Proceedings of the 3rd International Workshop on Distributed Statistical Computing (DSC 2003)
March 20–22, Vienna, Austria ISSN 1609-395X
Kurt Hornik, Friedrich Leisch & Achim Zeileis (eds.)
http://www.ci.tuwien.ac.at/Conferences/DSC-2003/

`geoR` and `geoRglm`: Software for Model-Based Geostatistics

Paulo J. Ribeiro Jr.
Univ. Federal do Paraná, Brasil

Ole F. Christensen
Aarhus Universitet, Denmark

Peter J. Diggle
Lancaster University, UK

Abstract

The packages `geoR` and `geoRglm` are contributed packages to the statistical software system R, implementing methods for model-based geostatistical data-analysis. In this paper we focus on the capabilities of the packages, the computational implementation and related issues, and indicate directions for future developments.

`geoR` implements methods for Gaussian and transformed Gaussian models. The package includes functions and methods for reading and preparing the data, exploratory analysis, inference on model parameters including variogram based and likelihood based methods, and spatial interpolation. The generic term *kriging* is used in the geostatistical literature in connection with several methods of spatial interpolation/prediction. `geoR` implements classical "kriging flavours", simple, ordinary, universal and external trend kriging and algorithms for conditional simulation. The package also implements Bayesian methods which take the parameter uncertainty into account when predicting at specified locations.

The package `geoRglm` is an extension of `geoR` for inference in generalised linear spatial models using Markov chain Monte Carlo (MCMC) methods. `geoRglm` implements conditional simulation and Bayesian inference for the Poisson and Binomial generalised linear models.

1 Introduction

Geostatistics is now recognised as one of the main branches of spatial statistics Cressie (1993). It deals with modelling and inference for spatially continuous phenomena, $S(x)$, where data $Y_1, \ldots, Y_n$ are obtained by sampling at a finite number of locations $x_1, \ldots, x_n$. In simple cases $Y_i = S(x_i)$. More generally, Y_i is stochastically dependent on $S(x_i)$, and can often be considered as a "noisy" version of an underlying "signal" $S(x_i)$.

The main motivation behind the packages `geoR` (Ribeiro Jr and Diggle, 2001) and `geoRglm` (Christensen and Ribeiro Jr, 2002) is the implementation of model-based methods geostatistical data-analysis (Diggle, Tawn, and Moyeed, 1998), by which we mean an approach to spatial prediction problems based on explicitly declared stochastic models and associated formal methods of statistical inference. Likelihood based methods are therefore implemented, in particular maximum likelihood estimation and Bayesian inference. This paper discusses the usage of the packages. The underlying theory and algorithms for its implementation are described in Diggle, Ribeiro Jr, and Christensen (2003) and Diggle and Ribeiro Jr (2004). The package web pages at `http://www.est.ufpr.br/geoR` and `http://www.maths.lancs.ac.uk/~christen/geoRglm` provide supplementary information, including illustrative sessions and tutorials.

Section 2 describes geostatistical data objects of the class `geodata`. Tools for descriptive spatial analysis are shown in Section 3. The models underlying the packages are presented in Section 4. Sections 5 to 7 discuss methods implemented in `geoR` for inference and prediction under the Gaussian and transformed Gaussian models. Section 8 presents inference for Binomial and Poisson models implemented in `geoRglm`.

Most of the functionality in the two packages is presented here. We refer the reader to the web-pages, help-files for the functions, and list of references for further details. The data sets `ca20`, `b64` and `rongelap` included in the distributions illustrate the usage of the functions. The versions at the time of writing are `geoR_1.3-12` and `geoRglm_0.7-0`.

2 Preparing data

In their most basic format geostatistical data consists of n pairs $(y_i, x_i), i = 1 \ldots n$, where each pair is a location x_i, and the observed data value y_i measured at this location. We will assume here that x_i is a 2-dimensional vector. Therefore, a typical and minimal data structure would consist of a vector and a matrix with two columns.

The class `geodata` is defined for objects containing geostatistical data. A `geodata` object is a list with at least 2 components: `coords` which is an $n \times 2$ matrix and `data` which is an n-dimensional vector. The function `as.geodata` is used to convert data from a data-frame format to a `geodata` object. The function `read.geodata` is used to create a `geodata` object directly from an external text file, which is done by calling internally first the standard R function `read.table` and then `as.geodata`. For example, the command `foo <- read.geodata("foo.txt")` reads data, assuming the existence of an ASCII file `foo.txt` in the working directory with three columns, the first two containing the coordinates and the third the data values. Further arguments such as headers, character separation, etc, can be passed to `read.table` using the `...` mechanism. If the columns are not ordered as assumed above, the arguments `coords.col` and `data.col` can be used to indicate the column numbers. The function call above assumes default values and is equivalent to `foo <- read.geodata("foo.txt", coords.col=1:2, data.col=3)`.

Storing data in a `geodata` object is convenient to facilitate the usage of other functions in geoR and geoRglm, but it is not required since arguments coords and data are available in most of the functions.

The data component of a geodata object can alternatively be a matrix to accommodate multivariate response data. In this case, each column contains the values of one variable. However, just a few functions such as variog make use of this data format to compute variograms for all variables at once. Multivariate methods like cross-covariograms and co-kriging are not implemented in geoR/geoRglm. We refer to the package gstat (Pebesma, 2003) for an implemetation of these methods. Another more usual form of multivariate data is to have measurements of one or more covariates at the data locations. In this case the argument coords.col is used to indicate the column number(s) containing the value(s) of the covariate(s) and the geodata object will also have another element named covariate.

The data preparation functions also include options for handling NA's using the optional argument na.action. Using the argument realisations we can specify a column, containing numbers indicating different realisations of the process. This can be used for example to accommodate data collected at different times.

3 Descriptive analysis

Descriptive tools for spatial data allow visualisation of the data variability over the region and exploration of the spatial correlation. Methods for displaying geostatistical data are provided for functions such as plot and points. Furthermore smoothing techniques, as implemented in the R function loess, can be used to show the main spatial pattern of the data.

A method for summary computes the range of the coordinates and borders (if present) and a standard summary for the data and covariates. For instance the command summary(ca20) would give a summary of the data set ca20.

Typically, an analysis would start by plotting the point locations, the data values against the coordinates, and a histogram of the data. A 2 × 2 display with these plots is produced by a method for plot. For example, the command plot(ca20) will display the ca20 data.

Data transformation is implemented through the argument lambda which takes a numerical value for the parameter of the Box-Cox family of transformations (Box and Cox, 1964). If provided to the plot function, the transformed variable is shown in the plots. For instance, plot(ca20, lambda=0) produces plots of the logarithm of the data.

A common issue in geostatistical data analysis is *trend-removal*. If the argument trend is provided to the plot function, a linear regression is fitted using the R function lm and the residuals are used to produce the plot. Input for this argument is quite flexible: it can be a vector, a matrix or data-frame with the covariate values, a formula, or a string "1st" or "2nd" which indicates first or second order polynomials on the coordinates, respectively.

Finally the optional argument borders takes a matrix with the coordinates of the borders of the region which are added to the first plot in the graphical display. Figure 1 shows the output of the command plot(ca20, trend=~altitude+region, bor=borders) which plots residuals of a regression against the covariates altitude and region available for this data-set.

Another method to visualise the data is provided for points, which besides the arguments lambda, trend and borders described above, takes other arguments to

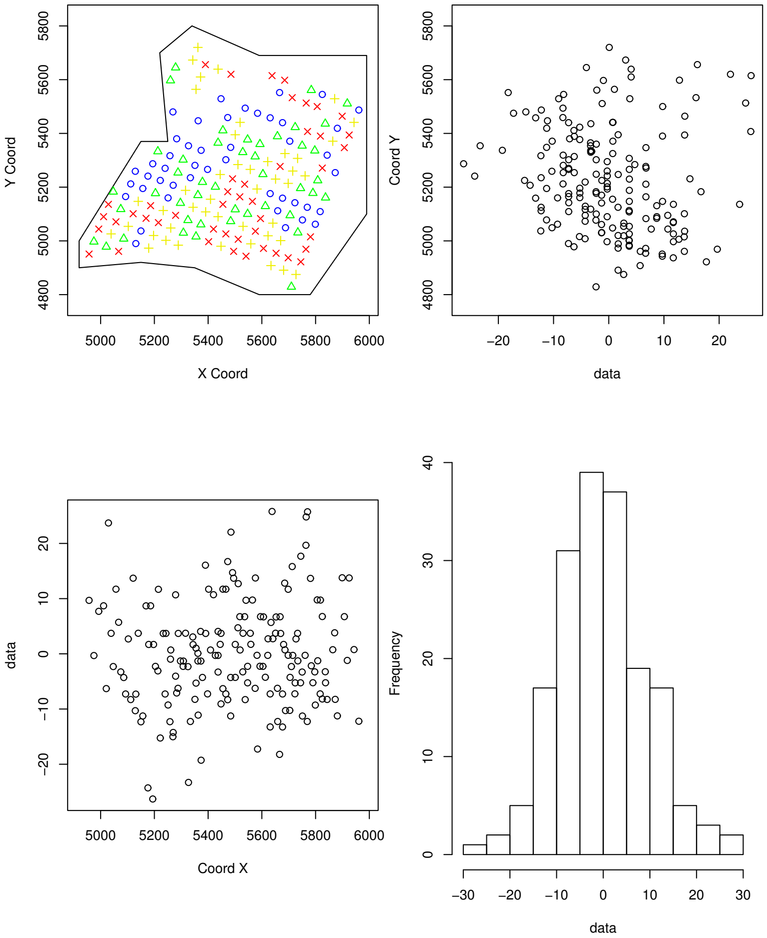

Figure 1: A typical output of the function `plot.geodata` for the `ca20` data-set.

control the colours and sizes of the points in the plot. Making these proportional to the data, ranks or specific quantiles of the data, allows for a visualisation of the spatial pattern. The plots in Figure 2 are produced with the commands:

```
> points(ca20, borders=borders, xl="W-E", yl="S-N")
> points(ca20, borders=borders, xl="W-E", yl="S-N", pt.d="qui",
         cex.max=1, cex.min=1, col="gray")
```

A common tool to describe the spatial dependence in geostatistics is the *empirical variogram* which describes the spatial association as a function of the separation distance, and is computed as follows. For each pair of data points (x_i, y_i) and (x_j, y_j) we compute $u = ||x_i - x_j||$, the distance between x_i and x_j, and $v = (y_i - y_j)^2/2$. A scatterplot of v against u for all pairs is the *variogram cloud*. In practice it is common to group the points in classes of distances ("bins"), averaging the corresponding u and v values. The averaged v's results in the *binned variogram* or *sample*

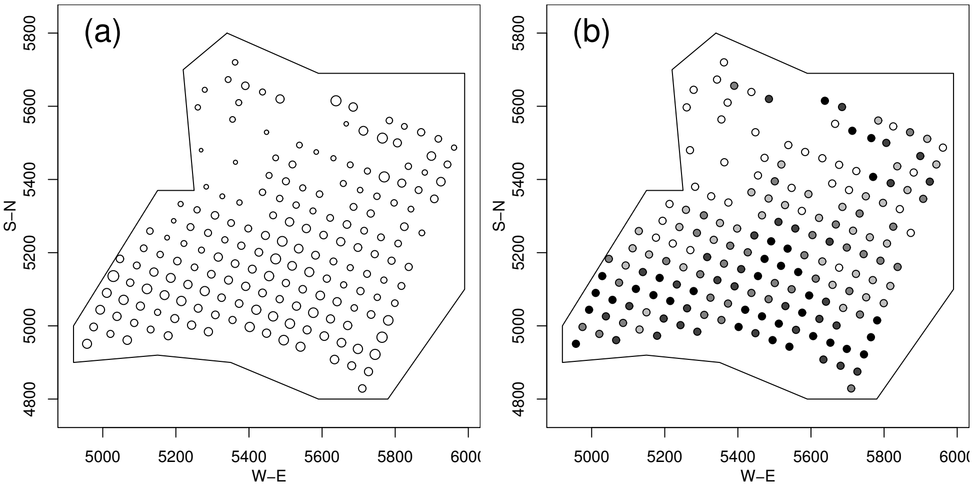

Figure 2: Graphics produced by the function `points.geodata` for the `ca20` data-set with points sizes (a) and shades of gray (b) proportional to the data.

variogram

$$\hat{\gamma}(u) = \frac{1}{2|N_u|} \sum_{(i,j)\in N_u} (y_i - y_j)^2, \tag{1}$$

where u is a distance class, N_u are the pairs in this class, and $|N_u|$ is the number of pairs in N_u.

The function `variog` is used to compute empirical variograms. It can produce a variogram cloud or a binned variogram and can either use the estimator (1) or the Hawkins-Cressie's modulus estimator (Cressie and Hawkins, 1980). The arguments `lambda` and `trend` can be used for data transformation and trend removal as described in Section 3. There are also arguments to define the bins. By default the function computes an omnidirectional variogram but the arguments `direction` and `tolerance` can be used to specify directional variograms. The function `variog4` automatically produces variograms for four different directions. The commands below produce the plots shown in Figure 3.

```
> ca20.v <- variog(ca20, max.dist=600, trend=~altitude+area)
> plot(ca20.v)
> plot(variog4(ca20, max.dist=600, trend=~altitude+area))
```

An object computed by the function `variog` is of the class `variogram`, and there are methods for `plot` and `lines` with options which include making point sizes proportional to the number of pairs and scaling the variogram, among others.

Sometimes it is useful to plot several variograms together, either to compare several simulations from a model, or to compare variograms of different variables. If a matrix is provided to the `data` argument, the function `variog` computes variograms for each column and a method for `plot` draws all of these in the same plot. For example, the code below, where `grf` is a function for simulating from a geostatistical model, produces the plot in the left hand panel of Figure 4.

```
> sim <- grf(100, cov.pars=c(1, 0.15), nsim=10)
> plot(variog(sim, max.dist=0.7))
```

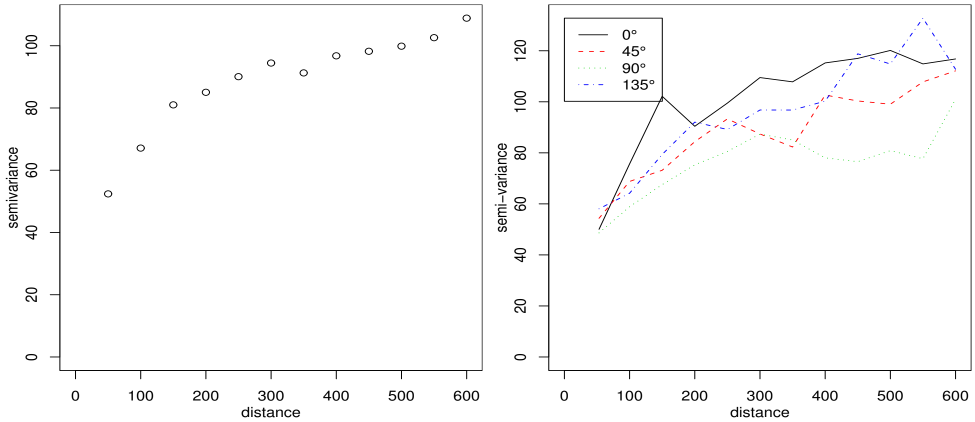

Figure 3: Empirical variograms computed by `variog` and `variog4`.

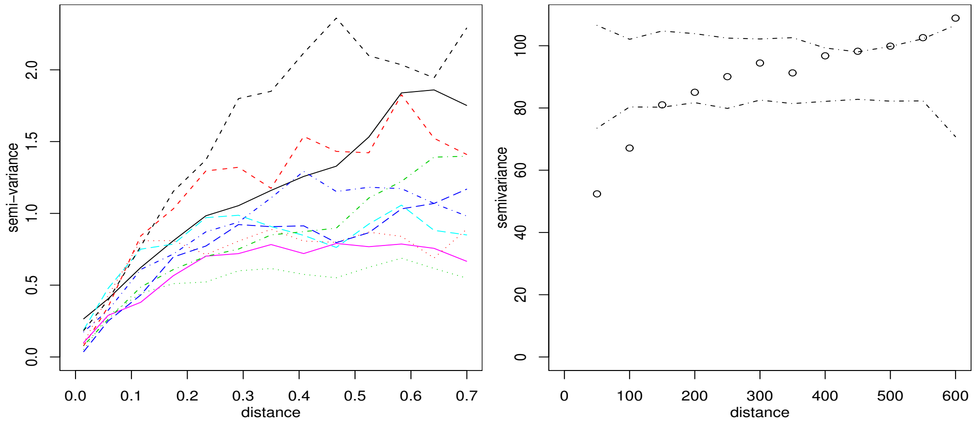

Figure 4: Some extra functionalities of the function `variog`: multiple variograms (left) and variogram envelopes for the `ca20` data (right).

A simple Monte Carlo test based on the variogram can be used to check for evidence of spatial correlation. Under the null hypothesis of no spatial correlation we can exchange the data values across the locations. Therefore to perform the test: (i) permute the data locations, (ii) for each permutation compute the variogram, (iii) calculate variogram "envelopes" using minimum and maximum values at each bin, (iv) plot the variogram of the original data and check whether it lies inside the envelopes. The function `variog.mc.env` is designed to perform this test as illustrated in the commands below and in the right hand panel of Figure 4.

```
> ca20.env <- variog.mc.env(ca20, obj=ca20.v)
> plot(ca20.v, env=ca20.env)
```

4 The geostatistical model

The model underlying both packages assumes that measurements represent a noisy version of a latent *signal process* which describes the variability of the random variable over the area. The typical goal of a geostatistical analysis is to predict either the signal over the area or some quantity which can be written as a functional of the signal.

We specify the geostatistical model as follows. Let $S(\cdot) = \{S(x) : x \in A\}$ be a Gaussian stochastic process with $\mathrm{E}[S(x)] = \sum_{j=1}^{p} f_j(x)\beta_j$, $\mathrm{Var}[S(x)] = \sigma^2$ and correlation function $\rho(u) = \mathrm{Corr}[S(x), S(x')]$ where $u = \|x - x'\|$ is the Euclidean distance between x and x'. Assume that $Y_1, \ldots, Y_n$ are conditionally independent given $S(\cdot)$, with conditional expectations μ_i and $h(\mu_i) = S(x_i), i = 1, \ldots, n$, for a known link function h. The signal process is given by $\{h^{-1}(S(x)) : x \in A\}$. To complete the model specification above we need to define the conditional distribution of Y_i given $S(\cdot)$, and the link function h. The package `geoR` implements methods which are appropriated when the conditional distribution is Gaussian and the link is the identity function with options to include a transformation of the Y-variable chosen within the Box-Cox family. The Gaussian model is specified by setting `lambda = 1` which is the default option for all functions in `geoR`. The package `geoRglm` implements two types of models: the Poisson model with link function in the Box-Cox class, and the Binomial logistic model. Table 4 summarises the models which are implemented at the time of writing.

Table 1: Models implemented by the packages `geoR` and `geoRglm`

Model	Sampling distribution	h(μ_i)	package
Gaussian	$[Y_i \| S(\cdot)] \sim \mathrm{Normal}(\mu_i, \tau^2)$	μ_i	`geoR`
Transf. Gaussian	$[(Y_i^\lambda - 1)/\lambda \| S(\cdot)] \sim \mathrm{Normal}(\mu_i, \tau^2)$	μ_i	`geoR`
Poisson	$[Y_i \| S(\cdot)] \sim \mathrm{Poisson}(\mu_i)$	Box-Cox(μ_i)	`geoRglm`
Binomial	$[Y_i \| S(\cdot)] \sim \mathrm{Binomial}(\mathrm{N}, \mu_i)$	logit(μ_i)	`geoRglm`

The correlation function $\rho(u)$ is specified by a parametric model. There are several correlation models currently implemented and documented in `cov.spatial`. The default is the Matérn model given by

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

where $\kappa > 0$ and $\phi > 0$ are model parameters, and K_κ denotes a modified Bessel function of order κ. Special cases of this family include the *exponential* correlation function, $\rho(u) = \exp(-u/\phi)$, when $\kappa = 0.5$, and the *squared exponential* or *Gaussian* correlation function, $\rho(u) = \exp(-(u/\tilde{\phi})^2)$, when $\phi = \tilde{\phi}/(2\sqrt{\kappa + 1})$ and $\kappa \to \infty$. The Matérn family is particularly attractive because the value of the parameter κ controls the smoothness of the underlying signal process.

The model described above is *isotropic* in the sense that the correlation depends only on the separation distance, but not on the orientation. A generalisation is given by the *geometric anisotropic* model which is isotropic only after some rotation and stretching of the original coordinates. This adds two extra parameters to the correlation function, the anisotropy angle ψ_A and anisotropy ratio ψ_R.

The possible model parameters can therefore be grouped as follows: transformation (λ), mean (β's), variance (σ^2, τ^2) and correlation ($\phi, \kappa, \psi_A, \psi_R$) parameters.

5 Inference for the Gaussian model

The function `likfit` is designed to find maximum likelihood estimates for the parameters of the (transformed) Gaussian model described in Section 4. The log-likelihood function is obtained from the density of the multivariate Gaussian:

$$\begin{aligned} l(\beta, \tau^2, \sigma^2, \phi, \kappa) \quad \propto \quad & -0.5\{\log|(\sigma^2 R + \tau^2 I)| \\ & +(y - F\beta)^{\mathrm{T}}(\sigma^2 R + \tau^2 I)^{-1}(y - F\beta)\}. \end{aligned} \tag{2}$$

As mentioned in the previous section, generalisations add parameters to the model. We refer to Diggle et al. (2003) and Christensen, Diggle, and Ribeiro Jr (2001) for further details.

Numerical optimisation of (3) is required to calculate the maximum likelihood estimates of the model parameters. The function `likfit` implements this using the R optimisation function `optim`. There are options to fix values for some of the model parameters $(\tau^2, \lambda, \psi_A, \psi_R)$. The output is an object of the class variomodel and there are associated methods for `summary`, `print` and `lines`. Along with the parameter estimates, the output includes the values of the maximised likelihood, AIC, BIC, fitted values and residuals. Alternatively, restricted maximum likelihood estimates (REML) can be obtained by setting the argument `method = "REML"`. The usage of `likfit` is illustrated below where, for brevity, we ommit the output of `summary`.

```
> ca20.ml <- likfit(ca20, trend=~altitude+area, ini=c(100,30))
> ca20.ml
likfit: estimated model parameters:
  beta0   beta1   beta2   beta3   tausq sigmasq     phi
33.0295  1.3439  7.8171 12.9089  5.4749 97.9126 77.2224

likfit: maximised log-likelihood = -629.0581

> summary(ca20.ml)
```

The output of `likfit` can be passed to the function `proflik`, which computes profile likelihoods for the model parameters as illustrated in Figure 5. This diagram is produced with the following command.

```
> plot(proflik(ca20.ml, geodata=ca20, sill.val=seq(50, 180, l=11),
       range.val=seq(30, 300, l=11), nugget.val=seq(0, 42, l=11))).
```

Another procedure for parameter estimation which is commonly used consists of fitting the curve of a valid variogram model to the empirical variogram. The model parameters can then be read directly from the fitted model as illustrated in Figure 6 with the exponential model fitted to the variogram of the `ca20` data. A common jargon is to refer to the intercept of the curve as the *nugget*, the difference between the asymptote and the nugget as the *sill*, and the distance at which the theoretical variogram curve reaches its maximum as the *range*. For models with an infinite range the value of u at which the variogram reaches 95% of the asymptote, is called the *practical range*. These names correspond to the parameters τ^2, σ^2 and ϕ respectively, where the latter is usually multiplied by a constant depending on the model. For instance, the practical range is 3ϕ for the exponential, $\sqrt{3}\phi$ for the Gaussian, 4ϕ and 5.37ϕ for the Matérn model with $\kappa = 1$ and 2, respectively, and ϕ for the spherical model.

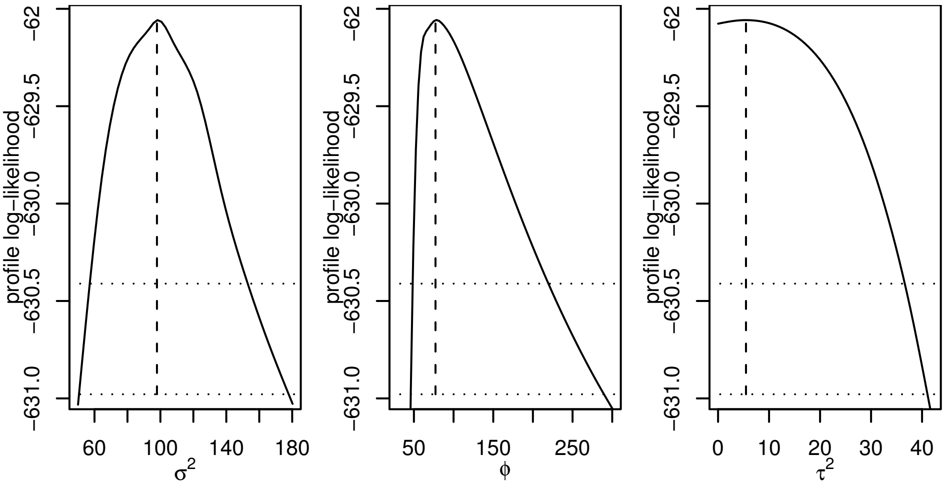

Figure 5: Profile likelihood for the parameters of the model fitted to the `ca20` data.

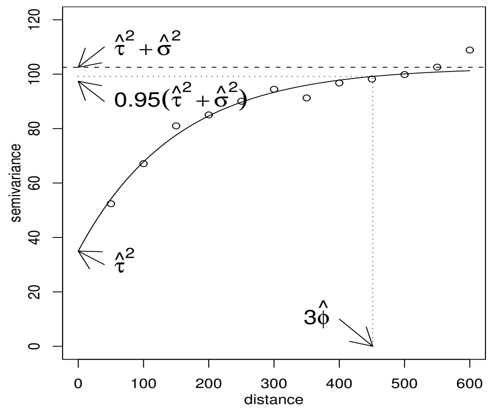

Figure 6: Exponential variogram model fitted to the `ca20` data and estimates of model parameters.

The function `variofit` implements non-linear least squares methods for this curve fitting procedure with options for ordinary and weighted least squares. Alternativelly, fitting a variogram "by eye" interactively can be done using the function `lines.variomodel`.

Regardless the method of parameter estimation, the results can be passed to the function `variog.model.env` which simulates from the fitted model and generates envelopes illustrates the statistical uncertainty in the empirical variogram.

6 Spatial prediction for the Gaussian model

Typically in geostatistical applications, interest lies in making inference about $S(\cdot)$, such as predicting the value of $S(\cdot)$ over the area, or estimating the probability that $S(x)$ is above a certain threshold value c.

The prediction problem then reduces to studying the conditional distribution of $S(\cdot)$ given the observed data y. To do this we consider a grid over the region. Considering a single value S_0 at a generic location x_0, under the Gaussian model (S_0, Y) is multivariate Gaussian with mean vector $(\mu_0, \mu) = (F_0\beta, F\beta)$ and covariance matrix

$$V = \left[\begin{array}{cc} \sigma^2 & \sigma^2\mathbf{r}^{\mathrm{T}} \\ \sigma^2\mathbf{r} & \tau^2 I + \sigma^2 R \end{array} \right]$$

where the vector r has elements $r_i = \rho(||x_0 - x_i||)$, $i = 1, ..., n$. The predictive distribution $[S_0 \mid y]$ is then Gaussian with mean and variance

$$\begin{aligned} \mathrm{E}[S_0 \mid y] &= \mu_0 + \sigma^2\mathbf{r}^{\mathrm{T}}(\tau^2 I + \sigma^2 R)^{-1}(y - \mu) \\ \mathrm{Var}[S_0 \mid y] &= \sigma^2 - \sigma^2\mathbf{r}^{\mathrm{T}}(\tau^2 I + \sigma^2 R)^{-1}\sigma^2\mathbf{r}. \end{aligned}$$

With all model parameters considered known the equations above are called *simple kriging* equations. If the mean vector parameter β is replaced by its generalised least squares estimator, $\hat{\beta} = (F'V^{-1}F)^{-1}F'V^{-1}y$, the resulting expression corresponds to *ordinary kriging* when F equals a vector or one's, to *universal kriging* or *kriging with a trend model* when F corresponds to a polynomial in the coordinates, and to *kriging with external trend* when F is given by a set of covariates.

These formulas are implemented in the function `krige.conv`, whose name stands for "conventional kriging". The inputs for the function are: the `geodata` object, coordinates of the locations for prediction, and model information, including parameter values which are often given by an output object from `likfit` or `variofit`. The model information is typically passed using `krige.control`. There is also an argument `border` allowing the user to compute predictions on a grid within a non-rectangular area. The function computes estimates of the means and variances at the prediction locations, the latter called *kriging variances*. Output options include simulations from the predictive distribution $[S_0|y]$, estimation of quantiles and probabilities of being below a specified threshold.

For the transformed model the predictions are returned on the untransformed scale. In general back-transformation is done by simulation, but for the log transformation (`lambda=0`), the back-transformation is done analytically using standard expressions for log-Gaussian kriging.

The output of `krige.conv` is of the class `kriging` and associated methods for `image`, `contour` and `persp` can be used to visualise the predictions as illustrated in Figure 7. Legends can be added to image plots either by using arguments in `image.kriging` or by calling the function `legend.krige`.

Two cross-validation methods for assessing the goodness of fit are implemented by the function `xvalid`. The first is the "leave-one-out" method where each point is removed from the data set and predicted using the remaining points, with an option for re-fitting the model at each run. The second splits the data into two sets, where one set is used to fit the model and to predict at the locations of the other set. For each validation location, observed and predicted values and their differences can be used in different ways to assess the model fitting as, for instance, by plotting cross-validated residuals against predicted values. The output is an object of class `xvalid`, and a method for plot is implemented for displaying the results.

7 Bayesian analysis for the Gaussian model

Bayesian inference treats the model parameters as random variables, adding to the model the specification of a *prior* distribution for the parameters. The predictive

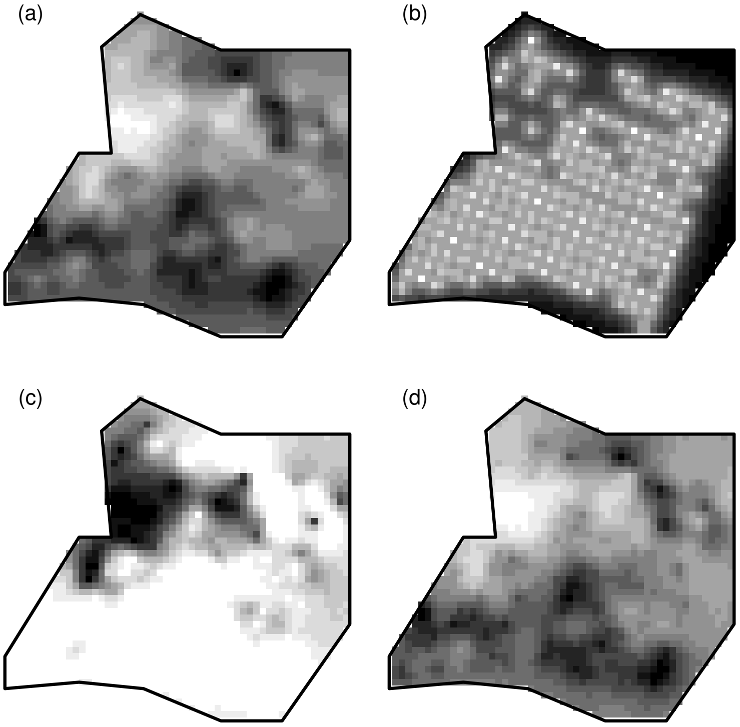

Figure 7: Kriging results for the `ca20` data: (a) predicted values, (b) prediction std. errors, (c) map of $P(Y < 40)$, (d) map of q, $P(Y < q) = 0.10$.

distribution $[S_0 \mid y]$ is then obtained by averaging the predictive distribution given in Section 6 over the different parameter values, with respect to their posterior distribution. This incorporates uncertainty about the model parameters into the predictive distribution.

The function `krige.bayes` implements Bayesian methods for the Gaussian model, sampling from the posterior distribution of the model parameters and, optionally, performing prediction. The input for this function includes: the `geodata` object; locations for prediction (if any); borders as described in Section 6; and three *control* functions. The function `model.control` defines the model to be fitted, i.e. choice of correlation function, covariates, etc., `prior.control` sets prior distributions for the model parameters, with options to fix some of them, and `output.control` defines what should be returned, as for `krige.conv`.

The output is of the class `kriging`. As for `krige.conv`, methods are provided for `image`, `contour` and `persp`. However, since this function also enables inference for the model parameters, methods are also provided for `plot`, `hist`, `print` and `lines`. Extra functions such as `sample.prior`, `sample.posterior` and `statistics.predictive` can be used to explore the output.

Figure 8 shows posterior distributions of the model parameters obtained with the following commands:

```
MC <- model.control(trend.l=~altitude+region)
```

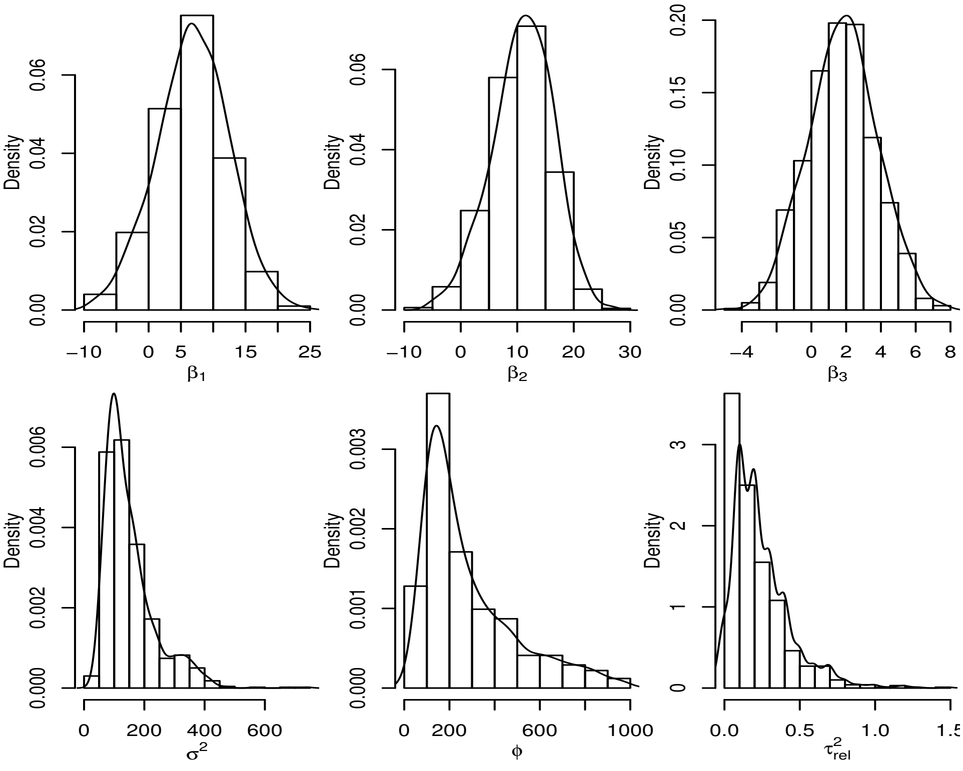

Figure 8: Posterior distributions for the covariance parameters.

```
PC <- prior.control(tausq.rel.prior = "uniform",
                    tausq.rel.discrete = seq(0,1,l=11),
                    phi.dis=seq(0,500,l=51))
ca20.kb <- krige.bayes(ca20, model = MC, prior = PC)
par(mfrow=c(1,3))
hist(ca20.kb)
```

For computational reasons the algorithm implements discrete priors for the parameters ϕ and $\tau^2_{rel} = \tau^2/\sigma^2$. The current version does not implement priors for either anisotropy or transformation parameters. Prediction results can be explored and maps produced as for the output of `krige.conv`. In addition, the function `post2prior` facilites sequential Bayesian updating, whereby the posterior from one stage of a data-analysis can be used as the prior for the next stage. This can be useful for computations on data which are divided into batches.

8 Inference for the generalised linear spatial model

The classical geostatistical model assumes that data are Gaussian, which may be an unrealistic assumption for some data sets. An example is the simulated data set shown below, which consists of binomial data of size 4 at 64 locations.

The generalised linear spatial model provides a framework for analysing such data; in particular Binomial and Poisson data. The likelihood for a model of this

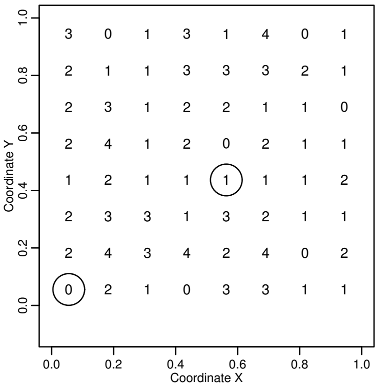

Figure 9: Simulated binomial data of size 4. Circles indicates locations for which traceplots from the MCMC algorithm will be shown.

kind is in general not expressible in closed form, but only as a high-dimensional integral

$$L(\beta, \sigma^2, \phi) = \int \prod_{i=1}^{n} f(y_i; h^{-1}(s_i)) p(s; \beta, \sigma^2, \phi) ds,$$

where $f(y; \mu)$ denotes the density of the error distribution parameterised by the mean μ, $p(s; \beta, \sigma^2, \phi)$ is the multivariate Gaussian density for the vector S of random effects at the data locations and $h(\cdot)$ is the link function. In practice, the high dimensionality of this integral prevents direct calculation, and inference relies on Markov chain Monte Carlo (MCMC) methods.

Most of the functionality presented in Section 6 and 7 for the Gaussian model is implemented for generalised linear models with Poisson error distribution and link function in the Box-Cox class, or with Binomial error distribution and logistic link function. Below we describe this briefly.

Here we consider MCMC simulation and prediction, with a focus on pratical issues related to the MCMC. We will first consider the case where parameters are known. We use the functions `pois.krige` and `binom.krige`.

A Langevin-Hastings algorithm is used to obtain MCMC simulations. The user must provide a value for the proposal variance `S.scale`. Optional inputs include the starting value, `S.start`, the length of the burn in, `burn.in`, the thinning, `thin`, and the number of iterations, `n.iter`. This input is provided via the argument `mcmc.input`, either as a list or by using the function `mcmc.control`. An example using the Rongelap data set is given below

```
ron1 <- pois.krige(rongelap, krige=list(cov.pars=c(0.2654,151.5885),
                                        beta=1.8212, nugget = 0.1337),
          mcmc.input=mcmc.control(S.scale=0.5, thin=1, n.iter=10000))
```

As a rule of thumb, `S.scale` should be chosen such that the acceptance rates for updating the random effects are approximately 60%. Also, we recommend studing the autocorrelations of the output and thinning such that the stored sample is approximately uncorrelated. For prediction, the argument `locations` must be provided, similarly to Sections 6 and 7.

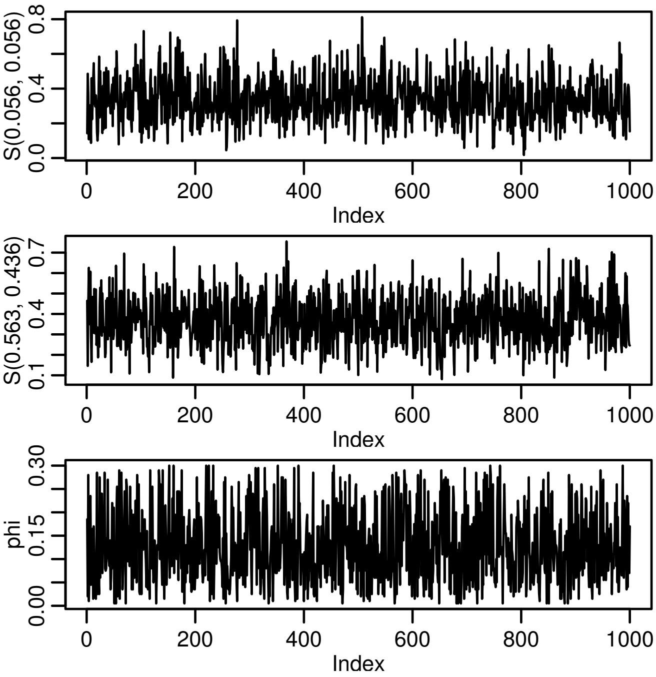

Figure 10: Traceplots for the parameter ϕ and for the two random effects at locations marked with a circle in Figure 9.

The procedure above can be extended to Bayesian inference which is implemented by the functions `pois.krige.bayes` and `binom.krige.bayes`. Priors are specified using the `prior.glm.control` function. If the parameter ϕ is not fixed, the user must provide for the MCMC-algorithm a scaling for the proposal distribution of ϕ using the argument `phi.scale` in the `mcmc.control` function. The update for ϕ is a random walk Metropolis type, and `phi.scale` should be chosen so that approximately 25% of the proposals are accepted. An example is given below for the simulated data set shown in Figure 9.

```
prior.sim <- prior.glm.control(beta.prior="normal", beta=0,
                beta.var=1, phi.prior="exponential", phi=0.2,
                phi.discrete=seq(0.005,0.3, l=60),
                sigmasq.prio="sc.inv.chisq", df.sigmasq=5,
                sigmasq=0.5)
mcmc.sim <- mcmc.control(S.scale=0.05, phi.scale=0.015, thin=100,
             burn.in = 10000)
b.sim <- binom.krige.bayes(b64, prior=prior.sim, mcmc.input=mcmc.sim)
```

Output from the MCMC-algorithm is presented in Figure 10 for the parameter ϕ and for the two random effects at locations marked with a circle in Figure 9.

9 Closing remarks

A *ToDo* file included in the `geoR` package distribution lists features to be implemented in the near future. Implementation of a wider class of models and methods may include tools for spatio-temporal and multivariate models. Functions for Markov chain Monte Carlo maximum likelihood are planned for `geoRglm`. The number of data points the packages are able to handle is limited by the ability to perform matrix operations on full covariance matrices. With the resources currently available to us, we can handle up to about 1000 points. One way to improve efficiency is to incorporate algorithms based on the work of Rue and Tjelmeland (2002). Another is to use approximate methods, such as model-based analogues of kriging with local neighbourhoods.

This is machine and memory dependent and with currently available resources our experience is that the maximum number of points ranges from 500-1000. Therefore approximate methods of inference for large data sets need also to be implemented.

Another direction is the development of functions for analysing marked point processes, an area which combines aspects of geostatistical and point processes methodology (Schlather, Ribeiro Jr, and Diggle, 2003). Spatial statistics can also take advantage of resources implemented in geographical information systems (GIS) by interfacing with software such as TERRALIB (`http://www.terralib.org`) and GRASS (`http://grass.itc.it`).

Acknowledgements

We thank Edzer Pebesma for his comments on the manuscript.

References

G.E.P. Box and D.R. Cox. An analysis of transformations (with discussion). *Journal of the Royal Statistical Society, Series B*, 26:211–252, 1964.

O.F. Christensen, P.J. Diggle, and P.J. Ribeiro Jr. Analysing positive-valued spatial data: the transformed Gaussian model. In P. Monestiez, D. Allard, and Froidevaux, editors, *GeoENV III - Geostatistics for Environmental Applications*, volume 11 of *Quantitative Geology and Geostatistics*, pages 287–298. Kluwer, 2001.

O.F. Christensen and P.J. Ribeiro Jr. geoRglm: a package from generalised linear spatial models. *R-NEWS*, pages 26–28, June 2002. ISSN 1609-3631. URL `http://cran.R-project.org/doc/Rnews`.

N. Cressie. *Statistics for spatial data - revised edition.* Wiley, New York, 1993.

N. Cressie and D.M. Hawkins. Robust estimation of the variogram. *Mathematical Geology*, 12:115–125, 1980.

P.J. Diggle and P.J. Ribeiro Jr. *Model-based geostatistics.* Springer, 2004. (in preparation).

P.J. Diggle, P.J. Ribeiro Jr, and O.F. Christensen. An introduction to model-based geostatistics. In J. Møller, editor, *Spatial Statistics and Computational Methods*, volume 173 of *Lecture Notes in Statistics*, pages 43–86. Springer, 2003.

P.J. Diggle, J.A. Tawn, and R.A. Moyeed. Model based geostatistics (with discussion). *Applied Statistics*, 47(3):299–350, 1998.

E.J. Pebesma. `gstat`: Multivariable statistics for S. In Kurt Hornik, Friedrich Leisch, and Achim Zeileis, editors, *Proceedings of the 3rd International Workshop on Distributed Statistical Computing, Technische Universität Wien, Vienna, Austria, March 20-22*, 2003. URL http://www.ci.tuwien.ac.at/Conferences/DSC-2003/Proceedings/. ISSN 1609-395X.

P.J. Ribeiro Jr and P.J. Diggle. geoR: a package from geostatistical analysis. *R-NEWS*, 1(2):15–18, June 2001. ISSN 1609-3631. URL http://cran.R-project.org/doc/Rnews.

H. Rue and H. Tjelmeland. Fitting Gaussian random fields to Gaussian fields. *Scandinavian Journal of Statistics*, 29(1):31–49, 2002.

M. Schlather, P.J. Ribeiro Jr, and P.J. Diggle. Detecting dependence between marks and locations of marked point processes. *Journal of the Royal Statistical Society, Series B*, 2003. (to appear).

Corresponding author

Paulo Justiniano Ribeiro Junior
Depto de Estatística
Universidade Federal do Paraná
Cx. Postal 19.081
CEP 81.531-990, Curitiba, PR
Brasil
E-mail: pj@est.ufpr.br