Package 'automap'

September 3, 2024

Version 1.1-12

Date 2024-09-03

Title Automatic Interpolation Package

Maintainer Jon Olav Skoien <jon.skoien@gmail.com>

Description
An automatic interpolation is done by automatically estimating the variogram and then calling gstat. An overview is given by Hiemstra et al (2008) <doi:10.1016/j.cageo.2008.10.011>.

Depends R (>= 2.10.0)

Imports gstat, lattice, reshape, methods, ggplot2, sp, sf, stars, graphics

License GPL

NeedsCompilation no

Repository CRAN

Author Paul Hiemstra [aut], Jon Olav Skoien [aut, cre]

Date/Publication 2024-09-03 15:50:06 UTC

Contents

autofitVariogram *Automatically fitting a variogram*

Description

Automatically fitting a variogram to the data on which it is applied. The automatic fitting is done through fit.variogram. In fit.variogram the user had to supply an initial estimate for the sill, range etc. autofitVariogram provides this estimate based on the data and then calls fit.variogram.

Usage

```
autofitVariogram(formula,
         input_data,
         model = c("Sph", "Exp", "Gau", "Ste"),
         kappa = c(0.05, seq(0.2, 2, 0.1), 5, 10),
         fix.values = c(NA,NA,NA),
         verbose = FALSE,
         GLS.model = NA,
         start_vals = c(NA,NA,NA),
         miscFitOptions = list(),
         ...)
```

Arguments

formula	formula that defines the dependent variable as a linear model of independent variables; suppose the dependent variable has name 'z', for ordinary and simple kriging use the formula 'z~1'; for simple kriging also define 'beta' (see below); for universal kriging, suppose 'z' is linearly dependent on 'x' and 'y', use the formula 'z~x+y'.
input_data	An object of SpatialPointsDataFrame-class or sf .
model	The list of variogrammodels that will be tested.
kappa	Smoothing parameter of the Matern model. Provide a list if you want to check more than one value.
fix.values	Can be used to fix a variogram parameter to a certain value. It consists of a list with a length of three. The items describe the fixed value for the nugget, range and sill respectively. They need to be given in that order. Setting the value to NA means that the value is not fixed.
verbose	logical, if TRUE the function will give extra feedback on the fitting process
GLS.model	If a variogram model is passed on through this parameter a Generalized Least Squares sample variogram is calculated.
start_vals	Can be used to give the starting values for the variogram fitting. The items describe the fixed value for the nugget, range and sill respectively. They need to be given in that order. Setting the value to NA means that the value will be automatically chosen.

`miscFitOptions` A list with named arguments that provide additional control over the fitting process. For example: `list(merge.small.bins = TRUE)`. If the list is empty, autofitVariogram uses default values. The following parameters can be set:

`merge.small.bins`**:** logical, when TRUE, the function checks if there are bins with less than 5 points. If so, the first two bins are merged and the check is repeated. This is done until all bins have more than `min.np.bin` points.

`min.np.bin`**:** integer, the minimum number of points allowed in a bin before we start merging bins. See also `merge.small.bins`.

`...` parameters that are passed on to variogram when calculating the sample variogram.

Details

Geostatistical routines are used from package `gstat`.

A few simple choices are made when estimating the inital guess for `fit.variogram`. The initial sill is estimated as the `mean` of the `max` and the `median` of the semi-variance. The inital range is defined as 0.10 times the diagonal of the bounding box of the data. The initial nugget is defined as the `min` of the the semi-variance.

There are five different types of models that are often used:

Sph A shperical model.

Exp An exponential model.

Gau A gaussian model.

Mat A model of the Matern familiy

Ste Matern, M. Stein's parameterization

A list of all permitted variogram models is available by typing vgm() into the R console. `autofitVariogram` iterates over the variogram models listed in `model` and picks the model that has the smallest residual sum of squares with the sample variogram. For the Matern model, all the kappa values in `kappa` are tested.

Note that when using the power model, and not specifying starting values yourself, the sill is set to 1, the range to 1 and the nugget to 0. This is because the normal initial values for those paramters don't work well with the power model. I consider this a temporary solution, any suggestions are appreciated.

It is possible to pass anisotropy parameters to `autofitVariogram`. However, `autofitVariogram` does not fit anisotropic variogram models. The function sees the anisotropic sample variogram as one big sample variogram. So it fits an average isotropic variogram model from the anisotropic sample variogram. A warning is issued when a users passes `alpha` to `autofitVariogram`.

Value

An object of type `autofitVariogram` is returned. This object contains the experimental variogram, the fitted variogram model and the sums of squares (`sserr`) between the sample variogram and the fitted variogram model.

Note

autofitVariogram is mostly used indirectly through the function autoKrige

Author(s)

Paul Hiemstra, <paul@numbertheory.nl>

See Also

fit.variogram, autoKrige, posPredictionInterval

Examples

```
library(sp)
data(meuse)
coordinates(meuse) =~ x+y
variogram = autofitVariogram(zinc~1,meuse)
plot(variogram)

# Residual variogram
data(meuse)
coordinates(meuse) =~ x+y
variogram = autofitVariogram(zinc ~ soil + ffreq + dist, meuse)
plot(variogram)

# Settings additional fitting options
variogram = autofitVariogram(zinc ~ soil + ffreq + dist, meuse,
    miscFitOptions = list(merge.small.bins = FALSE))
plot(variogram)

# Settings the minimum number of pairs per bin quite high
# to see the effect of merging bins
variogram = autofitVariogram(zinc ~ soil + ffreq + dist, meuse,
    miscFitOptions = list(min.np.bin = 500))
plot(variogram)

# ...and disable the merging, note the difference between the two plots
variogram = autofitVariogram(zinc ~ soil + ffreq + dist, meuse,
    miscFitOptions = list(min.np.bin = 500, merge.small.bins = FALSE))
plot(variogram)

# An example of autofitVariogram with anisotropic sample variogram.
# This is not supported, see details section.
vm.isotropic = autofitVariogram(log(zinc) ~ dist, meuse)
# The following line might not work, depending on version of R and gstat
# vm.anisotropic = autofitVariogram(log(zinc) ~ dist, meuse, alpha = c(0,45,90,135))
```

autoKrige *Performs an automatic interpolation*

Description

This function performs automatic kriging on the given dataset. The variogram is generated automatically using autofitVariogram.

Usage

```
autoKrige(formula,
     input_data,
  new_data,
    data_variogram = input_data,
  block = 0,
  model = c("Sph", "Exp", "Gau", "Ste"),
  kappa = c(0.05, seq(0.2, 2, 0.1), 5, 10),
  fix.values = c(NA,NA,NA),
  remove_duplicates = TRUE,
  verbose = FALSE,
  GLS.model = NA,
      start_vals = c(NA,NA,NA),
      miscFitOptions = list(),
  ...)
```

Arguments

formula	formula that defines the dependent variable as a linear model of independent variables; suppose the dependent variable has name 'z', for ordinary and simple kriging use the formula 'z~1'; for simple kriging also define 'beta' (see below); for universal kriging, suppose 'z' is linearly dependent on 'x' and 'y', use the formula 'z~x+y'.
input_data	An object of the SpatialPointsDataFrame-class or sf containing the data to be interpolated.
new_data	A sp, sf or stars (st_as_stars) object containing the prediction locations. new_data can be a points set, a grid or a polygon. Must not contain NA's. If this object is not provided a default is calculated. This is done by taking the convex hull of input_data and placing around 5000 gridcells in that convex hull.
data_variogram	An optional way to provide a different dataset for the building of the variogram then for the spatial interpolation.
block	Use this parameter to pass on a specification for the block size. e.g. c(1000,1000)
model	List of models that will be tested during automatic variogram fitting.
kappa	List of values for the smoothing parameter of the Matern model that will be tested during automatic variogram fitting.

fix.values	Can be used to fix a variogram parameter to a certain value. It consists of a list with a length of three. The items describe the fixed value for the nugget, range and sill respectively. Setting the value to NA means that the value is not fixed. Is passed on to autofitVariogram.
remove_duplicates	logical, remove duplicate points from the input_data. This can take some time on large datasets.
verbose	logical, if TRUE autoKrige will give extra information on the fitting process
GLS.model	If a variogram model is passed on through this parameter a Generalized Least Squares sample variogram is calculated.
start_vals	Can be used to give the starting values for the variogram fitting. The items describe the fixed value for the nugget, range and sill respectively. They need to be given in that order. Setting the value to NA means that the value will be automatically chosen.
miscFitOptions	Additional options to set the behavior of autofitVariogram. For details see the documentation of autofitVariogram.
...	arguments that are passed on to the gstat function krige.

Details

autoKrige calls the function autofitVariogram that fits a variogram model to the given dataset. This variogram model and the data are used to make predictions on the locations in new_data. The only compulsory argument is input_data. So the most simple call would of the form:

```
autoKrige(meuse)
```

autoKrige now assumes that you want to perform ordinary kriging on the first column of input_data.

autoKrige performs some checks on the coordinate systems of input_data and new_data. If one of both is NA, it is assigned the projection of the other. If they have different projections, an error is raised. If one of both has a non-projected system (i.e. latitude-longitude), an error is raised. This error is raised because 'gstat does use spherical distances when data are in geographical coordinates, however the usual variogram models are typically not non-negative definite on the sphere, and no appropriate models are available' (Edzer Pebesma on r-sig-geo).

When the user specifies the power model (Pow) as the model, the initial range is set to one. Note that when using the power model, the initial range is the initial power.

Value

This function returns an autoKrige object containing the results of the interpolation (prediction, variance and standard deviation), the sample variogram, the variogram model that was fitted by autofitVariogram and the sums of squares between the sample variogram and the fitted variogram model. The attribute names are krige_output, exp_var, var_model and sserr respectively.

Author(s)

Paul Hiemstra, <paul@numbertheory.nl>

See Also

autofitVariogram, krige

Examples

```
# Data preparation

library(sp)
library(sf)
library(stars)
data(meuse)
coordinates(meuse) =~ x+y
data(meuse.grid)
gridded(meuse.grid) =~ x+y


# Ordinary kriging, no new_data object
kriging_result = autoKrige(zinc~1, meuse)
plot(kriging_result)

# Ordinary kriging
kriging_result = autoKrige(zinc~1, meuse, meuse.grid)
plot(kriging_result)

# Fixing the nugget to 0.2
kriging_result = autoKrige(zinc~1, meuse,
meuse.grid, fix.values = c(0.2,NA,NA))
plot(kriging_result)

# Universal kriging
kriging_result = autoKrige(zinc~soil+ffreq+dist, meuse, meuse.grid)
plot(kriging_result)

# Block kriging
kriging_result_block = autoKrige(zinc~soil+ffreq+dist,
meuse, meuse.grid, block = c(400,400))
plot(kriging_result_block)

# Dealing with duplicate observations
data(meuse)
meuse.dup = rbind(meuse, meuse[1,]) # Create duplicate
coordinates(meuse.dup) = ~x+y
kr = autoKrige(zinc~dist, meuse.dup, meuse.grid)

# Extracting parts from the autoKrige object
prediction_spdf = kr$krige_output
sample_variogram = kr$exp_var
variogram_model = kr$var_model

coordinates(meuse) = ~x + y
meuse = st_as_sf(meuse)
meuse.grid = st_as_stars(meuse.grid)
```

```
kriging_result = autoKrige(zinc~1, meuse,
meuse.grid, fix.values = c(0.2,NA,NA))
```

---

autoKrige.cv *Automatic cross-validation*

Description

Uses autofitVariogram to fit a variogram model to the data and then calls krige.cv to perform cross-validation.

Usage

```
autoKrige.cv(formula,
     input_data,
     data_variogram = input_data,
     model = c("Sph", "Exp", "Gau", "Ste"),
     kappa = c(0.05, seq(0.2, 2, 0.1), 5, 10),
     fix.values = c(NA,NA,NA),
     verbose = c(FALSE, interactive()),
     GLS.model = NA,
         start_vals = c(NA,NA,NA),
         miscFitOptions = list(),
     ...)
```

Arguments

formula	formula that defines the dependent variable as a linear model of independent variables; suppose the dependent variable has name 'z', for ordinary and simple kriging use the formula 'z~1'; for simple kriging also define 'beta' (see below); for universal kriging, suppose 'z' is linearly dependent on 'x' and 'y', use the formula 'z~x+y'.
input_data	An object of the SpatialPointsDataFrame-class containing the data to be interpolated.
data_variogram	An optional way to provide a different dataset for the building of the variogram.
model	List of models that will be tested during automatic variogram fitting.
kappa	List of values for the smoothing parameter of the Matern model that will be tested during automatic variogram fitting.
fix.values	Can be used to fix a variogram parameter to a certain value. It consists of a list with a length of three. The items describe the fixed value for the nugget, range and sill respectively. Setting the value to NA means that the value is not fixed. Is passed on to autofitVariogram.

verbose	vector of 2 logicals. The first element sets the verbosity of autofitVariogram, see its documentation for more information. The second element sets the verbosity level of krige.cv, see its documentation for more information.
GLS.model	If a variogram model is passed on through this parameter a Generalized Least Squares sample variogram is calculated.
start_vals	Can be used to give the starting values for the variogram fitting. The items describe the fixed value for the nugget, range and sill respectively. They need to be given in that order. Setting the value to NA means that the value will be automatically chosen.
miscFitOptions	Additional options to set the behavior of autofitVariogram. For details see the documentation of autofitVariogram.
...	arguments passed to krige.cv

Value

autoKrige.cv returns an object of class autoKrige.cv. This is a list containing one object of class SpatialPointsDataFrame with the results of the cross-validation, see krige.cv for more details. The attribute name is krige.cv_output.

Author(s)

Paul Hiemstra, <paul@numbertheory.nl>

See Also

krige.cv, autofitVariogram, compare.cv

Examples

```
library(sp)
data(meuse)
coordinates(meuse) = ~x+y
data(meuse.grid)
gridded(meuse.grid) = ~x+y

kr.cv = autoKrige.cv(log(zinc)~1, meuse, model = c("Exp"), nfold = 10)
kr_dist.cv = autoKrige.cv(log(zinc)~sqrt(dist), meuse,
       model = c("Exp"), nfold = 10)
kr_dist_ffreq.cv = autoKrige.cv(log(zinc)~sqrt(dist)+ffreq,
       meuse, model = c("Exp"), nfold = 10)
```

automapPlot *Special plot function for automap*

Description

This function wraps around spplot and creates a blue-to-whitish colorscale instead of the standard bpy colorscale.

Usage

```
automapPlot(plot_data, zcol, col.regions, sp.layout, points,  ...)
```

Arguments

plot_data	A spatial object that is to be plotted, sp or sf
zcol	The name of the column from plot_data you want to use. Can also be a list.
col.regions	Choose a colors that specify the fill colours.
sp.layout	An sp.layout object that can be passed to spplot, to be added to the plot
points	Points that can be added to the plot
...	other possible arguments that can be passed on to spplot.

Details

The classIntervals function from the classInt package is a good function to calculate the position of the colorbreaks.

Author(s)

Paul Hiemstra, <paul@numbertheory.nl>

See Also

spplot, plot.autoKrige, plot.posPredictionInterval

Examples

```
# Ordinary kriging
library(sp)
data(meuse)
coordinates(meuse) =~ x+y
data(meuse.grid)
gridded(meuse.grid) =~ x+y

kriging_result = autoKrige(zinc~1, meuse, meuse.grid)

# Adding the sp.layout parameter shows the locations of the measurements
automapPlot(kriging_result$krige_output, "var1.pred",
sp.layout = list("sp.points", meuse))
```

compare.cv *Comparing the results of cross-validations*

Description

Allows comparison of the results from several outcomes of autoKrige.cv in both statistics and spatial plots (bubble plots).

Usage

```
compare.cv(...,
   col.names,
   bubbleplots = FALSE,
   zcol = "residual",
   layout,
   key.entries,
   reference = 1,
   plot.diff = FALSE,
       digits = 4,
       ggplot = FALSE,
       addPoly = NULL)
```

Arguments

...	autoKrige.cv objects that are compared to each other. Also accepts the output form krige.cv, these objects are transformed to autoKrige.cv objects.
col.names	Names for the different objects in .... This defaults to the names of the objects in ....
bubbleplots	logical, if TRUE then bubble plots of the objects in ... are drawn using the same value for the color breaks.
zcol	Which column in the objects in ... is going to be drawn in the bubbleplots. Options are: var1.pred, var1.var, observed, residual and zscore.
layout	layout of the bubbleplot, e.g. c(2,2). The argument gives the number of rows and columns in which the set of bubbleplots is to be drawn. Useful defaults are selected.
key.entries	A list of numbers telling what the key entries in the bubbleplots are. See bubble for more details.
reference	An integer telling which of the objects should be taken as a reference if plot.diff equals TRUE. reference equal to 1 means that the first object is the reference, reference equal to 2 means that the second object is the reference etc.
plot.diff	logical, if plot.diff is TRUE the number specified in reference defines the CV object that is taken as a reference What is shown in the plot is reference data squared minus the other data squared. So the color red means that the CV is doing worse than the reference, vice-versa for green. This is very useful to see where the differences between the results are spatially and if there is a pattern.

digits	The number of significant digits in the resulting data.frame.
ggplot	logical, determines if spplot or ggplot2 is used to make the spatial plot of the cross-validation residuals. Note that the plot.diff and reference arguments are obsolete when ggplot equals TRUE.
addPoly	if this object contains a SpatialPolygons* object, it is added to the plot as layout. Note that this only works when ggplot equals TRUE.

Value

A data.frame with for each cross-validation result a number of diagnostics:

mean_error	The mean of the cross-validation residual. Ideally small.
me_mean	mean error divided by the mean of the observed values, measure for how large the mean_error is in contrast to the mean of the dataset
MSE	Mean Squared error.
MSNE	Mean Squared Normalized Error, mean of the squared z-scores. Ideally small.
cor_obspred	Correlation between the observed and predicted values. Ideally 1.
cor_predres	Correlation between the predicted and the residual values. Ideally 0.
RMSE	Root Mean Squared Error of the residual. Ideally small.
RMSE_sd	RMSE divided by the standard deviation of the observed values. Provides a measure variation of the residuals vs the variation of the observed values.
URMSE	Unbiased Root Mean Squared Error of the residual. Ideally small.
iqr	Interquartile Range of the residuals. Ideally small.

Author(s)

Paul Hiemstra, <paul@numbertheory.nl>

See Also

krige.cv, bubble, autofitVariogram, autoKrige.cv,

Examples

```
# Load the data
library(sp)
data(meuse)
coordinates(meuse) = ~x+y
data(meuse.grid)
gridded(meuse.grid) = ~x+y

# Perform cross-validation
kr.cv = autoKrige.cv(log(zinc)~1, meuse, model = c("Exp"), nfold = 10)
kr_dist.cv = autoKrige.cv(log(zinc)~sqrt(dist), meuse,
       model = c("Exp"), nfold = 10)
kr_dist_ffreq.cv = autoKrige.cv(log(zinc)~sqrt(dist)+ffreq,
       meuse, model = c("Exp"), nfold = 10)
```

```
# Compare the results
compare.cv(kr.cv, kr_dist.cv, kr_dist_ffreq.cv)
compare.cv(kr.cv, kr_dist.cv, kr_dist_ffreq.cv,
           bubbleplots = TRUE)
compare.cv(kr.cv, kr_dist.cv, kr_dist_ffreq.cv,
           bubbleplots = TRUE, col.names = c("OK","UK1","UK2"))
compare.cv(kr.cv, kr_dist.cv, kr_dist_ffreq.cv,
           bubbleplots = TRUE, col.names = c("OK","UK1","UK2"),
           plot.diff = TRUE)

library(ggplot2)
compare.cv(kr.cv, kr_dist.cv, kr_dist_ffreq.cv,
           bubbleplots = TRUE, col.names = c("OK","UK1","UK2"),
           ggplot = TRUE)
```

`plot.autoKrige` *Plot methods in automap*

Description

Defines methods to plot objects in automap.

Usage

```
## S3 method for class 'autoKrige'
plot(x, sp.layout = NULL, ...)
## S3 method for class 'posPredictionInterval'
plot(x, sp.layout = NULL, justPosition = TRUE, main = "Position prediction interval", ...)
```

Arguments

`x`	the object to plot (of class `autoKrige` or `posPredictionInterval`)
`sp.layout`	An object that can contain lines, points and polygons that function as extra layout.
`justPosition`	logical, if FALSE: not only the plot with the position of the prediction interval is plotted, but also plots with the upper and lower limits of the prediction interval.
`main`	Title of the plot for the position of the prediction interval.
`...`	arguments passed to lattice functions xyplot, spplot or plot.sf

Details

For a detailed description of how `sp.layout` is constructed see spplot.

Author(s)

Paul Hiemstra, <paul@numbertheory.nl>

See Also

spplot, autoKrige, posPredictionInterval

Examples

```
# Ordinary kriging
library(sp)
library(sf)
data(meuse)
coordinates(meuse) =~ x+y
data(meuse.grid)
gridded(meuse.grid) =~ x+y

kriging_result = autoKrige(log(zinc)~1, meuse, meuse.grid)
# Adding the sp.layout parameter shows the locations of the measurements
plot(kriging_result, sp.layout = list(pts = list("sp.points", meuse)))

meuse = as(meuse, "sf")
meuse.grid = as(meuse.grid, "sf")

kriging_result = autoKrige(log(zinc)~1, meuse, meuse.grid)
# Adding the meuse points shows the locations of the measurements
plot(kriging_result, points = meuse)
```

---

posPredictionInterval *Determines the position of the p% prediction interval*

Description

This function calculates the p% prediction interval and determines the position of this interval relative to value. This can be higher, lower or not distinguishable.

Usage

```
posPredictionInterval(krige_object,
                      p = 95,
                      value = median(krige_object$krige_output$var1.pred))
```

Arguments

krige_object	The result of from the autoKrige procedure. This is expected to be a autoKrige-object.
p	The p% percent prediction interval is compared to value
value	The value to which the the p% prediction interval compared

Value

The output object is of class `posPredictionInterval` and contains the results of the function in an Spatial-class object similar to the one in the input object. This means that if the input object containes a grid, the results are also returned on that same grid. Also included in the return object are the values for `p` and `value`.

Author(s)

Paul Hiemstra, <paul@numbertheory.nl>

See Also

autoKrige, autofitVariogram

Examples

```
library(sp)
data(meuse)
coordinates(meuse) =~ x+y
data(meuse.grid)
gridded(meuse.grid) =~ x+y

kriging_result = autoKrige(zinc~1, meuse, meuse.grid)
pos = posPredictionInterval(kriging_result, 95, 75)
plot(pos)
```

Index