogram.com/publications/publication-FR307-DHS-Final-Reports.cfm.

Gneiting, Tilmann, Fadoua Balabdaoui, and Adrian E Raftery. 2007. "Probabilistic Forecasts, Calibration and Sharpness." *Journal of the Royal Statistical Society: Series B (Statistical Methodology)* 69 (2): 243–68.

Gneiting, Tilmann, and Adrian E Raftery. 2007. "Strictly Proper Scoring Rules, Prediction, and Estimation." *Journal of the American Statistical Association* 102 (477): 359–78. https://doi.org/10.1198/016214506000001437.

Harrell, Frank E. 2015. *Regression Modeling Strategies*. 2nd ed. New York: Springer.

Hastie, Trevor, Robert Tibshirani, and Jerome Friedman. 2001. *The Elements of Statistical Learning*. Springer Series in Statistics. New York, NY, USA: Springer New York Inc.

Hastings, W. K. 1970. "Monte Carlo Sampling Methods Using Markov Chains and Their Applications." *Biometrika* 57 (1): 97–109.

Johnson, Olatunji, Claudio Fronterre, Benjamin Amoah, Antonio Montresor, Emanuele Giorgi, Nicholas Midzi, Masceline Jenipher Mutsaka-Makuvaza, et al. 2021. "Model-Based Geostatistical Methods Enable Efficient Design and Analysis of Prevalence Surveys for Soil-Transmitted Helminth Infection and Other Neglected Tropical Diseases." *Clinical Infectious Diseases* 72 (Supplement_3): S172–79. https://doi.org/10.1093/cid/ciab192.

Jones, Roderick C., Kingsley N. Weaver, Shamika Smith, Claudia Blanco, Cristina Flores, Kevin Gibbs, Daniel Markowski, and John-Paul Mutebi. 2011. "Use of the Vector Index and Geographic Information System to Prospectively Inform West Nile Virus Interventions." *Journal of the American Mosquito Control Association* 27 (3): 315–19. https://doi.org/10.2987/10-6098.1.

Katz, Elizabeth, and Bill & Melinda Gates Foundation. 2020. "Gender and

Malaria Evidence Reivew." Bill & Melinda Gates Foundation. https://www.gatesgenderequalitytoolbox.org/wp-content/uploads/BMGF_Malaria-Review_FC.pdf.

Kilpatrick, A. Marm, and W. John Pape. 2013. "Predicting Human West Nile Virus Infections with Mosquito Surveillance Data." *American Journal of Epidemiology* 178 (5): 829–35. https://doi.org/10.1093/aje/kwt046.

Kramer, Laura D., Jun Li, and Pei-Yong Shi. 2007. "West Nile Virus." *The Lancet Neurology* 6 (2): 171–81. https://doi.org/10.1016/S1474-4422(07)70030-3.

Krige, D. G. 1951. "A Statistical Approach to Some Basic Mine Valuation Problems on the Witwatersrand." *Journal of the Chemical, Metallurgical and Mining Society of South Africa* 52: 119–39.

Kyomuhangi, Irene, Tarekegn A. Abeku, Matthew J. Kirby, Gezahegn Tesfaye, and Emanuele Giorgi. 2021. "Understanding the Effects of Dichotomization of Continuous Outcomes on Geostatistical Inference." *Spatial Statistics* 42: 100424. https://doi.org/https://doi.org/10.1016/j.spasta.2020.100424.

Lovelace, Robin, Jakub Nowosad, and Jannes Muenchow. 2020. *Geocomputation with R*. London, England: CRC Press. https://r.geocompx.org/.

Lucas, Tim C. D., Anita K. Nandi, Rosalind E. Howes, Daniel J. Weiss, Ewan Cameron, Nick Golding, and Peter W. Gething. 2020. "malariaAtlas: An r Interface to Global Malariometric Data Hosted by the Malaria Atlas Project." *Wellcome Open Research* 5: 74. https://doi.org/10.12688/wellcomeopenres.15987.1.

Mahoney, Michael J, Lucas K Johnson, Julia Silge, Hannah Frick, Max Kuhn, and Colin M Beier. 2023. "Assessing the Performance of Spatial Cross-Validation Approaches for Models of Spatially Structured Data." https://doi.org/10.48550/arXiv.2303.07334.

Matern, B. 2013. *Spatial Variation*. Lecture Notes in Statistics. Springer New York. https://books.google.co.uk/books?id=HrbSBwAAQBAJ.

Matheron, G. 1963. "Principles of Geostatistics." *Economic Geology* 58: 1246–66.

Metropolis, Nicholas, Arianna W. Rosenbluth, Marshall N. Rosenbluth, Augusta H. Teller, and Edward Teller. 1953. "Equation of State Calculations by Fast Computing Machines." *The Journal of Chemical Physics* 21 (6): 1087–92.

Neal, Radford M. 2003. "Slice Sampling." *The Annals of Statistics* 31 (3): 705–67.

Nelder, J. A., and R. W. M. Wedderburn. 1972. "Generalized Linear Models." *Journal of the Royal Statistical Society A* 135: 370–84.

Organization, World Health. 2024. "Indicator Metadata Registry: Child Malnutrition—Underweight Among Children Under Five Years of Age (Weight-for-Age <-2 SD)." https://www.who.int/data/gho/indicator-metadata-registry/imr-details/27.

Pawitan, Yudi. 2001. *In All Likelihood : Statistical Modelling and Inference Using Likelihood*. Oxford ; New York: Clarendon Press : Oxford University

Press.

Petersen, Lyle R., Aaron C. Brault, and Roger S. Nasci. 2013. "West Nile Virus: Review of the Literature." *JAMA* 310 (3): 308–15. https://doi.org/10.1001/jama.2013.8042.

Puranik, Amitha, Peter J. Diggle, Maurice R. Odiere, Katherine Gass, Stella Kepha, Collins Okoyo, Charles Mwandawiro, et al. 2024. "Understanding the Impact of Covariates on the Classification of Implementation Units for Soil-Transmitted Helminths Control: A Case Study from Kenya." *BMC Medical Research Methodology* 24 (1): 294. https://doi.org/10.1186/s12874-024-02420-1.

Ripley, B. D. 1981. *Spatial Statistics*. New York: Wiley.

Robert, Christian P, and George Casella. 2004. *Monte Carlo Statistical Methods*. 2nd ed. Springer.

Roberts, Gareth O., and Richard L. Tweedie. 1996. "Exponential Convergence of Langevin Distributions and Their Discrete Approximations." *Bernoulli* 2 (4): 341–63.

Ross, Sheldon. 2013. *First Course in Probability, a*. 9th ed. Harlow: Pearson Education UK.

Rossky, Peter J., J. D. Doll, and Harold L. Friedman. 1978. "Brownian Dynamics as Smart Monte Carlo Simulation." *The Journal of Chemical Physics* 69 (10): 4628–33.

Rue, H., S. Martino, and N. Chopin. 2009. "Approximate Bayesian Inference for Latent Gaussian Models by Using Integrated Nested Laplace Approximations." *Journal of the Royal Statistical Society: Series B (Statistical Methodology)* 71 (2): 319–92. https://doi.org/10.1111/j.1467-9868.2008.00700.x.

Sacramento-Yolo Mosquito and Vector Control District. 2018. "2018 Annual Report." Online.

Shmueli, Galit. 2010. "To Explain or to Predict?" *Statistical Science* 25 (3): 289–310.

Smith, David L, Carlos A Guerra, Robert W Snow, and Simon I Hay. 2007. "Standardizing Estimates of the Plasmodium Falciparum Parasite Rate." *Malaria Journal* 6 (1): 131–31.

Stein, Michael L. 1999. *Interpolation of Spatial Data Some Theory for Kriging*. 1st ed. 1999. Springer Series in Statistics. New York, NY: Springer New York : Imprint: Springer.

Stevenson, Gillian H. AND Gitonga, Jennifer C. AND Stresman. 2013. "Reliability of School Surveys in Estimating Geographic Variation in Malaria Transmission in the Western Kenyan Highlands." *PLOS ONE* 8 (10). https://doi.org/10.1371/journal.pone.0077641.

Tene Fossog, Billy, Diego Ayala, Pelayo Acevedo, Pierre Kengne, Ignacio Ngomo Abeso Mebuy, Boris Makanga, Julie Magnus, et al. 2015. "Habitat Segregation and Ecological Character Displacement in Cryptic African Malaria Mosquitoes." *Evolutionary Applications* 8 (4): 326–45. https://doi.org/10.1111/eva.12242.

Tobler, W. R. 1970. "A Computer Movie Simulating Urban Growth in the Detroit Region." *Economic Geography* 46: 234–40.

United Nations Statistics Division, World Health Organization, and UNICEF. 2025. "SDG Indicator 2.2.1 Metadata: Prevalence of Stunting (Height-for-Age <-2 SD) Among Children Under Five Years of Age." https://unstats.un.org/sdgs/metadata/files/Metadata-02-02-01.pdf.

Watson, G. S. 1971. "Trend -Surface Analysis." *Mathematical Geology* 3: 215–26.

———. 1972. "Trend Surface Analysis and Spatial Correlation." *Geological Society of America Special Paper* 146: 39–46.

Weisberg, Sanford. 2014. *Applied Linear Regression*. Fourth. Hoboken NJ: Wiley. http://z.umn.edu/alr4ed.

WHO. 2006. "WHO Child Growth Standards: Length/Height-for-Age, Weight-for-Age, Weight-for-Length, Weight-for-Height and Body Mass Index-for-Age: Methods and Development." Geneva: WHO. https://www.who.int/publications/i/item/924154693X.

Yin, Hui, Yutong Wang, Yuxin Wang, Zihan Li, Yujie Li, and Yuxin Wang. 2024. "A Rapid Review of Clustering Algorithms." *arXiv Preprint arXiv:2401.07389*.

Zhang, Hao. 2002. "On Estimation and Prediction for Spatial Generalized Linear Mixed Models." *Biometrics* 58 (1): 129–36.

———. 2004. "Inconsistent Estimation and Asymptotically Equal Interpolations in Model-Based Geostatistics." *Journal of the American Statistical Association* 99 (465): 250–61.

Zouré, Honorat GM, Mounkaila Noma, Afework H Tekle, Uche V Amazigo, Peter J Diggle, Emanuele Giorgi, and Jan HF Remme. 2014. "Geographic Distribution of Onchocerciasis in the 20 Participating Countries of the African Programme for Onchocerciasis Control: (2) Pre-Control Endemicity Levels and Estimated Number Infected." *Parasites & Vectors* 7 (1): 326–26.

# *Appendix*

## An appendix section

Tempor justo turpis amet egestas inceptos. Conubia tristique mattis cras dui fames nisi curae. Dictumst convallis egestas, elementum viverra ex morbi. Sem libero est curae cursus, tempor rutrum etiam. Convallis scelerisque metus primis mus scelerisque vehicula potenti congue augue. Lacus taciti ullamcorper dapibus elit vulputate natoque erat. Nisi facilisi nisl urna ullamcorper imperdiet at nullam ac imperdiet.