Package 'GeoAdjust'

May 7, 2026

Type Package

Title Accounting for Random Displacements of True GPS Coordinates of Data

Version 2.0.1

Description The purpose is to account for the random displacements (jittering) of true survey household cluster center coordinates in geostatistical analyses of Demographic and Health Surveys program (DHS) data. Adjustment for jittering can be implemented either in the spatial random effect, or in the raster/distance based covariates, or in both. Detailed information about the methods behind the package functionality can be found in our two papers.
Umut Altay, John Paige, Andrea Riebler, Geir-Arne Fuglstad (2024) <doi:10.32614/RJ-2024-027>.
Umut Altay, John Paige, Andrea Riebler, Geir-Arne Fuglstad (2023) <doi:10.1177/1471082X231219847>.

Encoding UTF-8

RoxygenNote 7.3.2

Suggests knitr, rmarkdown, testthat (>= 3.0.0),

Config/testthat/edition 3

Depends R (>= 3.5)

LinkingTo TMB, RcppEigen

License GPL (>= 2)

Imports fmesher, terra, sf, stats, SUMMER, Matrix, ggplot2, fields, TMB

NeedsCompilation yes

Author Umut Altay [cre, aut],
John Paige [aut],
Geir-Arne Fuglstad [aut],
Andrea Riebler [aut]

Maintainer Umut Altay <altayumut.ua@gmail.com>

Repository CRAN

Date/Publication 2025-10-02 05:10:28 UTC

Contents

`dummy` *Roxygen commands*

Description

Roxygen commands

Usage

```
dummy()
```

`estimateModel` *Estimates model parameters*

Description

Estimates model parameters

Usage

```
estimateModel(
  data = NULL,
  options = NULL,
  priors = NULL,
  n.sims = NULL,
  log_tau = NULL,
  log_kappa = NULL,
  USpatial = 1,
  alphaSpatial = 0.05,
  UNugget = 1,
  alphaNug = 0.05
)
```

Arguments

data	A data input list that is created by prepareInput() function.
options	A list containing two components, namely, random and covariates. The function accounts for jittering both in the spatial random effect and in covariates (if there are any) by default. However, the jittering adjustment can be turned off in either the random effect or in covariates, or both, by setting the corresponding component of the list to zero.
priors	A list of two components. Beta is a vector of two elements and passes the parameters of the Gaussian prior that will be assigned to the covariates (including the intercept). The first element of it is the mean and the second one is the standard deviation of Gaussian prior. Range is a value representing the median range in kilometers, which will be used for constructing the PC (Penalized-complexity) priors.
n.sims	number of samples to be drawn for each model parameter
log_tau	SPDE parameter related to the spatial precision
log_kappa	SPDE parameter related to the range and spatial precision
USpatial	The threshold that is crossed by the the variance prior.
alphaSpatial	The probability of crossing the threshold for the variance prior.
UNugget	The threshold that is crossed by the prior on the nugget standard deviation. It will only be used when the likelihood is Gaussian.
alphaNug	The probability of crossing the threshold for the prior on the nugget standard deviation. It will only be used when the likelihood is Gaussian.

Value

Model estimation results of class GAmodel. The output consists of four elements: A data frame containing the estimated model parameters and the corresponding 95 from autodifferentiation of TMB, A matrix containing the sampled coefficient effect sizes and the random effect coefficients, A character string indicating the likelihood type in the model.

Examples

```
path1 <- system.file("extdata", "exampleInputData.rda", package = "GeoAdjust")
path2 <- system.file("extdata", "exampleMesh.rda", package = "GeoAdjust")
load(path1)
load(path2)
results <- estimateModel(data = exampleInputData, priors = list(beta = c(0,1),
range = 114), USpatial = 1, alphaSpatial = 0.05, UNugget = 1, alphaNug = 0.05, n.sims = 1000)
```

`gridCountry` *Creates a grid of locations within the bounding box of the national borders of a country of interest.*

Description

Creates a grid of locations within the bounding box of the national borders of a country of interest.

Usage

```
gridCountry(admin0 = NULL, res = NULL, target_crs = NULL)
```

Arguments

`admin0`	An sf class MULTIPOLYGON representing the country borders.
`res`	A value representing the resolution in kilometers.
`target_crs`	A projection string representing the desired coordinate reference system according to which the prediction grid will be constructed. The measurement unit of the target_crs should be in kilometers.

Value

A list. The first element of the list, predRast, is a SpatRaster object. The second element of the list, loc.pred, is an sf class POINT object containing coordinates of the cell centers (in target_crs) of the prediction raster.

Examples

```
path1 <- system.file("extdata", "geoData.rda", package = "GeoAdjust")
load(path1)
crs_KM = "+units=km +proj=utm +zone=37 +ellps=clrk80 +towgs84=-160,-6,-302,0,0,0,0 +no_defs"
grid = gridCountry(admin0 = adm0, res = 5, target_crs = crs_KM)
```

`meshCountry` *Creates a constrained refined Delaunay triangulation mesh based on the country borders.*

Description

Creates a constrained refined Delaunay triangulation mesh based on the country borders.

Usage

```
meshCountry(
  admin0 = NULL,
  max.edge = NULL,
  cutoff = NULL,
  offset = NULL,
  target_crs = NULL
)
```

Arguments

admin0	An sf class multipolygon representing the country borders.
max.edge	A vector of two values. The first and the second elements of the vector represent the largest allowed triangle lengths for the inner and outer mesh, respectively.
cutoff	The minimum allowed distance of the vertices to each other.
offset	A value representing the extension distance for the mesh.
target_crs	A projection string representing the desired coordinate reference system according to which the mesh will be constructed. The measurement unit of the target_crs should be in kilometers.

Value

A constrained refined Delaunay triangulation mesh created based on the country borders.

Examples

```
path1 <- system.file("extdata", "geoData.rda", package = "GeoAdjust")
load(path1)
crs_KM = "+units=km +proj=utm +zone=37 +ellps=clrk80
+towgs84=-160,-6,-302,0,0,0,0 +no_defs"
mesh.s <- meshCountry(admin0= adm0, max.edge = c(25, 50), offset = -.08,
cutoff=4, target_crs = crs_KM)
```

---

plotPred *Plots the predictions and the corresponding uncertainty (coefficient of variation)*

Description

Plots the predictions and the corresponding uncertainty (coefficient of variation)

Usage

```
plotPred(
  pred = NULL,
  predRaster = NULL,
  admin0 = NULL,
  admin1 = NULL,
  admin2 = NULL,
  rmPoly = NULL,
  target_crs = NULL
)
```

Arguments

pred	A matrix that is the output of predRes() function.
predRaster	The prediction raster that is constructed by the gridCountry() function.
admin0	An sf class MULTIPOLYGON representing the national level (admin0) borders of the country.
admin1	An sf class MULTIPOLYGON representing the first level (admin1) subnational borders of the country.
admin2	An sf class MULTIPOLYGON representing the second level (admin2) subnational borders of the country.
rmPoly	A number referring to the ID number of the admin2 level polygon that needs to be left uncolored. It can be set to NULL as well.
target_crs	A projection string representing the desired coordinate reference system according to which the maps will be created.

Value

A list of two ggplot objects. One of them (ggPred) shows the median predictions and the other one (ggUncertainty) shows the corresponding coefficient of variations across the country, respectively.

Examples

```
path1 <- system.file("extdata", "examplePredictionResults.rda", package = "GeoAdjust")
path2 <- system.file("extdata", "geoData.rda", package = "GeoAdjust")
load(path1)
load(path2)
crs_KM = "+units=km +proj=utm +zone=37 +ellps=clrk80 +towgs84=-160,-6,-302,0,0,0,0 +no_defs"
exampleGrid <- gridCountry(admin0 = adm0, res = 5, target_crs = crs_KM)
plots = plotPred(pred = examplePredictionResults,
predRaster = exampleGrid[["predRast"]], admin0 = adm0,
admin1 = adm1, target_crs = crs_KM)
```

predRes *Predicts model outcomes at new locations.*

Description

Predicts model outcomes at new locations.

Usage

```
predRes(
  obj = NULL,
  predCoords = NULL,
  draws = NULL,
  covariateData = NULL,
  mesh.s = NULL,
  flag = NULL
)
```

Arguments

obj	The optimized core model object returned by estimateModel() function.
predCoords	An sf class POINT object containing the coordinates of the prediction locations. It should contain crs information.
draws	A matrix containing 10.000 sampled values for each covariate effect size and 10.000 sampled values of random effect coefficients for each mesh node. It is one of the elements of the returning output list of estimateModel() function.
covariateData	A list containing the covariate rasters. Each covariate raster should be SpatRast object and contain crs information.
mesh.s	A mesh created based on the country borders.
flag	A value indicating the type of the likelihood that will be used. Pass 0 for Gaussian, 1 for binomial and 2 for Poisson likelihoods.

Value

A matrix containing the mean, median,standard deviation and the lower and the upper bounds of 95

Examples

```
path1 <- system.file("extdata", "exampleInputData.rda", package = "GeoAdjust")
path2 <- system.file("extdata", "exampleMesh.rda", package = "GeoAdjust")
path3 <- system.file("extdata", "geoData.rda", package = "GeoAdjust")
load(path1)
load(path2)
load(path3)
results <- estimateModel(data = exampleInputData,
options = list(random = 1, covariates = 1), priors = list(beta = c(0,1),
```

```
range = 114), USpatial = 1, alphaSpatial = 0.05, UNugget = 1, alphaNug = 0.05, n.sims = 1000)
crs_KM = "+units=km +proj=utm +zone=37 +ellps=clrk80 +towgs84=-160,-6,-302,0,0,0,0 +no_defs"
exampleGrid <- gridCountry(admin0 = adm0, res = 5, target_crs = crs_KM)
pred = predRes(obj = results[["obj"]],
predCoords = exampleGrid[["loc.pred"]],
draws = results[["draws"]],
mesh.s = exampleMesh, covariateData = NULL, flag = 1)
```

`prepareInput` *Prepares input data list for the model estimation with estimateModel() function.*

Description

Prepares input data list for the model estimation with estimateModel() function.

Usage

```
prepareInput(
  response = NULL,
  locObs = NULL,
  likelihood,
  jScale = 1,
  urban = NULL,
  mesh.s = NULL,
  adminMap = NULL,
  covariateData = NULL,
  target_crs
)
```

Arguments

`response`	A list containing the number of trials (ns) and number of successes (ys) for the binomial model, response values (ys) for the Gaussian model or the Poisson counts for the Poisson model.
`locObs`	An sf class multipoint object containing the coordinates of DHS survey cluster centers. (It should contain the crs information.)
`likelihood`	A value indicating which likelihood model should be used (0, 1 or 2 for Gaussian, binomial or Poisson, respectively).
`jScale`	Jittering scale, where 1 represents the default DHS jittering scheme. It allows experimenting with larger jittering scales. Otherwise should remain as 1.
`urban`	A vector containing the urbanization classification of the administrative area that each cluster center is initially located within (U for urban and R for rural).
`mesh.s`	A triangulation mesh. It should be constructed in the target_crs.

adminMap	An sf class multipolygon object. It contains the borders of the administrative area level that was respected while the cluster centers were initially being jittered (can be obtained from https://gadm.org). It should contain crs information.
covariateData	A list containing the covariates as SpatRaster objects. Each SpatRaster should contain the crs information.
target_crs	A projection string representing the desired coordinate reference system according to which, the integration rings and integration points will be located. The measurement unit of the target_crs should be in kilometers.

Value

A list containing the data input for estimateModel() function.

Examples

```
path1 <- system.file("extdata", "geoData.rda", package = "GeoAdjust")
path2 <- system.file("extdata", "exampleMesh.rda", package = "GeoAdjust")
load(path1)
load(path2)
crs_KM = "+units=km +proj=utm +zone=37 +ellps=clrk80 +towgs84=-160,-6,-302,0,0,0,0 +no_defs"
crs_Degrees = "+proj=longlat +datum=WGS84"
locObs = data.frame(long = surveyData$long, lat = surveyData$lat)
locObs = sf::st_as_sf(locObs, coords=c("long","lat"), crs = crs_Degrees)
inputData <- prepareInput(response = list(ys = surveyData$ys, ns = surveyData$ns),
locObs = locObs, likelihood = 1, jScale = 1,
urban = surveyData$urbanRural, mesh.s = exampleMesh, adminMap = adm1,
covariateData = NULL, target_crs = crs_KM)
```

---

print.res *Prints the output of estimateModel() function.*

Description

Prints the output of estimateModel() function.

Usage

```
## S3 method for class 'res'
print(x, ...)
```

Arguments

x	A list containing the model estimation output, returned by estimateModel() function.
...	not used

Value

Prints the model estimation results of class GAmodel as a table that shows the estimated model parameters and the corresponding 95 interval lengths.

Examples

```
path1 <- system.file("extdata", "exampleInputData.rda", package = "GeoAdjust")
load(path1)
results <- estimateModel(data = exampleInputData,
options = list(random = 1, covariates = 1), priors = list(beta = c(0,1),
range = 114, USpatial = 1, alphaSpatial = 0.05, UNugget = 1, alphaNug = 0.05), n.sims = 1000)
print(results)
```

Index