Journal of Statistical Software

MMMMMM YYYY, Volume VV, Issue II. *http://www.jstatsoft.org/*

PrevMap: An R Package for Prevalence Mapping

Emanuele Giorgi
Lancaster University

Peter J. Diggle
Lancaster University

Abstract

In this paper we introduce a new R package, **PrevMap**, for the analysis of spatially referenced prevalence data, including both classical maximum likelihood and Bayesian approaches to parameter estimation and plug-in or Bayesian prediction. More specifically, the new package implements fitting of geostatistical models for binomial data, based on two distinct approaches. The first approach uses a generalized linear mixed model with logistic link function, binomial error distribution and a Gaussian spatial process as a stochastic component in the linear predictor. A simpler, but approximate, alternative approach consists of fitting a linear Gaussian model to empirical-logit-transformed data. The package also includes implementations of convolution-based low-rank approximations to the Gaussian spatial process to enable computationally efficient analysis of large spatial data-sets. We illustrate the use of the package through the analysis of *Loa loa* prevalence data from Cameroon and Nigeria. We illustrate the use of the low rank approximation using a simulated geostatistical data-set.

Keywords: Bayesian analysis, geostatistics, low-rank approximations, Monte Carlo maximum likelihood, prevalence data, R.

1. Introduction

This article introduces **PrevMap**, an R package for classical and Bayesian inference on spatially referenced prevalence data. The package implements fitting and spatial prediction for the standard geostatistical model used in the context of prevalence mapping (Diggle, Tawn, and Moyeed 1998). This model falls within the generalized linear mixed model framework with binomial error distribution, logistic-link function and a latent Gaussian spatial process in the linear predictor. For classical analysis, we estimate parameters by Monte Carlo maximum likelihood (MCML), which uses importance sampling techniques so as to approximate the high-dimensional intractable integral that defines the likelihood function; see for example Christensen (2004). Plug-in spatial prediction is then carried out by fixing the model pa-

	PrevMap	**geoR**	**geoRglm**	**geostatsp**	**geoBayes**	**spBayes**
Binomial models	✓	✗	✓	✓	✓	✓
Likelihood-based inference (binomial)	✓	✗	✓	✗	✗	✗
Bayesian inference (binomial)	✓	✗	✓	✓	✓	✓
Nugget effect (binomial)	✓	✗	✓	✓	✓	✗
Low-rank approximations.	✓	✗	✗	✗	✗	✓
Fitting of two-level models.	✓	✗	✗	✓	✗	✗
Non-linear prediction	✓	✓*	✓	✗	✓	✓
Multivariate prediction	✓	✓*	✓	✗	✓	✓
Anisotropy	✗	✓*	✓	✓*	✗	✗
Non-Matérn correlation functions	✗	✓*	✓	✗	✓	✓

Table 1: List of functionalities that are currently available (✓) and not available (✗) in **PrevMap** and other R packages for geostatistical analysis.

* Available only for the linear model.

rameters at the corresponding MCML estimates. In order to account for uncertainty in the model parameter estimates, we also consider a Bayesian approach in which plug-in predictive distributions at different values of the model parameters are weighted according to their posterior probabilities. A simpler, approximate procedure consists of fitting a geostatistical linear Gaussian model to empirical-logit-transformed prevalences.

Table 1 summarises the common functionalities required for prevalence mapping that are available in **PrevMap** and the existing packages **geoR** (Diggle and Ribeiro 2007; Ribeiro and Diggle 2001), **geoRglm** (Christensen and Ribeiro 2002), **geostatsp** (Brown 2015), **geoBayes** and **spBayes** (Finley, Banerjee, and Carlin 2007; Finley, Banerjee, and Gelfand 2015). Overall, **PrevMap** provides the most extensive functionality. Specifically, **PrevMap** provides the following features: implementation of a convolution-based low-rank approximation that can be used to reduce the computational burden when analysing large spatial data-sets; accurate numerical computation of MCML standard errors for both regression and covariance parameters estimates; inclusion of both individual-level and location-level explanatory variables with random effects defined at location-level when repeated observations are made at the same location; more flexible prior specifications for the covariance parameters; implementation of an efficient Hamiltonian Markov chain Monte Carlo algorithm for Bayesian parameter estimation.

The paper is structured as follows. In Section 2, we briefly introduce the geostatistical binomial logistic (henceforth BL) model, describe methods for classical and Bayesian inference, and outline approximate procedures based on the empirical logit transformation and low-rank approximations. Section 3 describes geostatistical analyses of *Loa loa* prevalence data using either the approximate linear model for the empirical logit transformation of prevalence or the exact BL model using Monte Carlo methods, both for classical and Bayesian analysis. In Section 4, we illustratethe use of the low-rank approximation by fitting a BL model to a simulated geostatistical data-set. Section 5 is a concluding discussion on planned extensions to the package.

2. Methodological framework

The ingredients of a geostatistical BL model are: random variables Y_i of positive counts, binomial denominators m_i, explanatory variables $d_i \in \mathbb{R}^p$ and associated sampling locations $x_i : i = 1, \ldots, n$ in a given region of interest $A \subseteq \mathbb{R}^2$. Conditionally on a zero-mean Gaussian process $S(x)$ and mutually independent zero-mean Gaussian variables Z_i, Y_i follows a binomial distribution with mean $E[Y_i|S(x_i), Z_i] = m_i p_i$ such that

$$\log\left\{\frac{p_i}{1-p_i}\right\} = T_i = d(x_i)^\top \beta + S(x_i) + Z_i, \tag{1}$$

where we set $d_i = d(x_i)$ to emphasize the spatial context. In (1), we write τ^2 for the variance of Z_i and model $S(x)$ as a stationary isotropic Gaussian process with variance σ^2 and Matérn (1986) correlation function given by

$$\rho(u; \phi, \kappa) = \{2^{k-1}\Gamma(\kappa)\}^{-1}(u/\phi)^\kappa \mathcal{K}_\kappa(u/\phi), u > 0,$$

where $\phi > 0$ is a scale parameter, $\mathcal{K}_\kappa(\cdot)$ is the modified Bessel function of the second kind of order $\kappa > 0$ and u is the distance between two sampling locations. The shape parameter κ determines the smoothness of $S(x)$, in the sense that $S(x)$ is $\lceil \kappa \rceil - 1$ times mean-square differentiable, with $\lceil \kappa \rceil$ denoting the smallest integer greater than or equal to κ.

In most of the functions available in **PrevMap**, the Matérn shape parameter κ is treated as fixed. One reason for this is that, as shown by Zhang (2004), not all of the three parameters σ^2, ϕ and κ can be consistently estimated under in-fill asymptotics, and in practice this translates to κ often being poorly identified. Additionally, the parameter κ is rarely of direct scientific interest. We therefore recommend either fixing κ at a plausible value, or considering a discrete set of values e.g., $\{1/2, 3/2, 5/2\}$ corresponding to different levels of smoothness, and profiling on κ.

2.1. Monte Carlo maximum likelihood

The likelihood function for the parameters β and $\theta^\top = (\sigma^2, \phi, \tau^2)$ is obtained by integrating out the random effects in T_i as defined by Equation 1. Let D denote the n by p matrix of explanatory variables and $y^\top = (y_1, \ldots, y_n)$ the vector of binomial observations. The marginal distribution of T is multivariate Gaussian with mean vector $D\beta$ and covariance matrix $\Sigma(\theta)$ with diagonal elements $\sigma^2 + \tau^2$ and off-diagonal elements $\sigma^2\rho(u_{ij})$, where u_{ij} is the distance between locations x_i and x_j. The conditional distribution of $Y^\top = (Y_1, \ldots, Y_n)$ given $T^\top = t^\top = (t_1, \ldots, t_n)$ is

$$f(y|t) = \prod_{i=1}^{n} f(y_i|t_i), \tag{2}$$

a product of independent binomial probability functions. The likelihood function for β and θ follows as

$$L(\beta, \theta) = f(y; \beta, \theta) = \int_{\mathbb{R}^n} N(t; D\beta, \Sigma(\theta)) f(y|t)\, dt \tag{3}$$

where $N(\cdot; \mu, \Sigma)$ is the density function of a multivariate Gaussian distribution with mean vector μ and covariance matrix Σ.

The MCML method (Geyer and Thompson 1992; Geyer 1994, 1996, 1999) uses conditional simulation from the distribution of T given $Y = y$ to approximate the high-dimensional integral in Equation 3. Specifically, the likelihood function can be rewritten as

$$\begin{aligned} L(\beta,\theta) &= \int_{\mathbb{R}^n} \frac{N(t;D\beta,\Sigma(\theta))f(y|t)}{N(t;D\beta_0,\Sigma(\theta_0))f(y|t)} f(y,t)\,dt \\ &\propto \int_{\mathbb{R}^n} \frac{N(t;D\beta,\Sigma(\theta))}{N(t;D\beta_0,\Sigma(\theta_0))} f(t|y)\,dt = E_{T|y}\left[\frac{N(t;D\beta,\Sigma(\theta))}{N(t;D\beta_0,\Sigma(\theta_0))}\right] \end{aligned} \tag{4}$$

where $f(y,t) = N(t;D\beta_0,\Sigma(\theta_0))f(y|t)$ is the joint distribution of Y and T for pre-defined, fixed values of β_0 and θ_0. We use a Markov Chain Monte Carlo (MCMC) algorithm to obtain m samples $t_{(i)}$ from the conditional distribution of T given $Y = y$ under β_0 and θ_0 and approximate Equation 4 with

$$L_m(\beta,\theta) = \frac{1}{m}\sum_{i=1}^{n} \frac{N(t_{(i)};D\beta,\Sigma(\theta))}{N(t_{(i)};D\beta_0,\Sigma(\theta_0))}. \tag{5}$$

Note that $L_m(\beta,\theta)$ is a consistent estimator of $L(\beta,\theta)$, whether or not the samples $t_{(i)}$ are correlated. The optimal choices for β_0 and θ_0 are the maximum likelihood estimates of β and θ, for which $\max_{\beta,\theta} L_m(\beta,\theta) \to 1$ as $m \to \infty$. Since our choices for β_0 and θ_0 will necessarily differ from the actual maximum likelihood estimates, the distance of $L_m(\hat{\beta}_m,\hat{\theta}_m)$ from 1, where $\hat{\beta}_m$ and $\hat{\theta}_m$ are the MCML estimates, provides a measure of the quality of the Monte Carlo approximation. In practice, we embed the maximisation of $L_m(\beta,\theta)$ within the following iterative procedure. Let $\hat{\beta}_m$ and $\hat{\theta}_m$ denote the values that maximise $L_m(\beta,\theta)$ using an initial guess at suitable values β_0 and θ_0, repeat the maximisation with $\beta_0 = \hat{\beta}_m$ and $\theta_0 = \hat{\theta}_m$ and continue until convergence.

For maximization of the approximation to the log-likelihood $l_m(\beta,\theta) = \log L_m(\beta,\theta)$ in **PrevMap**, the user can choose between a BFGS algorithm or unconstrained optimization with PORT routines. Let $\psi = \log\theta$; analytical expressions for the first and second derivatives of $l_m(\beta,\psi)$ with respect to β and ψ are internally passed to the optimization functions `maxBFGS` of the **maxLik** package (Henningsen and Toomet 2011) in the former case and to the `nlminb` function in the latter. This can be very useful in order better to locate the global maximum on a relatively flat likelihood surface, as it is often the case for the ψ parameter. The MCML standard errors are then estimated by taking the square-roots of the diagonal elements of the inverse of the negative Hessian of $l_m(\hat{\beta}_m,\hat{\psi}_m)$. The inherent accuracy of this approximation for the standard errors is context-specific in addition to being affected by the Monte Carlo error. As a partial check, the resulting standard errors for β are typically larger than those estimated using an ordinary logistic regression. In the examples of Section 3.3 and Section 4, the number of simulated samples is sufficiently large to make the Monte Carlo error negligible.

In the **PrevMap** package, conditional simulation of T given y with fixed parameters β and θ is implemented by the function `Laplace.sampling`. This function uses a Langevin-Hastings algorithm to update the random variable $\hat{T} = \hat{\Sigma}^{1/2}(T-\hat{t})$, where $\hat{t}$ and $\hat{\Sigma}$ are, respectively, the mode and the inverse of the negative Hessian of the density of the conditional distribution. The objective of this linear transformation is to break the dependence between the different components of T so as to allow for faster convergence of the MCMC algorithm. However, when using the function `binomial.logistic.MCML` for parameter estimation, conditional simulation is carried out internally; see Section 3.3.1.

Function	Model	Method of inference	Type of use
`binomial.logistic.MCML`	Binomial	Classical	Parameter estimation
`binomial.logistic.Bayes`	Binomial	Bayesian	Parameter estimation
`linear.model.MLE`	Linear	Classical	Parameter estimation
`linear.model.Bayes`	Linear	Bayesian	Parameter estimation
`spatial.pred.binomial.MCML`	Binomial	Classical	Spatial prediction
`spatial.pred.binomial.Bayes`	Binomial	Bayesian	Spatial prediction
`spatial.pred.linear.MLE`	Linear	Classical	Spatial prediction
`spatial.pred.linear.Bayes`	Linear	Bayesian	Spatial prediction

Table 2: Some of the main functions available in the **PrevMap** package. Note that all of the listed functions include an option to use a low-rank approximation procedure.

2.2. Bayesian inference

In the Bayesian framework, a joint prior distribution for β and θ is combined with the likelihood function through Bayes' theorem so as to obtain the corresponding posterior distribution. We assume that the prior distributions for θ and β are of the form

$$\begin{aligned} \theta &\sim g(\cdot), \\ \beta|\sigma^2 &\sim N(\cdot; \xi, \sigma^2\Omega) \end{aligned}$$

where $g(\cdot)$ can be any distribution for θ, and ξ and Ω are the mean vector and a p by p covariance matrix for the Gaussian prior of β. The posterior distribution for β, θ and T is given by

$$\pi(\beta, \theta, t|y) \propto g(\theta)N(\beta; \xi, \sigma^2\Omega)N(t; D\beta, \Sigma(\theta))f(y|t). \tag{6}$$

The function `binomial.logistic.Bayes` can be used to obtain samples from the above posterior distribution. This uses an MCMC algorithm, where θ, β and T are updated in turn using the following procedure.

1. Initialise β, θ and T.

2. Following the procedure proposed by Christensen, Roberts, and Sköld (2006), use the following re-parametrization for the covariance parameters

$$(\tilde{\theta}_1, \tilde{\theta}_2, \tilde{\theta}_3) = (\log\sigma, \log(\sigma^2/\phi^{2\kappa}), \log\tau^2)$$

and update each of them in turn using a random-walk Metropolis Hastings (RWMH). In each of the three RWMH for $\tilde{\theta}_1$, $\tilde{\theta}_2$ and $\tilde{\theta}_3$, the standard deviation, h say, of the Gaussian proposal at i-th iteration is given by

$$h_i = h_{i-1} + c_1 i^{-c_2}(\alpha_i - 0.45), \tag{7}$$

where $c_1 > 0$ and $c_2 \in (0, 1]$ are pre-defined constants, α_i is the acceptance probability at the i-th iteration and 0.45 is the optimal acceptance probability for a univariate Gaussian distribution.

3. Update β using a Gibbs step. The required conditional distribution of β given θ and T is Gaussian, independent of y and with mean $\tilde{\xi}$ and covariance matrix $\sigma^2\tilde{\Omega}$ given by

$$\begin{aligned} \tilde{\xi} &= \tilde{\Omega}(\Omega^{-1}\xi + D^\top R(\theta)^{-1}T) \\ \sigma^2\tilde{\Omega} &= \sigma^2(\Omega^{-1} + D^\top R(\theta)^{-1}D)^{-1}, \end{aligned}$$

where $\sigma^2 R(\theta) = \Sigma(\theta)$.

4. Update the distribution of T given β, θ and y using a Hamiltonian Monte Carlo algorithm (Neal 2011). Specifically, let $H(t, u)$ be the Hamiltonian function

$$H(t, u) = u^\top u/2 - \log f(t|y, \beta, \theta),$$

where $u \in \mathbb{R}^n$ is the vector of the momentum variables and $f(t|y, \beta, \theta)$ is the conditional density of T given β, θ and y. The partial derivatives of $H(u, t)$ determine how u and t change over time v according to the Hamiltonian equations

$$\begin{aligned} \frac{dt_i}{dv} &= \frac{\partial H}{\partial u_i}, \\ \frac{du_i}{dv} &= -\frac{\partial H}{\partial t_i} \end{aligned}$$

for $i = 1, \ldots, n$. In order to implement the Hamiltonian dynamic, the above differential equations are discretized using the *leapfrog* method (Neal 2011, pages 121-122) and approximate solutions are then found.

Two auxiliary functions, `control.prior` and `control.mcmc.Bayes`, define prior distributions and tuning parameters for the above MCMC scheme.

2.3. Empirical logit transformation

An alternative approach to exact fitting methods is to use a trans-Gaussian approximation of the model in Equation 1. This consists of fitting a linear model to the empirical logit transformation of the data,

$$\tilde{Y}_i = \log\left(\frac{Y_i + 1/2}{m_i - Y_i + 1/2}\right) : i = 1, \ldots, n. \tag{8}$$

The method then assumes that $\tilde{Y}_i|S(x_i) \sim N(d(x_i)^\top\beta + S(x_i), \tau^2)$ with $S(x)$ having the same properties as previously defined. Guidance on when this model can be used safely is given by Stanton and Diggle (2013).

In the **PrevMap** package the empirical logit transformation is implemented both for classical and Bayesian inference in the functions `linear.model.MLE` and `linear.model.Bayes`.

2.4. Low-rank approximation

The Gaussian process $S(x)$ in Equation 1 can be represented as a convolution of Gaussian noise (Higdon 1998, 2002) ,

$$S(x) = \int_{\mathbb{R}^2} K(\|x - t\|; \phi, \kappa)\, dB(t) \tag{9}$$

where B is Brownian motion, $\|\cdot\|$ is the Euclidean distance and $K(\cdot)$ is the Matérn kernel given by the following expression

$$K(u;\phi,\kappa) = \frac{\Gamma(\kappa+1)^{1/2}\kappa^{(\kappa+1)/4}u^{(\kappa-1)/2}}{\pi^{1/2}\Gamma((\kappa+1)/2)\Gamma(\kappa)^{1/2}(2\kappa^{1/2}\phi)^{(\kappa+1)/2}}\mathcal{K}_\kappa(u/\phi), u>0. \tag{10}$$

Let $(\tilde{x}_1,\ldots,\tilde{x}_r)$ be a grid of spatial knots. By discretization of Equation 9, and for r sufficiently large, we obtain a low-rank approximation

$$S(x) \approx \sum_{i=1}^{r} K(\|x-\tilde{x}_i\|;\phi,\kappa)U_i, \tag{11}$$

where the U_i are independent zero-mean Gaussian variables with variance σ^2. This approximation is particularly beneficial for relatively large values of the scale parameter ϕ, when a small number of spatial knots is required to give a good approximation over the study-region. Note also that the number of spatial knots r is independent of the sample size n, making this approach computationally attractive when n is large.

Since the resulting approximation in Equation 11 is no longer a stationary process, we adjust the value of σ^2 by multiplying it by the following quantity

$$\frac{1}{n}\sum_{i=1}^{n}\sum_{j=1}^{m} K(\|\tilde{x}_j-\tilde{x}_i\|;\phi,\kappa)^2.$$

The adjusted value of σ^2 is then a closer approximation to the actual variance of the Gaussian process $S(x)$.

Different implementations of this method are possible, depending on whether we use an exact fitting method or an empirical logit approximation. In the **PrevMap** package, low-rank approximations can be used in each of the fitting functions listed in Table 2; we give an example in Section 4.

Implementations of the low-rank approximation for the BL and linear model are as follows.

- *BL model.* In this implementation the nugget effect is not included, hence $\tau^2=0$. For both the classical and Bayesian analysis, conditional simulation from the distribution of the random effect U given the data y (and the model parameters in the Bayesian case) is used, hence avoiding matrix inversion.

- *Linear model.* The low-rank approximation is here used for the empirical logit transformation of prevalence. In this case $\tau^2>0$, since the nugget effect is now a proxy for binomial sampling variation. Inversion of the covariance matrix and computation of the determinant are simplified as follows. Let $K(\theta)$ denote the n by r kernel matrix. The covariance matrix now assumes the form

  $$\Sigma(\theta) = \sigma^2 K(\theta)K(\theta)^\top + \tau^2 I_n$$

  where I_n is the n by n identity matrix. The Woodbury identity for matrix inversion gives

  $$\Sigma(\theta)^{-1} = \sigma^2\nu^{-2}(I_n - \nu^{-2}K(\theta)(\nu^{-2}K(\theta)^\top K(\theta)+I_r)^{-1}K(\theta)^\top)$$

where $\nu^2 = \tau^2/\sigma^2$. Inversion of $\Sigma(\theta)$ now requires inversion of an r by r matrix. Computation of the determinant, denoted by $|\cdot|$, can also be simplified by using Sylvester's determinant theorem. This gives

$$\begin{aligned} |\Sigma(\theta)| &= |\sigma^2 K(\theta) K(\theta)^\top + \tau^2 I_n| \\ &= \tau^{2n} |\nu^{-2} K(\theta)^\top K(\theta) + I_r|, \end{aligned}$$

which again reduces the dimensionality of the required matrix operations from n by n to r by r.

2.5. Spatial prediction

We now consider the prediction of $T^* = (T(x_{n+1}), \ldots, T(x_{n+q}))^\top$ at q additional locations not included in the data. This requires all relevant explanatory variables to be available at the prediction locations. We do not include the mutually independent random variables Z_i in Equation 1 as part of our target for prediction, hence $T(x_{n+i}) = d(x_{n+i})^\top \beta + S(x_{n+i})$ for $i = 1, \ldots, q$.

Conditionally on $T^\top = (T_1, \ldots, T_n)$, β, θ and y, the target for prediction T^* follows a multivariate Gaussian distribution with mean and covariance matrix

$$\mu^*(T) = D^*\beta + C\Sigma^{-1}(T - D\beta), \tag{12}$$

$$\Sigma^* = V - C\Sigma^{-1}C^\top, \tag{13}$$

where C is the cross-covariance matrix between T and T^*, V is the covariance matrix of T^* and D^* is a q by p matrix of explanatory variables at the prediction locations. Let $T^*_{(j)}$ denote the j-th simulated sampled from the posterior distribution of T^* for $j = 1, \ldots, m$. If the sample mean is to be used as a point predictor of T, the package uses the following result to reduce the associated Monte Carlo error,

$$E_{T^*|y}E[T^*] = E_{T,\beta,\theta|y}[E_{T^*|T,\beta,\theta,y}[T^*]] = E_{T,\beta,\theta|y}[\mu^*(T)] \approx \frac{1}{m}\sum_{j=1}^{m} \mu^*(T_{(j)}).$$

Prediction of the functional $W(T^*)^\top = (W(T(x_{n+1})), \ldots, W(T(x_{n+q})))$ follows immediately by computing $W_{(j)} = W(T^*_{(j)})$ for $j = 1, \ldots, m$. The **PrevMap** package provides automatic computation of the following functionals.

- *Prevalence:* $W(T(x_{n+i})) = \exp\{T(x_{n+i})\}/(1 + \exp\{T(x_{n+i})\})$.
- *Odds:* $W(T_{n+i}) = \exp\{T(x_{n+i})\}$. Let $\sigma^{2*} = \mathrm{diag}(\Sigma^*)$ denote the vector of conditional variances. In this case, the Monte Carlo error in the computation of the posterior mean is reduced by noticing that $E_{T^*|T,\beta,\theta,y}[\exp\{T^*\}] = \exp\{\mu^*(T) + \sigma^{2*}/2\}$, hence

$$E_{T^*|y}[\exp\{T^*\}] \approx \frac{1}{m}\sum_{j=1}^{m} \exp\{\mu^*(T_{(j)}) + \sigma^{2*}_{(j)}/2\}.$$

Another summary of the posterior distribution that is often relevant, particularly in problems of hotspot detection, is the exceedance probability $P(T(x_{n+i}) > l \mid y)$ for a given threshold l and $i = 1, \ldots, q$. We estimate this as

$$\frac{1}{m} \sum_{j=1}^{m} I\left(T_{(j)}(x_{n+i}) > l\right),$$

where $I(a > l)$ is 1 if $a > l$ and 0 otherwise, and $T_{(j)}(x_{n+i})$ is the i-th element of $T^*_{(j)}$.

The `spatial.pred.binomial.MCML` and `spatial.pred.binomial.Bayes` functions can be used for classical and Bayesian spatial prediction, respectively. As we later illustrate, one of the available options is also the computation of either joint or marginal predictions. For example, joint predictions are needed when the target for prediction is an average over a sub-region. Spatial prediction for the empirical logit transformation using classical and Bayesian approaches is implemented in the `spatial.pred.linear.MLE` and `spatial.pred.linear.Bayes` functions, respectively. Low-rank approximations for each of the above functions are also available; see Section 3.3.

3. Example: Loa loa prevalence mapping

The data for this example relate to a study of the prevalence of *Loa loa* (eyeworm) in a series of surveys undertaken in 197 villages in Cameroon and southern Nigeria; see Diggle, Thomson, Christensen, Rowlingson, Obsomer, Gardon, Wanji, Takougang, Enyong, Kamgno, Remme, Boussinesq, and Molyneux (2007) for more details. Figure 2(a) shows the locations of the sampled villages.

3.1. Exploratory analysis

Exploratory analysis is useful under both classical and Bayesian inferential frameworks, to identify a provisional model for the data. under the classical framework, initial values for the model parameters are also needed for numerical optimisation of the likelihood. Initial values for the regression coefficients can be easily obtained from an ordinary logistic regression fit. Choosing initial values for the covariance parameters is less straightforward. The shape parameter κ of the Matérn function is typically chosen from a discrete set of candidate values, which can be compared by evaluating a profile likelihood for κ based on the empirical logit transformation of the observed prevalence, as in the following example.

```
R> library("PrevMap")
R> data("loaloa")
R> loaloa$logit <- log((loaloa$NO_INF + 0.5)/
+  (loaloa$NO_EXAM - loaloa$NO_INF + 0.5))
R> profile.kappa <- shape.matern(formula = logit ~ 1,
+  coords = ~ LONGITUDE + LATITUDE,
+  data = loaloa, set.kappa = seq(0.2,1.5, length = 15),
+  start.par = c(0.2,0.05), coverage = 0.95)
R>
R> c(profile.kappa$lower, profile.kappa$upper)
```

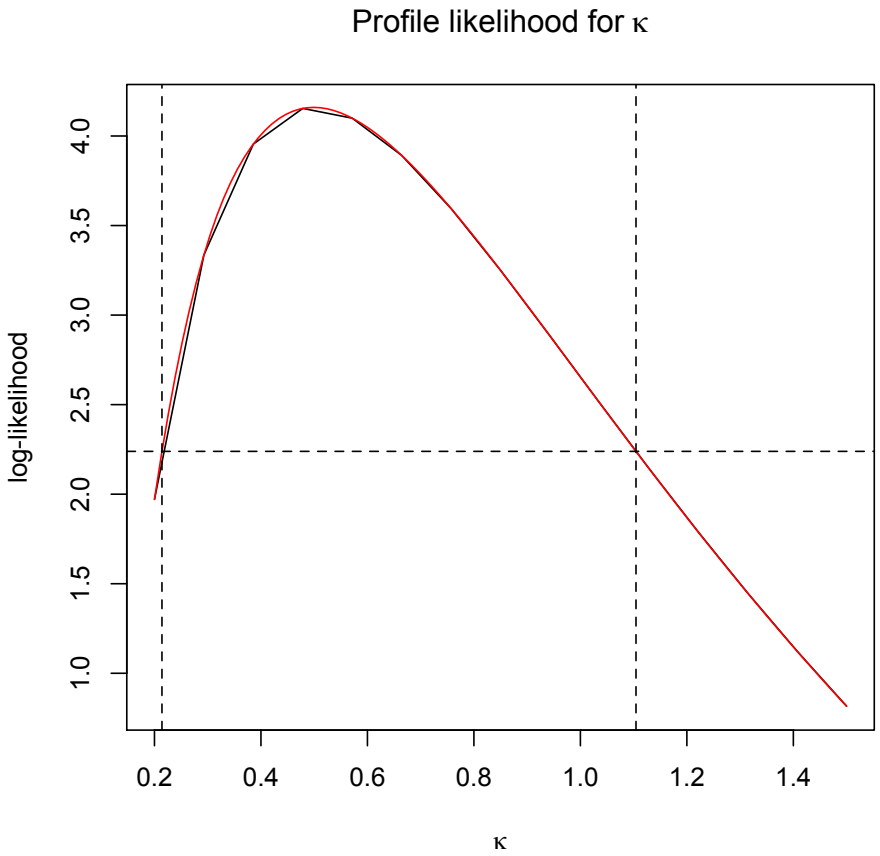

Figure 1: Profile likelihood for the shape parameter κ of the Matérn covariance function, obtained using the function `shape.matern`; the profile likelihood (black solid line) is interpolated by a spline (red solid line), which is then used to obtain a confidence interval of coverage 95% (vertical dashed lines).

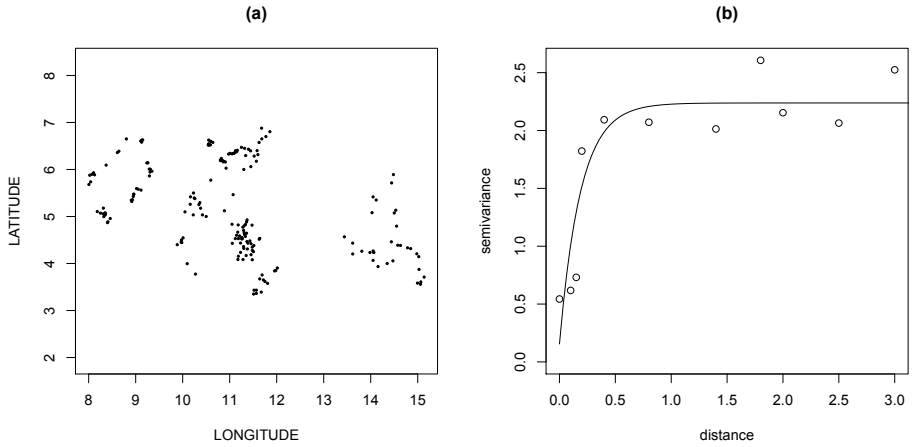

Figure 2: (a) Sampling locations for the *Loa loa* data. (b) Empirical variogram for the empirical logit transformation of the observed prevalence with theoretical variogram (solid line) obtained by least-squares estimation.

```
[1] 0.2140705 1.1044392
```

```
R> profile.kappa$kappa.hat
```

```
[1] 0.4991899
```

The `shape.matern` function evaluates the profile likelihood for κ and obtains a corresponding confidence interval with coverage specified by the argument `coverage`. The set of values that are used for evaluation of the profile likelihood is specified through the `set.kappa` argument. Computation of the confidence interval uses the interpolated profile log-likelihood as shown in Figure 1: the red line corresponds to an interpolating spline and the likelihood threshold, denoted by the horizontal dashed line, is obtained using the asymptotic distribution of a chi-squared with one degree of freedom. Since the maximum likelihood estimate is very close to 1/2, we then fix the shape parameter κ at this value for the subsequent analysis.

The package **geoR** provides several functions that are useful for an initial exploratory analysis of geostatistical data. For example, using the function `variofit`, a least-squares estimation of the empirical variogram can be used in order to choose initial values for the covariance parameters of the Gaussian spatial process.

```
R> library("geoR")
R> coords <- as.matrix(loaloa[, c("LONGITUDE", "LATITUDE")])
R> vari <- variog(coords = coords, data = loaloa$logit,
+  uvec = c(0, 0.1, 0.15, 0.2, 0.4, 0.8, 1.4, 1.8, 2, 2.5, 3))
R> vari.fit <- variofit(vari, ini.cov.pars = c(2, 0.2),
+  cov.model = "matern",
+  fix.nugget = FALSE, nugget = 0 ,
+  fix.kappa = TRUE, kappa = 0.5)
R> par(mfrow = c(1,2))
R> plot(coords, pch = 20, asp = 1, cex = 0.5, main = "(a)")
R> plot(vari, main = "(b)")
R> lines(vari.fit)
R> vari.fit

variofit: model parameters estimated by WLS (weighted least squares):
covariance model is: matern with fixed kappa = 0.5 (exponential)
parameter estimates:
  tausq sigmasq     phi
 0.1554  2.0827  0.1890
Practical Range with cor = 0.05 for asymptotic range: 0.5662674

variofit: minimised weighted sum of squares = 780.6663
```

The above code computes the empirical logit transformation of the observed *Loa loa* prevalence, uses this to obtain the empirical variogram with the `variog` function and fits an exponential correlation function to the empirical variogram with the `variofit` function, which uses a least squares curve-fitting criterion. The results are shown in Figure 2(b).

3.2. Linear model

In this section we show how to fit a linear model with Matérn correlation function to the empirical logit transformation of the *Loa loa* data using the maximum likelihood method. The `linear.model.MLE` function has its counterpart in the `likfit` function in **geoR** but, unlike `likfit`, uses analytic expressions for the gradient function and Hessian matrix, and delivers an estimated covariance matrix of the maximum likelihood estimates accordingly. As shown in the next section, the `binomial.logistic.MCML` function uses the same approach in fitting a BL model.

```
R> fit.MLE <- linear.model.MLE(formula = logit ~ 1,
+  coords = ~ LONGITUDE + LATITUDE, data = loaloa,
+  start.cov.pars = c(0.2, 0.15), kappa = 0.5)
R>
R> summary(fit.MLE, log.cov.pars = FALSE)

Geostatistical linear Gaussian model
Call:
geo.linear.MLE(formula = formula, coords = coords, data = data,
    kappa = kappa, fixed.rel.nugget = fixed.rel.nugget,
    start.cov.pars = start.cov.pars,
    method = method)

            Estimate  StdErr z.value   p.value
(Intercept)  -2.2986  0.5469  -4.203 2.634e-05 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Log-likelihood: -94.34047

Covariance parameters Matern function (kappa = 0.5)
        Estimate StdErr
sigma^2  2.45148 0.1393
phi      0.84398 0.4933
tau^2    0.36865 1.1717

Legend:
sigma^2 = variance of the Gaussian process
phi = scale of the spatial correlation
tau^2 = variance of the nugget effect
```

The first argument of `linear.model.MLE` specifies the covariates used in the regression as a `formula` object; in this case `formula = logit ~ 1` since we only fit an intercept. The argument `start.cov.pars` provides the initial values of ϕ and $\nu^2 (= \tau^2/\sigma^2)$, respectively, used in the optimization algorithm. The argument `fixed.rel.nugget` allows the relative variance of the nugget effect ν^2 to be fixed if desired. Additionally, two different maximisation algorithms are available: if `method = "BFGS"` (set by default), the `maxBFGS` function in the

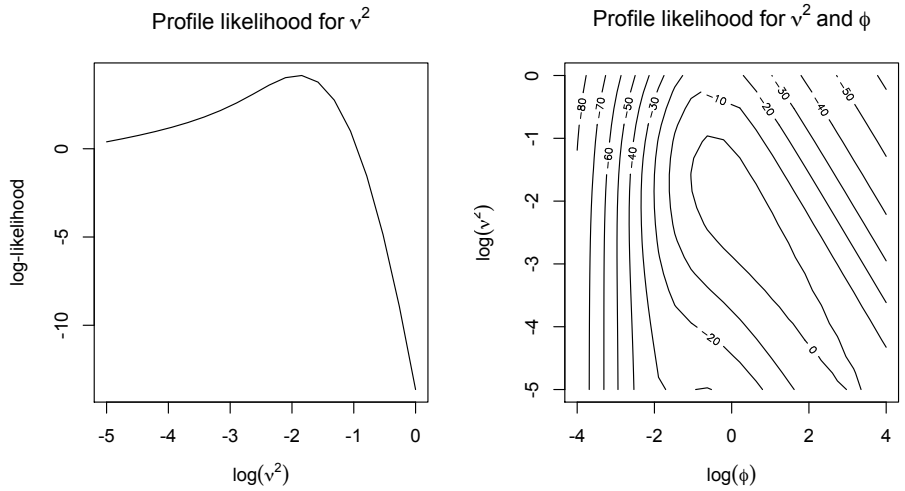

Figure 3: Profile log-likelihood for ν^2 (left panel) and (ν^2, ϕ) (right panel) obtained using the function `loglik.linear.model`.

`maxLik` package is used, otherwise `method = "nlminb"` and the `nlminb` function is then used for unconstrained optimization using PORT routines. When calling a summary of the fitted model, estimates and standard errors of the covariance parameters are given on the log-scale by default. Setting `log.scale = FALSE` gives estimates and standard errors on the original scale.

The function `loglik.linear.model` can be used either for computation of the profile likelihood for ϕ and/or ν^2 or for evaluation of the likelihood keeping the other parameters fixed. The auxiliary function `control.profile` is used to define the set of values for ϕ and/or ν^2 used in the evaluation of the likelihood, and the fixed values for β and σ^2, if necessary. The shape parameter κ is also fixed at the value defined in the fitted model object that must be specified as first argument of `loglik.linear.model`.

```
R> cp1 <- control.profile(rel.nugget = exp(seq(-5, 0, length = 20)))

Control profile: parameters have been set for
evaluation of the profile log-likelihood.

R> cp2 <- control.profile(rel.nugget = exp(seq(-5, 0, length = 20)),
+  phi = exp(seq(-4, 4, length = 20)))

Control profile: parameters have been set for
evaluation of the profile log-likelihood.

R> lp1 <- loglik.linear.model(fit.MLE, cp1, plot.profile = FALSE)
R>
R> lp2 <- loglik.linear.model(fit.MLE, cp2, plot.profile = FALSE)
```

```
R>
R> par(mfrow = c(1, 2))
R> plot(lp1, type = "l", log.scale = TRUE, xlab = expression(log(nu^2)),
+  ylab = "log-likelihood",
+  main = expression("Profile likelihood for" ~ nu^2))
R> plot(lp2, log.scale = TRUE, xlab = expression(log(phi)),
+  ylab = expression(log(nu^2)),
+  main = expression("Profile likelihood for" ~ nu^2 ~ "and" ~ phi))
```

The resulting plots of the profile log-likelihood for ν^2 and the profile log-likelihood surface of (ν^2, ϕ) are shown in Figure 3. These are generated using the function `plot.profile.PrevMap` as an S3 method, in which the logical argument `log.scale` can be set to `TRUE` in order to plot the profile likelihood on the log-scale of the chosen parameters. Likelihood-based confidence intervals for ϕ or ν^2 can also be obtained by using the `loglik.ci` function. As with the `shape.matern` function, the `loglik.ci` function uses a spline to interpolate the univariate profile likelihood and obtain a confidence interval of coverage specified by `coverage`.

```
R> ci0.95 <- loglik.ci(lp1, coverage = 0.95, plot.spline.profile = FALSE)
```

```
Likelihood-based 95% confidence interval: (0.04460758, 0.2936487)
```

3.3. Binomial logistic model

We now show how to fit a BL model to the *Loa loa* data using either the MCML method (Section 3.3.1) or a Bayesian approach (Section 3.3.2).

Likelihood-based analysis

For the MCML method, we set the parameters of the importance sampling distribution, β_0 and θ_0, to the estimates reported in Section 3.1 using ordinary logistic regression and a least squares fit to the variogram, respectively.

```
R> fit.glm <- glm(cbind(NO_INF, NO_EXAM - NO_INF) ~ 1, data = loaloa,
+  family = binomial)
R> par0 <- c(coef(fit.glm), vari.fit$cov.pars, vari.fit$nugget)
R> c.mcmc <- control.mcmc.MCML(n.sim = 10000, burnin = 2000,
+  thin = 8, h = (1.65)/(nrow(loaloa) ^ (1/6)))
R> fit.MCML1 <- binomial.logistic.MCML(formula = NO_INF ~ 1,
+  units.m = ~ NO_EXAM, par0 = par0,
+  coords = ~ LONGITUDE + LATITUDE, data = loaloa,
+  control.mcmc = c.mcmc,
+  kappa = 0.5, start.cov.pars = c(par0[3], par0[4]/par0[2]))
R> fit.MCML1$log.lik
```

```
[1] 24.24903
```

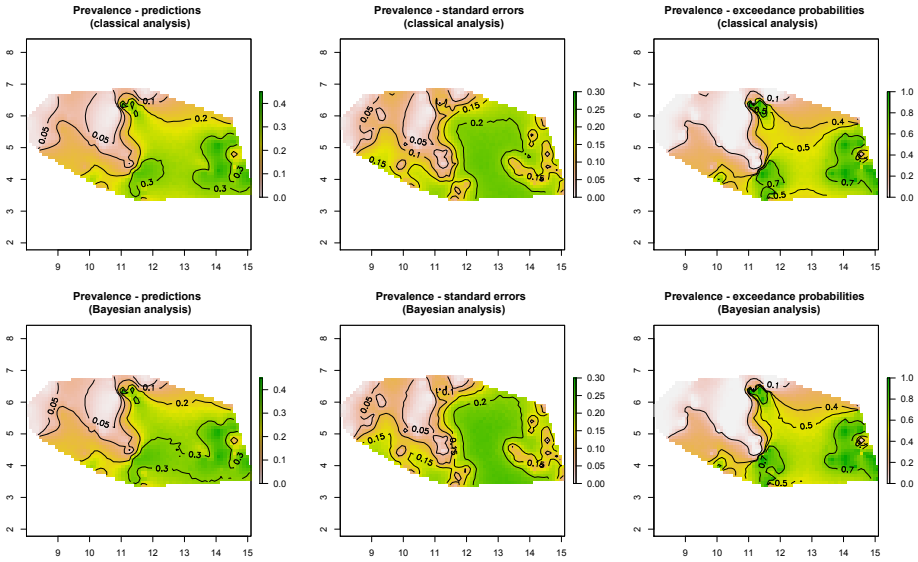

Figure 4: Plots of the prevalence estimates, standard errors and exceedance probabilities for the *Loa loa* data from the MCML (upper panels) and Bayesian (lower panels) analyses.

The above code fits a BL model by simulating 10,000 samples and retaining every eighth sample after a burn-in of 2,000 values to approximate the likelihood integral. The function `control.mcmc.MCMCL` sets the control parameters of the MCMC algorithm. The argument `h` represents the proposal density of the Langevin-Hastings (see Section 2.1). Our suggestion is to set this to $1.65/n^{1/6}$, where n is the sample size, which corresponds to the optimal value for sampling from a standard multivariate Gaussian distribution (Roberts and Rosenthal 2001).

We now repeat the MCML procedure twice, but with new values for β_0 and θ_0 set as the MCML estimates each time; in the last iteration, we also increase the number of retained simulated samples to 10,000.

```
R> par0 <- coef(fit.MCML1)
R> start <- c(par0[3], par0[4]/par0[2])
R> fit.MCML2 <- binomial.logistic.MCML(formula = NO_INF ~ 1,
+  units.m = ~ NO_EXAM, par0 = par0,
+  coords = ~ LONGITUDE + LATITUDE, data = loaloa,
+  control.mcmc = c.mcmc, kappa = 0.5,
+  start.cov.pars = c(par0[3], par0[4]/par0[2]))
R> fit.MCML2$log.lik

[1] 1.287294

R> c.mcmc <- control.mcmc.MCML(n.sim = 65000, burnin = 5000,
+  thin = 6, h = (1.65)/(nrow(loaloa)^(1/6)))
R> par0 <- coef(fit.MCML2)
R> fit.MCML3 <- binomial.logistic.MCML(formula = NO_INF ~ 1,
+  units.m = ~ NO_EXAM,par0=par0,
```

```
+  coords = ~LONGITUDE+LATITUDE,data=loaloa,
+  control.mcmc = c.mcmc,
+  kappa = 0.5, start.cov.pars = c(par0[3],par0[4]/par0[2]))
R> summary(fit.MCML3)

Binomial geostatistical model
Call:
binomial.logistic.MCML(formula = NO_INF ~ 1, units.m = ~NO_EXAM,
    coords = ~LONGITUDE + LATITUDE, data = loaloa, par0 = par0,
    control.mcmc = c.mcmc, kappa = 0.5, start.cov.pars = c(par0[3],
        par0[4]/par0[2]))

            Estimate   StdErr z.value   p.value
(Intercept) -2.30556  0.51743 -4.4558 8.358e-06 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Objective function: 0.1366855

Covariance parameters Matern function (kappa=0.5)
             Estimate StdErr
log(sigma^2)  0.92408 0.3215
log(phi)     -0.28736 0.3804
log(tau^2)   -3.23648 1.5796

Legend:
sigma^2 = variance of the Gaussian process
phi = scale of the spatial correlation
tau^2 = variance of the nugget effect
```

Note that updating β_0 and θ_0 with the resulting MCML estimates at each iteration results in the maximum value of the approximation to the log-likelihood function approaching zero. This is an indication that the MCML estimates are converging towards the actual maximum likelihood estimates of β and θ, for which the value of the Monte Carlo likelihood is exactly zero. We now carry out spatial predictions over a 0.1 by 0.1 degree regular grid, fixing the model parameters at the MCML estimates, and summarise the predictive distribution of prevalence in each grid cell through its mean, standard deviation and probability that the estimated prevalence is above 20%.

```
R> library("splancs")
R> poly <- coords[chull(coords),]
R> grid.pred <- gridpts(poly, xs = 0.1, ys = 0.1)
R> pred.MCML <- spatial.pred.binomial.MCML(fit.MCML3, grid.pred,
+  control.mcmc = c.mcmc, type = "marginal",
+  scale.predictions = "prevalence",
+  standard.errors = TRUE, thresholds = 0.2,
+  scale.thresholds = "prevalence")
```

```
R>
R> par(mfrow = c(1,3))
R> plot(pred.MCML, type = "prevalence",
+  summary = "predictions", zlim = c(0,0.45),
+  main = "Prevalence - predictions \n (classical analysis)")
R> contour(pred.MCML, type = "prevalence",
+  summary = "predictions",
+  levels = c(0.05,0.1,0.2,0.3), add = TRUE)
R> plot(pred.MCML, type = "prevalence",
+  summary = "standard.errors", zlim = c(0,0.3),
+  main = "Prevalence - standard errors  \n (classical analysis)")
R> contour(pred.MCML, type = "prevalence",
+  summary = "standard.errors",
+  levels = c(0.05,0.1,0.15,0.2), add = TRUE)
R> plot(pred.MCML, summary = "exceedance.prob",
+  zlim = c(0,1),
+  main = "Prevalence - exceedance probabilities \n (classical analysis)")
R> contour(pred.MCML, summary = "exceedance.prob",
+  levels = c(0.1,0.4,0.5,0.7), add = TRUE)
```

Using the argument `type` in `spatial.pred.binomial.MCML`, we can specify either marginal (`type = "marginal"`) or joint (`type = "joint"`) predictions. Through `scale.predictions`, we can also specify the scale on which predictions are required: `"logit"`, `"prevalence"` or `"odds"`. Exceedance probability thresholds and the scale on which they are provided are specified through the arguments `thresholds` and `scale.thresholds`, respectively. Figure 4 shows the images of prevalence estimates, standard errors and exceedance probabilities with associated contours. These plots are obtained using the methods `plot.pred.PrevMap` and `contour.pred.PrevMap`, whose arguments `type` and `summary` can be used to specify which summaries should be displayed.

The following code generates a set of diagnostic plots, shown in Figure 5, that provide checks on convergence of the MCMC.

```
R> par(mfrow=c(3,3))
R> S.mean <- apply(pred.MCML$samples, 2, mean)
R> acf(S.mean,main = "")
R> plot(S.mean,type = "l")
R> plot(ecdf(S.mean[1:5000]), main = "")
R> lines(ecdf(S.mean[5001:10000]), col = 2, lty = "dashed")
+
R> ind.S <- sample(1:nrow(grid.pred), 2)
R> acf(pred.MCML$samples[ind.S[1],], main = "")
R> plot(pred.MCML$samples[ind.S[1], ],
+  ylab = paste("Component n.", ind.S[1]), type = "l")
R> plot(ecdf(pred.MCML$samples[ind.S[1], 1:5000]), main = "")
R> lines(ecdf(pred.MCML$samples[ind.S[1], 5001:10000]),
+  col = 2, lty = "dashed")
+
```

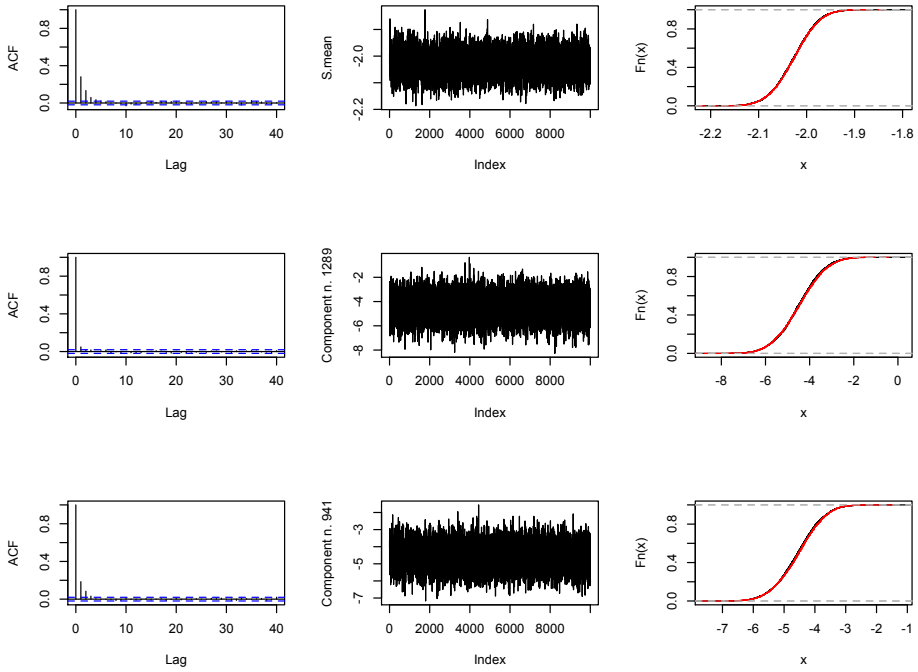

Figure 5: Autocorrelation plot of a thinned sequence of 10000 MCMC samples (left panels), trace plot of the same sequence (central panels) and empirical cumulative distribution plots for the first 5000 and second 5000 samples (right panels), for the spatial average of predicted logit-transformed prevalence (first row) and for the predicted logit-transformed prevalence at two randomly selected locations (second and third rows).

```
R> acf(pred.MCML$samples[ind.S[2],], main = "")
R> plot(pred.MCML$samples[ind.S[2], ],
+  ylab = paste("Component n.", ind.S[2]), type = "l")
R> plot(ecdf(pred.MCML$samples[ind.S[2], 1:5000]), main = "")
R> lines(ecdf(pred.MCML$samples[ind.S[2], 5001:10000]),
+  col = 2, lty = "dashed")
```

In the first row of Figure 5, the target for prediction is the spatial average of logit-transformed prevalence, in the second and third rows the target is logit-transformed prevalence at each of two randomly sampled location. The three columns show: the autocorrelation plot of a thinned sequence of 10000 MCMC samples; the trace plot of these same 10000 samples; the empirical cumulative distribution functions of the first 5000 and the second 5000 of these 10000 samples. None of these plots show any evidence of non-convergence.

Bayesian analysis

For a Bayesian analysis of the *Loa loa* data, we use the following prior specification:

$$\begin{aligned} \phi &\sim \text{Uniform}(0, 8), \\ \log(\sigma^2) &\sim N(\cdot; 1, 25), \\ \log(\tau^2) &\sim N(\cdot; -3, 1), \\ \beta|\sigma^2 &\sim N(\cdot; 0, \sigma^2 100^2). \end{aligned}$$

In the **PrevMap** package, the `control.prior` function can be used to set a Gaussian prior on β and any required prior distribution for the covariance parameters σ^2, ϕ and τ^2. The arguments `beta.mean` and `beta.covar` are the mean vector and the covariance matrix of the Gaussian prior for β. Log-Gaussian and uniform priors can also be directly defined for each covariance parameter by using the corresponding arguments. For example, `log.normal.sigma2` and `uniform.sigma2` define log-Gaussian and uniform priors, respectively, for σ^2. In both cases a vector of length two must be provided. If the prior is log-Gaussian the two elements are the mean and standard deviation of the distribution on the log scale. If the prior is uniform the two elements are the lower and upper limits of the support of the uniform distribution.

```
R> cp <-control.prior(beta.mean = 0, beta.covar = 100^2,
+  log.normal.sigma2 = c(1,5),
+  uniform.phi  = c(0,8),
+  log.normal.nugget = c(-3,1))
```

If different priors are required for the covariance parameters, user-defined functions of the prior log-density can be specified through the arguments `log.prior.sigma2`, `log.prior.phi` and `log.prior.nugget`.

Control parameters for the MCMC algorithm (see Section 2.2) are specified with the function `control.mcmc.Bayes`.

```
R> mcmc.Bayes <- control.mcmc.Bayes(n.sim = 6000, burnin = 1000, thin = 1,
+  h.theta1 = 1, h.theta2 = 0.7, h.theta3 = 0.05,
+  L.S.lim = c(5,50), epsilon.S.lim = c(0.03,0.06),
```

```
+  start.beta  = -2.3, start.sigma2 = 2.6,
+  start.phi = 0.8, start.nugget = 0.05,
+  start.S = predict(fit.glm))
```

The arguments `h.theta1`, `h.theta2` and `h.theta3` are the starting values for the standard deviations of the Gaussian proposals; these are then tuned according to the adaptive scheme given by Equation 7. The control parameters for the Hamiltonian Monte Carlo procedure, used to update the random effects, are `L.S.lim` and `epsilon.S.lim`. These represent, respectively, the intervals used to randomly generate from a uniform distribution the number of steps and the step size in the *leapfrog* method at each iteration of the MCMC (see Section 2.2).

```
R> fit.Bayes <- binomial.logistic.Bayes(formula = NO_INF ~ 1,
+  units.m = ~ NO_EXAM,
+  coords = ~ LONGITUDE + LATITUDE,
+  data = loaloa, control.prior = cp,
+  control.mcmc = mcmc.Bayes, kappa = 0.5)
R>
R> summary(fit.Bayes, hpd.coverage = 0.95)

Bayesian binomial geostatistical logistic model
Call:
binomial.logistic.Bayes(formula = NO_INF ~ 1, units.m = ~ NO_EXAM,
    coords = ~ LONGITUDE + LATITUDE, data = loaloa, control.prior = cp,
    control.mcmc = mcmc.Bayes, kappa = 0.5)

                Mean   Median      Mode   StdErr HPD 0.025 HPD 0.975
(Intercept) -2.696243 -2.48606 -2.305288 1.827606 -7.253424 0.5964536

Covariance parameters Matern function (kappa = 0.5)

              Mean     Median       Mode     StdErr   HPD 0.025  HPD 0.975
sigma^2 7.66349058 5.27116856 3.28389063 5.86998256 1.650528734 20.5858584
phi     2.58509412 1.79584603 1.04951602 1.98215672 0.440133492  6.9920447
tau^2   0.05250712 0.04516296 0.02365498 0.03371813 0.003049963  0.1190124

Legend:
sigma^2 = variance of the Gaussian process
phi = scale of the spatial correlation
tau^2 = variance of the nugget effect
```

The above code fits a Bayesian BL model and returns summaries of the posterior distribution for each of the model parameters. In the output, high posterior density credible intervals are also computed, with associated coverage specified through the argument `hpd.coverage`.

```
R> par(mfrow = c(2,4))
R> autocor.plot(fit.Bayes, param = "beta", component.beta = 1)
```

```
R> autocor.plot(fit.Bayes, param = "sigma2")
R> autocor.plot(fit.Bayes, param = "phi")
R> autocor.plot(fit.Bayes, param = "tau2")
R> i <- sample(1:nrow(loaloa),4)
R> autocor.plot(fit.Bayes, param = "S", component.S = i[1])
R> autocor.plot(fit.Bayes, param = "S", component.S = i[2])
R> autocor.plot(fit.Bayes, param = "S", component.S = i[3])
R> autocor.plot(fit.Bayes, param = "S", component.S = i[4])
```

Autocorrelation plots can be obtained with the `autocor.plot` function, whose argument `param` specifies the model component for which the autocorrelation plot is required. If `param = "beta"`, then `component.beta` must be used to specify the component of the regression coefficients. To display autocorrelation plots for the random effect, then `param = "S"` and `component.S` must be either a positive integer indicating the component of the random effect, or `"all"` in order to display the autocorrelation for all components in a single plot. Using a similar syntax, the functions `trace.plot` and `dens.plot` are also available for visualization of trace-plots and kernel density estimates based on the posterior samples.

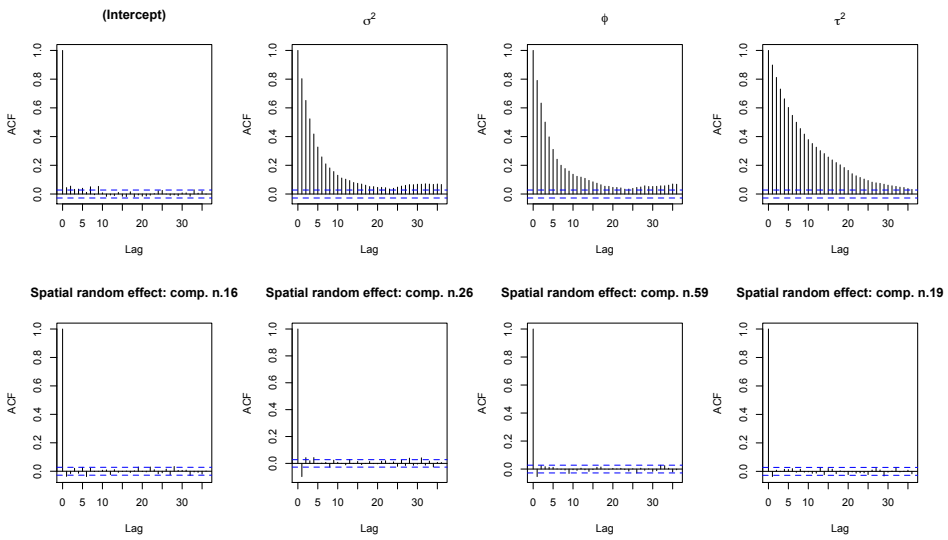

Figure 6: Autocorrelation plots for the posterior samples of β (the intercept), σ^2, ϕ, τ^2 and four randomly chosen components of the spatial random effect.

```
R> pred.Bayes <- spatial.pred.binomial.Bayes(fit.Bayes, grid.pred,
+  type = "marginal",
+  scale.predictions = "prevalence", quantiles = NULL,
+  standard.errors = TRUE, thresholds = 0.2,
+  scale.thresholds = "prevalence")
R>
R> par(mfrow = c(1,3))
R> plot(pred.Bayes, type = "prevalence", summary = "predictions",
```

```
+  zlim = c(0,0.45),
+  main = "Prevalence - predictions \n (Bayesian analysis)")
R> contour(pred.Bayes, type = "prevalence", summary = "predictions",
+  levels = c(0.05,0.1,0.2,0.3), add = TRUE)
R> plot(pred.Bayes, type = "prevalence", summary = "standard.errors",
+  zlim = c(0,0.3),
+  main = "Prevalence - standard errors \n (Bayesian analysis)")
R> contour(pred.Bayes, type = "prevalence",
+  summary = "standard.errors",
+  levels = c(0.05,0.1,0.15,0.2), add = TRUE)
R> plot(pred.Bayes, type = "exceedance.prob", zlim = c(0,1),
+  main = "Prevalence - exceedance probabilities \n
+  (Bayesian analysis)")
R> contour(pred.Bayes, type = "exceedance.prob",
+  levels = c(0.1,0.4,0.5,0.7), add = TRUE)
```

The function `spatial.pred.binomial.Bayes` generates spatial Bayesian predictions using the same syntax as `spatial.pred.binomial.MCML`. The resulting plots of the prevalence estimates, standard errors and exceedance probabilities are shown in Figure 4.

4. Example: simulated data

In this example, we use a simulated binomial data-set, available in the package as `data_sim`. For these data, a zero-mean Gaussian process was generated over a 30 by 30 grid covering the unit square, with parameters $\sigma^2 = 1$, $\phi = 0.15$ and $\kappa = 2$; the nugget effect was not included, hence $\tau^2 = 0$. Binomial observations, with 10 trials at each grid point and probabilities given by the anti-logit of the simulated values of the Gaussian process, constitute the variable `y` in the data. To illustrate the accuracy of the low-rank approximation, we analyse these data using three different grids covering the square $[-0.2, 1.2] \times [-0.2, 1.2]$ with 25, 100 and 225 spatial knots, respectively. By letting some knots lie outside of the unit square, we avoid the presence of edge-effects due to the restriction of the integral in Equation 9 to a sub-region of the real plane.

```
R> data("data_sim")
R> knots1 <- expand.grid(seq(-0.2,1.2, length = 5),
+  seq(-0.2,1.2, length = 5))
R> knots2 <- expand.grid(seq(-0.2,1.2, length = 10),
+  seq(-0.2,1.2, length = 10))
R> knots3 <- expand.grid(seq(-0.2,1.2, length = 15),
+  seq(-0.2,1.2, length = 15))
```

We use the MCML method to fit a BL model using both exact and approximate approaches. We then use the resulting binomial fits to generate spatial predictions of prevalence at each of the 900 sampling locations.

```
R> par0.exact <- c(0,1,0.15)
R> exact.mcmc <- control.mcmc.MCML(n.sim = 65000, burnin = 5000, thin = 12,
```

```
+  h = 1.65/(nrow(data_sim)^(1/6)))
R> system.time(fit.MCML.exact <- binomial.logistic.MCML(y ~ 1,
+  units.m = ~ units.m, coords = ~ x1 + x2,
+  data = data_sim, par0 = par0.exact,
+  start.cov.pars = 0.15,
+  control.mcmc = exact.mcmc,
+  kappa = 2, fixed.rel.nugget = 0, method = "nlminb",
+  plot.correlogram = FALSE))

    user   system  elapsed
 2401.530  297.871 2714.146

R> par0.lr <- c(-0.219294,0.97945,0.21393)
R> lr.mcmc <- control.mcmc.MCML(n.sim = 65000, burnin = 5000, thin = 12,
+  h = 1.65/(nrow(knots1)^(1/6)))
R> system.time(fit.MCML.lr1 <- binomial.logistic.MCML(y ~ 1,
+  units.m = ~ units.m, coords = ~ x1 + x2,
+  data = data_sim, par0 = par0.lr,
+  start.cov.pars = par0.lr[3], control.mcmc = lr.mcmc,
+  low.rank = TRUE, knots = knots1, kappa = 2,
+  method = "nlminb", plot.correlogram  = FALSE))

   user  system elapsed
 72.893   2.785  77.157

R> lr.mcmc$h <- 1.65/(nrow(knots2)^(1/6))
R> par0.lr <- c(-0.017333,0.16490,0.16971)
R> system.time(fit.MCML.lr2 <- binomial.logistic.MCML(y ~ 1,
+  units.m = ~ units.m, coords = ~ x1 + x2,
+  data = data_sim, par0 = par0.lr,
+  start.cov.pars = par0.lr[3], control.mcmc = lr.mcmc,
+  low.rank = TRUE, knots = knots2, kappa = 2,
+  method = "nlminb", plot.correlogram = FALSE))

   user  system elapsed
172.864  20.973 194.625

R> lr.mcmc$h <- 1.65/(nrow(knots3)^(1/6))
R> par0.lr <- c(-0.031759,0.30572, 0.18854)
R> system.time(fit.MCML.lr3 <- binomial.logistic.MCML(y ~ 1,
+  units.m = ~ units.m, coords = ~ x1 + x2,
+  data = data_sim, par0 = par0.lr,
+  start.cov.pars = par0.lr[3], control.mcmc = lr.mcmc,
+  low.rank = TRUE, knots = knots3, kappa = 2,
+  method = "nlminb", plot.correlogram = FALSE))

   user  system elapsed
407.376  14.397 423.235
```

To fit a low-rank approximation, we only need to specify `low.rank = TRUE` and define the set of spatial knots through the argument `knots`. For parameter estimation, this approach was about 35, 13 and 6 times faster than the exact method when using 5, 100 and 225 knots, respectively.

```
R> par.hat <- coef(fit.MCML.exact)
R> Sigma.hat <- varcov.spatial(coords = data_sim[c("x1","x2")],
+  cov.pars = par.hat[2:3], kappa = 2)$varcov
R> mu.hat <- rep(par.hat[1], nrow(data_sim))
R> system.time(S.cond.sim <- Laplace.sampling(mu = mu.hat,
+  sigma = Sigma.hat,
+  y = data_sim$y, units.m = data_sim$units.m,
+  control.mcmc = exact.mcmc, plot.correlogram = FALSE))

    user   system  elapsed
 1275.890  134.015 1393.457

R> prevalence.sim <- exp(S.cond.sim$samples)/(1 + exp(S.cond.sim$samples))
R> prevalence.exact <- apply(prevalence.sim,2, mean)
R>
R> lr.mcmc$h <- 1.65/(nrow(knots1)^(1/6))
R> system.time(pred.MCML.lr1 <- spatial.pred.binomial.MCML(fit.MCML.lr1,
+  grid.pred = data_sim[c("x1","x2")], control.mcmc = lr.mcmc,
+  type = "joint", scale.predictions = "prevalence",
+  plot.correlogram = FALSE))

   user  system elapsed
 34.571   2.954  37.664

R> lr.mcmc$h <- 1.65/(nrow(knots2)^(1/6))
R> system.time(pred.MCML.lr2 <- spatial.pred.binomial.MCML(fit.MCML.lr2,
+  grid.pred = data_sim[c("x1","x2")], control.mcmc = lr.mcmc,
+  type = "joint", scale.predictions = "prevalence",
+  plot.correlogram = FALSE))

   user  system elapsed
 75.035   6.008  81.399

R> lr.mcmc$h <- 1.65/(nrow(knots3)^(1/6))
R> system.time(pred.MCML.lr3 <- spatial.pred.binomial.MCML(fit.MCML.lr3,
+  grid.pred = data_sim[c("x1","x2")], control.mcmc = lr.mcmc,
+  type = "joint", scale.predictions = "prevalence",
+  plot.correlogram = FALSE))

   user  system elapsed
169.352  21.975 192.218
```

```
R> par(mfrow = c(2,2), mar = c(3,4,3,4))
R> r.exact <- rasterFromXYZ(cbind(data_sim[, c("x1","x2")],
+  prevalence.exact))
R> plot(r.exact, zlim = c(0,1), main = "Exact method")
R> contour(r.exact, levels = seq(0.1,0.9,0.1), add = TRUE)
R>
R> plot(pred.MCML.lr1,"prevalence","predictions", zlim = c(0,1),
+  main = "Low-rank: 25 knots")
R> contour(pred.MCML.lr1,"prevalence","predictions", zlim = c(0,1),
+  levels = seq(0.1,0.9,0.1), add = TRUE)
R>
R> plot(pred.MCML.lr2,"prevalence","predictions", zlim = c(0,1),
+  main = "Low-rank: 100 knots")
R> contour(pred.MCML.lr2,"prevalence","predictions", zlim = c(0,1),
+  levels = seq(0.1,0.9,0.1), add = TRUE)
R>
R> plot(pred.MCML.lr3,"prevalence","predictions", zlim = c(0,1),
+  main = "Low-rank: 225 knots")
R> contour(pred.MCML.lr3,"prevalence","predictions", zlim = c(0,1),
+  levels = seq(0.1,0.9,0.1), add = TRUE)
```

The above code generates and plots spatial predictions of prevalence at the 900 sample locations using exact and approximate methods. In the exact case, we first use the function `Laplace.sampling` to sample from the predictive distribution of $T^\top = (T_1, \ldots, t_{900})$, where T_i is given by (1). The arguments `mu` and `Sigma` of this function represents the mean vector and covariance matrix of the unconditional distribution of T. We post-process the simulation output to obtain estimates of prevalence by using the anti-logit transformation of each simulated sample and taking the average of these values at each sampling location. Figure 7 shows the resulting estimates of prevalence. As expected, the accuracy of the low-rank approximation increases as more knots are included: while using 5 knots leads to a computationally fast but poor approximation, 100 and 225 knots give progressive improvements in accuracy which might be considered sufficient in practice.

5. Conclusions and future developments

We have illustrated the use the **PrevMap** package for geostatistical modelling of spatially referenced prevalence data. The package is intended to be compatible with the existing **geoR** and **geoRglm** packages, but with increased functionality. By comparison with these earlier packages, **PrevMap** provides more accurate numerical procedures for maximum likelihood estimation of the geostatistical linear and BL models, as well as routines for evaluation of the profile likelihood. Computationally faster approximations of the likelihood function for geostatistical BL models can be obtained using the Laplace approximation (LA). However, the resulting parameter estimates can be substantially biased in the case of binomial observations with small denominators (Joe 2008), whereas the MCML method delivers asymptotically unbiased estimates.

For likelihood-based inference we have used a Langevin-Hastings MCMC algorithm because

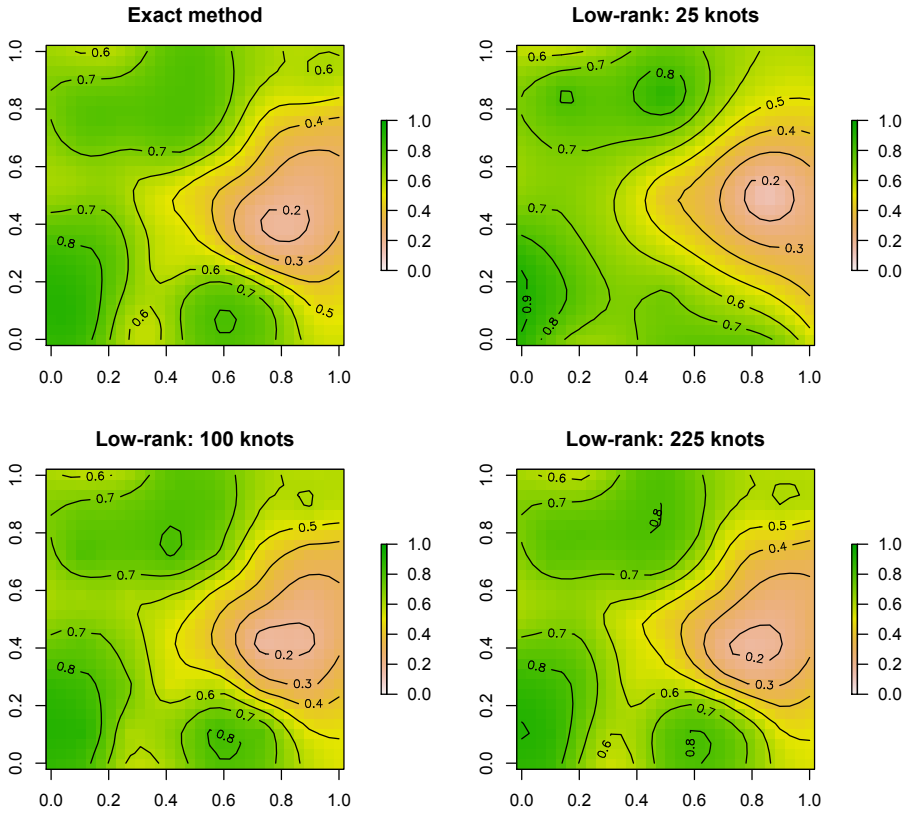

Figure 7: Images of the estimated surfaces of prevalence obtained for the simulated data using the exact method (upper left panel) and the low-rank approximation using 25 (upper right panel), 100 (lower left panel) and 225 (lower right panel) spatial knots.

the availability of optimal scaling results makes it easier to tune than the Hamiltonian MCMC. However, for Bayesian inference where model parameters are also updated at each iteration, the required computation of the mode of the random effects conditional distribution (see Section 2.1) would have been computationally too demanding. For Bayesian analysis, we have therefore implemented an efficient Hamiltonian MCMC scheme that updates the random effects on their original scale and allows a more flexible prior specification for the model parameters.

The sampling-based approach to inference allows the user to access any predictive target through post-processing of the samples from the joint predictive distribution of the values of the latent field at all prediction points. Within the package, the user can specify whether marginal or joint predictions are required for different predictive targets: logit, prevalence, odds and exceedance probabilities. Software based on analytical approximations to the marginal predictive distributions, such as the **geostatsp** package that uses Integrated nested Laplace approximations (Rue, Martino, and Chopin 2009), cannot routinely calculate predictive distributions for arbitrary functionals of the latent field.

The package includes several functions for automatic post-processing of the results, such as the diagnostic plots illustrated in Figure 5. As is the case for any MCMC application, these can only reveal or fail to reveal non-convergence rather than guarantee convergence, but are nevertheless useful as partial checks. We therefore considered it important to make them easily accessible to users.

The accuracy of the low-rank approximations that are incorporated into the package is context-specific. However, used with care they offer computationally efficient procedures for analysing large data-sets. The **spBayes** package implements a low-rank procedure based on Gaussian predictive process models (Banerjee, Gelfand, Finley, and Sang 2008). In this approach, the latent field $S(x)$ in Equation 1 is replaced by the conditional expectation of $S(x)$ given $S(\tilde{x}_i)$ for $i = 1, \ldots, r < n$, where $\tilde{x}_i$ is a set of pre-defined spatial knots. This is particularly useful and computationally advantageous when spatial interpolation is the sole objective of the analysis. In this context, other computationally efficient procedures could also be considered, such as low-rank spline smoothers (Wood 2003). However, for applications that involve a range of inferential objectives, including both spatial prediction and estimation of covariate effects, it is desirable that the low-rank method approximates the same probabilistic model that would be used were computational burden not an issue, rather than changing the model specification. For this reason, we consider our version of low-rank approximation (Section 2.4) to be more suitable for disease mapping applications where, typically, the objectives include inference for regression parameters, both to assess the importance of hypothesised risk-factors and to enable spatial prediction under a range of scenarios. A specific example is the construction of predictive maps for malaria under different climate scenarios, or before and after widespread distribution of insecticide-treated bed-nets.

Another feature not illustrated in the present paper is the possibility of fitting a BL model to prevalence data from household surveys so as to include information at both household and individual level. More specifically, let i and j identify the i-th household and the j-th individual within that household; in this case the linear predictor is

$$\log\left\{\frac{p_{ij}}{1-p_{ij}}\right\} = d_{ij}^{\top}\beta + S(x_i) + Z_i, \tag{14}$$

where the random effects are now defined at household level. With the exception of the

geostatsp package, the model in Equation 14 can not be fitted in the other packages reported in Table 1 other than by replacing individual-level explanatory variables d_{ij} by their location-level averages. However, this would invalidate inferences on the regression coefficients β by introducing ecological bias (Wakefield and Lyons 2010).

Possible extensions of the package include the implementation of functions for spatio-temporal analyses, for geostatistical modelling of zero-inflated data, for combining data from multiple spatially referenced prevalence surveys (Giorgi, Sesay, Terlouw, and Diggle 2015) and for open-ends count data through a Poisson log-linear formulation. We will report these extensions separately in due course.

References

Banerjee S, Gelfand AE, Finley AO, Sang H (2008). "Gaussian Predictive Process Models for Large Spatial Data Sets." *Journal of the Royal Statistical Society B*, **70**(4), 825–848.

Brown PE (2015). "Model-Based Geostatistics the Easy Way." *Journal of Statistical Software*, **63**(12), 1–24. URL http://www.jstatsoft.org/v63/i12/.

Christensen O, Ribeiro PJ (2002). "geoRglm - A Package for Generalised Linear Spatial Models." *R-NEWS*, **2**(2), 26–28. ISSN 1609-3631, URL http://cran.R-project.org/doc/Rnews.

Christensen OF (2004). "Monte Carlo Maximum Likelihood in Model-Based Geostatistics." *Journal of Computational and Graphical Statistics*, **13**(3), 702–718.

Christensen OF, Roberts GO, Sköld M (2006). "Robust Markov Chain Monte Carlo Methods for Spatial Generalized Linear Mixed Models." *Journal of Computational and Graphical Statistics*, **15**(1), 1–17.

Diggle P, Thomson M, Christensen O, Rowlingson B, Obsomer V, Gardon J, Wanji S, Takougang I, Enyong P, Kamgno J, Remme H, Boussinesq M, Molyneux D (2007). "Spatial Modelling and Prediction of Loa Loa Risk: Decision Making Under Uncertainty." *Annals of Tropical Medicine and Parasitology*, **101**(6), 499–509.

Diggle PJ, Ribeiro PJ (2007). *Model Based Geostatistics*. Springer-Verlag, New York.

Diggle PJ, Tawn JA, Moyeed RA (1998). "Model-based Geostatistics (with discussion)." *Applied Statistics*, **47**(3), 299–350.

Finley AO, Banerjee S, Carlin BP (2007). "spBayes: An R Package for Univariate and Multivariate Hierarchical Point-Referenced Spatial Models." *Journal of Statistical Software*, **19**(4), 1–24. URL http://www.jstatsoft.org/v19/i04/.

Finley AO, Banerjee S, Gelfand AE (2015). "spBayes for Large Univariate and Multivariate Point-Referenced Spatio-Temporal Data Models." *Journal of Statistical Software*, **63**(13), 1–28. URL http://www.jstatsoft.org/v63/i13/.

Geyer CJ (1994). "On the Convergence of Monte Carlo Maximum Likelihood Calculations." *Journal of the Royal Statistical Society B*, **56**(1), 261–274.

Geyer CJ (1996). "Estimation and Optimization of Functions." In W Gilks, S Richardson, D Spiegelhalter (eds.), *Markov Chain Monte Carlo in Practice*, pp. 241–258. London: Chapman and Hall.

Geyer CJ (1999). "Likelihood Inference for Spatial Point Processes." In OE Barndorff-Nielsen, W SKendall, MNM van Lieshout (eds.), *Stochastic Geometry, Likelihood and Computation*, pp. 79–140. Boca Raton, FL: Chapman and Hall/CRC.

Geyer CJ, Thompson EA (1992). "Constrained Monte Carlo Maximum Likelihood for Dependent Data." *Journal of the Royal Statistical Society B*, **54**(3), 657–699.

Giorgi E, Sesay SSS, Terlouw DJ, Diggle PJ (2015). "Combining Data From Multiple Spatially Referenced Prevalence Surveys Using Generalized Linear Geostatistical Models." *Journal of the Royal Statistical Society A*, **178**(2), 445–464.

Henningsen A, Toomet O (2011). "maxLik: A Package for Maximum Likelihood Estimation in R." *Computational Statistics*, **26**(3), 443–458.

Higdon D (1998). "A Process-Convolution Approach to Modeling Temperatures in the North Atlantic Ocean." *Environmental and Ecological Statistics*, **5**(2), 173–190.

Higdon D (2002). "Space and Space-Time Modeling Using Process Convolutions." In CW Anderson, V Barnett, PC Chatwin, AH El-Shaarawi (eds.), *Quantitative methods for current environmental issues*, pp. 37–56. Springer-Verlag, New York.

Joe H (2008). "Accuracy of Laplace Approximation for Discrete Response Mixed Models." *Computational Statistics & Data Analysis*, **52**(12), 5066–5074.

Matérn B (1986). *Spatial Variation*. Second edition. Springer-Verlag, Berlin.

Neal RM (2011). "MCMC using Hamiltonian Dynamics." In S Brooks, A Gelman, G Jones, XL Meng (eds.), *Handbook of Markov Chain Monte Carlo*, chapter 5, pp. 113–162. Chapman & Hall, CRC Press.

Ribeiro PJ, Diggle PJ (2001). "geoR: a Package for Geostatistical Analysis." *R-NEWS*, **1**(2), 14–18. ISSN 1609-3631, URL http://CRAN.R-project.org/doc/Rnews/.

Roberts GO, Rosenthal JS (2001). "Optimal Scaling for Various Metropolis-Hastings Algorithms." *Statistical Science*, **16**(4), 351–367.

Rue H, Martino S, Chopin N (2009). "Approximate Bayesian Inference for Latent Gaussian Models by Using Integrated Nested Laplace Approximations." *Journal of the Royal Statistical Society B*, **71**(3), 319–392.

Stanton MC, Diggle PJ (2013). "Geostatistical Analysis of Binomial Data: Generalised Linear or Transformed Gaussian Modelling?" *Environmetrics*, **24**(3), 158–171.

Wakefield J, Lyons H (2010). "Spatial aggregation and the ecological fallacy." In AE Gelfand, PJ Diggle, M Fuentes, P Guttorp (eds.), *Handbook of Spatial Statistics*, pp. 541–558. CRC press.

Wood SN (2003). "Thin Plate Regression Splines." *Journal of the Royal Statistical Society B*, **65**(1), 95–114.

Zhang H (2004). "Inconsistent Estimation and Asymptotically Equal Interpolations in Model-Based Geostatistics." *Journal of the American Statistical Association*, **99**(465), 250–261.

Affiliation:

Emanuele Giorgi
Lancaster Medical School
Faculty of Health and Medicine
Lancaster University
LA1 4YB Lancaster, UK
E-mail: e.giorgi@lancaster.ac.uk
URL: http://www.lancs.ac.uk/staff/giorgi

Journal of Statistical Software http://www.jstatsoft.org/
published by the American Statistical Association http://www.amstat.org/
Volume VV, Issue II *Submitted:* yyyy-mm-dd
MMMMMM YYYY *Accepted:* yyyy-mm-dd