# The geoR package

March 29, 2001

## R topics documented:


as.geodata . . . . . 1
coords.aniso . . . . . 2
cov.spatial . . . . . 4
grf . . . . . 6
image.grf . . . . . 9
image.krige.bayes . . . . . 10
image.kriging . . . . . 11
krige.bayes . . . . . 12
krige.conv . . . . . 17
ksline . . . . . 20
likfit.old . . . . . 23
likfit . . . . . 27
lines.grf . . . . . 30
lines.krige.bayes . . . . . 31
lines.variogram.envelope . . . . . 32
lines.variogram . . . . . 33
lines.variomodel . . . . . 34
wrappers . . . . . 35
matern . . . . . 36
nlmP . . . . . 38
olsfit . . . . . 39
parana . . . . . 41
plot.geodata . . . . . 42
plot.grf . . . . . 43
plot.proflik . . . . . 44
plot.variogram . . . . . 46
points.geodata . . . . . 47
polygrid . . . . . 49
proflik . . . . . 50
read.geodata . . . . . 53
s100 and s121 . . . . . 54
summary.likGRF . . . . . 54
summary.variomodel . . . . . 55
trend.spatial . . . . . 56
varcov.spatial . . . . . 57
variog.mc.env . . . . . 59
variog.model.env . . . . . 60

variog . . . . . 62
wlsfit . . . . . 65

---

`as.geodata` *Converts an Object to the Class "geodata"*

---

## Description

Converts a matrix or a data-frame to an object of the `class` `"geodata"`. An object of the class `"geodata"` is a list with two obligatory components, `coords` and `data` and an optional component typically with values of the covariate(s).

## Usage

```
as.geodata(obj, coords.col = 1:2, data.col = 3, data.names = NULL,
           covar.col = NULL, covar.names = "obj.names")
```

## Arguments

| | |
|---|---|
| `obj` | a matrix or data-frame. The lines must have coordinates, data values and (optionally) covariates for each of the data locations. |
| `coords.col` | a vector with the numbers of the columns containing the coordinates. |
| `data.col` | a scalar or vector with the number of the column(s) containing the data. |
| `data.names` | a string or vector of strings with names for the data columns. Only valid if there is more than one column of data. By default the names in the original object are used. |
| `covar.col` | optional. A scalar or vector with the number of the column(s) with the values of the covariate(s). |
| `covar.names` | a string or vector of strings with the name(s) of the covariates. By default the names in the original object are used. |

## Details

Objects of the class `"geodata"` contain data for geostatistical analysis using the package **geoR**. Storing data in this format facilitates the usage of the functions in **geoR**. However this is not is not obligatory to carry out the analysis.

The component `coords` is always an $n \times 2$ matrix where $n$ is the number of data points.
The component `data` can be a vector, for the univariate case or, a matrix or data-frame for the multivariate case.
The third component is optional, does not have a fixed name, and contains the values of the covariate(s).

## Value

An object of the `class` `"geodata"` which is a list with two obligatory components:

| | |
|---|---|
| `coords` | data coordinates |
| `data` | data values |
| `covariate` | covariates values |

### Author(s)

Paulo Justiniano Ribeiro Jr. ⟨paulojus@est.ufpr.br⟩
Peter J. Diggle ⟨p.diggle@lancaster.ac.uk⟩.

### References

Further information about **geoR/geoS** can be found at:
`http://www.maths.lancs.ac.uk/~ribeiro/geoR.html`

### See Also

`read.geodata` for reading data from an *ASCII* file and `list` for information about lists in R.

---

## `coords.aniso` *Geometric Anisotropy Correction*

---

### Description

Transforms or back-transforms a set of coordinates according to the geometric anisotropy parameters.

### Usage

```
coords.aniso(coords, aniso.pars, reverse = FALSE)
```

### Arguments

| | |
|---|---|
| `coords` | an $n \times 2$ matrix with the coordinates to be transformed. |
| `aniso.pars` | a vector with two elements, $\psi_A$ and $\psi_R$, the *anisotropy angle* and the *anisotropy ratio*, respectively. The parameters must be provided in this order. See section DETAILS below for more information about these parameters. |
| `reverse` | logical. Defaults to `FALSE`. If `TRUE` the reverse transformation is performed. |

### Details

The parameters defining the geometric anisotropy are:

Anisotropy angle defined here as the azimuth angle of the direction with greater spatial continuity, i.e. the angle between the y-axis and the direction with the maximum range.

Anisotropy ratio defined here as the ratio between the ranges of the directions with greater and smaller continuity, i.e. the ratio between maximum and minimum ranges. Therefore, it must be greater or equal to one.

If `reverse = FALSE` (the default) the coordinates are transformed from the *anisotropic space* to the *isotropic space*. The transformation is done by multiplying the original coordinates by a rotation matrix $R$ and a shrinking matrix $T$, as follows:

$$X_m = XRT$$

where $X_m$ is a matrix with the modified coordinates (isotropic space) , $X$ is a matrix with original coordinates (anisotropic space), $R$ rotates coordinates according to the anisotropy angle $\psi_A$ and $T$ shrinks the coordinates according to the anisotropy ratio $\psi_R$.

If `reverse = TRUE`, the back-transformation is performed transforming the coordinates from the *isotropic space* to the *anisotropic space* by computing:

$$X = X_m(RT)^{-1}$$

## Value

An $n \times 2$ matrix with the transformed coordinates.

## Author(s)

Paulo Justiniano Ribeiro Jr. ⟨paulojus@est.ufpr.br⟩
Peter J. Diggle ⟨p.diggle@lancaster.ac.uk⟩.

## References

Further information about **geoR/geoS** can be found at:
`http://www.maths.lancs.ac.uk/~ribeiro/geoR.html`

## Examples

```
par.mf <- par()$mfrow
par(mfrow=c(3,2))
par(mar=c(2.5,0,0,0))
par(mgp=c(2,.5,0))
par(pty="s")
## Defining a set of coordinates
coords <- expand.grid(seq(-1, 1, l=3), seq(-1, 1, l=5))
plot(c(-1.5, 1.5), c(-1.5, 1.5), xlab="", ylab="", type="n")
text(coords[,1], coords[,2], 1:nrow(coords))
## Transforming coordinates according to some anisotropy parameters
coordsA <- coords.aniso(coords, aniso.pars=c(0, 2))
plot(c(-1.5, 1.5), c(-1.5, 1.5), xlab="", ylab="", type="n")
text(coordsA[,1], coordsA[,2], 1:nrow(coords))
##
coordsB <- coords.aniso(coords, aniso.pars=c(pi/2, 2))
plot(c(-1.5, 1.5), c(-1.5, 1.5), xlab="", ylab="", type="n")
text(coordsB[,1], coordsB[,2], 1:nrow(coords))
##
coordsC <- coords.aniso(coords, aniso.pars=c(pi/4, 2))
plot(c(-1.5, 1.5), c(-1.5, 1.5), xlab="", ylab="", type="n")
text(coordsC[,1], coordsC[,2], 1:nrow(coords))
##
coordsD <- coords.aniso(coords, aniso.pars=c(3*pi/4, 2))
plot(c(-1.5, 1.5), c(-1.5, 1.5), xlab="", ylab="", type="n")
text(coordsD[,1], coordsD[,2], 1:nrow(coords))
##
coordsE <- coords.aniso(coords, aniso.pars=c(0, 5))
plot(c(-1.5, 1.5), c(-1.5, 1.5), xlab="", ylab="", type="n")
text(coordsE[,1], coordsE[,2], 1:nrow(coords))
##
par(mfrow=par.mf)
```

`cov.spatial` *Computes Value of the Covariance Function for Given Distances*

## Description

Computes the covariances for pairs of points, given their separation distance. Several correlation functions can be used, see section DETAILS below. The results can be seen as a change of metric, from the *Euclidean distances* to *covariances*.

## Usage

```
cov.spatial(obj, cov.model=c("exponential", "matern", "gaussian",
            "spherical", "circular", "cubic", "wave",
            "powered.exponential", "cauchy",
            "gneiting", "gneiting.matern", "pure.nugget"),
            cov.pars=stop("no cov.pars argument provided"), kappa)
```

## Arguments

| | |
|---|---|
| `obj` | an object (vector or matrix) with values of distances, tipicaly between pairs of data locations. See section DETAILS below. |
| `cov.model` | a string indicating the type of the correlation function. See the DETAILS below for options and expressions of the correlation functions. |
| `cov.pars` | a vector with 2 elements or an $ns \times 2$ matrix with the covariance parameters. The first element (if a vector) or first column (if a matrix) corresponds to the variance parameter $\sigma^2$. The second element or column corresponds to the parameter $\phi$ of the correlation function. If a matrix is provided each row corresponds to the parameters of one *spatial structure*. Models with several structures are sometimes called *nested models* in the geostatistical literature. |
| `kappa` | value of an additional parameter. Only needed when using one of the following correlation functions: `"matern"`, `"powered.exponential"`, `"gneiting"` and `"gneiting.matern"`. |

## Details

The expressions of the covariance functions used in **geoR** can be found in Ribeiro and Diggle (1999).

The covariance functions can also be identified by numbers. The association between names and numbers is done by the function `cor.number`.

## Value

The function returns covariances values for the given distances. The type of object is the same as the type of the object provided in the argument `obj`, typically a vector, matrix or array.

## Note

This is an auxiliary function called by a number of other functions in the package.

## Author(s)

Paulo J. Ribeiro Jr. ⟨paulojus@est.ufpr.br⟩
Peter J. Diggle ⟨p.diggle@lancaster.ac.uk⟩.

## References

Ribeiro, P.J. Jr. and Diggle, P.J. (1999) geoR/geoS*: a geostatistical package/library for R/S-PLUS*. Tech. Report ST-99-09, Dept Maths and Stats, Lancaster University. Available on-line at: `http://www.maths.lancs.ac.uk/~ribeiro/publications.html`

Further information about **geoR/geoS** can be found at:
`http://www.maths.lancs.ac.uk/~ribeiro/geoR.html`

## See Also

`matern` for computation of the Matern model and `varcov.spatial` for computing the data covariance matrix.

## Examples

```
#
# Variogram models with the same "practical" range:
#
xval <- seq(0,1,l=101)
vexp <- cov.spatial(xval, cov.pars=c(1, .2))
vsph <- cov.spatial(xval, cov.pars=c(1, .60), cov.model="sph")
vgau <- cov.spatial(xval, cov.pars=c(1, .60/sqrt(3)),
                    cov.model="gau")
plot(0:1, 0:1, type="n", xlab="distance",
     ylab=expression(gamma(h)),
     main="variograms with equivalent \"practical range\"")
lines(xval, (1-vexp))
lines(xval, (1-vsph), lty=2)
lines(xval, (1-vgau), lwd=2)
legend(0.5,.3, c("exponential", "spherical", "gaussian"),
       lty=c(1,2,1), lwd=c(1,1,2))
#
# Matern models with equivalent "practical range"
# and varying smoothness parameter
#
dval <- seq(0,1,l=101)
mat.5 <- cov.spatial(dval, cov.pars = c(1, 0.25), kappa = 0.5)
mat1 <-  cov.spatial(dval, cov.pars = c(1, 0.188), kappa = 1,
                     cov.model="mat")
mat2 <-  cov.spatial(dval, cov.pars = c(1, 0.14), kappa = 2,
                     cov.model="mat")
mat3 <-  cov.spatial(dval, cov.pars = c(1, 0.117), kappa = 3,
                     cov.model="mat")
plot(0:1, 0:1, type="n", xlab="distance",
     ylab=expression(gamma(h)),
     main="models with equivalent \"practical\" range")
lines(dval, 1-mat.5, lty=2)
lines(dval, 1-mat1)
lines(dval, 1-mat2, lwd=2, lty=2)
lines(dval, 1-mat3, lwd=2)
legend(0.5,.3, c(expression(paste(kappa == 0.5, "  and  ",
       phi == 0.250)),
```

```
expression(paste(kappa == 1, "  and  ", phi == 0.188)),
expression(paste(kappa == 2, "  and  ", phi == 0.140)),
expression(paste(kappa == 3, "  and  ", phi == 0.117))),
lty=c(2,1,2,1), lwd=c(1,1,2,2))
```

---

grf *Simulation of Gaussian Random Fields*

---

## Description

Generates simulations of Gaussian random fields for given covariance parameters.

## Usage

```
grf(n, grid = "irreg", nx = round(sqrt(n)), ny = round(sqrt(n)),
    xlims = c(0, 1), ylims = c(0, 1), nsim = 1,
    cov.model = c("exponential", "matern", "gaussian",
                  "spherical", "circular", "cubic", "wave",
                  "powered.exponential", "cauchy", "gneiting",
                  "gneiting.matern", "pure.nugget"),
    cov.pars = stop("cov. parameters (sigmasq and phi) needed"),
    kappa = 0.5, nugget = 0, lambda = 1, aniso.pars,
    method = c("cholesky", "svd", "eigen", "circular.embedding"),
    messages.screen = TRUE)
```

## Arguments

| | |
|---|---|
| `n` | number of points (spatial locations) for each simulations. |
| `grid` | optional. An $n \times 2$ matrix with coordinates of the simulated data. |
| `nx` | optional. Number of points in the X direction. |
| `ny` | optional. Number of points in the X direction. |
| `xlims` | optional. Limits of the area in the X direction. Defaults to $[0, 1]$. |
| `ylims` | optional. Limits of the area in the Y direction. Defaults to $[0, 1]$. |
| `nsim` | Number of simulations. Defaults to 1. |
| `cov.model` | correlation function. See `cov.spatial` for details on the available choices of correlation functions. Defaults to `"exponential"`. |
| `cov.pars` | a vector with 2 elements or an $n \times 2$ matrix with values of the covariance parameters $\sigma^2$ (partial sill) and $\phi$ (range parameter). If a vector, the sill must be the first element and the range is the second. If a matrix, which corresponding to a model with several structures, the sill values are placed in the first column and the range values in the second column. |
| `kappa` | additional parameter for the following correlation functions: `"matern"`, `"matern""powered.exponential"`, `"matern""gneiting"` and `"gneiting.matern"`. For more details see documentation for the function `cov.spatial`. |
| `nugget` | the value of the nugget effect parameter $\tau^2$. |
| `lambda` | Box-Cox transformation parameter. The value $\lambda = 1$ corresponds to no transformation, the default. For any other value of $\lambda$ the simulated data is transformed. |

`aniso.pars` geometric anisotropy parameters. By default an isotropic field is assumed and this argument is ignored.
If a vector with 2 values (anisotropy angle, in radians, and anisotropy ratio) is provided, the coordinates are transformed, the simulation is performed on the isotropic (transformed) space and then the coordinates are back-transformed such that the resulting field is anisotropic. Coordinates transformation is performed by the function `coords.aniso`.

`method` simulation method. Defaults to the Cholesky decomposition.

`messages.screen` logical, indicating whether or not messages are printed on the screen (or output device) while the function is running. Defaults to `TRUE`.

## Details

For the methods `"cholesky"`, `"svd"` and `"eigen"` the simulation consists of multiplying a vector of standardized normal deviates by a square root of the covariance matrix. The square root of a matrix is not uniquely defined. These three methods differs in the way they compute the square root of the (positive definite) covariance matrix.

For `method = "circular.embedding"` the algorithm implements the method described by Wood & Chan (1994) which is based on Fourier transforms. Only regular and equally spaced grids can be generated using this method.
The code for the `"circular.embedding"` method was provided by Martin Schlather, University of Bayreuth
(`http://btgyn8.geo.uni-bayreuth.de/~martin/`).

**WARNING:** The code for the `"circular.embedding"` method is no longer being maintained. Martin will soon release a package for unconditional simulation of random fields. This will be announced on the R (contributed packages) and **geoR** home page. When this new package is released the current implementation of the `"circular.embedding"` method will become obsolete.

## Value

A list with the components:

`coords` an $n \times 2$ matrix with the coordinates of the simulated data.

`data` a vector (if `nsim = 1`) or a matrix with the simulations. For the latter each column corresponds to one simulation.

`cov.model` a string with the name of the correlation used.

`nugget` the value of the nugget parameter.

`cov.pars` a vector with the value of the partial sill and range parameter, respectively.

`kappa` value of the parameter $\kappa$.

`lambda` value of the Box-Cox transformation parameter $\lambda$.

`aniso.pars` a vector with values of the anisotropy parameters (if provided in the input).

`method` a string with the name of the simulation method used.

`.Random.seed` the random seed at the time the function was called.

`messages` messages produced by the function describing the simulation.

`call` the function call.

### Author(s)

Paulo Justiniano Ribeiro Jr. ⟨paulojus@est.ufpr.br⟩,
Peter J. Diggle ⟨p.diggle@lancaster.ac.uk⟩

### References

Wood, A.T.A. and Chan, G. (1994) Simulation of stationary Gaussian process in $[0,1]^d$. *Journal of Computatinal and Graphical Statistics*, **3**, 409–432.

Schlather, M. (1999) *Introduction to positive definite functions and to unconditional simulation of random fields.* Tech. Report ST-99-10, Dept Maths and Stats, Lancaster University.

Further information about **geoR** can be found at:
`http://www.maths.lancs.ac.uk/~ribeiro/geoR.html`

### See Also

`plot.grf` and `image.grf` for graphical output, `coords.aniso` for anisotropy coordinates transformation and, `chol`, `svd` and `eigen` for methods of matrix decomposition.

### Examples

```
sim1 <- grf(100, cov.pars=c(1, .25))
# a display of simulated locations and values
points.geodata(sim1)
# empirical and theoretical variograms
plot(sim1)
#
# a "smallish" simulation
sim2 <- grf(441, grid="reg", cov.pars=c(1, .25))
image.grf(sim2)
#
# a "bigger" one
sim3 <- grf(40401, grid="reg", cov.pars=c(10, .2), met="circ")
image.grf(sim3)
```

---

## `image.grf` *Image/Perspective Plot of Simulated Gaussian Random Field*

---

### Description

Plots an image or perspective plot with a realization of a Gaussian random field, simulated using the function `grf`.

### Usage

```
image.grf(obj, sim.number = 1, ...)

persp.grf(obj, sim.number = 1, ...)
```

### Arguments

| | |
|---|---|
| `obj` | an object with the output of the function `grf`. Typically an object of the class `grf`. |
| `sim.number` | number of simulations. Indicates the simulation number when the object contains more than one. |
| `...` | further arguments to be passed to the functions `image` or `persp`. |

### Value

An image or perspective plot is produced on the current graphics device.

### Author(s)

Paulo Justiniano Ribeiro Jr. ⟨paulojus@est.ufpr.br⟩
Peter J. Diggle ⟨p.diggle@lancaster.ac.uk⟩.

### References

Further information about **geoR/geoS** can be found at:
`http://www.maths.lancs.ac.uk/~ribeiro/geoR.html`

### See Also

`grf` for simulation of Gaussian random fields, `image` and `persp` for the generic plotting functions.

### Examples

```
# generating 4 simulations of a Gaussian random field
sim <- grf(225, grid="reg", cov.pars=c(1, .25), nsim=4)
par.ori <- par()
par(mfrow=c(2,2))
par(pty="s")
for (i in 1:4)
  image.grf(sim, sim.n=i)
```

---

## image.krige.bayes *Plots Results of the Predictive Distribution*

---

### Description

This function produces an image or perspective plot of a selected element of the predictive distribution returned by the function **geoR** function `krige.bayes`.

### Usage

```
image.krige.bayes(obj, locations,
           values.to.plot=c("moments.mean", "moments.variance",
                     "mean.simulations", "variance.simulations",
                     "quantiles", "probability", "simulation"),
           number.col, ...)
```

```
persp.krige.bayes(obj, locations,
           values.to.plot=c("moments.mean", "moments.variance",
                       "mean.simulations", "variance.simulations",
                        "quantiles", "probability", "simulation"),
           number.col, ...)
```

### Arguments

| | |
|---|---|
| `obj` | object of the class `krige.bayes`, typically an output of the function `krige.bayes`. |
| `locations` | an $n \times 2$ matrix with prediction locations. |
| `values.to.plot` | select an element of the predictive distribution to be plotted. See DETAILS below. |
| `number.col` | Only used if `values.to.plot = "quantile"`, `"probability"`, or `"simulation"`. Specifies the number of the column to be plotted. |
| `coords.data` | optional. If a matrix with the data coordinates is provided, points indicating the data locations are included in the plot. |
| `...` | extra arguments to be passed to the function `image` or `persp`. |

### Details

The function `krige.bayes` returns different summary and results of the predictive distribution. The argument `values.to.plot` specifies which result will be plotted. It can be passed to the function in two different forms:

- a vector with the object containing the values to be plotted or
- one of the following options: `"moments.mean"`, `"moments.variance"`, `"mean.simulations"`, `"variance.simulations"`, `"quantiles"`, `"probability"` and `"simulation"`.

For the last three options a column number must be provided using the argument `number.col` since the results are stored in matrices.

The documentation of the function `krige.bayes` provides mode details about each type of result.

### Value

An `image` or `persp` plot is produced on the current graphics device. No values are returned.

### Author(s)

Paulo J. Ribeiro Jr. ⟨paulojus@est.ufpr.br⟩
Peter J. Diggle ⟨p.diggle@lancaster.ac.uk⟩.

### References

Further information about **geoR/geoS** can be found at:
`http://www.maths.lancs.ac.uk/~ribeiro/geoR.html`

### See Also

`krige.bayes`.

### Examples

```
#See examples in the documentation of the function krige.bayes().
```

---

`image.kriging` *Image and Perspective Plots of Kriging Results*

---

### Description

Plots maps (image or perspective plots) with results of the kriging calculations.

### Usage

```
image.kriging(kriging.obj, location = kriging.obj$locations,
              values = kriging.obj$predict, ...)

persp.kriging(kriging.obj, locations=kriging.obj$locations,
              values=kriging.obj$predict, ...)
```

### Arguments

| | |
|---|---|
| `kriging.obj` | an object of the class `kriging` with the output of `krige.conv` or `ksline`. |
| `locations` | a matrix with the coordinates of the prediction locations. |
| `values` | a vector with values to be plotted. Defaults to `obj$predict`. |
| `coords.data` | optional. If a matrix with the data coordinates is provided, points indicating the data locations are included in the plot. |
| ... | further arguments to be passed to the functions `image` or `persp`. |

### Value

An image or perspective plot is produced in the current graphics device. No values are returned.

### Author(s)

Paulo J. Ribeiro Jr. ⟨paulojus@est.ufpr.br⟩
Peter J. Diggle ⟨p.diggle@lancaster.ac.uk⟩.

### References

Further information about **geoR/geoS** can be found at:
`http://www.maths.lancs.ac.uk/~ribeiro/geoR.html`

### See Also

`krige.conv`, `ksline`, `image`, `persp`.

### Examples

```
if(is.R()) data(s100)
loci <- expand.grid(seq(0,1,l=31), seq(0,1,l=31))
kc <- krige.conv(s100, loc=loci,
                 krige=krige.control(cov.pars=c(1, .25)))
par(mfrow=c(1,2))
image.kriging(kc, loc=loci)
image.kriging(kc, loc=loci, val=kc$krige.var)
```

---

`krige.bayes` *Bayesian Analysis for Gaussian Geostatistical Models*

---

### Description

The function `krige.bayes` performs Bayesian analysis of geostatistical data for different levels of uncertainty in the model parameters.
It returns results on the posterior distributions for the model parameters and on the predictive distributions for prediction locations (if provided).

### Usage

```
krige.bayes(geodata, coords = geodata$coords, data = geodata$data,
            locations = "no",
     model = model.control(trend.d = "cte", trend.l = "cte",
                  cov.model = "exponential", kappa = 0.5,
                  aniso.pars = NULL, lambda = 1),
     prior = prior.control(beta.prior = c("flat", "normal",
                                          "fixed"),
                  beta = NULL, beta.var = NULL,
                  sill.prior = c("reciprocal", "fixed"),
                  sill = NULL,
                  range.prior = c("uniform", "exponential",
                                  "fixed", "squared.reciprocal",
                                  "reciprocal"),
                  exponential.prior.par = 1, range = NULL,
                  range.discrete = NULL,
                  nugget.prior = c("fixed", "uniform"),
                  nugget = 0, nugget.discrete = NULL),
     output = output.control(n.posterior = 1000,
                  n.predictive = NULL, moments = TRUE,
                  simulations.predictive = TRUE,
                  keep.simulations = TRUE, mean.estimator = TRUE,
                  quantile.estimator =  NULL,
                  probability.estimator = NULL,
                  signal = FALSE, messages.screen = TRUE))
```

### Arguments

| | |
|---|---|
| `geodata` | a list containing elements `coords` and `data` as described next. Typically an object of the class `"geodata"` - a **geoR** data-set. If not provided the arguments `coords` and `data` must be provided instead. |

| | |
|---|---|
| `coords` | an $n \times 2$ matrix where each row has the 2-D coordinates of the $n$ data locations. By default it takes the component `coords` of the argument `geodata`, if provided. |
| `data` | a vector with $n$ data values. By default it takes the component `data` of the argument `geodata`, if provided. |
| `locations` | an $N \times 2$ matrix or data-frame with the 2-D coordinates of the $N$ prediction locations. Defaults to `"no"` in which case the function returns only results on the posterior distributions of the model parameters. |
| `model` | Defines model components. See section DETAILS below. |
| `prior` | Specification of priors for the model parameters. See section DETAILS below. |
| `output` | Defines output options. See section DETAILS below. |

## Details

### CONTROL FUNCTIONS

`krige.bayes` is a generic function for Bayesian geostatistical analysis where predictions can take into account the parameter uncertainty.

It can also be set to run conventional kriging methods which use known parameters or *plug-in* estimates. However, the functions `krige.conv` and `ksline` are preferable for prediction with fixed parameters.

The basis for the Bayesian algorithm is to discretize the prior distribution for the parameters $\phi$ and $\tau^2_{rel} = \frac{\tau^2}{\sigma^2}$. The Tech. Report referenced below provides the results used on the implementation.

The function call includes auxiliary control functions which allows the user to specify and/or change the model components (using `model.control`), prior distributions (using `prior.control`) and output options (using `output.control`). There are default options. The arguments for the control functions are as follows:

### ARGUMENTS FOR CONTROL FUNCTIONS

*Arguments for `model = model.control(...)` :*

**trend.d** information about the trend (covariates) values at the data locations. Possible values are: \ `"cte"` - model with constant mean, \ `"1st"` - trend is defined as a first degree polynomial on the coordinates, \ `"2nd"` - trend is defined as a second degree polynomial on the coordinates, \ a formula of the type `~X`, where `X` is a matrix with covariates (external trend) at data locations. \ Defaults to `"cte"`.

**trend.l** Specifies the trend (covariates) at prediction locations. Must be the same type as defined in `trend.d`. Only used if prediction locations are provided in the argument `locations`.

**cov.model** model for the correlation function. See options in the function call and/or documentation for the function `cov.spatial`.

**kappa** additional parameter for the following correlation functions: (`"matern"`, `"powered.exponential"`, `"gneiting"` and `"gneiting.matern"`). For more details see documentation for the function `cov.spatial`. This parameter is always regarded as fixed during the Bayesian analysis.

**aniso.pars** fixed parameters for geometric anisotropy correction. If `aniso.pars = FALSE` no correction is made otherwise a vector with 2 values, anisotropy angle and ratio of ranges, must be provided. Anisotropy correction consists of a transformation in the data and prediction coordinates performed by the function `coords.aniso`.

**lambda** Box-Cox transformation parameter. If $\lambda = 1$ no transformation is used, otherwise the Box-Cox data transformation is performed before the analysis. The transformation parameter $\lambda$ is regarded as a fixed parameter during the Bayesian analysis. Prediction results are returned is the same scale as for the original data. If $\lambda = 0$ the log-transformation is performed. If $\lambda < 0$ the mean predictor doesn't make sense (the resulting distribution has no expectation).

*Arguments for* `output = output.control(...)` :

**n.posterior** number of samples to be taken from the posterior distribution.

**n.predictive** number of samples to be taken from the predictive distribution.

**moments** logical. If `TRUE` moments of the predictive distribution are computed without sampling. This is valid only if `lambda = 1` or `lambda = 0`.

**simulations.predictive** logical. Defines whether simulations are drawn from the predictive distribution. Only valid if prediction locations are provided on the argument `locations`.

**keep.simulations** logical. Indicates whether or not the samples of the predictive distributions are kept and returned. Only valid if prediction locations are provided on the argument `locations`.

**mean.estimator** logical. Indicates whether or not the mean and variances of the predictive distributions are computed and returned. If `TRUE` objects `predict.mean` and `krige.var` are included in the output. Only valid prediction locations are provided on the argument `locations`.

**quantile.estimator** indicates whether or not quantiles of the predictive distributions must be computed and stored in the output. If a vector with numbers in the interval $[0, 1]$ is provided, the corresponding estimated quantile(s) are included in the output in the object `quantiles`. For example, if `estimator = c(0.25,0.50,0.75)` the function returns the quartiles for each location. If `quantile.estimator = TRUE`, the default is the vector `c(0.025, 0.5, 0.975)` and the corresponding percentiles are computed. An measure of uncertainty for the predictions, which is analogous to the kriging standard error, can be computed by $(quantile0.975 - quantile0.025)/4$. Only used if prediction locations are provided in the argument `locations`.

**probability.estimator** the default is `FALSE`. If some cutoff values are provided instead, an object called `probability` is included in the output providing, for each prediction location, the probability that the variable is less than or equal to the cutoff value given in the argument.

**signal** logical. If `TRUE` the signal is predicted, otherwise the variable is predicted. If no transformation is performed the expectations are the same in both cases and the kriging variances are different, if the nugget is different of zero.

**messages.screen** a flag `TRUE` or `FALSE` indicating whether or not messages are printed on the screen (or output file) while the function is running.

## Value

An object of the `class "krige.bayes"` which is a list with the following components:

`bold{posterior}`
: A list of results for the posterior distribution of the model parameters. The components are:

`beta.summary` summary of the posterior distribution of the mean parameter $\beta$.

`sigmasq.summary`
: summary of the posterior distribution of the variance parameter (partial sill) $\sigma^2$.

`phi.summary`
: summary of the posterior distribution of the correlation parameter (range parameter) $\phi$.

`tausq.summary`
: summary of the posterior distribution of the nugget variance parameter $\tau^2$.

`beta.samples`
: samples from the posterior distribution of the mean parameter $\beta$.

`sigmasq.samples`
: samples from the posterior distribution of the variance parameter (partial sill) $\sigma^2$.

`phi.samples`
: samples from the posterior distribution of the correlation parameter (range parameter) $\phi$.

`tausq.samples`
: samples from the posterior distribution of the nugget variance parameter $\tau^2$.

`phi.marginal`
: samples from the marginal posterior distribution of the correlation parameter $\phi$, resulting from averaging the posterior over the distribution of $(\beta, \sigma^2)$.

`nugget.marginal`
: samples from the marginal posterior distribution of the nugget variance parameter $\tau^2$, resulting from averaging the posterior over the distribution of $(\beta, \sigma^2)$.

`bold{predictive}`
: A list of results for the predictive distribution of the prediction locations (if provided). The components are:

`moments`
: a numerical matrix. The columns are moments of the predictive distribution at each prediction location

`simulations`
: a numerical matrix. Each column has a simulation drawn from the predictive distribution. Returned only if `keep.simulations = TRUE`.

`mean.simulations`
: a vector with the mean at the prediction locations computed by averaging over the simulations. Returned only if `mean.estimator = TRUE`.

`variance.simulations`
: a vector with the variance at the prediction locations computed from the simulations. Returned only if `mean.estimator = TRUE`.

`quantile`
: A matrix or vector with quantile estimators. Returned only if the argument `quantile.estimator` is used.

`probability`
: A matrix or vector with probability estimators. Returned only if the argument `probability.estimator` is used.

`type.prediction`
: information on the type of prediction performed.

`bold{message.prediction}`
: information about the type of inference, with or whitout allowing for parameter uncertainty, which has been performed.

`bold{.Random.seed}`
: system random seed before run the function. Allows reproduction of results. If the system `.Random.seed` is set to this value and the function is run again, it will produce exactly the same results.

`bold{call}`
: the function call.

### Auxiliary functions

The functions of type `krige.bayes.aux` and `control` are auxiliary functions called by `krige.bayes`.

### Author(s)

Paulo J. Ribeiro Jr. ⟨paulojus@est.ufpr.br⟩
Peter J. Diggle ⟨p.diggle@lancaster.ac.uk⟩.

### References

The technical details about the implementation of `krige.bayes` can be found at:

Ribeiro, P.J. Jr. and Diggle, P.J. (1999) *Bayesian inference in Gaussian model-based geostatistics*. Tech. Report ST-99-08, Dept Maths and Stats, Lancaster University. Available at: `http://www.maths.lancs.ac.uk/~ribeiro/publications.html`

Further information about **geoR/geoS** can be found at:
`http://www.maths.lancs.ac.uk/~ribeiro/geoR.html`

### See Also

`lines.krige.bayes`, `image.krige.bayes` and `persp.krige.bayes` for graphical output. `krige.conv` and `ksline` for conventional kriging methods.

### Examples

```
# generating a simulated data-set
ex.data <- grf(50, cov.pars=c(10, .25))
#
# defining the prediction grid:
ex.grid <- as.matrix(expand.grid(seq(0,1,l=11), seq(0,1,l=11)))
#
# computing Bayesian posterior and predictive distributions



ex.bayes <- krige.bayes(ex.data, loc=ex.grid, prior =
                 prior.control(range.discrete=seq(0, 2, l=51)))

#
# Ploting theoretical amd empirical variograms
plot(ex.data)
# adding lines with fitted variograms
lines(ex.bayes, max.dist=1.2)
lines(ex.bayes, max.dist=1.2, summ="median", lty=2)
lines(ex.bayes, max.dist=1.2, summ="mean", lwd=2, lty=2)
#
# Ploting prediction some results
par.mf <- par()$mfrow
par(mfrow=c(2,2))
image.krige.bayes(ex.bayes, loc=ex.grid, main="predicted values")
image.krige.bayes(ex.bayes, val="moments.variance",
                  loc=ex.grid, main="prediction variance")
image.krige.bayes(ex.bayes, val= "simulation", number.col=1,
                  loc=ex.grid,
```

```
      main="a simulation from the \npredictive distribution")
image.krige.bayes(ex.bayes, val= "simulation", number.col=2,
                  loc=ex.grid,
     main="another simulation from \nthe predictive distribution")
par(mfrow=par.mf)
```

---

`krige.conv` *Spatial Prediction - Conventional Kriging*

---

## Description

This function performs spatial prediction for fixed covariance parameters, using a global neighborhood.

Options corresponding to the following kriging types: *SK* (simple kriging), *OK* (ordinary kriging), *KTE* (external trend kriging), *UK* (universal kriging).

## Usage

```
krige.conv(geodata, coords = geodata$coords, data = geodata$data,
           locations,
           krige = krige.control(type.krige, beta = NULL,
                     trend.d, trend.l, cov.model, cov.pars,
                     kappa = 0.5, nugget = 0, micro.scale = 0,
                     dist.epsilon = 1e-10, aniso.pars = NULL,
                     lambda = 1, signal = FALSE,
                     n.samples.backtransform = 500, n.sim = 0),
           messages.screen = TRUE)
```

## Arguments

| | |
|---|---|
| `geodata` | a list containing elements `coords` and `data` as described next. Typically an object of the `class "geodata"` - a **geoR** data-set. If not provided the arguments `coords` and `data` must be provided instead. |
| `coords` | an $n \times 2$ matrix where each row have the 2-D coordinates of the $n$ data locations. By default it takes the component `coords` of the argument `geodata`, if provided. |
| `data` | a vector with $n$ data values. By default it takes the component `data` of the argument `geodata`, if provided. |
| `locations` | an $N \times 2$ matrix or data-frame with the 2-D coordinates of the $N$ locations to be predicted. |
| `krige` | defines the model components and several options for kriging. ATTENTION: the argument `cov.pars` is obligatory whilst the others have default options. See section DETAILS below. |
| `messages.screen` | logical. Indicates whether or not messages are printed on the screen (or other output device) while the function is running. |

**Details**

One of the different types of kriging (*SK*, *OK*, *UK*, *KTE*) is performed according to the input options. Defaults corresponds to ordinary kriging.

**Arguments for `krige = krige.control(...)` :**

type.krige type of kriging to be performed. Options are `"SK", "OK"` corresponding to simple or ordinary kriging. Kriging with external trend (universal kriging) can be defined setting `type.krige = "OK"` and specifying the trend model using the arguments `trend.d` and `trend.l`.

beta value of the mean (vector) parameter. Only used if `type.krige="SK"`.

trend.d information about the trend (covariates) values at the data locations. Possible values are: \ `"cte"` - model with constant mean, i.e. no trend \ `"1st"` - trend is defined as a first degree polynomial on the coordinates \ `"2nd"` - trend is defined as a second degree polynomial on the coordinates \ a formula of the type `~X`, where `X` is a matrix with covariates (external trend) at data locations. \ Defaults to `"cte"`.

trend.l Specifies the trend (covariate) at prediction locations. It must be the same type as defined in `trend.d`. Only used if prediction locations are provided in the argument `locations`.

cov.model model for the correlation function. See options in the function call and/or documentation for the function `cov.spatial`.

cov.pars a vector with 2 elements or an $n \times 2$ matrix with the covariance parameters $\sigma^2$ (partial sill) and $\phi$ (range parameter). If a vector, the sill must be the first element and the range is the second. If a matrix, corresponding to a model with several structures, the sill values are placed in the first column and the range values in the second column.

kappa additional parameter needed the the following correlation functions: (`"matern"`,\ `"powered.exponential"`, `"gneiting"` and `"gneiting.matern"`). For more details see documentation for the function `cov.spatial`.

nugget the value of the nugget variance. Defaults to zero.

micro.scale micro-scale variance. If different from zero, the nugget variance is divided into 2 terms: *micro-scale variance* and *measurement error*. This might affect the precision of the estimates. In practice, these two are usually indistinguishable but the distinction can be made here if justifiable.

dist.epsilon a distance such that points closer the the value of `dist.epsilon` are considered co-located.

aniso.pars fixed parameters for geometric anisotropy correction. If `aniso.pars = FALSE` no correction is made otherwise a vector with 2 values, anisotropy angle and ratio of ranges, must be provided. Anisotropy correction consists of a transformation in the data and prediction coordinates performed by the function `coords.aniso`.

lambda Box-Cox transformation parameter. If $\lambda = 1$ no transformation is used, otherwise the Box-Cox data transformation is performed before the analysis. The kriging results are returned in the same scale as for the original data. If $\lambda = 0$ the log-transformation is performed. If $\lambda < 0$ the mean predictor doesn't make sense (the resulting distribution has no expectation).

signal logical. If `TRUE` the signal is predict, otherwise the variable is predict. If no transformation is performed the expectations are the same in both cases and the difference is only in values of the kriging variance, if the value of the nugget is different from zero.

n.samples.backtransform When transformations are used (specified by the argument `lambda`), back-transformations are usually performed by sampling the predictive distribution and back-transforming the sampled values. The exceptions are for $\lambda = 0$ (log-transformation) and $\lambda = 1$ (no transformation). This option defines how many samples are taken to perform the back-transformation.

n.sim optional. Number of simulations to be drawn from the posterior distribution. If `n.sim` is provided, samples are taken from the predictive distribution. This corresponds to realizations of conditional simulations, given the data.

## Value

An object of the `class` `kriging` which is a list with the following components:

| | |
|---|---|
| `predict` | a vector with predicted value at the prediction locations. |
| `krige.var` | a vector with predicted variances at the prediction locations. |
| `beta.est` | estimates of the $\beta$ parameter implicitly made during the kriging procedure. Not used if `type.krige = "SK"`. |
| `simulations` | an $ni \times n.sim$ matrix where $ni$ is the number of prediction locations. Each column corresponds to a conditional simulation of the predictive distribution. Only returned if `n.sim > 0`. |
| `message` | messages about the type of prediction performed. |
| `call` | the function call. |

## Author(s)

Paulo J. Ribeiro Jr. ⟨paulojus@est.ufpr.br⟩
Peter J. Diggle ⟨p.diggle@lancaster.ac.uk⟩.

## References

Further information about **geoR/geoS** can be found at:
`http://www.maths.lancs.ac.uk/~ribeiro/geoR.html`

## See Also

`image.kriging`, `krige.bayes`, `ksline`.

## Examples

```
if(is.R()) data(s100)
loci <- expand.grid(seq(0,1,l=31), seq(0,1,l=31))
kc <- krige.conv(s100, loc=loci,
                 krige=krige.control(cov.pars=c(1, .25)))
par(mfrow=c(1,2))
image.kriging(kc, loc=loci, main="kriging estimates")
image.kriging(kc, loc=loci, val=sqrt(kc$krige.var),
              main="kriging std. errors")
```

`ksline` *Spatial Prediction - Conventional Kriging*

## Description

This function performs spatial prediction for fixed covariance parameters. Possible results which corresponds to the following kriging types: *SK* (simple kriging), *OK* (ordinary kriging), *KTE* (external trend kriging), *UK* (universal kriging).

The function `krige.conv` should be preferred for most kriging calculations. The only exception is for the case where the moving neighborhood is to be used.

## Usage

```
ksline(geodata, coords = geodata$coords, data = geodata$data,
       locations,
       cov.pars=stop("cov. parameters (sigmasq and phi) needed"),
       nugget=0, micro.scale=0,
       cov.model = c("exponential", "matern", "gaussian",
                     "spherical", "circular", "cubic", "wave",
                     "powered.exponential", "cauchy", "gneiting",
                     "gneiting.matern", "pure.nugget"),
       kappa=0.5, lambda=1, m0="ok", nwin="full",
       n.samples.backtransform=500, trend=1, d=2,
       ktedata=NULL, ktelocations=NULL, aniso.pars=NULL,
       signal=FALSE, dist.epsilon=1e-10, messages.screen=TRUE)
```

## Arguments

| | |
|---|---|
| `geodata` | a list containing elements `coords` and `data` as described next. Typically an object of the `class` `"geodata"` - a **geoR** data-set. If not provided the arguments `coords` and `data` must be provided instead. |
| `coords` | an $n \times 2$ matrix where each row has the 2-D coordinates of the $n$ data locations. By default it takes the component `$coords` of the argument `geodata`, if provided. |
| `data` | a vector with $n$ data values. By default it takes the component `data` of the argument `geodata`, if provided. |
| `locations` | an $N \times 2$ matrix or data-frame with the 2-D coordinates of the $N$ locations to be predicted. |
| `cov.pars` | a vector with 2 elements or an $n \times 2$ matrix with the values of the covariance parameters $\sigma^2$ (partial sill) and $\phi$ (range parameter). If a vector, the partial sill must be the first element and the range parameter is the second. If a matrix, corresponding to a model with several structures, the partial sill values are placed in the first column and the values for range parameter in the second column. |
| `nugget` | the value of the nugget variance. |
| `micro.scale` | micro-scale variance. If different from zero, the nugget variance is divided in 2 terms: *micro-scale variance* and *measurement error*. This might affect the precision of the estimates. In practice, these two are usually indistinguishable but the distinction can be made here if justifiable. |

`cov.model` model for the correlation function. See options in the function call and/or documentation for the function `cov.spatial`.

`kappa` additional parameter needed for the following correlation functions: (`"matern"`, `"powered.exponential"`, `"gneiting"` and `"gneiting.matern"`). For more details see documentation for the function `cov.spatial`.

`lambda` Box-Cox transformation parameter. If $\lambda = 1$ no transformation is used, otherwise the Box-Cox data transformation is performed before the analysis. The kriging results are returned in the same scale as the original data. If $\lambda = 0$ the log-transformation is performed. If $\lambda < 0$ the mean predictor doesn't make sense (the resulting distribution has no expectation).

`m0` The default value `"ok"` indicates that ordinary kriging will be performed. Other options are `"kt"` for kriging with a trend model (universal kriging) and `"kte"` for kriging with external trend (covariates). If a numeric value is provided, this value is assumed to be the know mean and simple kriging is performed. If `"av"` the arithmetic mean of the data is assumed to be the know mean for simple kriging algorithm.

`nwin` If `"full"`, all data values are used in the prediction at each prediction locations. If an integer is provided, it defines the number closest neighbors to be used for the prediction.

`n.samples.backtransform`
when transformations are used (specified by the argument `lambda`), back-transformations are usually performed by sampling the predictive distribution and back-transforming the sampled values. The exceptions are for $\lambda = 0$ (log-transformation) and $\lambda = 1$ (no transformation). This options defines how many samples are taken when performing the back-transformation.

`trend` only required only if `m0="kt"` (universal kriging). Possible values are 1 or 2 corresponding to a first or second degree polynomial trend on the coordinates, respectively.

`d` spatial dimension, 1 defines a prediction on a line, 2 on a plane (the default).

`ktedata` only required if `m0="kte"`. Values of the external trend (covariates) at the data locations.

`ktelocations` only required if `m0="kte"`. Values of the external trend (covariates) at the prediction locations.

`aniso.pars` fixed parameters for geometric anisotropy correction. If `aniso.pars = FALSE` no correction is made otherwise a vector with 2 values, anisotropy angle and ratio of ranges, must be provided. Anisotropy correction consists of a transformation in the data and prediction coordinates performed by the function `coords.aniso`.

`signal` logical. If `TRUE` the signal is predict, otherwise the variable is predict. If no transformation is performed the expectations are the same in both cases and the difference is only in values of the kriging variance, if the value of the nugget is different from zero.

`dist.epsilon` a distance such that points closer the the value of `dist.epsilon` are considered co-located.

`messages.screen`
logical. Indicates whether or not messages are printed on the screen (or other output device) while the function is running.

### Value

An object of the `class` `kriging` which is a list with the following components:

| | |
|---|---|
| `predict` | the predicted values. |
| `krige.var` | the kriging variances. |
| `dif` | the difference between the predicted value and the global mean. Represents the contribution to the neighboring data to the prediction at each point. |
| `summary` | values of the arithmetic and weighted mean of the data and standard deviations. The weighted mean corresponds to the estimated value of the global mean. |
| `ktrend` | the matrix with trend if `mo="kt"` (universal kriging). |
| `ktetrend` | the matrix with trend if `mo="kte"` (external trend kriging). |
| `beta` | the value of the mean which is implicitly estimated for `m0="ok"`, `"kte"` or `"kt"`. |
| `wofmean` | weight of mean. The predicted value is an weighted average between the global mean and the values at the neighboring locations. The value returned is the weight of the mean. |
| `locations` | the coordinates of the prediction locations. |
| `message` | status messages returned by the algorithm. |
| `call` | the function call. |

### Note

This is a preliminary and inefficient function. For predictions using global neighborhood the function `krige.conv` should be used instead.

### Author(s)

Paulo J. Ribeiro Jr. ⟨paulojus@est.ufpr.br⟩
Peter J. Diggle ⟨p.diggle@lancaster.ac.uk⟩.

### References

Further information about **geoR/geoS** can be found at:
`http://www.maths.lancs.ac.uk/~ribeiro/geoR.html`

### See Also

`krige.conv`, `krige.bayes`.

### Examples

```
if(is.R()) data(s100)
loci <- expand.grid(seq(0,1,l=31), seq(0,1,l=31))
kc <- ksline(s100, loc=loci, cov.pars=c(1, .25))
par(mfrow=c(1,2))
image.kriging(kc, loc=loci, main="kriging estimates")
image.kriging(kc, loc=loci, val=sqrt(kc$krige.var),
              main="kriging std. errors")
```

`likfit.old` *Likelihood Estimation for Gaussian Random Fields*

## Description

This function estimates the parameters of a (transformed) Gaussian random field using either *maximum likelihood* (ML) or *restricted maximum likelihood* (REML). Includes option for the estimation the Box-Cox transformation parameter.

This is a previous version of the function `likfit` and it will be made obsolete.

## Usage

```
likfit(geodata, coords=geodata$coords, data=geodata$data,
       trend = "cte", ini, fix.nugget = FALSE, nugget = 0,
       cov.model = c("exponential", "matern", "gaussian",
                     "spherical", "circular", "cubic", "wave",
                     "powered.exponential", "cauchy", "gneiting",
                     "gneiting.matern", "pure.nugget"),
       kappa = 0.5, fix.lambda = TRUE, lambda = 1, method = "ML",
       predicted = FALSE, residuals = FALSE,
       minimisation.function = c("optim","nlmP", "nlm"),
       automatic.refit = FALSE, range.limits,
       messages.screen = TRUE, ...)
```

## Arguments

| | |
|---|---|
| `geodata` | a list containing the elements `coords` and `data` as described next. If not provided the arguments `coords` and `data` must be provided instead. |
| `coords` | an $n \times 2$ matrix containing in each row Euclidean coordinates of the $n$ data locations. By default it takes the element `$coords` of the argument `geodata`. |
| `data` | a vector with data values. By default it takes the element `$data` of the argument `geodata`. |
| `trend` | specifies the mean part of the model. The options are: `"cte"` (constant mean - default option), `"1st"` (a first degree polynomial on the coordinates), `"2nd"` (a second degree polynomial on the coordinates), or a formula of the type `~X` where `X` is a matrix with the covariates (external trend). |
| `ini` | initial values of the covariance parameters. These values are used to initiate the numerical minimizer. If the model to be fitted has 3 parameters: nugget ($\tau^2$), sill ($\sigma^2$) and range ($\phi$), `ini` is a vector with initial values for these tree parameters. If the model to be fitted does not have has nugget `ini` is a vector with 2 components, the initial values for $\sigma^2$ and $\phi$. If `ini` is a matrix with several initial values on the rows a search for the better initial value is performed. The values of the likelihood function is computed for each of them and the row with maximum value is used to initiate the numerical maximization. |
| `fix.nugget` | logical, indicating whether or not the nugget parameter must be included in the estimation. If `fix.nugget = T` the argument `ini` must be a vector of length 2 or a matrix with 2 columns |

`nugget` Value of the nugget parameter when `fix.nugget = TRUE`. Defaults to zero.

`cov.model` model for the correlation function. See options in the function call and/or documentation for the function `cov.spatial`.

`kappa` extra parameter needed for some of the correlation functions: `("matern")`, `code("powered.exponential")`, `code("gneiting"` and `"gneiting.matern")`. For more details see documentation for the function `cov.spatial`. This parameter is always regarded as fixed during the ML/REML parameter estimation.

`fix.lambda` If `TRUE` (default option) the transformation parameter is regarded as fixed (known) during the estimation process otherwise the ML/REML for this parameter is also computed.

`lambda` Box-Cox transformation parameter. Two particular cases are $\lambda = 1$ which corresponds to no transformation and $\lambda = 0$ corresponding to the log-transformation. The value of the transformation parameter $\lambda$ is regarded as a fixed value for the parameter if `fix.lambda = TRUE` or is used as a initial value in the search if `fix.lambda = FALSE`.

`method` `ML` for maximum likelihood or `REML` for restricted maximum likelihood.

`predicted` a matrix with the several compnents of the predicted values, at the data locations.

`residuals` a matrix with several components of the residuals (random parsto of the model), at the data locations.

`automatic.refit`
Defaults to `FALSE`. If set to `TRUE` the model is automatically changed and fitted again as follows:

- In the case of a model with nugget, if the relative nugget is less than 0.01 the model without nugget is fitted.
- If the range parameters is less than $1e-12$ or a value provided in the argument `range.limits` a model without spatial correlation is fitted.

`minimisation.func`
minimization function to be used

`range.limits` minimum and maximum values allowed for the correlation function parameter (range). Defaults to `c(0, +Inf)`.

`messages.screen`
a flag `TRUE` or `FALSE` indicating whether or not messages are printed on the screen (or output device) while the function is running.

`...` additional parameters to be passed to the minimization function. Typically `control` type arguments which controls the behavior of the minimization algorithm. See documentation of the minimization functions for further details.

## Details

The characterization of the random field model considered here is given by the following parameters:

- the mean parameter $\beta$,
- the covariance parameters: sill $\sigma^2$ and range $\phi$,

- the smoothness parameter $\kappa$,
- the nugget parameter $\tau^2$,
- the Box-Cox transformation parameter $\lambda$,
- and the anisotropy parameters: anisotropy angle $\alpha$ and anisotropy ratio $\psi$,

However not all of them are estimated by this function. In the current implementation:

1. the parameters $\beta$, $\sigma^2$, $\phi$ are always estimated;
2. the parameters $\tau^2$, $\lambda$ can either be estimated or be considered fixed;
3. the parameters $\kappa$, $\alpha$, $\psi$.

**Value**

An object of the class `variomodel` which is a list with the following components:

| | |
|---|---|
| `cov.model` | a string with the name of the correlation function. See function call for the options. |
| `nugget` | value of the nugget parameter $\tau^2$. This is the estimated value if `fix.nugget = FALSE` or the fixed (known) value if `fix.nugget = TRUE`. |
| `cov.pars` | a 2 elements vector with the estimated sill and range parameters, respectively. |
| `kappa` | (fixed) value of the smoothness parameter required by some of the correlation functions. |
| `beta` | estimated mean parameters. Can be either a scalar or vector depending on the trend model (covariates) used. |
| `beta.var` | variance (or covariance matrix) of the mean parameter. |
| `lambda` | Box-Cox transformation parameter. A fixed (know) value if `fix.lambda = TRUE` or the estimated value if `fix.lambda = FALSE`. |
| `loglik` | the value of the maximized likelihood. |
| `npars` | number of estimated parameters. |
| `AIC` | Value of the Akaike information criteria. |
| `BIC` | Value of the Bayesian information criteria. |
| `trend.ols` | Trend (mean parameters) estimated by ordinary least squares (i.e. ignoring the spatial correlation). |
| `info.lambda` | information about the Box-Cox transformation parameter to be used by other functions which uses the output of `likfit()`. |
| `method` | Estimation method used, `"ML"` (maximum likelihood) or `"REML"` (restricted maximum likelihood). |
| `s2` | estimated residual variance. Only returned if `predicted = TRUE` or `residuals = TRUE`. |
| `predicted` | predicted values at data locations. Only returned if `predicted = TRUE`. |
| `residuals` | estimated residuals at data locations. Only returned if `residuals = TRUE`. |
| `info` | results returned by the minimization function. |
| `max.dist` | maximum distance found between 2 data points. This information is used by other functions which uses results from `likfit()`. |
| `trend.matrix` | trend matrix (covariates). |
| `call` | the function call. |

### Note

`proflik.phi`, `proflik.ftau`, `proflik.nug` are auxiliary functions which computed the value of the likelihood for models without nugget, with a fixed nugget and including nugget, respectively. These functions are internally called by the minimization function when estimating the model parameters.

### Author(s)

Paulo Justiniano Ribeiro Jr. ⟨paulojus@est.ufpr.br⟩
Peter J. Diggle ⟨p.diggle@lancaster.ac.uk⟩.

### References

Further information about **geoR/geoS** can be found at:
`http://www.maths.lancs.ac.uk/~ribeiro/geoR.html`

### See Also

`summary.variomodel`, `olsfit`, `wlsfit`, `plot.variogram`, `lines.variogram`, `lines.variomodel`, `proflik`, `optim`, `nlm`.

### Examples

```
if(is.R()) data(s100)
ml <- likfit(s100, ini=c(.5, .5), fix.nug=T)
summary(ml)
reml <- likfit(s100, ini=c(.5, .5), fix.nug=T, met="REML")
summary(reml)
plot(variog(s100))
lines(ml)
lines(reml, lty=2)
```

---

## `likfit` *Likelihood Estimation for Gaussian Random Fields*

---

### Description

Parameter estimation for (transformed) Gaussian random fields by *maximum likelihood* (ML) or *restricted maximum likelihood* (REML).

### Usage

```
likfit(geodata, coords = geodata$coords, data = geodata$data,
       trend = "cte", ini.cov.pars, fix.nugget = FALSE, nugget = 0,
       fix.kappa = TRUE, kappa = 0.5, fix.lambda = TRUE, lambda = 1,
       fix.psiA = TRUE, psiA = 0, fix.psiR = TRUE, psiR = 1,
       cov.model = c("matern", "exponential", "gaussian",
                     "spherical", "circular","cubic", "wave",
                     "powered.exponential", "cauchy", "gneiting",
                     "gneiting.matern", "pure.nugget"),
       method = "ML", components = FALSE, nospatial = TRUE,
       limits = likfit.limits(), messages.screen = TRUE, ...)
```

**Arguments**

| | |
|---|---|
| `geodata` | a list containing elements `coords` and `data` as described next. Typically an object of the class `"geodata"` - a **geoR** data-set. If not provided the arguments `coords` and `data` must be provided instead. |
| `coords` | an $n \times 2$ matrix, each row containing Euclidean coordinates of the $n$ data locations. By default it takes the element `$coords` of the argument `geodata`. |
| `data` | a vector with data values. By default it takes the element `$data` of the argument `geodata`. |
| `trend` | specifies the mean part of the model. The options are: `"cte"` (constant mean), `"1st"` (a first degree polynomial on the coordinates), `"2nd"` (a second degree polynomial on the coordinates), or a formula of the type `~X` where `X` is a matrix with the covariates (external trend). Defauts to `"cte"`. |
| `ini.cov.pars` | initial values for the covariance parameters: $\sigma^2$ (partial sill) and $\phi$ (range parameter). |
| `fix.nugget` | logical, indicating whether the parameter $\tau^2$ (nugget variance) should be regarded as fixed (`fix.nugget = TRUE`) or is to be estimated (`fix.nugget = FALSE`). Defaults to `FALSE`. |
| `nugget` | value for the nugget parameter. Regarded as a fixed values if `fix.nugget = TRUE` or as a initial value for the minimization algorithm if `fix.nugget = FALSE`. Defaults to zero. |
| `fix.kappa` | logical, indicating whether the extra parameter $\kappa$ is regarded as fixed (`fix.kappa = TRUE`) or is to be estimated (`fix.kappa = FALSE`). Defaults to `TRUE`. |
| `kappa` | value for the extra parameter $\kappa$. Regarded as a fixed value if `fix.kappa = TRUE` or as a initial value for the minimization algorithm if `fix.kappa = FALSE`. Defaults to 0.5. This parameter is valid only if the covariance function is one of: `"matern"`, `"powered.exponential"`, `"gneiting"` or `"gneiting.matern"`. For more details on covariance functions see documentation for `cov.spatial`. |
| `fix.lambda` | logical, indicating whether the Box-Cox transformation parameter $\lambda$ should be regarded as fixed (`fix.lambda = TRUE`) or is to be estimated (`fix.lambda = FALSE`). Defaults to `TRUE`. |
| `lambda` | value for the Box-Cox transformation parameter $\lambda$. Regarded as a fixed value if `fix.lambda = TRUE` or as a initial value for the minimisation algorithm if `fix.lambda = FALSE`. Defaults to 1. Two particular cases are $\lambda = 1$ indicationg no transformation, and $\lambda = 0$ indicating the log-transformation. |
| `fix.psiA` | logical, indicating whether the anisotropy angle parameter $\psi_R$ should be regarded as fixed (`fix.psiA = TRUE`) or is to be estimated (`fix.psiA = FALSE`). Defaults to `TRUE`. |
| `psiA` | value (in radians) for the anisotropy angle parameter $\psi_A$. Regarded as a fixed value if `fix.psiA = TRUE` or as a initial value for the minimisation algorithm if `fix.psiA = FALSE`. Defaults to 0. See `coords.aniso` for further details on anisotropy correction. |
| `fix.psiR` | logical, indicating whether the anisotropy ratio parameter $\psi_R$ should be regarded as fixed (`fix.psiR = TRUE`) or is to be estimated (`fix.psiR = FALSE`). Defaults to `TRUE`. |

`psiR` value (greater than 1) for the anisotropy ratio parameter $\psi_R$. Regarded as a fixed value if `fix.psiR = TRUE` or as a initial value for the minimisation algorithm if `fix.psiR = FALSE`. Defaults to 1. See `coords.aniso` for further details on anisotropy correction.

`cov.model` model for the correlation function. See options in the function call and/or in the documentation for `cov.spatial`.

`method` `"ML"` for maximum likelihood or `"REML"` for restricted maximum likelihood. Defaults to `"ML"`.

`components` a data-frame with fitted values for the three model components: trend, spatial and residuals. See the section `DETAILS` below for the model specification.

`nospatial` logical. If `TRUE` parameter estimates for the model without spatial component are returned.

`limits` lower and upper limits for the model parameters used in the numerical minimisation. Uses the auxiliary function `likfit.limits()`.

`messages.screen`
logical. `TRUE` indicates that messages will be printed on the screen (or output device) while the function is running.

`...` additional parameters to be passed to the minimization function. Typically `control()` type of arguments which controls the behavior of the minimization algorithm. For further datails see documentation for the minimization function `optim`.

## Details

This function estimate the parameters of the Gaussian random field model specifyed here as:

$$Y(x) = \mu(x) + S(x) + e$$

where

- $\mu(x) = X\beta$ is the mean component of the model (trend).
- $S(x)$ is a Gaussian process with variance $\sigma^2$ (partial sill) and a correlation function parametrized by $\phi$ (the range parameter). Possible extra parameters for the correlation function are the smoothness parameter $\kappa$ and the anisotropy parameters $\phi_R$ and $\phi_A$ (anisotropy ratio and angle, respectively).
- $e$ is the error term with variance parameter $\tau^2$ (nugget variance).

The additional parameter $\lambda$ allows the Box-Cox transformation.

Parameter estimation is performed numerically using the R function `optim` to minimize the negative log-likelihood computed by `negloglik.GRF`.

Lower and upper limits for the values of the parameters can be individually specified using the function `likfit.limits()`. For example:
`likfit.limits(phi=c(0, 10), lambda=c(-2.5, 2.5))`,
change the limits for the paramaters $\phi$ and $\lambda$. If not provided default values are used.

If the `fix.lambda = FALSE` and `nospatial = FALSE` the Box-Cox parameter for the model without the spatial component is obtained numerically, with log-likelihood computed by the function `boxcox.ns`.

## Value

An object of the `class "likGRF"` and `"variomodel"`.
The function `summary.likGRF` is used to print a summary of the fitted model.
The object is a list with the following components:

| | |
|---|---|
| `cov.model` | a string with the name of the correlation function. |
| `nugget` | value of the nugget parameter $\tau^2$. This is an estimate if `fix.nugget = FALSE` or a fixed value if `fix.nugget = TRUE`. |
| `cov.pars` | a vector with the estimates of the partial sill and range parameters, respectively. |
| `kappa` | value of the smoothness parameter. Valid only if the correlation function is one of: `"matern"`, `"powered.exponential"`, `"gneiting"` or `"gneiting.matern"` |
| `beta` | estimate of mean parameter $\beta$. Can be either a scalar or vector depending on the trend (covariates) used in the model. |
| `beta.var` | estimated variance (or covariance matrix) for the mean parameter $\beta$. |
| `lambda` | Box-Cox transformation parameter. A fixed value if `fix.lambda = TRUE` or the estimate if `fix.lambda = FALSE`. |
| `aniso.pars` | fixed values or estimates of the anisotropy parameters. |
| `method` | Estimation method used, `"ML"` (maximum likelihood) or `"REML"` (restricted maximum likelihood). |
| `loglik` | the value of the maximized likelihood. |
| `npars` | number of estimated parameters. |
| `AIC` | value of the Akaike information criteria. |
| `BIC` | value of the Bayesian information criteria. |
| `parameters.summary` | a data-frame with all model parameters, their status (estimated or fixed) and values. |
| `info.minimisation` | results returned by the minimisation function. |
| `max.dist` | maximum distance between 2 data points. This information relevant for other functions which use outputs from `likfit`. |
| `trend.matrix` | the trend (covariates) matrix $X$. |
| `log.jacobian` | numerical value of the logarithm of the Jacobian of the transformation. |
| `nospatial` | estimates for the model without the spatial component. |
| `call` | the function call. |

## Author(s)

Paulo Justiniano Ribeiro Jr. ⟨paulojus@est.ufpr.br⟩
Peter J. Diggle ⟨p.diggle@lancaster.ac.uk⟩.

## References

Further information about **geoR/geoS** can be found at:
`http://www.maths.lancs.ac.uk/~ribeiro/geoR.html`

### See Also

`summary.likGRF` for summary of the results, `plot.variogram`, `lines.variogram` and `lines.variomodel` for graphical output, `proflik` for computing profile likelihoods, `olsfit` and `wlsfit` for other estimation metods, and `optim` for the numerical minimization function.

### Examples

```
if(is.R()) data(s100)
ml <- likfit(s100, ini=c(0.5, 0.5), fix.nug = TRUE)
ml
summary(ml)
reml <- likfit(s100, ini=c(0.5, 0.5), fix.nug = TRUE, met = "REML")
summary(reml)
plot(variog(s100))
lines(ml)
lines(reml, lty = 2)
```

---

## `lines.grf` *Lines with True Variogram for Simulated Data*

---

### Description

This functions adds to the current plot the theoretical (true) variogram of simulations generated by the function `grf`.

### Usage

```
lines.grf(obj, max.dist = max(dist(obj$coords)), length = 100,
          lwd = 2, ...)
```

### Arguments

| | |
|---|---|
| `obj` | an object from the class `grf` with simulated data generated by the function `grf`. |
| `max.dist` | maximum distance to compute and plot the true variogram. |
| `length` | number of points to compute and draw the variogram line. |
| `lwd` | width of the line. |
| `...` | further arguments to be passed to the function `lines`. |

### Value

A line with the true variogram model used to simulate the data is added to the current plot on the graphics device. No values returned.

### Author(s)

Paulo J. Ribeiro Jr. ⟨paulojus@est.ufpr.br⟩
Peter J. Diggle ⟨p.diggle@lancaster.ac.uk⟩.

### References

Further information about **geoR/geoS** can be found at:
`http://www.maths.lancs.ac.uk/~ribeiro/geoR.html`

### See Also

`grf`, `plot.grf`, `lines`.

### Examples

```
sim <- grf(100, cov.pars=c(1, .25)) # simulates data
plot(variog(sim, max.dist=1))       # plot empirical variogram
lines(sim, max.dist=1)              # plot true variogram
```

---

## `lines.krige.bayes` *Add a (Bayesian) Estimated Variogram to a Plot*

---

### Description

Add a variogram estimate to an empirical variogram plot. The estimate is given by a summary (mean, mode or mean) of the posterior distribution returned by the funbction `krige.bayes`.

### Usage

```
lines.krige.bayes(obj, max.dist, length = 100,
             summary.posterior = c("mode", "median", "mean"), ...)
```

### Arguments

| | |
|---|---|
| `obj` | an object with the output of the function `krige.bayes`. |
| `max.dist` | maximum distance in the distance axis (X-axis). |
| `length` | number of points to compute the variogram values. |
| `summary.posterior` | specify which summary of the posterior distribution should be used as parameter estimation (mean, median or mode). |
| `...` | arguments to be passed to the function `lines`. |

### Value

A line with the estimated variogram plot is added to the plot in the current graphics device.

### Author(s)

Paulo J. Ribeiro Jr. ⟨paulojus@est.ufpr.br⟩
Peter J. Diggle ⟨p.diggle@lancaster.ac.uk⟩.

### References

Further information about **geoR/geoS** can be found at:
`http://www.maths.lancs.ac.uk/~ribeiro/geoR.html`

### See Also

`krige.bayes`, `lines`.

### Examples

```
#See examples in the documentation of the function krige.bayes().
```

---

`lines.variogram.envelope`
*Adds Envelopes Lines to a Variogram Plot*

---

### Description

Envelopes computed by `variog.model.env` or `variog.mc.env` are added to the current variogram plot.

### Usage

```
lines.variogram.envelope(obj, lty = 3, ...)
```

### Arguments

| | |
|---|---|
| `obj` | an object of the calss `"variogram.envelope"`, typically an output of the functions `variog.model.env` or `variog.mc.env`. |
| `lty` | line type. Defaults to 3. |
| `...` | arguments to be passed to the function `lines`. |

### Value

Lines with the lower and upper limits defining the variogram envelope are added to a plot in the current graphics device.

### Author(s)

Paulo Justiniano Ribeiro Jr. ⟨paulojus@est.ufpr.br⟩
Peter J. Diggle ⟨p.diggle@lancaster.ac.uk⟩.

### References

Further information about **geoR/geoS** can be found at:
`http://www.maths.lancs.ac.uk/~ribeiro/geoR.html`

### See Also

`variog` for variogram computation, `variog.model.env` and `variog.mc.env` for variogram envelopes computation, and `lines` for the generic function.

### Examples

```
if(is.R()) data(s100)
s100.vario <- variog(s100, max.dist = 1)
s100.ml <- likfit(s100, ini=c(.5, .5))
s100.mod.env <- variog.model.env(s100, obj.variog = s100.vario,
   model = s100.ml)
s100.mc.env <- variog.mc.env(s100, obj.variog = s100.vario)
plot(s100.vario)
lines(s100.mod.env)
lines(s100.mc.env, lwd=2)
```

---

## lines.variogram *Line with a Empirical Variogram*

---

### Description

A sample (empirical) variogram from an object computed using the geoR function `variog` is added to the current plot.

### Usage

```
lines.variogram(obj, max.dist = max(obj$u), type = "o",
                scaled = F, ...)
```

### Arguments

| | |
|---|---|
| `obj` | an object of the class `"variogram"`, typically an output from the function `variog`. |
| `max.dist` | maximum distance in the x-axis. The default is the maximum distance for which the sample variogram was computed. |
| `type` | type of line for the empirical variogram. The default is `"o"` (dots and lines). |
| `scaled` | If `TRUE` the variogram values are divided by the sample variance. This allows comparison between variograms from different data and/or variables. |
| `...` | other arguments to be passed to the function `lines`. |

### Details

Allows sample variogram(s) be added to a plot. Together with `lines.variogram` can be used to compare sample variograms for different data and/or variables.

### Value

A line with the empirical variogram is added to a plot in the current graphics device.

### Author(s)

Paulo Justiniano Ribeiro Jr. ⟨paulojus@est.ufpr.br⟩
Peter J. Diggle ⟨p.diggle@lancaster.ac.uk⟩.

### References

Further information about **geoR/geoS** can be found at:
`http://www.maths.lancs.ac.uk/~ribeiro/geoR.html`

### See Also

`variog`, `lines.variogram`, `lines.variomodel`, `variog.model.env`, codevariog.mc.env, codeplot.grf, `lines`.

---

## `lines.variomodel` *Line with a Variogram Model*

---

### Description

This function adds a theoretical and/or fitted variogram to the current plot (typically a sample variogram). The variogram model to be added is typically an output of a variogram estimation function, or a model specified by the user.

### Usage

```
lines.variomodel(obj, max.dist = obj$max.dist, length = 100, ...)
```

### Arguments

| | |
|---|---|
| `obj` | an object of the class `variomodel` which is a list containing information about the variogram model parameters. |
| `max.dist` | maximum distance (x-axis) to compute and draw the line with the variogram model. The default is the distance defined in the object `obj`. |
| `length` | number of points between `0` and `max.dist` to compute the values of the variogram model. |
| `...` | other arguments to be passed to the function `lines`. |

### Details

Allows theoretical and/or fitted variogram(s) be added to a plot. Together with `plot.variogram` can be used to compare sample variograms against fitted models returned by `olsfit`, `wlsfit` or `likfit`.

### Value

A line with a variogram model is added to a plot on the current graphics device. No values returned.

### Author(s)

Paulo Justiniano Ribeiro Jr. ⟨paulojus@est.ufpr.br⟩
Peter J. Diggle ⟨p.diggle@lancaster.ac.uk⟩.

### References

Further information about **geoR/geoS** can be found at:
`http://www.maths.lancs.ac.uk/~ribeiro/geoR.html`

### See Also

`plot.variogram`, `lines.variogram`, `olsfit`, `wlsfit`, `likfit`, `lines.`

### Examples

```
sim <- grf(100, cov.pars=c(1, .3))
# generates simulated data
vario <- variog(sim)
# computes empirical variogram
plot(vario)
# plots the empirical variogram
vario.wls <- wlsfit(vario, ini=c(1, .3), fix.nugget = T)
# estimate parameters
lines(vario.wls)
# adds fitted model to the plot
```

---

## wrappers *Wrappers for C functions in geoR*

---

### Description

These functions are *wrappers* for some of the C functions included in the **geoR** package. Typically the C code is directly called from the **geoR** functions.

### Usage

```
distdiag(coords)
loccoords(coords, locations)
diagquadraticformXAX(X, lowerA, diagA)
bilinearform(X, lowerA, diagA, Y)
```

### Arguments

| | |
|---|---|
| `coords` | an $n \times 2$ matrix with the data coordinates. |
| `locations` | an $n \times 2$ matrix with the coordinates of the prediction locations. |
| `lowerA` | a vector with the diagonal terms of the symmetric matrix A. |
| `diagA` | a vector with the diagonal terms of the symmetric matrix A. |
| `X` | a matrix with conforming dimensions. |
| `Y` | a matrix with conforming dimensions. |

### Value

| | |
|---|---|
| `loccoords` | returns a vector with distances between data points and prediction locations. |
| `distdiag` | returns a vector with distances between data locations, including the zero values for the distances between the point with itself. |
| `diagquadraticformXAX` | returns a vector with the diagonal term of the quadratic form of the type $XAX$. |

`bilinearform` returns a vector or a matrix with the terms of the quadratic form of the type $X'AX$.

normal-bracket41bracket-normal

### Author(s)

Paulo Justiniano Ribeiro Jr. ⟨paulojus@est.ufpr.br⟩
Peter J. Diggle ⟨p.diggle@lancaster.ac.uk⟩.

### References

Further information about **geoR/geoS** can be found at:
`http://www.maths.lancs.ac.uk/~ribeiro/geoR.html`

---

## `matern` *Computer Values of the Matern Correlation Function*

---

### Description

This function computes the values of the Matern correlation function for given distances and parameters.

### Usage

```
matern(u, phi, kappa)
```

### Arguments

| | |
|---|---|
| `u` | a vector, matrix or array with values of the distances between data points. |
| `phi` | value of the range parameter $\phi$. |
| `kappa` | value of the smoothness parameter $\kappa$. |

### Details

The Matern model is defined as:

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

where $\phi, \kappa$ are parameters and $K_{\kappa}(\cdot)$ denotes the modified Bessel function of the third kind of order $\kappa$. The family is valid for $\phi > 0$ and $\kappa > 0$.

### Value

A vector matrix or array, according to the argument `u`, with the values of the Matern correlation function for the given distances.

### Author(s)

Paulo J. Ribeiro Jr. ⟨paulojus@est.ufpr.br⟩
Peter J. Diggle ⟨p.diggle@lancaster.ac.uk⟩.

### References

Further information about **geoR/geoS** can be found at:
`http://www.maths.lancs.ac.uk/~ribeiro/geoR.html`

### See Also

`cov.spatial`, `besselK`.

### Examples

```
dval <- seq(0,1,l=101)
#
# Models with fixed range and varying smoothness parameter
#
mat.5 <-  matern(dval, phi = .25, kappa = 0.5)
mat1 <-  matern(dval, phi = .25, kappa = 1)
mat2 <-  matern(dval, phi = .25, kappa = 2)
mat3 <-  matern(dval, phi = .25, kappa = 3)
plot(0:1, 0:1, type="n", xlab="distance", ylab=expression(rho(h)),
main=expression(paste("varying  ", kappa, "  and fixed  ", phi)))
lines(dval, mat.5, lty=2)
lines(dval, mat1)
lines(dval, mat2, lwd=2, lty=2)
lines(dval, mat3, lwd=2)
legend(0.6,1, c(expression(kappa == 0.5), expression(kappa == 1),
    expression(kappa == 2), expression(kappa == 3)),
    lty=c(2,1,2,1), lwd=c(1,1,2,2))
#
# Correlations with equivalent "practical range"
# and varying smoothness parameter
#
mat.5 <-  matern(dval, phi = .25, kappa = 0.5)
mat1 <-  matern(dval, phi = .188, kappa = 1)
mat2 <-  matern(dval, phi = .140, kappa = 2)
mat3 <-  matern(dval, phi = .117, kappa = 3)
plot(0:1, 0:1, type="n", xlab="distance", ylab=expression(rho(h)),
    main="models with the equivalent \"practical\" range")
lines(dval, mat.5, lty=2)
lines(dval, mat1)
lines(dval, mat2, lwd=2, lty=2)
lines(dval, mat3, lwd=2)
legend(0.5,1, c(expression(paste(kappa == 0.5, "  and  ",
    phi == 0.250)),
    expression(paste(kappa == 1, "  and  ", phi == 0.188)),
    expression(paste(kappa == 2, "  and  ", phi == 0.140)),
    expression(paste(kappa == 3, "  and  ", phi == 0.117))),
    lty=c(2,1,2,1), lwd=c(1,1,2,2))
```

### Description

This function adapts the R function `nlm` to allow for constraints (upper and/or lower bounds) in the values of the parameters.

### Usage

```
nlmP(objfunc, params, lower=rep(-Inf, length(params)),
     upper=rep(+Inf, length(params)), ...)
```

### Arguments

| | |
|---|---|
| `objfunc` | the function to be minimized. |
| `params` | starting parameter values for the minimization. |
| `lower` | lower bounds for the variables. Defaults to $-Inf$. |
| `upper` | upper bounds for the variables. Defaults to $-Inf$. |
| ... | further arguments to be passed to the function `nlm`. |

### Details

Constraints on the parameter values are internally imposed by using exponential, logarithmic, and logit transformation of the parameter values.

### Value

The output is the same as for the function `nlm`.

### Author(s)

Patrick E. Brown ⟨p.brown@lancaster.ac.uk⟩
and adapted and included in **geoR** by
Paulo Justiniano Ribeiro Jr. ⟨paulojus@est.ufpr.br⟩ and
Peter J. Diggle ⟨p.diggle@lancaster.ac.uk⟩

### References

Further information about **geoR** can be found at:
`http://www.maths.lancs.ac.uk/~ribeiro/geoR.html`

### See Also

`nlm`, `optim`.

`olsfit` *Variogram Fitting by Ordinary Least Squares*

## Description

Fits a variogram model to a empirical variogram. Variogram parameters are estimated by ordinary least squares.

## Usage

```
olsfit(vario, ini.cov.pars,
       cov.model = c("exponential", "matern", "gaussian",
                     "spherical", "circular", "cubic", "wave",
                     "powered.exponential", "cauchy", "gneiting",
                     "gneiting.matern", "pure.nugget"),
       fix.nugget = FALSE, nugget = 0, kappa = NULL,
       simul.number = NULL, max.dist = "all",
       minimisation.function = c("optim", "nlm", "nls"), lower = 0,
       messages.screen = TRUE)
```

## Arguments

| | |
|---|---|
| `vario` | an object of the class `"variogram"`, typically an output of the function `variog`. The object is a list with information about the empirical variogram. |
| `ini.cov.pars` | initial values for the covariance parameters: $\sigma^2$ (partial sill) and $\phi$ (range parameter). See `DETAILS` below. |
| `cov.model` | a string with the name of the correlation function. See function call for the options. Defaults to `"exponential"`. |
| `fix.nugget` | logical, indicating whether the parameter $\tau^2$ (nugget variance) should be regarded as fixed (`fix.nugget = TRUE`) or is to be estimated (`fix.nugget = FALSE`). Defaults to `FALSE`. |
| `nugget` | value for the nugget parameter. Regarded as a fixed values if `fix.nugget = TRUE` or as a initial value for the minimization algorithm if `fix.nugget = FALSE`. Defaults to zero. |
| `kappa` | fixed value of the smoothness parameter, only required by the following correlation functions: `"matern"`, `"powered.exponential"`, `"gneiting"` and `"gneiting.matern"`. |
| `simul.number` | number of simulation. To be used when the object passed to the argument `vario` has empirical variograms for more than one data-set (or simulation). Indicates to which one to model will be fitted. |
| `max.dist` | maximum distance considered when fitting the variogram. Defaults to the maximum distance found in the object defined in the argument `vario`. |
| `minimisation.function` | minimization function used to estimate the parameters. See function call for options. Defaults to the first option. |
| `lower` | lower limits for the parameters. Defaults to zero. |

`messages.screen`
: logical. Indicates whether or not messages are printed on the screen (or other output device) while the function is running.

`...`
: further parameters to be passed to the minimization function. Typically `control` type arguments which controls the behavior of the minimization algorithm. See documentation for the selected minimisation function for further details.

## Details

### INITIAL VALUES

The algorithms for minimisation functions require initial values for the parameters.

A unique initial value is used if a vector is provided in the argument `ini.cov.pars`. The elements are initial values for $\sigma^2$ and $\phi$, respectively. This vector is concatenated with the value of the argument `nugget` if `fix.nugget = FALSE`.

Several initial values can also be provided. If so, first the function finds the one which minimizes the loss function and this is used as the initial value for the minimization algorithm. Multiple initial values can be passed by providing a matrix in the argument `ini.cov.pars` and/or, if `fix.nugget = FALSE`, providing a vector with length greater than one for the argument `nugget`. If `ini.cov.pars` is a matrix the first column has values of $\sigma^2$ and the second has values of $\phi$.

## Value

An object of the `class "variomodel"` which is list with the following components:

`nugget`
: value of the nugget parameter. An estimated value if `fix.nugget = FALSE` or a fixed value if `fix.nugget = TRUE`.

`cov.pars`
: a 2 elements vector with estimated values of the covariance parameters, partial sill and range parameter, respectively.

`cov.model`
: a string with the name of the correlation function.

`kappa`
: fixed value of the smoothness parameter.

`value`
: minimized value of the loss function.

`max.dist`
: maximum distance considered in the variogram fitting.

`minimisation.function`
: minimization function used.

`message`
: status messages returned by the function.

`method`
: a string `"OLS"` indicating the estimation method which was used.

`call`
: the function call.

## Author(s)

Paulo Justiniano Ribeiro Jr. ⟨paulojus@est.ufpr.br⟩
Peter J. Diggle ⟨p.diggle@lancaster.ac.uk⟩.

## References

Further information about **geoR** can be found at:
`http://www.maths.lancs.ac.uk/~ribeiro/geoR.html`

### See Also

`cov.spatial` for a detailed description of the available correlation (veriogram) functions, `olsfit` for weighted least squares variogram fit, `likfit` for maximum and restricted maximum likelihood estimation, `lines.variomodel` for graphical output of the fitted model. For details on the minimization functions see `optim`, `nlm` and `nls`.

### Examples

```
if(is.R()) data(s100)
vario100 <- variog(s100, max.dist=1)
ini.vals <- expand.grid(seq(0,1,l=5), seq(0,1,l=5))
ols <- olsfit(vario100, ini=ini.vals, fix.nug=TRUE)
summary(ols)
plot(vario100)
lines(ols)
```

---

## parana *Rainfall Data from Parana State, Brasil*

---

### Description

This data-set was used by Diggle and Ribeiro (2001) to illustrate the methods discussed in the paper. The analysis was performed using the package `geoR`.

The data refers to average rainfall over different years for the period May-June (dry-season). It was collected at 143 recording stations throughout Parana State, Brasil.

### Usage

```
data(parana)
```

### Format

Three objects are provided:

maijun a list with coordinates of the recording stations and the May-June average data.

borders a matrix with the coordinates defining the borders of Parana state.

- loci.papera matrix with the coordinates of the four prediction locations discussed in the paper.

### Source

The data were collected at several recording stations at Parana State, Brasil. The stations belongs to the following companies: COPEL, IAPAR, DNAEE, SUREHMA and INEMET.

The data base was organized by Laura Regina Bernardes Kiihl (IAPAR, Instituto Agronomico do Parana, Londrina, Brazil) and the fraction of the data included in this data-set was provided by Jacinta Loudovico Zamboti (Universidade Estadual de Londrina, Brazil). The coordinates of the borders of Parana State were provided by Joao Henrique Caviglione (IAPAR).

### References

Diggle, P.J. and Ribeiro Jr, P.J. (2001). Bayesian inference in Gaussian model-based geostatistics. *Geographical and Environmental Modeling* (to appear)

Further information about **geoR/geoS** can be found at:
`http://www.maths.lancs.ac.uk/~ribeiro/geoR.html`

---

`plot.geodata` *Exploratory Geostatistical Plots*

---

### Description

This function produces an $2 \times 2$ graphics display with four plots: the first indicates spatial locations, the second is a 3-D plot with spatial locations and associated data values (or a histogram of the data), and the last two are plots of the data against the $X$ and $Y$ coordinates.

### Usage

```
plot.geodata(geodata, coords=geodata$coords, data = geodata$data,
     trend="cte", lambda = 1, col.data = 1, weights.divide = FALSE,
     window.new = FALSE, ...)
```

### Arguments

| | |
|---|---|
| `geodata` | a list containing elements `coords` and `data` described next. Typically an object of the class `"geodata"` - a **geoR** data-set. If not provided the arguments `coords` and `data` must be provided instead. |
| `coords` | an $n \times 2$ matrix containing in each row Euclidean coordinates of the $n$ data locations. By default it takes the element `coords` of the argument `geodata`. |
| `data` | a vector with data values. By default it takes the element `data` of the argument `geodata`. |
| `trend` | specifies the mean part of the model. The options are: `"cte"` (constant mean - default option), `"1st"` (a first degree polynomial on the coordinates), `"2nd"` (a second degree polynomial on the coordinates), or a formula of the type `~X` where `X` is a matrix with the covariates (external trend). If provided the trend is "removed" using the function `lm` and the residuals are plotted. |
| `lambda` | Box-Cox transformation parameter. Two particular cases are $\lambda = 1$ which corresponds to no transformation and $\lambda = 0$ corresponding to the log-transformation. |
| `col.data` | indicates the number of the column of the data to be plotted. Only valid if more than one data-set is available i.e., if the argument `data` ins a matrix. |
| `weights.divide` | if a vector of weights with the same length as the data is provided each data is divided by the corresponding element in this vector. Defaults to `NULL`. |
| `window.new` | logical. If `TRUE` a new graphic device is openned, otherwise the current one is used. Defaults to `FALSE`. |
| `...` | further arguments to be passed to the function `scatterplot3d`. |

### Details

By default, this function requires the package *scatterplot3d* in order to produce a 3-D plot with data locations and coordinates. If this package is not available an histogram replaces the 3-D plot.

### Value

A plot is produced on the graphics device. No values returned.

### Author(s)

Paulo J. Ribeiro Jr. ⟨paulojus@est.ufpr.br⟩
Peter J. Diggle ⟨p.diggle@lancaster.ac.uk⟩.

### References

Further information about **geoR** can be found at:
`http://www.maths.lancs.ac.uk/~ribeiro/geoR.html`

### See Also

`points.geodata`, `scatterplot3d`.

### Examples

```
if(is.R()) data(s100)
plot(s100)
```

---

## plot.grf *Plots Variograms for Simulated Data*

---

### Description

This function plots variograms for simulated geostatistical data generated by the function `grf`.

### Usage

```
plot.grf(obj, model.line = TRUE, plot.grid = FALSE,
         ylim = "default", ...)
```

### Arguments

| | |
|---|---|
| `obj` | an object of the class `grf`, typically an output of the function `grf`. |
| `model.line` | if `TRUE` the true variogram model is added to the plot with the sample variogram(s). |
| `plot.grid` | if `TRUE` a plot with data locations is shown |
| `ylim` | limits for the y-axis. The default is the maximum value of the variogram among all of the simulations. |
| `...` | further arguments to be passed to the functions `variog` and `plot`. |

### Details

Plot with sample variogram(s) for simulated data are produced. True variogram model used to generate the simulations is also plotted if `model = TRUE`. Data locations are plotted first if `grid = TRUE`.

### Value

A plot with the variogram(s) is produced in the output device. No values returned.

### Author(s)

Paulo Justiniano Ribeiro Jr. ⟨paulojus@est.ufpr.br⟩
Peter J. Diggle ⟨p.diggle@lancaster.ac.uk⟩

### References

Further information about **geoR/geoS** can be found at:
`http://www.maths.lancs.ac.uk/~ribeiro/geoR.html`

### See Also

`grf`, `plot.variogram`, `variog`, `plot`.

### Examples

```
par(mfrow=c(2,1))
sim1 <- grf(100, cov.pars=c(10, .25))
# generates simulated data
plot(sim1, grid=T)
# plots the locations and the sample true variogram model
#
par(mfrow=c(1,1))
sim2 <- grf(100, cov.pars=c(10, .25), nsim=10)
# generates 10 simulated data
plot(sim1)
# plots sample variograms for all simulations with the true model
```

---

## `plot.proflik` *Plots Profile Likelihoods*

---

### Description

This function produces plots with the profile likelihoods computed by the function `proflik`.

### Usage

```
plot.proflik(obj.proflik, pages = c("user", "one", "two"),
     uni.only = FALSE, bi.only = F,
     type.bi = c("contour", "persp"), conf.int = c(0.9, 0.95),
     yaxis.lims = c("conf.int", "as.computed"),
     by.col = TRUE, log.scale = FALSE,
     par.mar.persp = c(0, 0, 0, 0), ...)
```

### Arguments

| | |
|---|---|
| `obj.proflik` | An object with results from the function `proflik`. |
| `pages` | specify whether the results will be display in the graphics device: as it is set (default option), with all the profiles in one page or in two pages with 1-D profiles in the first and 2-D profiles in the second. |
| `uni.only` | only 1-D profile are plotted even if the object contains results of the 2-D profiles. |
| `bi.only` | only 2-D profile are plotted even if the object contains results of the 1-D profiles. |
| `type.bi` | Type of plot for the 2-D profiles: contour plot (the default) or perspective plot. |
| `conf.int` | levels of the (approximated) confidence intervals. Defaults to 90% and 95%. |
| `yaxis.lims` | defines the lower limits for the y-axis in the 1-D plots. If `"conf.int"` the limit is determined by the level of the confidence interval (the default) otherwise will be determined by the lowest computed value. |
| `by.col` | plots arranged by columns in a multiple graphics device. |
| `log.scale` | plots the x-axis in the logarithmic scale. Defaults to `FALSE`. |
| `...` | additional arguments to be passed to the plot functions `plot`, `contour` or `persp`. |

### Value

Produces a plot of the profile likelihoods on the graphics output device. No value is returned.

### Author(s)

Paulo Justiniano Ribeiro Jr. ⟨paulojus@est.ufpr.br⟩
Peter J. Diggle ⟨p.diggle@lancaster.ac.uk⟩.

### References

Further information about **geoR/geoS** can be found at:
`http://www.maths.lancs.ac.uk/~ribeiro/geoR.html`

### See Also

`proflik`, `contour`, `persp`.

### Examples

```
# see examples for the function proflik()
```

`plot.variogram` *Plot Empirical Variogram*

## Description

Plots sample (empirical) variogram from an object computed using the function `variog`.

## Usage

```
plot.variogram(obj, max.dist = max(obj$u), ylim = "default",
               type = "b", scaled = F, var.lines = F,
               envelope.obj = NULL, bin.cloud = F, ...)
```

## Arguments

| | |
|---|---|
| `obj` | an object of the class `"variogram"`, typically an output from the function `variog`. |
| `max.dist` | maximum distance in the x-axis. The default is the maximum distance for which the sample variogram was computed. |
| `ylim` | limits for the variogram values in the y-axis. The default is from 0 to the maximum value in `obj$v`. |
| `type` | type of line for the empirical variogram. The default is `"b"` (dots and lines). |
| `scaled` | If `TRUE` the variogram values are divided by the sample variance. This allows comparison between variograms from different data and/or variables. |
| `var.lines` | If `TRUE` a horizontal line is drawn at the value of the variance of the data (if `scaled = F`) or at 1 (if `scaled = T`). |
| `envelope.obj` | adds a variogram envelope previously computed by the functions `variog.model.env` or `variog.mc.env`. |
| `bin.cloud` | plots the variogram cloud if the sample variogram from `variog` was computed with the option `option = "cloud"`. |
| `...` | other arguments to be passed to the function `plot`. |

## Details

This function allows visualization of sample variogram and together with `lines.variogram` can be used to compare sample variograms for different data and/or variables. Furthermore, together with `lines.variomodel` can be used to compare theoretical and/or fitted variogram models against the empirical variogram.

## Value

Produces a plot with the sample variogram on the graphics device.

## Author(s)

Paulo Justiniano Ribeiro Jr. ⟨paulojus@est.ufpr.br⟩
Peter J. Diggle ⟨p.diggle@lancaster.ac.uk⟩

### References

Further information about **geoR/geoS** can be found at:
`http://www.maths.lancs.ac.uk/~ribeiro/geoR.html`

### See Also

`variog` for variogram calculations, `lines.variogram` and `lines.variomodel` for adding lines to the current plot, `variog.model.env` and `variog.mc.env` for variogram envelops computation, and `plot` for generic plot function.

### Examples

```
sim <- grf(100, cov.pars=c(1, .2))# generates simulated data
vario <- variog(sim, max.dist=1)  # computes sample variogram
par(mfrow=c(1,2))
plot(vario)                       # the sample variogram
plot(vario, scaled=T)             # the scaled sample variogram
```

---

## points.geodata *Plots Spatial Locations and Data Values*

---

### Description

This function produces a plot with points indicating the data locations. Arguments can control the points sizes, patterns and colors. These can be set to be proportional to data values, ranks or quantiles. Alternatively, points can be added to the current plot.

### Usage

```
points.geodata(geodata, coords=geodata$coords, data=geodata$data,
               data.col = 1,
               pt.sizes=c("data.proportional","rank.proportional",
                    "quintiles", "quartiles", "deciles", "equal"),
               cex.min, cex.max, pch.seq, col.seq,
               add.to.plot = FALSE,
               round.quantiles = FALSE, graph.pars = FALSE, ...)
```

### Arguments

| | |
|---|---|
| `geodata` | a list containing elements `coords` and `data` described next. Typically an object of the class `"geodata"` - a **geoR** data-set. If not provided the arguments `coords` and `data` must be provided instead. |
| `coords` | an $n \times 2$ matrix containing coordinates of the $n$ data locations in each row. Defaults to `geodata$coords`. |
| `data` | a vector or matrix with data values. If a matrix is provided each column is regarded as one variable or realization. Defaults to `geodata$data`. |
| `data.col` | the number of the data column. Only used if `data` is a matrix with columns corresponding to different variables or simulations. |
| `pt.sizes` | defines the point sizes. See function call and `DETAILS` below for the available options. Default to `pt.sizes = "data.proportional"`. |

| | |
|---|---|
| `cex.min` | minimum value for the graphical parameter `cex`. Defines the size of the point corresponding the minimum of the data. Defaults to 0.5. |
| `cex.max` | maximum value for the graphical parameter `cex`. Defines the size of the point corresponding the maximum of the data. If `pt.sizes = "equal"` it is used to set the value for the graphical parameter `cex`. Defaults to 1.5. |
| `pch.seq` | number(s) defining the graphical parameter `pch`. |
| `col.seq` | number(s) defining the colors in the graphical parameter `cor`. |
| `add.to.plot` | logical. If `TRUE` the points are added to a current plot, otherwise a new plot is created. Defaults to `FALSE`. |
| `round.quantiles` | logical. Defines whether or not the values of the quantiles should be rounded. Defaults to `FALSE`. |
| `graph.pars` | logical. If `TRUE` the graphics parameters used to produce the plots are returned. Defaults to `FALSE`. |
| `...` | further arguments to be passed to the function `plot`, if `add.to.plot = FALSE`, or for the function `points`, if `add.to.plot = TRUE`. |

## Details

The points can have different sizes according to the argument `pt.sizes`. The options are:

| | |
|---|---|
| "data.proportional" | sizes proportional to the data values. |
| "rank.proportional" | sizes proportional to the rank of the data. |
| "quintiles" | uses five different sizes according to the quintiles of the data. |
| "quartiles" | uses four different sizes according to the quartiles of the data. |
| "deciles" | uses ten different sizes according to the deciles of the data. |
| "equal" | all point with the same size. |

For cases where points have different sizes the arguments `cex.min` and `cex.max` set the minimum and the maximum point sizes. Additionally, `pch.seq` can set different patterns for the points and `col.seq` can define colors. For example, different colors can be used for quartiles, quintiles and deciles while a sequence of gray tones (or s color sequence) can be used for point sizes proportional to the data or their ranks.

## Value

A plot is created or points are added on the current graphics device.
By default no value is returned. However, if `graph.pars = TRUE`, a list with graphical parameters used to produce the plot is returned. According to the input, this is a list contains some or all of the following components:

| | |
|---|---|
| `quantiles` | the values of the quantiles used to divide the data (if required). |
| `cex` | the values of the graphics expansion parameter `cex`. |
| `col` | the values of the graphics color parameter `col`. |
| `pch` | the values of the graphics pattern parameter `pch`. |

## Author(s)

Paulo J. Ribeiro Jr. ⟨paulojus@est.ufpr.br⟩
Peter J. Diggle ⟨p.diggle@lancaster.ac.uk⟩.

### References

Further information about geoR can be found at:
http://www.maths.lancs.ac.uk/~ribeiro/geoR.html

### See Also

plot.geodata for another display of the data and points and plot for information on the generic R functions. The documentation of par provides details on graphical parameters. For a diversity of color sequences check the functions gray, rainbow.

### Examples

```
if(is.R()) data(s100)
points.geodata(s100, xlab="Coord X", ylab="Coord Y")
points.geodata(s100, xlab="Coord X", ylab="Coord Y",
               pt.size="rank.prop")
points.geodata(s100, xlab="Coord X", ylab="Coord Y", cex.max=1.7,
               col=gray(seq(1, 0.1, l=100)), pt.siz="equal")
               # function gray() works only for R
points.geodata(s100, pt.sizes="quintile", xlab="Coord X",
               ylab="Coord Y")
```

---

## polygrid *Coordinates of Points Inside a Polygon*

---

### Description

This function builds a rectangular grid and extracts points which are inside of an internal polygonal region.

### Usage

```
polygrid(xgrid, ygrid, poly, vec.inout = FALSE)
```

### Arguments

| | |
|---|---|
| xgrid | grid values at the $X$-direction. |
| ygrid | grid values at the $Y$-direction. |
| poly | a matrix with the polygon coordinates. |
| vec.inout | logical. If TRUE a logical vector is included in the output indicating whether each point of the grid is inside the polygon. Defauts to FALSE. |

### Details

This function requires the package splancs.

First the function creates a grid using the function expand.grid and then it uses the function inout from the package splancs to extract the points of the grid which are inside the polygon.

Within the package geoR, this function is typically used to select points to perform spatial prediction using krige.bayes or krige.conv, and to produce plots with the results.

### Value

A list with components:

| | |
|---|---|
| `xypoly` | a matrix with the coordinates of the points inside the polygon. |
| `vec.inout` | logical, a vector indicating whether each point of the grid is inside the polygon. Only returned if `vec.inout = TRUE`. |

### Author(s)

Paulo Justiniano Ribeiro Jr. ⟨paulojus@est.ufpr.br⟩
Peter J. Diggle ⟨p.diggle@lancaster.ac.uk⟩.

### References

Further information about **geoR/geoS** can be found at:
`http://www.maths.lancs.ac.uk/~ribeiro/geoR.html`

### See Also

`expand.grid`, `inout`.

### Examples

```
poly <- matrix(c(.2, .8, .7, .1, .2, .1, .2, .7, .7, .1), ncol=2)
plot(0:1, 0:1, type="n")
lines(poly)
poly.in <- polygrid(seq(0,1,l=11), seq(0,1,l=11), poly, vec=T)
points(poly.in$xy)
```

---

## `proflik` *Computes Profile Likelihoods*

---

### Description

This function computes profile likelihoods for model parameters which should have been estimated previously using the function `likfit`.

### Usage

```
proflik(obj.likfit, geodata, coords = geodata$coords,
        data = geodata$data, sill.values, range.values,
        nugget.values, nugget.rel.values, lambda.values,
        sillrange.values = TRUE, sillnugget.values = TRUE,
        rangenugget.values = TRUE, sillnugget.rel.values = FALSE,
        rangenugget.rel.values = FALSE, silllambda.values = FALSE,
        rangelambda.values = TRUE,  nuggetlambda.values = FALSE,
        nugget.rellambda.values = FALSE,
        uni.only = TRUE, bi.only = FALSE,
        minimisation.function = c("optim", "nlmP"), ...)
```

**Arguments**

| | |
|---|---|
| `obj.likfit` | an object with the output from the function `likfit`. |
| `geodata` | a list containing elements `coords` and `data` described next. Typically an object of the class `"geodata"` - a **geoR** data-set. If not provided the arguments `coords` and `data` must be provided instead. |
| `coords` | an $n \times 2$ matrix containing in each row Euclidean coordinates of the $n$ data locations. By default it takes the element `$coords` of the argument `geodata`. |
| `data` | a vector with data values. By default it takes the element `$data` of the argument `geodata`. |
| `sill.values` | values of the sill parameter $\sigma^2$ for which the profile likelihood will be computed. |
| `range.values` | values of the range parameter $\phi$ for which the profile likelihood will be computed. |
| `nugget.values` | values of the nugget parameter $\tau^2$ for which the profile likelihood will be computed. Only to be used if the model was fitted using the function `likfit` with the option `fix.nugget = FALSE`. |
| `nugget.rel.values` | values of the relative nugget parameter $\tau^2_R$ for which the profile likelihood will be computed. Only to be used if the model was fitted using the function `likfit` with the option `fix.nugget = FALSE`. |
| `lambda.values` | values of the Box-Cox transformation parameter $\lambda$ for which the profile likelihood will be computed. Only to be used if the model was fitted using the function `likfit` with the option `fix.lambda = FALSE`. |
| `sillrange.values` | `TRUE` or `FALSE` indicating whether or not the 2-D profile likelihood should be computed. Only used if `uni.only = FALSE`. |
| `sillnugget.values` | as above. |
| `rangenugget.values` | as above. |
| `sillnugget.rel.values` | as above. |
| `rangenugget.rel.values` | as above. |
| `silllambda.values` | as above. |
| `rangelambda.values` | as above. |
| `nuggetlambda.values` | as above. |
| `nugget.rellambda.values` | as above. |
| `uni.only` | as above. |
| `bi.only` | as above. |
| `minimisation.function` | minimization function to be used. Defaults to `optim`. |
| `...` | additional parameters to be passed to the minimization function. |

### Details

The functions `proflik.*` are auxiliary functions to compute the profile likelihoods and are internally called by the minimization functions when estimating the model parameters.

### Value

An object of the class `"proflik"` which is a list where each element contains values of a parameter (or a pair of parameters for 2-D profiles) and the corresponding value of the profile likelihood. The components of the output will varies according to the input options.

A graphical display of the output with the profiles plots is produced by the function `plot.proflik`.

### Note

1. Profiles for Gaussian Random Fields are usually uni-modal. Unusual shapes (ups and downs) can be due to the lack of the convergence for particular values of the parameter(s). It might be necessary to pass `control` arguments to the minimization functions using the argument `....` See documentation of the functions `optim` and/or `nlm` for further details. It's also advisable to try the different options for the `minimisation.function` argument.
2. 2-D profiles can be computed by setting the argument `uni.only = FALSE`. However, be sure first that the 2-D profiles are really wanted since it can be time demanding. The reason for that is due to the fact that computation is performed on a grid determined by the cross-product of the values defining the 1-D profiles.
3. There is no "default" way to find reasonable values for the x-axis. They must be found in a "try-and-error" exercise. It's recommended to use short sequences in the initial attempts. This is illustrated in the examples below.

### Author(s)

Paulo Justiniano Ribeiro Jr. ⟨paulojus@est.ufpr.br⟩
Peter J. Diggle ⟨p.diggle@lancaster.ac.uk⟩.

### References

Further information about **geoR/geoS** can be found at:
`http://www.maths.lancs.ac.uk/~ribeiro/geoR.html`

### See Also

`plot.proflik` for graphical output, `likfit` for the parameter estimation, `optim` and `nlm` for further details about the minimization functions.

### Examples

```
if(is.R()) data(s100)
ml <- likfit(s100, ini=c(.5, .5), fix.nug=T)
# a first atempt to find reasonable values for the x-axis:
prof <- proflik(ml, s100, sill.values=seq(0.5, 1.5, l=4),
                range.val=seq(0.1, .5, l=4))
par(mfrow=c(1,2))
plot(prof)
# a nicer setting and now including 2-D profiles:
```

```
prof <- proflik(ml, s100, sill.values=seq(0.45, 2, l=16),
                range.val=seq(0.1, .55, l=16), uni.only=F)
par(mfrow=c(2,2))
plot(prof, nlevels=16)

par(mfrow=c(1,1))
```

---

`read.geodata` *Reads and Converts Data to geoR Format*

---

### Description

Reads data from a *ASCII* file and converts it to an object of the `class geodata`, the standard data format for the **geoR** package.

### Usage

```
read.geodata(file, header = FALSE, coords.col = 1:2, data.col = 3,
             data.names = NULL, covar.col = NULL,
             covar.names = "header", ...)
```

### Arguments

| | |
|---|---|
| `file` | a string with the name of the *ASCII* file. |
| `header` | logical. Indicates whether the variables names shoud be read from the first line of the input file. |
| `coords.col` | a vector with the numbers of the columns containing the coordinates. |
| `data.col` | a scalar or vector with the number of the column(s) containing the data. |
| `data.names` | a string or vector of strings with names for the data columns. Only valid if there is more than one column of data. By default the names in the original object are used. |
| `covar.col` | optional. A scalar or vector with the number of the column(s) with the values of the covariate(s). |
| `covar.names` | a string or vector of strings with the name(s) of the covariates. By default the names in the original object are used. |
| `...` | further arguments to `read.table`. |

### Details

The function `read.table` is used to read the data from the *ASCII* file and then `as.geodata` is used to convert to an object of the `class geodata`.

### Value

An object of the `class geodata` as described in the documentation for the function `as.geodata`.

### Author(s)

Paulo Justiniano Ribeiro Jr. ⟨paulojus@est.ufpr.br⟩
Peter J. Diggle ⟨p.diggle@lancaster.ac.uk⟩.

### References

Further information about **geoR/geoS** can be found at:
`http://www.maths.lancs.ac.uk/~ribeiro/geoR.html`

### See Also

`as.geodata` to convert existing R objects, `read.table`, the basic R for reading *ASCII* files, and `list`.

---

`s100 and s121` *Simulated Data-Sets Illustrating the Usage of the Package geoR*

---

### Description

These two simulated data sets are the ones used in the Technical Report which describes the package **geoR** (see reference below). They are also used in several examples from the package documentation.

### Usage

```
data(s100)

data(s121)
```

### References

Ribeiro Jr, P.J. and Diggle, P.J. (1999). geoS: A geostatistical library for S-PLUS. Technical report ST-99-09, Dept of Maths and Stats, Lancaster University.

Further information about **geoR/geoS** can be found at:
`http://www.maths.lancs.ac.uk/~ribeiro/geoR.html`

---

`summary.likGRF` *Summarizes Parameter Estimation Results for Gaussian Random Fields*

---

### Description

Summarizes results returned by the function `likfit`. Functions are *methods* for `class likGRF` and `summary.likGRF`.

### Usage

```
summary(obj, ...)
print(summary.likGRF.obj, ...)
print(obj, ...)
```

### Arguments

| | |
|---|---|
| `obj` | an object of `class` *likGRF*, typically a result of a call to `likfit`. |
| ... | extra arguments for `print`. A commonly used argument is `digits` which specifies the number of digits for the numerical output. The default for is given by `options()$digits`. |

### Details

The model specification with fixed and estimated parameters is printed by `summary.likGRF`. A simplified summary of the parameter estimation is printed by `print.summary.likGRF`.

### Value

`print.likGRF` prints the parameter estimates and the value of the maximized likelihood.
`summary.likGRF` returns a list with main results of a call to `likfit`.
`print.summary.likGRF` prints these results on the screen (or other output device) in a *"nice"* way.

### Author(s)

Paulo Justiniano Ribeiro Jr. ⟨paulojus@est.ufpr.br⟩
Peter J. Diggle ⟨p.diggle@lancaster.ac.uk⟩.

### References

Further information about **geoR/geoS** can be found at:
`http://www.maths.lancs.ac.uk/~ribeiro/geoR.html`

### See Also

`likfit`, `print`, `summary`.

### Examples

```
# See examples for the function likfit()
```

---

## summary.variomodel *Summarize Results of Variogram Estimation*

---

### Description

This function prints a summary of the parameter estimation results given by `olsfit` and `wlsfit`.

### Usage

```
summary.variomodel(obj)
```

### Arguments

| | |
|---|---|
| `obj` | an object which contains the output of one of the functions: `olsfit` or `wlsfit`. |

### Value

Prints a summary of the estimation results on the screen or any other current output device.

### Author(s)

Paulo J. Ribeiro Jr. ⟨paulojus@est.ufpr.br⟩
Peter J. Diggle ⟨p.diggle@lancaster.ac.uk⟩.

### References

Further information about **geoR/geoS** can be found at:
`http://www.maths.lancs.ac.uk/~ribeiro/geoR.html`

### See Also

The functions `olsfit`, `wlsfit` and `likfit` for parameter estimation.

### Examples

```
if(is.R()) data(s100)
s100.vario <- variog(s100, max.dist=1)
wls <- wlsfit(s100.vario, ini=c(.5, .5), fix.nugget = TRUE)
summary(wls)
```

---

## `trend.spatial` *Prepare Trend Matrix*

---

### Description

This function produces a trend matrix according to the specification of the mean part of the model.

### Usage

```
trend.spatial(trend, coords = NULL)
```

### Arguments

| | |
|---|---|
| `trend` | specifies the mean part of the model. The options are: `"cte"` (constant mean - default option), `"1st"` (a first degree polynomial on the coordinates), `"2nd"` (a second degree polynomial on the coordinates), or a formula of the type `~X` where `X` is a matrix with the covariates (external trend). |
| `coords` | an $n \times 2$ matrix containing in each row Euclidean coordinates. Needs to be provided only if `trend = "1st"` or `trend="2nd"`. |

### Value

An $n \times p$ *trend* matrix where $n$ is the number of spatial locations and $p$ is the number of mean parameters in the model.

### Note

This is an auxiliary function for the **geoR** functions. Typically, it is called by other functions in the package.

### Author(s)

Paulo J. Ribeiro Jr. ⟨paulojus@est.ufpr.br⟩
Peter J. Diggle ⟨p.diggle@lancaster.ac.uk⟩.

### References

Further information about **geoR/geoS** can be found at:
`http://www.maths.lancs.ac.uk/~ribeiro/geoR.html`

---

## varcov.spatial *Computes Covariance Matrix and/or Related Results*

---

### Description

This function computes the covariance matrix for a set of spatial locations, for given parameters. According to the input other results related to the covariance matrix, such as decompositions, determinants, inverse. etc are returned.

### Usage

```
varcov.spatial(coords = NULL, dists.lowertri = NULL,
               cov.model = c("exponential", "matern", "gaussian",
                      "spherical", "circular", "cubic", "wave",
                      "powered.exponential", "cauchy", "gneiting",
                      "gneiting.matern", "pure.nugget"),
               kappa = NULL, nugget = 0,
               cov.pars = stop("no cov.pars argument"),
               inv = FALSE, det = FALSE,
               func.inv = c("cholesky", "eigen", "svd", "solve"),
               scaled = FALSE,  only.decomposition = FALSE,
               sqrt.inv = FALSE, try.another.decomposition = TRUE,
               only.inv.lower.diag = FALSE)
```

### Arguments

| | |
|---|---|
| `coords` | an $n \times 2$ matrix with coordinates. If not provided the argument `dists.lowertri` should be provided instead. |
| `dists.lowertri` | a vector with the lower triangle of the matrix of distances between data points. If not provided the argument `coords` should be provided instead. |
| `cov.model` | a string indicating the type of the correlation function. See the documentation of `cov.spatial` for the options for the correlation function model. |
| `kappa` | additional parameter needed for some of the correlation functions, namely: `"matern"`, `"powered.exponential"`, `"gneiting"` and `"gneiting.matern"`. |

`nugget` the value of the nugget parameter.

`cov.pars` a vector with 2 elements or an $nsx2$ matrix with the covariance parameters. The first element (if a vector) or first column (if a matrix) is the variance parameter $\sigma^2$. The second element or column is the correlation function parameter $\phi$. If a matrix is provided, each row corresponds to the parameters of one *spatial structure*. Models with several structures are sometimes called *nested models* in the geostatistical literature.

`inv` if `TRUE` the inverse of covariance matrix is returned. Defaults to `FALSE`.

`det` if `TRUE` the logarithmic of the square root of the determinant of the covariance matrix is returned. Defaults to `FALSE`.

`func.inv` algorithm used for the decomposition and inversion of the covariance matrix. Options are `"chol"` for Cholesky decomposition, `"svd"` for singular value decomposition and `"eigen"` for eigenvalues/eigenvectors decomposition. Defaults to `"chol"`.

`scaled` scales the covariance matrix. If `TRUE` the sill parameter $\sigma^2$ is set to 1. Defaults to `FALSE`.

`only.decomposition` if `TRUE` only the decomposition is returned. Defaults to `FALSE`.

`sqrt.inv` if `TRUE` the square root of the inverse of covariance matrix is returned. Defaults to `FALSE`.

`try.another.decomposition` logical. If `TRUE` and the argument `func.inv` is one of `"cholesky"`, `"svd"` or `"solve"`, the matrix decomposition or inversion is tested and if it fails the argument `func.inv` is re-set to `"eigen"`.

`only.inv.lower.diag` if `TRUE` only the lower triangle and the diagonal of the inverse of the covariance matrix is returned. Defaults to `FALSE`.

## Details

The elements of the covariance matrix are computed by the function `cov.spatial`.

## Value

The result is always list. The components will vary according to the input options. The possible components are:

`varcov` the covariance matrix.

`sqrt.varcov` a square root of the covariance matrix.

`lower.inverse` the lower triangle of the inverse of covariance matrix.

`diag.inverse` the diagonal of the inverse of covariance matrix.

`inverse` the inverse of covariance matrix.

`sqrt.inverse` a square root of the inverse of covariance matrix.

`log.det.to.half` the logarithmic of the square root of the determinant of the covariance matrix.

## Author(s)

Paulo J. Ribeiro Jr. ⟨paulojus@est.ufpr.br⟩
Peter J. Diggle ⟨p.diggle@lancaster.ac.uk⟩.

### References

Further information about **geoR/geoS** can be found at:
`http://www.maths.lancs.ac.uk/~ribeiro/geoR.html`

### See Also

`cov.spatial` for more information on the correlation functions; `chol`, `solve` `svd` and `eigen` for matrix invertion and/or decomposition.

---

## variog.mc.env *Envelops for Empirical Variograms Based on Permutation*

---

### Description

This function computes emvelops for empirical variograms by permutating the data values at the spatial locations.

### Usage

```
variog.mc.env(geodata, coords = geodata$coords, data = geodata$data,
              obj.variog, nsim = 99, messages.screen = TRUE)
```

### Arguments

| | |
|---|---|
| `geodata` | a list containing elements `coords` and `data` as described next. Typically an object of the class `"geodata"` - a **geoR** data-set. If not provided the arguments `coords` and `data` must be provided instead. |
| `coords` | an $n \times 2$ matrix, each row containing Euclidean coordinates of the $n$ data locations. By default it takes the element `$coords` of the argument `geodata`. |
| `data` | a vector with data values. By default it takes the element `$data` of the argument `geodata`. |
| `obj.variog` | an object of the class `"variogram"`, typically an output of the function `variog`. |
| `nsim` | number of simulations to compute the envelope. Defaults to 99. |
| `messages.screen` | if `TRUE`, the default, status messages are printed while the function is running. |

### Details

The envelops are obtained by permutation. The data values are randomly allocated to the spatial locations generating the simulated data-sets. The empirical variogram is computed for each simulation using the same lags as for the origianl variogram of the data. The envelops are computed by taking, at each lag, the maximum and minimum values of the variograms for the simulated data.

### Value

An object of the `class "variogram.envelope"` which is a list with the components:

| | |
|---|---|
| `u` | a vector with distances. |
| `v.lower` | a vector with the minimum variogram values at each distance in `u`, across the simulations. |
| `v.upper` | a vector with the maximum variogram values at each distance in `u`, across the simulations. |

### Author(s)

Paulo Justiniano Ribeiro Jr. ⟨paulojus@est.ufpr.br⟩
Peter J. Diggle ⟨p.diggle@lancaster.ac.uk⟩.

### References

Further information about **geoR/geoS** can be found at:
`http://www.maths.lancs.ac.uk/~ribeiro/geoR.html`

### See Also

`variog.mc.env` for envelops computed by permutating data, codevariog for variogram calculations, `plot.variogram` and `variog.mc.env` for graphical output. The functions `likfit`,`wlsfit`, `olsfit` estimate model parameters.

### Examples

```
if(is.R()) data(s100)
s100.vario <- variog(s100, max.dist=1)
s100.env <- variog.mc.env(s100, obj.var = s100.vario)
plot(s100.vario, envelope = s100.env)
```

---

## `variog.model.env` *Envelops for Empirical Variograms Based on Model Parameters*

---

### Description

This function computes emvelops for empirical variograms by simulating data given the model parameters.

### Usage

```
variog.model.env(geodata, coords = geodata$coords, obj.variog,
                 model.pars, nsim = 99, messages.screen = TRUE)
```

### Arguments

| | |
|---|---|
| `geodata` | a list containing element `coords` as described next. Typically an object of the class `"geodata"` - a **geoR** data-set. If not provided the arguments `coords` must be provided instead. |
| `coords` | an $n \times 2$ matrix, each row containing Euclidean coordinates of the $n$ data locations. By default it takes the element `$coords` of the argument `geodata`. |
| `obj.variog` | an object of the class `"variogram"`, typically an output of the function `variog`. |
| `coords` | an $n \times 2$ matrix containing in each row the coordinates of the $n$ data locations. |
| `model.pars` | a list with model specification and parameter values. The input is typically an object of the class `variomodel` which is an output of `likfit`, `wlsfit` or `olsfit`. This required components of list are: |

- `beta`, the mean parameter. Defaults to zero.
- `cov.model`, the covariance model. Defaults to "exponential".
- `cov.pars`, the covariance parameters $\sigma^2$ and $\phi$.
- `kappa`, the extra covariance parameters for some of the covariance models. Defaults to 0.5.
- `nugget`, the error component variance. Defaults to zero.
- `estimator.type`, the type of variogram estimator. Options are `"classical"` or `"robust"`. Defaults to `"classical"`.

| | |
|---|---|
| `nsim` | number of simulations to compute the envelope. Defaults to 99. |
| `messages.screen` | if `TRUE`, the default, status messages are printed while the function is running. |

### Details

The envelops are computed assuming a (transformed) Gaussian random field. Simulations are generated at the data locations given the model parameters. The empirical variogram is computed for each simulation using the same lags as for the origianl variogram of the data. The envelops are computed by taking, at each lag, the maximum and minimum values of the variograms for the simulated data.

### Value

An object of the `class "variogram.envelope"` which is a list with the components:

| | |
|---|---|
| `u` | a vector with distances. |
| `v.lower` | a vector with the minimum variogram values at each distance in `u`, across the simulations. |
| `v.upper` | a vector with the maximum variogram values at each distance in `u`, across the simulations. |

### Author(s)

Paulo Justiniano Ribeiro Jr. ⟨paulojus@est.ufpr.br⟩
Peter J. Diggle ⟨p.diggle@lancaster.ac.uk⟩.

### References

Further information about **geoR/geoS** can be found at:
`http://www.maths.lancs.ac.uk/~ribeiro/geoR.html`

### See Also

`variog.mc.env` for envelops computed by permutating data, codevariog for variogram calculations, `plot.variogram` and `variog.mc.env` for graphical output. The functions `likfit`,`wlsfit`, `olsfit` estimate model parameters.

### Examples

```
if(is.R()) data(s100)
s100.ml <- likfit(s100, ini = c(0.5, 0.5), fix.nugget = TRUE)
s100.vario <- variog(s100, max.dist = 1)
s100.env <- variog.model.env(s100, obj.v = s100.vario,
   model.pars = s100.ml)
plot(s100.vario, env = s100.env)
```

---

## `variog` *Compute Empirical Variograms*

---

### Description

Computes sample (empirical) variograms using either the *classical* or *robust* estimator. Output can be returned as a `binned variogram`, a `variogram cloud` or a `smoothed variogram`. Data transformation (Box-Cox) is allowed. Trends (fitted by ordinary least squares) can be removed and, in this case, variogram for the residuals are computed.

### Usage

```
variog(geodata, coords=geodata$coords, data=geodata$data,
       uvec = "default", trend = "cte", lambda = 1,
       option = c("bin", "cloud", "smooth"),
       estimator.type = c("classical", "robust"),
       nugget.tolerance = 0, max.dist = NULL, pairs.min = 2,
       bin.cloud = FALSE, ...)
```

### Arguments

| | |
|---|---|
| `geodata` | a list containing elements `coords` and `$data` described below. If not provided the arguments `coords` and `data` must be provided instead. |
| `coords` | an $n \times 2$ matrix containing coordinates of the $n$ data locations in each row. Defaults to `geodata$coords`, if provided. |
| `data` | a vector or matrix with data values. If a matrix is provided each column is regarded as one variable or realization. Defaults to `geodata$data`, if provided. |

`uvec` a vector with values defining the variogram binning when `option = "bin"`. The values of `uvec` defines the middle points of the bins.
If $uvec[1] > 0$ the first bin is: $0 < u <= uvec[2] - 0.5 * (uvec[2] - uvec[1])$.
If $uvec[1] = 0$ first bin is: $0 < u <= 0.5 * uvec[1]$ and $uvec[1]$ is replaced by the midpoint of this interval.

`trend` defines the mean part of the model (or trend). The default is `"cte"` implying constant mean. Other options are `"1st"` (trend is a first degree polynomial on the coordinates), `"2nd"` (trend is a second degree polynomial on the coordinates), or a formula of the type X where X is a matrix with external trend (covariates) measured at data locations. If trend is different from `"cte"` a linear model defined by the trend is fitted to the data and variogram are computed for the residuals.

`lambda` Box-Cox transformation parameter. Defaults to 1 (no transformation) otherwise the variogram is computed for transformed data. Notice that $\lambda = 0$ implies log-transformation.

`option` defines the output type: `bin` returns values of binned variogram, `cloud` returns the variogram cloud and `smooth` returns the kernel smoothed variogram. Defaults to `"bin"`.

`estimator.type`
`classical` computes the classical method of moments estimator and `robust` returns the robust variogram estimator. See Cressie (1993) for the expressions of the estimators. Defaults to `"classical"`.

`nugget.tolerance`
a number defining a distance. Distances between pairs of points lower than this values are set to zero and the pairs are used to estimate the nugget effect. Defaults to zero.

`max.dist` a number defining a distance. Pairs of points separated for distance greater than this value are ignored in the variogram calculation. Defaults to the maximum distance between to points.

`pairs.min` a integer number. If `option = "bin"`, bins with number of pairs smaller than this value are ignored. Defaults to NULL.

`bin.cloud` Logical. If `TRUE` and `option = "bin"` the cloud values for each class are also included in the output. Defaults to `FALSE`.

`...` passes arguments to the function `ksmooth`, if `option = "smooth"`.

## Details

Variograms are widely used in geostatistical analysis for exploratory purposes, to estimate covariance parameters, or to compare theoretical and/or fitted models against sample variograms. In this function, prior to the variogram computation, data can be transformed (Box-Cox) and/or trend can be removed (by ordinary least squares) if covariates are provided. For the later, variograms are computed for the residuals.

## Value

An object of the `class variogram` which is a list with the following components:

`u` a vector with distances.

`v` a vector with estimated variogram values at distances `u`.

`n` number of pairs in each bin if `option = "bin"`.

| | |
|---|---|
| `var.mark` | variance of the data. |
| `output.type` | echoes the `option` argument. |
| `estimator.type` | echoes the estimator used. |
| `n.data` | number of data points. |
| `call` | the function call. |

## Author(s)

Paulo J. Ribeiro Jr. ⟨paulojus@est.ufpr.br⟩
Peter J. Diggle ⟨p.diggle@lancaster.ac.uk⟩.

## References

Cressie, N.A.C.(1993) *Statistics for Spatial Data*, Wiley.

Further information about **geoR/geoS** can be found at:
`http://www.maths.lancs.ac.uk/~ribeiro/geoR.html`

## See Also

`variog.model.env` and `variog.mc.env` for variogram envelopes computation, `olsfit` and `wlsfit` for variogram model fitting, `plot.variogram` for graphical output.

## Examples

```
# Loading data:
if(is.R()) data(s100)

# computing variogram:
#
# binned variogram
vario.b <- variog(s100, max.dist=1)
# variogram cloud
vario.c <- variog(s100, max.dist=1, op="cloud")
#binned variogram and stores the cloud
vario.bc <- variog(s100, max.dist=1, bin.cloud=T)
# smoothed variogram
vario.s <- variog(s100, max.dist=1, op="sm", band=0.2)

# plotting the variograms:
par(mfrow=c(2,2))
plot(vario.b, main="binned variogram")
plot(vario.c, main="variogram cloud")
plot(vario.bc, bin.cloud=T, main="clouds for binned variogram")
plot(vario.s, main="smoothed variogram")
```

`wlsfit` *Variogram Fitting by Weighted Least Squares*

## Description

Fits a variogram model to a empirical variogram. Variogram parameters are estimated by weighted least squares.

## Usage

```
wlsfit(vario, ini.cov.pars,
       cov.model = c("exponential", "matern", "gaussian",
                     "spherical", "circular", "cubic", "wave",
                     "powered.exponential", "cauchy", "gneiting",
                     "gneiting.matern", "pure.nugget"),
       fix.nugget = FALSE, nugget = 0, kappa = NULL,
       simul.number = NULL, max.dist = "all",
       minimisation.function = c("optim", "nlm"),  lower = 0,
       weight = c("npairs", "cressie"), messages.screen = TRUE)
```

## Arguments

| | |
|---|---|
| `vario` | an object of the class `"variogram"`, typically an output of the function `variog`. The object is a list with information about the empirical variogram. |
| `ini.cov.pars` | initial values for the covariance parameters: $\sigma^2$ (partial sill) and $\phi$ (range parameter). See `DETAILS` below. |
| `cov.model` | a string with the name of the correlation function. See function call for the options. Defaults to `"exponential"`. |
| `fix.nugget` | logical, indicating whether the parameter $\tau^2$ (nugget variance) should be regarded as fixed (`fix.nugget = TRUE`) or is to be estimated (`fix.nugget = FALSE`). Defaults to `FALSE`. |
| `nugget` | value for the nugget parameter. Regarded as a fixed values if `fix.nugget = TRUE` or as a initial value for the minimization algorithm if `fix.nugget = FALSE`. Defaults to zero. |
| `kappa` | fixed value of the smoothness parameter, only required by the following correlation functions: `"matern"`, `"powered.exponential"`, `"gneiting"` and `"gneiting.matern"`. |
| `simul.number` | number of simulation. To be used when the object passed to the argument `vario` has empirical variograms for more than one data-set (or simulation). Indicates to which one to model will be fitted. |
| `max.dist` | maximum distance considered when fitting the variogram. Defaults to the maximum distance found in the object defined in the argument `vario`. |
| `minimisation.function` | minimization function used to estimate the parameters. See function call for options. Defaults to the first option. |
| `lower` | lower limits for the parameters. Defaults to zero. |

`weight` type weights used in the loss function. See `DETAILS` below.

`messages.screen`
logical. Indicates whether or not messages are printed on the screen (or other output device) while the function is running.

`...` further parameters to be passed to the minimization function. Typically `control` type arguments which controls the behavior of the minimization algorithm. See documentation for the selected minimisation function for further details.

## Value

An object of the `class "variomodel"` which is list with the following components:

`nugget` value of the nugget parameter. An estimated value if `fix.nugget = FALSE` or a fixed value if `fix.nugget = TRUE`.

`cov.pars` a 2 elements vector with estimated values of the covariance parameters, partial sill and range parameter, respectively.

`cov.model` a string with the name of the correlation function.

`kappa` fixed value of the smoothness parameter.

`value` minimized value of the loss function.

`max.dist` maximum distance considered in the variogram fitting.

`minimisation.function`
minimization function used.

`message` status messages returned by the function.

`method` a string `"WLS"` indicating the estimation method which was used.

`call` the function call.

## Author(s)

Paulo Justiniano Ribeiro Jr. ⟨paulojus@est.ufpr.br⟩
Peter J. Diggle ⟨p.diggle@lancaster.ac.uk⟩.

## References

Barry, J.T., Crowder, M.J. and Diggle, P.J. (1997) Parametric estimation of the variogram. *Tech. Report, Dept Maths & Stats, Lancaster University.*

Cressie, N.A.C (1993) *Statistics for Spatial Data.* New York: Wiley.

Further information about **geoR** can be found at:
`http://www.maths.lancs.ac.uk/~ribeiro/geoR.html`.

## See Also

`cov.spatial` for a detailed description of the available correlation (veriogram) functions, `olsfit` for ordinary least squares variogram fit, `likfit` for maximum and restricted maximum likelihood estimation, `lines.variomodel` for graphical output of the fitted model. For details on the minimization functions see `optim` and `nlm`.

### Examples

```
if(is.R()) data(s100)
vario100 <- variog(s100, max.dist=1)
ini.vals <- expand.grid(seq(0,1,l=5), seq(0,1,l=5))
wls <- wlsfit(vario100, ini=ini.vals, fix.nug=TRUE)
summary(wls)
plot(vario100)
lines(wls)
```

# Index


∗Topic **datasets, spatial**
parana, 41
s100 and s121, 54
∗Topic **spatial, aplot**
lines.grf, 30
lines.krige.bayes, 31
lines.variogram, 33
lines.variogram.envelope, 32
lines.variomodel, 34
∗Topic **spatial, class, manip**
as.geodata, 1
∗Topic **spatial, dplot, aplot**
points.geodata, 47
∗Topic **spatial, dplot**
image.grf, 9
image.kriging, 11
plot.geodata, 42
plot.grf, 43
plot.proflik, 44
plot.variogram, 46
∗Topic **spatial, manip**
read.geodata, 53
∗Topic **spatial, models**
krige.bayes, 12
likfit, 27
∗Topic **spatial, nonparametric**
variog.mc.env, 59
∗Topic **spatial, print**
summary.likGRF, 54
∗Topic **spatial, programming, interface**
wrappers, 35
∗Topic **spatial, smooth**
variog, 62
∗Topic **spatial**
coords.aniso, 2
cov.spatial, 4
grf, 6
image.krige.bayes, 10
krige.conv, 17
ksline, 20
likfit.old, 23
matern, 36
olsfit, 39
polygrid, 49
proflik, 50
summary.variomodel, 55
trend.spatial, 56
varcov.spatial, 57
variog.model.env, 60
wlsfit, 65

as.geodata, **1**, *53, 54*

besselK, *37*
bilinearformXAY *(wrappers)*, 35
borders *(parana)*, 41
boxcox.ns *(likfit)*, 27

chol, *8*, *59*
class, *1, 2*, *15*, *17*, *19, 20*, *22*, *29*, *40*, *53–55*, *60, 61*, *63*, *66*
contour, *45*
coords.aniso, **2**, *7, 8*, *14*, *19*, *22*, *28*
cor.number *(cov.spatial)*, 4
cov.spatial, **4**, *7*, *14*, *18*, *21*, *24*, *28*, *37*, *41*, *57–59*, *66*

diagquadraticformXAX *(wrappers)*, 35
distdiag *(wrappers)*, 35

eigen, *8*, *59*
expand.grid, *49, 50*

gray, *49*
grf, **6**, *9*, *30, 31*, *43, 44*
grfclass *(grf)*, 6

image, *9–12*
image.grf, *8*, **9**
image.krige.bayes, **10**, *16*
image.kriging, **11**, *19*
inout, *49, 50*

krige.bayes, *10, 11*, **12**, *19*, *23*, *31, 32*, *49*
krige.control *(krige.conv)*, 17
krige.conv, *11, 12*, *16*, **17**, *20*, *22, 23*, *49*

ksline, *11, 12*, *16*, *19*, **20**
ksline.aux.1 *(ksline)*, 20
ksmooth, *63*

likfit, *23*, **27**, *35*, *41*, *50–52*, *54–56*, *60–62*, *66*
likfit.nospatial *(likfit.old)*, 23
likfit.old, **23**
lines, *31–35*
lines.grf, **30**
lines.krige.bayes, *16*, **31**
lines.variogram, *26*, *30*, **33**, *34*, *35*, *46*, *47*
lines.variogram.envelope, **32**
lines.variomodel, *26*, *30*, **34**, *34*, *41*, *46*, *47*, *66*
list, *2*, *54*
lm, *42*
loccoords *(wrappers)*, 35
loci.paper *(parana)*, 41
loglik.spatial *(likfit.old)*, 23
loss.olsvario *(olsfit)*, 39
loss.wlsvario *(wlsfit)*, 65

maijun *(parana)*, 41
matern, *5*, **36**
model.control *(krige.bayes)*, 12

negloglik.GRF *(likfit)*, 27
nlm, *26*, *38*, *39*, *41*, *52*, *66*
nlmP, **38**
nls, *41*

olsfit, *26*, *30*, *35*, **39**, *41*, *55*, *56*, *60–62*, *64*, *66*
optim, *26*, *28–30*, *39*, *41*, *51*, *52*, *66*
output.control *(krige.bayes)*, 12

par, *49*
parana, **41**
persp, *9–12*, *45*
persp.grf *(image.grf)*, 9
persp.krige.bayes, *16*
persp.krige.bayes *(image.krige.bayes)*, 10
persp.kriging *(image.kriging)*, 11
plot, *43–49*
plot.geodata, **42**, *49*
plot.grf, *8*, *31*, 34, **43**
plot.proflik, **44**, *52*
plot.variogram, *26*, *30*, *35*, *44*, **46**, *60*, *62*, *64*
points, *48*, *49*
points.geodata, *43*, **47**
polygrid, **49**
print, *55*
print.likGRF *(summary.likGRF)*, 54
print.summary.likGRF *(summary.likGRF)*, 54
prior.control *(krige.bayes)*, 12
proflik, *26*, *30*, *44*, *45*, **50**
proflik.aux1.1 *(proflik)*, 50
proflik.aux10 *(proflik)*, 50
proflik.aux11 *(proflik)*, 50
proflik.aux12 *(proflik)*, 50
proflik.aux13 *(proflik)*, 50
proflik.aux14 *(proflik)*, 50
proflik.aux15 *(proflik)*, 50
proflik.aux16 *(proflik)*, 50
proflik.aux17 *(proflik)*, 50
proflik.aux18 *(proflik)*, 50
proflik.aux19 *(proflik)*, 50
proflik.aux2 *(proflik)*, 50
proflik.aux20 *(proflik)*, 50
proflik.aux21 *(proflik)*, 50
proflik.aux22 *(proflik)*, 50
proflik.aux23 *(proflik)*, 50
proflik.aux24 *(proflik)*, 50
proflik.aux27 *(proflik)*, 50
proflik.aux28 *(proflik)*, 50
proflik.aux3 *(proflik)*, 50
proflik.aux30 *(proflik)*, 50
proflik.aux31 *(proflik)*, 50
proflik.aux32 *(proflik)*, 50
proflik.aux33 *(proflik)*, 50
proflik.aux4 *(proflik)*, 50
proflik.aux5 *(proflik)*, 50
proflik.aux6 *(proflik)*, 50
proflik.aux7 *(proflik)*, 50
proflik.aux8 *(proflik)*, 50
proflik.aux9 *(proflik)*, 50
proflik.cov *(proflik)*, 50
proflik.ftau *(likfit.old)*, 23
proflik.lambda *(proflik)*, 50
proflik.main *(proflik)*, 50
proflik.nug *(likfit.old)*, 23
proflik.phi *(likfit.old)*, 23

rainbow, *49*
read.geodata, *2*, **53**
read.table, *53*, *54*
rfm.bin *(variog)*, 62

s100 *(s100 and s121)*, 54
s100 and s121, **54**
s121 *(s100 and s121)*, 54

`scatterplot3d`, *42, 43*
`solve`, *59*
`summary`, *55*
`summary.likGRF`, *29, 30*, **54**
`summary.variomodel`, *26*, **55**
`svd`, *8*, *59*

`trend.spatial`, **56**

`varcov.spatial`, *5*, **57**
`variog`, *33, 34*, *39*, *43, 44*, *46, 47*, *59*, 60, *61*, 62, **62**, *65*
`variog.mc.env`, *32–34*, *46, 47*, **59**, *60*, *62*, *64*
`variog.model.env`, *32–34*, *46, 47*, **60**, *64*

`wlsfit`, *26*, *30*, *35*, *55, 56*, *60–62*, *64*, **65**
`wrappers`, **35**