Package 'SUMMER'

February 14, 2022

Type Package

Title Small-Area-Estimation Unit/Area Models and Methods for Estimation in R

Version 1.2.1

Date 2022-02-12

Description Provides methods for spatial and spatio-temporal smoothing of demographic and health indicators using survey data, with particular focus on estimating and projecting under-five mortality rates, described in Mercer et al. (2015) <doi:10.1214/15-AOAS872>, Li et al. (2019) <doi:10.1371/journal.pone.0210645> and Li et al. (2020) <arXiv:2007.05117>.

URL https://github.com/richardli/SUMMER

BugReports https://github.com/richardli/SUMMER/issues

Depends R (>= 3.5)

License GPL (>= 2)

Imports survey, stats, spdep, survival, ggplot2, scales, utils, Matrix, reshape2, viridis, sp, shadowtext, ggridges, methods, data.table, RColorBrewer, grDevices

Encoding UTF-8

LazyData true

RoxygenNote 7.1.2

Additional_repositories https://inla.r-inla-download.org/R/testing/

Suggests INLA, knitr, rmarkdown, readstata13, patchwork, rdhs, R.rsp, fields

VignetteBuilder R.rsp, knitr

NeedsCompilation no

Config/build/clean-inst-doc FALSE

Author Zehang R Li [cre, aut], Bryan D Martin [aut], Yuan Hsiao [aut], Jessica Godwin [aut],

John Paige [aut],
Jon Wakefield [aut],
Samuel J Clark [aut],
Geir-Arne Fuglstad [aut],
Andrea Riebler [aut]

Maintainer Zehang R Li <lizehang@gmail.com>

Repository CRAN

Date/Publication 2022-02-14 08:00:09 UTC

R topics documented:

`SUMMER-package` *SUMMER package documentation.*

Description

SUMMER provides methods for spatial and spatio-temporal smoothing of demographic and health indicators using survey data, with particular focus on estimating and projecting under-five mortality rates.

Details

For details on the model implemented in this package, see Mercer et al. (2015) <doi:10.1214/15-AOAS872> and Li et al. (2019) <doi:10.1371/journal.pone.0210645>.

The development version of the package will be maintained on https://github.com/richardli/SUMMER.

`aggregateSurvey` *Aggregate estimators from different surveys.*

Description

Aggregate estimators from different surveys.

Usage

```
aggregateSurvey(data)
```

Arguments

`data`	Output from `getDirectList`

Value

Estimators aggregated across surveys.

Author(s)

Zehang Richard Li

Examples

```
## Not run:
data(DemoData)
data(DemoMap)
years <- levels(DemoData[[1]]$time)

# obtain direct estimates
data <- getDirectList(births = DemoData,
years = years,
regionVar = "region", timeVar = "time",
clusterVar = "~clustid+id",
ageVar = "age", weightsVar = "weights",
geo.recode = NULL)

# obtain maps
geo <- DemoMap$geo
mat <- DemoMap$Amat

# Simulate hyper priors
priors <- simhyper(R = 2, nsamp = 1e+05, nsamp.check = 5000, Amat = mat, only.iid = TRUE)

# combine data from multiple surveys
data <- aggregateSurvey(data)
utils::head(data)


## End(Not run)
```

BRFSS *The BRFSS dataset*

Description

The Behavioral Risk Factor Surveillance System (BRFSS) is an annual telephone health survey conducted by the Centers for Disease Control and Prevention (CDC) that tracks health conditions and risk behaviors in the United States and its territories since 1984. This BRFSS dataset contains 16124 observations. The 'diab2' variable is the binary indicator of Type II diabetes, 'strata' is the strata indicator and 'rwt_llcp' is the final design weight. Records with missing HRA code or diabetes status are removed from this dataset. See https://www.cdc.gov/brfss/annual_data/2013/pdf/Weighting_Data.pdf for more details of the weighting procedure.

Usage

```
data(BRFSS)
```

Format

A data.frame of 26 variables.

References

Washington State Department of Health, Center for Health Statistics. Behavioral Risk Factor Surveillance System, supported in part by the Centers for Disease Control and Prevention. Corporative Agreement U58/DP006066-01 (2015).

`ChangeRegion` *Map region names to a common set.*

Description

Map region names to a common set.

Usage

```
ChangeRegion(data, Bmat, regionVar = "region")
```

Arguments

`data`	Preprocessed data
`Bmat`	Matrix of changes. Each row corresponds to a region name possibly in the data files, and each column corresponds to a region after mapping. The values in the matrix are binary. The row names and column names need to be specified to the region names.
`regionVar`	String indicating the region variable. Defaults to 'region'.

Value

Data after changing region names

Author(s)

Zehang Richard Li

Examples

```
# Construct a small test data
testdata <- data.frame(region = c("north", "south", "east",
 "south", "east"), index = c(1:5))

# Construct a changing rule: combining south and east
Bmat <- matrix(c(1, 0, 0, 0, 1, 1), 3, 2)
colnames(Bmat) <- c("north", "south and east")
rownames(Bmat) <- c("north", "south", "east")
```

```
print(Bmat)

# New data after transformation
test <- ChangeRegion(testdata, Bmat, "region")
print(test)
```

DemoData *Simulated child mortality person-month dataset.*

Description

A small simulated dataset with 4 regions and 5 survey years. This does not represent any real country's data and are based on a subset of the model dataset provided by DHS.

Usage

```
data(DemoData)
```

Format

A list of with five components, named by survey year.

Source

https://dhsprogram.com/data/model-datasets.cfm

DemoData2 *Simulated dataset for prevalence mapping.*

Description

A small fake dataset with 8 regions and two response variables: age and tobacco.use. This does not represent any real country's data and are based on a subset of the model dataset provided by DHS.

Usage

```
data(DemoData2)
```

Format

A data.frame of 7 variables.

Source

https://dhsprogram.com/data/model-datasets.cfm

DemoMap *Uganda Admin-1 region map for illustration purpose*

Description

Shapefiles are from 1995 Uganda Admin 1 regions provided by DHS, but the data do not represent real information about any country.

Usage

```
data(DemoMap)
```

Format

An object of class `list` of length 2.

Details

- geo. Geographic map files
- Amat. Adjacency matrix for regions

Source

https://spatialdata.dhsprogram.com/boundaries/#view=table&countryId=UG

DemoMap2 *Kenya Admin-1 region map for illustration purpose*

Description

Shapefiles are from 2014 Kenya Admin 1 regions provided by DHS.

Usage

```
data(DemoMap2)
```

Format

An object of class `list` of length 2.

Details

- geo Geographic map files
- Amat Adjacency matrix for regions

Source

https://spatialdata.dhsprogram.com/boundaries/#view=table&countryId=KE

expit *Expit transformation*

Description

Expit transformation

Usage

```
expit(x)
```

Arguments

x	data

Value

expit of x

Examples

```
x <- .5
expit(x)
```

getAdjusted *Adjust direct estimates and their associated variances*

Description

Adjust direct estimates and their associated variances

Usage

```
getAdjusted(
  data,
  ratio,
  time = "years",
  region = "region",
  est = "mean",
  logit = "logit.est",
  logit.var = "var.est",
  logit.prec = "logit.prec",
  logit.lower = "lower",
  logit.upper = "upper",
  prob.lower = NULL,
```

```
  prob.upper = NULL,
  adj = "ratio",
  verbose = FALSE,
  lower = NULL,
  upper = NULL
)
```

Arguments

data	data frame of the adjusted estimates and the associated uncertainties, see the arguments below for specific columns.
ratio	the ratio of unadjusted mortality rates to the true mortality rates. It can be either a data frame with the following three columns (region, time, and adj) if adjustment factor differ by region; or a data frame with the following two columns (time and adj) if adjustment factor only varies over time. The column names specifying region, time, and adjustment are specified by the arguments in the function call.
time	the column name for time in the data and adjustment ratio.
region	the column name for region in the data and adjustment ratio.
est	the column name for unadjusted mortality rates in the data
logit	the column name for the logit of the unadjusted mortality rates in the data
logit.var	the column name for the variance of the logit of the unadjusted mortality rates in the data
logit.prec	the column name for the precision of the logit of the unadjusted mortality rates in the data
logit.lower	the column name for the 95% lower bound of the logit of the unadjusted mortality rates in the data
logit.upper	the column name for the 95% lower bound of the logit of the unadjusted mortality rates in the data
prob.lower	the column name for the 95% lower bound of the unadjusted mortality rates in the data. If this is provided instead of logit.lower, the logit scale lower bound will be created.
prob.upper	the column name for the 95% lower bound of the unadjusted mortality rates in the data. if this is provided instead of logit.upper, the logit scale upper bound will be created.
adj	the column name for the adjustment ratio
verbose	logical indicator for whether to print out unadjusted row index
lower	previous argument name for prob.lower. Will be removed in the next update
upper	previous argument name for prob.upper. Will be removed in the next update

Value

adjusted dataset of the same columns.

Author(s)

Zehang Richard Li

Examples

```
## Not run:
years <- levels(DemoData[[1]]$time)

# obtain direct estimates
data <- getDirectList(births = DemoData,
years = years,
regionVar = "region", timeVar = "time",
clusterVar = "~clustid+id",
ageVar = "age", weightsVar = "weights",
geo.recode = NULL)
# obtain direct estimates
data_multi <- getDirectList(births = DemoData, years = years,
  regionVar = "region",  timeVar = "time", clusterVar = "~clustid+id",
  ageVar = "age", weightsVar = "weights", geo.recode = NULL)
data <- aggregateSurvey(data_multi)

# randomly simulate adjustment factor
adj <- expand.grid(region = unique(data$region), years = years)
adj$ratio <- runif(dim(adj)[1], min = 0.5, max = 0.8)
data.adj <- getAdjusted(data = data, ratio = adj)

## End(Not run)
```

---

getAmat *Extract adjacency matrix from the map*

Description

Extract adjacency matrix from the map

Usage

```
getAmat(geo, names)
```

Arguments

geo	SpatialPolygonsDataFrame of the map
names	character vector of region ids to be added to the neighbours list

Value

Spatial djacency matrix.

Author(s)

Zehang Richard Li

Examples

```
## Not run:
data(DemoMap)
mat <- getAmat(geo = DemoMap$geo, names = DemoMap$geo$REGNAME)
mat
DemoMap$Amat

## End(Not run)
```

---

getBirths *Reformat full birth records into person-month format*

Description

Reformat full birth records into person-month format

Usage

```
getBirths(
  filepath = NULL,
  data = NULL,
  surveyyear = NA,
  variables = c("caseid", "v001", "v002", "v004", "v005", "v021", "v022", "v023",
    "v024", "v025", "v139", "bidx"),
  strata = c("v024", "v025"),
  dob = "b3",
  alive = "b5",
  age = "b7",
  age.truncate = 24,
  date.interview = "v008",
  month.cut = c(1, 12, 24, 36, 48, 60),
  year.cut = seq(1980, 2020, by = 5),
  min.last.period = 0,
  cmc.adjust = 0,
  compact = FALSE,
  compact.by = c("v001", "v024", "v025", "v005")
)
```

Arguments

filepath	file path of raw .dta file from DHS. Only used when data frame is not provided in the function call.
data	data frame of a DHS survey
surveyyear	year of survey. Observations after this year will be excluded from the analysis.
variables	vector of variables to be used in obtaining the person-month files. The variables correspond the the DHS recode manual VI. For early DHS data, the variable names may need to be changed.

`strata` vector of variable names used for strata. If a single variable is specified, then that variable will be used as strata indicator If multiple variables are specified, the interaction of these variables will be used as strata indicator.

`dob` variable name for the date of birth.

`alive` variable name for the indicator of whether child was alive or dead at the time of interview. It should be factor or character variable with levels "no" or "yes". Other coding scheme will not be recognized and can lead to errors.

`age` variable name for the age at death of the child in completed months.

`age.truncate` the smallest age in months where only full years are reported. The default value is 24, which corresponds to the DHS practice of recording only age in full years for children over 2 years old. That is, for children with age starting from 24 months old, we assume the age variable reported in multiples of 12 are truncated from its true value. For example, children between age 24 to 35 months are all recorded as 24. To account for the truncation of age, 5 months are added to all ages recorded in multiples of 12 starting from 24. To avoid this adjustment, set this argument to NA.

`date.interview` variable name for the date of interview.

`month.cut` the cutoff of each bins of age group in the unit of months. Default values are 1, 12, 24, 36, 48, and 60, representing the age groups (0, 1), [1, 12), [12, 24), ..., [48, 60).

`year.cut` The cutoff of each bins of time periods, including both boundaries. Default values are 1980, 1985, ..., 2020, representing the time periods 80-84, 85-89, ..., 15-19. Notice that if each bin contains one year, the last year in the output is max(year.cut)-1. For example, if year.cut = 1980:2020, the last year in the output is 2019.

`min.last.period` The cutoff for how many years the last period must contain in order to be counted in the output. For example, if the last period is 2015-2019 and min.last.period = 3, person-months for the last period will only be returned if survey contains observations at least in 2017. This argument avoids the situation that estimates for the last period being based on only a small number of initial years, if applicable. Default to be 0.

`cmc.adjust` number of months to add to the recorded month in the dataset. Some DHS surveys does not use Gregorian calendar (the calendar used in most of the world). For example, the Ethiopian calendar is 92 months behind the Gregorian calendar in general. Then we can set cmc.adjust to 92, which adds 92 months to all dates in the dataset, effectively transforming the Ethiopian calendar to the Gregorian calendar.

`compact` logical indicator of whether the compact format is returned. In the compact output, person months are aggregated by cluster, age, and time. Total number of person months and deaths in each group are returned instead of the raw person-months.

`compact.by` vector of variables to summarize the compact form by.

Value

This function returns a new data frame where each row indicate a person-month, with the additional variables specified in the function argument.

Author(s)

Zehang Richard Li, Bryan Martin, Laina Mercer

References

Li, Z., Hsiao, Y., Godwin, J., Martin, B. D., Wakefield, J., Clark, S. J., & with support from the United Nations Inter-agency Group for Child Mortality Estimation and its technical advisory group. (2019). *Changes in the spatial distribution of the under-five mortality rate: Small-area analysis of 122 DHS surveys in 262 subregions of 35 countries in Africa.* PloS one, 14(1), e0210645.

Mercer, L. D., Wakefield, J., Pantazis, A., Lutambi, A. M., Masanja, H., & Clark, S. (2015). *Space-time smoothing of complex survey data: small area estimation for child mortality.* The annals of applied statistics, 9(4), 1889.

Examples

```
## Not run:
my_fp <- "/myExampleFilepath/surveyData.DTA"
DemoData <- getBirths(filepath = my_fp, surveyyear = 2015)

## End(Not run)
```

---

getCounts *Aggregate person-month data into counts and totals by groups.*

Description

Aggregate person-month data into counts and totals by groups.

Usage

```
getCounts(data, variables, by, ignore = NULL, addtotal = TRUE, drop = TRUE)
```

Arguments

data	dataset in person-month format
variables	a character vector of the variables to aggregate
by	a character vector of columns that specifies which groups to aggregate by.
ignore	list of conditions not to impute 0. If left unspecified, any group levels not in the data will be imputed to have 0 counts.
addtotal	logical indicator of whether to add a column of group total counts.
drop	logical indicator of whether to drop all rows with total = 0.

Value

data.frame of the ggregated counts.

Author(s)

Zehang Richard Li

Examples

```
# a toy dataset with 4 time periods but one missing in data
timelist <- factor(1:4)
data = data.frame(died = c(0,0,0,1,1,0,0),
area = c(rep(c("A", "B"), 3), "A"),
time = timelist[c(1,1,2,3,3,3,3)])
data
# without ignore argument, all levels will be imputed
getCounts(data, variables = "died", by = c("area", "time"))

# ignoring time = 4, the ignored level will not be imputed (but still in the output)
getCounts(data, variables = "died", by = c("area", "time"),
ignore = list("time"=c(4)) )
```

`getDiag` *Extract posterior summaries of random effects*

Description

Extract posterior summaries of random effects

Usage

```
getDiag(
  inla_mod,
  field = c("space", "time", "spacetime")[1],
  CI = 0.95,
  draws = NULL,
  ...
)
```

Arguments

`inla_mod`	output from `smoothDirect`
`field`	which random effects to plot. It can be one of the following: space, time, and spacetime.

CI	Desired level of credible intervals
draws	Posterior samples drawn from the fitted model. This argument allows the previously sampled draws (by setting save.draws to be TRUE) be used in new aggregation tasks.
...	Unused arguments, for users with fitted object from the package before v1.0.0, arguments including Amat, year_label, and year_range can still be specified manually.

Value

List of diagnostic plots

Author(s)

Zehang Richard Li

Examples

```
## Not run:
  data(DemoMap)
  years <- levels(DemoData[[1]]$time)

  # obtain direct estimates
  data <- getDirectList(births = DemoData,
  years = years,
  regionVar = "region", timeVar = "time",
  clusterVar = "~clustid+id",
  ageVar = "age", weightsVar = "weights",
  geo.recode = NULL)
  # obtain direct estimates
  data_multi <- getDirectList(births = DemoData, years = years,
    regionVar = "region",  timeVar = "time", clusterVar = "~clustid+id",
    ageVar = "age", weightsVar = "weights", geo.recode = NULL)
  data <- aggregateSurvey(data_multi)

  #  national model
  years.all <- c(years, "15-19")
  fit1 <- smoothDirect(data = data, geo = DemoMap$geo, Amat = DemoMap$Amat,
    year_label = years.all, year_range = c(1985, 2019),
    rw = 2, is.yearly=FALSE, m = 5)
random.time <- getDiag(fit1, field = "time")
  random.space <- getDiag(fit1, field = "space")
  random.spacetime <- getDiag(fit1, field = "spacetime")

## End(Not run)
```

getDirect *Obtain the Horvitz-Thompson direct estimates and standard errors using delta method for a single survey.*

Description

Obtain the Horvitz-Thompson direct estimates and standard errors using delta method for a single survey.

Usage

```
getDirect(
  births,
  years,
  regionVar = "region",
  timeVar = "time",
  clusterVar = "~v001+v002",
  ageVar = "age",
  weightsVar = "v005",
  Ntrials = NULL,
  geo.recode = NULL,
  national.only = FALSE
)
```

Arguments

births	A matrix child-month data from getBirths
years	String vector of the year intervals used
regionVar	Variable name for region in the input births data.
timeVar	Variable name for the time period indicator in the input births data.
clusterVar	Variable name for cluster, typically '~v001 + v002'
ageVar	Variable name for age group. This variable need to be in the form of "a-b" where a and b are both ages in months. For example, "1-11" means age between 1 and 11 months, including both end points. An exception is age less than one month can be represented by "0" or "0-0".
weightsVar	Variable name for sampling weights, typically 'v005'
Ntrials	Variable for the total number of person-months if the input data (births) is in the compact form.
geo.recode	The recode matrix to be used if region name is not consistent across different surveys. See ChangeRegion.
national.only	Logical indicator to obtain only the national estimates

Value

a matrix of period-region summary of the Horvitz-Thompson direct estimates by region and time period specified in the argument, the standard errors using delta method for a single survey, the 95% confidence interval, and the logit of the estimates.

Author(s)

Zehang Richard Li, Bryan Martin, Laina Mercer

References

Li, Z., Hsiao, Y., Godwin, J., Martin, B. D., Wakefield, J., Clark, S. J., & with support from the United Nations Inter-agency Group for Child Mortality Estimation and its technical advisory group. (2019). *Changes in the spatial distribution of the under-five mortality rate: Small-area analysis of 122 DHS surveys in 262 subregions of 35 countries in Africa.* PloS one, 14(1), e0210645.

Mercer, L. D., Wakefield, J., Pantazis, A., Lutambi, A. M., Masanja, H., & Clark, S. (2015). *Space-time smoothing of complex survey data: small area estimation for child mortality.* The annals of applied statistics, 9(4), 1889.

See Also

getDirectList

Examples

```
## Not run:
data(DemoData)
years <- c("85-89", "90-94", "95-99", "00-04", "05-09", "10-14")
mean <- getDirect(births = DemoData[[1]],  years = years,
regionVar = "region", timeVar = "time", clusterVar = "~clustid+id",
ageVar = "age", weightsVar = "weights", geo.recode = NULL)

## End(Not run)
```

`getDirectList` *Obtain the Horvitz-Thompson direct estimates and standard errors using delta method for multiple surveys.*

Description

Obtain the Horvitz-Thompson direct estimates and standard errors using delta method for multiple surveys.

Usage

```
getDirectList(
  births,
  years,
  regionVar = "region",
  timeVar = "time",
  clusterVar = "~v001+v002",
  ageVar = "age",
  weightsVar = "v005",
  Ntrials = NULL,
  geo.recode = NULL,
  national.only = FALSE
)
```

Arguments

births	A list of child-month data from multiple surveys from getBirths. The name of the list is used as the identifier in the output.
years	String vector of the year intervals used
regionVar	Variable name for region, typically 'v024', for older surveys might be 'v101'
timeVar	Variable name for the time period indicator in the input births data.
clusterVar	Variable name for the IDs in the second-stage cluster sampling, typically '~v001 + v002', i.e., the cluster number and household number. When no cluster sampling design exists, this variable usually is the household ID.
ageVar	Variable name for age group. This variable need to be in the form of "a-b" where a and b are both ages in months. For example, "1-11" means age between 1 and 11 months, including both end points. An exception is age less than one month can be represented by "0" or "0-0".
weightsVar	Variable name for sampling weights, typically 'v005'
Ntrials	Variable for the total number of person-months if the input data (births) is in the compact form.
geo.recode	The recode matrix to be used if region name is not consistent across different surveys. See ChangeRegion.
national.only	Logical indicator to obtain only the national estimates

Value

This is the extension to the getDirect function that returns estimates from multiple surveys. Additional columns in the output (survey and surveyYears) specify the estimates from different surveys.

Author(s)

Zehang Richard Li, Bryan Martin, Laina Mercer

References

Li, Z., Hsiao, Y., Godwin, J., Martin, B. D., Wakefield, J., Clark, S. J., & with support from the United Nations Inter-agency Group for Child Mortality Estimation and its technical advisory group. (2019). *Changes in the spatial distribution of the under-five mortality rate: Small-area analysis of 122 DHS surveys in 262 subregions of 35 countries in Africa.* PloS one, 14(1), e0210645.

Mercer, L. D., Wakefield, J., Pantazis, A., Lutambi, A. M., Masanja, H., & Clark, S. (2015). *Space-time smoothing of complex survey data: small area estimation for child mortality.* The annals of applied statistics, 9(4), 1889.

See Also

getDirect

Examples

```
## Not run:
data(DemoData)
years <- c("85-89", "90-94", "95-99", "00-04", "05-09", "10-14")
mean <- getDirectList(births = DemoData, years = years,
regionVar = "region", timeVar = "time", clusterVar = "~clustid+id",
ageVar = "age", weightsVar = "weights", geo.recode = NULL)

## End(Not run)
```

---

getSmoothed *Extract smoothed estimates.*

Description

Extract smoothed estimates.

Usage

```
getSmoothed(
  inla_mod,
  nsim = 1000,
  weight.strata = NULL,
  weight.frame = NULL,
  verbose = FALSE,
  mc = 0,
  include_time_unstruct = FALSE,
  CI = 0.95,
  draws = NULL,
  save.draws = FALSE,
  include_subnational = TRUE,
  ...
)
```

Arguments

inla_mod	output from smoothDirect or smoothCluster
nsim	number of simulations, only applicable for the cluster-level model. The smooth direct model always draws 1e5 samples from the marginal distribution since the computation is faster.
weight.strata	a data frame with two columns specifying time and region, followed by columns specifying proportion of each strata for each region. This argument specifies the weights for strata-specific estimates on the probability scale.
weight.frame	a data frame with three columns, years, region, and the weight of each frame for the corresponding time period and region. This argument specifies the weights for frame-specific estimates on the logit scale. Notice this is different from weight.strata argument.
verbose	logical indicator whether to print progress messages from inla.posterior.sample.
mc	number of monte carlo draws to approximate the marginal prevalence/hazards for binomial model. If mc = 0, analytical approximation is used. The analytical approximation is invalid for hazard modeling with more than one age groups.
include_time_unstruct	Indicator whether to include the temporal unstructured effects (i.e., shocks) in the smoothed estimates from cluster-level model. The argument only applies to the cluster-level models (from smoothCluster). Default is FALSE which excludes all unstructured temporal components. If set to TRUE all the unstructured temporal random effects will be included. Alternatively, if this is specified as a vector of subset of year labels (as in the year_label argument), only the unstructured terms in the corresponding time periods will be added to the prediction.
CI	Desired level of credible intervals
draws	Posterior samples drawn from the fitted model. This argument allows the previously sampled draws (by setting save.draws to be TRUE) be used in new aggregation tasks.
save.draws	Logical indicator whether the raw posterior draws will be saved. Saved draws can be used to accelerate aggregations with different weights.
include_subnational	logical indicator whether to include the spatial and space-time interaction components in the smoothed estimates. If set to FALSE, only the main temporal trends are returned.
...	Unused arguments, for users with fitted object from the package before v1.0.0, arguments including Amat, year_label, and year_range can still be specified manually.

Value

A data frame or a list of data frames of S3 class SUMMERproj, which contains the smoothed estimates.

Author(s)

Zehang Richard Li

See Also

plot.SUMMERproj

Examples

```
## Not run:
years <- levels(DemoData[[1]]$time)

# obtain direct estimates
data <- getDirectList(births = DemoData,
years = years,
regionVar = "region", timeVar = "time",
clusterVar = "~clustid+id",
ageVar = "age", weightsVar = "weights",
geo.recode = NULL)
# obtain direct estimates
data_multi <- getDirectList(births = DemoData, years = years,
  regionVar = "region",  timeVar = "time", clusterVar = "~clustid+id",
  ageVar = "age", weightsVar = "weights", geo.recode = NULL)
data <- aggregateSurvey(data_multi)

#  national model
years.all <- c(years, "15-19")
fit1 <- smoothDirect(data = data, Amat = NULL,
  year_label = years.all, year_range = c(1985, 2019),
  rw = 2, is.yearly=FALSE, m = 5)
out1 <- getSmoothed(fit1)
plot(out1, is.subnational=FALSE)

#  subnational model
fit2 <- smoothDirect(data = data, Amat = mat,
  year_label = years.all, year_range = c(1985, 2019),
  rw = 2, is.yearly=TRUE, m = 5, type.st = 4)
out2 <- getSmoothed(fit2)
plot(out2, is.yearly=TRUE, is.subnational=TRUE)



## End(Not run)
```

---

hatchPlot *Plot maps with uncertainty hatching.*

Description

This function visualizes the map with different variables. The input data frame can be either the long or wide format.

Usage

```
hatchPlot(
  data,
  variables,
  values = NULL,
  labels = NULL,
  geo,
  by.data,
  by.geo,
  is.long = FALSE,
  lower,
  upper,
  lim = NULL,
  lim.CI = NULL,
  breaks.CI = NULL,
  ncol = 4,
  hatch = NULL,
  border = NULL,
  size = 1,
  legend.label = NULL,
  per1000 = FALSE,
  direction = 1,
  ...
)
```

Arguments

data	a data frame with variables to be plotted
variables	vector of variables to be plotted. If long format of data is used, only one variable can be selected
values	the column corresponding to the values to be plotted, only used when long format of data is used
labels	vector of labels to use for each variable, only used when wide format of data is used
geo	SpatialPolygonsDataFrame object for the map
by.data	column name specifying region names in the data
by.geo	variable name specifying region names in the data
is.long	logical indicator of whether the data is in the long format, default to FALSE
lower	column name of the lower bound of the CI
upper	column name of the upper bound of the CI
lim	fixed range of values for the variables to plot
lim.CI	fixed range of the CI widths to plot
breaks.CI	a vector of numerical values that decides the breaks in the CI widths to be shown
ncol	number of columns for the output tabs

hatch	color of the hatching lines.
border	color of the polygon borders.
size	line width of the polygon borders.
legend.label	Label for the color legend.
per1000	logical indicator to plot mortality rates as rates per 1,000 live births. Note that the added comparison data should always be in the probability scale.
direction	Direction of the color scheme. It can be either 1 (smaller values are darker) or -1 (higher values are darker). Default is set to 1.
...	unused.

Author(s)

Zehang Richard Li, Katie Wilson

Examples

```
## Not run:
years <- levels(DemoData[[1]]$time)

# obtain direct estimates
data <- getDirectList(births = DemoData,
years = years,
regionVar = "region", timeVar = "time",
clusterVar = "~clustid+id",
ageVar = "age", weightsVar = "weights",
geo.recode = NULL)
# obtain direct estimates
data_multi <- getDirectList(births = DemoData, years = years,
  regionVar = "region",  timeVar = "time", clusterVar = "~clustid+id",
  ageVar = "age", weightsVar = "weights", geo.recode = NULL)
data <- aggregateSurvey(data_multi)

fit2 <- smoothDirect(data = data, geo = geo, Amat = mat,
  year_label = years.all, year_range = c(1985, 2019),
  rw = 2, is.yearly=TRUE, m = 5, type.st = 4)
out2 <- getSmoothed(fit2)

plot(out2, is.yearly=TRUE, is.subnational=TRUE)

hatchPlot(data = subset(out2, is.yearly==FALSE), geo = geo,
variables=c("years"), values = c("median"),
by.data = "region", by.geo = "REGNAME",
lower = "lower", upper = "upper", is.long=TRUE)


## End(Not run)
```

iid.new *New random IID models for m-year to period random effects*

Description

New random IID models for m-year to period random effects

Usage

```
iid.new(
  cmd = c("graph", "Q", "mu", "initial", "log.norm.const", "log.prior", "quit"),
  theta = NULL
)
```

Arguments

cmd	list of model components
theta	log precision

iid.new.pc *New random IID models for m-year to period random effects*

Description

New random IID models for m-year to period random effects

Usage

```
iid.new.pc(
  cmd = c("graph", "Q", "mu", "initial", "log.norm.const", "log.prior", "quit"),
  theta = NULL
)
```

Arguments

cmd	list of model components
theta	log precision

KenData *Auxiliary data for Kenya 2014 DHS.*

Description

The list contains several data frames.

Usage

```
data(KenData)
```

Format

An object of class `list` of length 4.

Details

- HIV2014, a data frame with three columns: years (in five year periods), region (8 Admin-1 region groups), and the estimated bias of the reported U5MR due to HIV for each 5 year period from 1990-1994 to 2010-2014. The bias is represented as the ratio of the reported U5MR to the true U5MR.
- HIV2014.yearly, a data frame with three columns: years (in one year interval), region (8 Admin-1 region groups), and the estimated bias of the reported U5MR due to HIV for each year from 1980 to 2014. The bias is represented as the ratio of the reported U5MR to the true U5MR.
- IGME2019. Yearly Estimates of national under-5 child mortality in Kenya from the 2019 UN-IGME estimates.
- UrbanProp. Proportion of urban population by county and total population by county. Source: 2009 Kenya Population and Housing Census, and Table A2 of Kenya 2014 DHS report.

References

Neff Walker, Kenneth Hill, and Fengmin Zhao (2012) *Child mortality estimation: methods used to adjust for bias due to aids in estimating trends in under-five mortality.*,
PLoS Medicine, 9(8):e1001298.

KingCounty *Map of King County*

Description

Shapefiles are King County in the Washington States.

Usage

```
KingCounty
```

Format

An object of class SpatialPolygonsDataFrame with 48 rows and 9 columns.

`logit` *Logit transformation*

Description

Logit transformation

Usage

```
logit(x)
```

Arguments

x	data

Value

logit of x

Examples

```
x <- .5
logit(x)
```

`MalawiData` *Auxiliary data for Malawi 2000, 2004, 2010, and 2015 DHS.*

Description

The list contains several data frames.

Usage

```
data(MalawiData)
```

Format

An object of class `list` of length 4.

Details

- HIV, a data frame with three columns: years (in five year periods), survey, and the estimated bias of the reported U5MR due to HIV for each 5 year period. The bias is represented as the ratio of the reported U5MR to the true U5MR.
- HIV.yearly, a data frame with three columns: years (in one year interval), survey, and the estimated bias of the reported U5MR due to HIV for each year. The bias is represented as the ratio of the reported U5MR to the true U5MR.
- IGME2019. Yearly Estimates of national under-5 child mortality in Malawi from the 2019 UN-IGME estimates.
- IGME2019.nmr. Yearly Estimates of national neonatal mortality in Malawi from the 2019 UN-IGME estimates.

References

Neff Walker, Kenneth Hill, and Fengmin Zhao (2012) *Child mortality estimation: methods used to adjust for bias due to aids in estimating trends in under-five mortality.*,
PLoS Medicine, 9(8):e1001298.

`MalawiMap` *Malawi Admin-2 map*

Description

SpatialPolygonsDataFrame objects that reflect the Admin 2 regions in Malawi, including the Likoma island. The Admin 2 region names are in the ADM2_EN field.

Usage

```
data(MalawiMap)
```

Format

An object of class `SpatialPolygonsDataFrame` with 28 rows and 14 columns.

`mapPlot` *Plot region-level variables on a map*

Description

This function visualizes the map with different variables. The input data frame can be either the long or wide format.

Usage

```
mapPlot(
  data = NULL,
  variables,
  values = NULL,
  labels = NULL,
  geo,
  by.data,
  by.geo,
  is.long = FALSE,
  size = 0.5,
  removetab = FALSE,
  border = "gray20",
  ncol = NULL,
  ylim = NULL,
  legend.label = NULL,
  per1000 = FALSE,
  clean = TRUE,
  size.label = 2,
  add.adj = FALSE,
  color.adj = "red",
  alpha.adj = 0.85,
  direction = 1,
  cut = NULL
)
```

Arguments

data	a data frame with variables to be plotted. When it is null, a map is produced.
variables	vector of variables to be plotted. If long format of data is used, only one variable can be selected
values	the column corresponding to the values to be plotted, only used when long format of data is used
labels	vector of labels to use for each variable, only used when wide format of data is used
geo	SpatialPolygonsDataFrame object for the map
by.data	column name specifying region names in the data
by.geo	variable name specifying region names in the data
is.long	logical indicator of whether the data is in the long format, default to FALSE
size	size of the border
removetab	logical indicator to not show the tab label, only applicable when only one tab is present.
border	color of the border
ncol	number of columns for the output tabs
ylim	range of the values to be plotted.

legend.label	Label for the color legend.
per1000	logical indicator to plot mortality rates as rates per 1,000 live births. Note that the added comparison data should always be in the probability scale.
clean	remove all coordinates for a cleaner layout, default to TRUE.
size.label	size of the label of the regions.
add.adj	logical indicator to add edges between connected regions.
color.adj	color of the adjacency matrix edges.
alpha.adj	alpha level (transparency) of the adjacency matrix edges.
direction	Direction of the color scheme. It can be either 1 (smaller values are darker) or -1 (higher values are darker). Default is set to 1.
cut	a vector of values to cut the continuous scale color to discrete intervals.

Author(s)

Zehang Richard Li

Examples

```
## Not run:
data(DemoMap)
# Plotting data in the long format
dat <- data.frame(region = rep(c("central",  "eastern", "northern", "western"), 3),
year = rep(c(1980, 1990, 2000), each = 4),
values = stats::rnorm(12))
utils::head(dat)
mapPlot(dat, variables = "year", values = "values",
by.data = "region", geo = DemoMap$geo,
by.geo = "NAME_final", is.long = TRUE)
dat <- data.frame(region = c("central",  "eastern", "northern", "western"),
Year1 = stats::rnorm(4), Year2 = stats::rnorm(4),
Year3 = stats::rnorm(4))
utils::head(dat)
mapPlot(dat, variables = c("Year1", "Year2", "Year3"),
 labels = c(1980, 1990, 2000),
by.data = "region", geo = DemoMap$geo,
by.geo = "NAME_final", is.long = FALSE)


## End(Not run)
```

---

mapPoints *Map GPS points to polygon regions*

Description

Map GPS points to polygon regions

Usage

```
mapPoints(data, geo, long, lat, names)
```

Arguments

data	point data with two columns of GPS locations.
geo	SpatialPolygonsDataFrame of the map
long	column name for longitudinal coordinate in the data
lat	column name for latitude coordinate in the data
names	character vector of region ids to be added to the neighbours list

Value

Spatial djacency matrix.

Author(s)

Zehang Richard Li

Examples

```
data(DemoMap)
dat <- data.frame(ID = c(1,2,3), lon = c(32.2, 33.7, 33), lat = c(0.1, 0.9, 2.8))
dat2 <- mapPoints(dat, DemoMap$geo, long = "lon", lat = "lat", names = "REGNAME")
dat2
```

---

plot.SUMMERproj *Plot projection output.*

Description

Plot projection output.

Usage

```
## S3 method for class 'SUMMERproj'
plot(
  x,
  year_label = c("85-89", "90-94", "95-99", "00-04", "05-09", "10-14", "15-19"),
  year_med = c(1987, 1992, 1997, 2002, 2007, 2012, 2017),
  is.subnational = TRUE,
  proj_year = 2015,
  data.add = NULL,
  option.add = list(point = NULL, lower = NULL, upper = NULL, by = NULL),
  color.add = "black",
```

```
  label.add = NULL,
  dodge.width = 1,
  plot.CI = NULL,
  per1000 = FALSE,
  color.CI = NULL,
  alpha.CI = 0.5,
  ...
)
```

Arguments

x	output from getSmoothed
year_label	labels for the periods
year_med	labels for the middle years in each period, only used when both yearly and period estimates are plotted. In that case, year_med specifies where each period estimates are aligned.
is.subnational	logical indicator of whether the data contains subnational estimates
proj_year	the first year where projections are made, i.e., where no data are available.
data.add	data frame for the Comparisons data points to add to the graph. This can be, for example, the raw direct estimates. This data frame is merged to the projections by column 'region' and 'years'. Except for these two columns, this dataset should not have Comparisons columns with names overlapping the getSmoothed output.
option.add	list of options specifying the variable names for the points to plot, lower and upper bounds, and the grouping variable. This is intended to be used to add Comparisons estimates on the same plot as the smoothed estimates. See examples for details.
color.add	the color of the Comparisons data points to plot.
label.add	the label of the Comparisons data points in the legend.
dodge.width	the amount to add to data points at the same year to avoid overlap. Default to be 1.
plot.CI	logical indicator of whether to plot the error bars.
per1000	logical indicator to plot mortality rates as rates per 1,000 live births. Note that the added comparison data should always be in the probability scale.
color.CI	the color of the error bars of the credible interval.
alpha.CI	the alpha (transparency) of the error bars of the credible interval.
...	optional arguments, see details

Details

Examples of some arguments:

- year_labelstring of year labels, e.g., c("85-89","90-94","95-99","00-04","05-09","10-14","15-19") or c(1985:2019)
- proj_yearthe year projection starts, e.g., 2015
- year_med median of year intervals, e.g., c(1987,1992,1997,2002,2007,2012,2017)

Author(s)

Zehang Richard Li

See Also

getSmoothed

Examples

```
## Not run:
years <- levels(DemoData[[1]]$time)

# obtain direct estimates
data <- getDirectList(births = DemoData,
years = years,
regionVar = "region", timeVar = "time",
clusterVar = "~clustid+id",
ageVar = "age", weightsVar = "weights",
geo.recode = NULL)
# obtain direct estimates
data_multi <- getDirectList(births = DemoData, years = years,
  regionVar = "region",  timeVar = "time", clusterVar = "~clustid+id",
  ageVar = "age", weightsVar = "weights", geo.recode = NULL)
data <- aggregateSurvey(data_multi)

#  national model
years.all <- c(years, "15-19")
fit1 <- smoothDirect(data = data, geo = NULL, Amat = NULL,
  year_label = years.all, year_range = c(1985, 2019),
  rw = 2, is.yearly=FALSE, m = 5)
out1 <- getSmoothed(fit1)
plot(out1, is.subnational=FALSE)

#  subnational model
fit2 <- smoothDirect(data = data, geo = geo, Amat = mat,
  year_label = years.all, year_range = c(1985, 2019),
  rw = 2, is.yearly=TRUE, m = 5, type.st = 4)
out2 <- getSmoothed(fit2)
plot(out2, is.yearly=TRUE, is.subnational=TRUE)



## End(Not run)
```

---

print.SUMMERmodel *Print method for the smoothing models.*

Description

This function is the print method for class SUMMERmodel.

Usage

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

Arguments

x	output from smoothDirect or smoothCluster
...	not used

Author(s)

Zehang Li

See Also

summary.SUMMERmodel

Examples

```
## Not run:
  library(SUMMER)
  library(dplyr)
  data(DemoData)

  # Smooth Direct Model
  years <- levels(DemoData[[1]]$time)
  # obtain direct estimates
  data_multi <- getDirectList(births = DemoData, years = years,
  regionVar = "region",  timeVar = "time", clusterVar = "~clustid+id",
  ageVar = "age", weightsVar = "weights", geo.recode = NULL)
  data <- aggregateSurvey(data_multi)

  years.all <- c(years, "15-19")
  fit <- smoothDirect(data = data, Amat = NULL,
  year_label = years.all, year_range = c(1985, 2019),
  time.model = 'rw2', is.yearly=FALSE, m = 5)
  fit

  # Cluster-level Model
  counts.all <- NULL
  for(i in 1:length(DemoData)){
  counts <- getCounts(DemoData[[i]][, c("clustid", "time", "age", "died",
                                        "region", "strata")],
           variables = 'died', by = c("age", "clustid", "region",
                                      "time", "strata"))
  counts <- counts %>% mutate(cluster = clustid, years = time, Y=died)
  counts$strata <- gsub(".*\\.","",counts$strata)
```

```
  counts$survey <- names(DemoData)[i]
  counts.all <- rbind(counts.all, counts)
  }

  # fit cluster-level model on the periods
  periods <- levels(DemoData[[1]]$time)
  fit <- smoothCluster(data = counts.all,
     Amat = DemoMap$Amat,
     time.model = "rw2",
     st.time.model = "rw1",
     strata.time.effect =  TRUE,
     survey.effect = TRUE,
     family = "betabinomial",
     year_label = c(periods, "15-19"))
  fit

## End(Not run)
```

`print.SUMMERmodel.svy` *Print method for the smoothing models from* `smoothSurvey`.

Description

This function is the print method for class `SUMMERmodel.svy`.

Usage

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

Arguments

`x`	output from `smoothSurvey`.
`...`	not used

Author(s)

Zehang Li

See Also

`summary.SUMMERmodel.svy`

Examples

```
## Not run:
data(DemoData2)
data(DemoMap2)
fit0 <- smoothSurvey(data=DemoData2,
Amat=DemoMap2$Amat, responseType="binary",
responseVar="tobacco.use", strataVar="strata",
weightVar="weights", regionVar="region",
clusterVar = "~clustid+id", CI = 0.95)
fit0

## End(Not run)
```

`print.SUMMERprojlist` *Print method for the combined projection output.*

Description

This function is the print method for class `SUMMERprojlist`.

Usage

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

Arguments

`x`	output from `getSmoothed`
`...`	not used

Author(s)

Zehang Li

Examples

```
## Not run:
 library(SUMMER)
 library(dplyr)
 data(DemoData)
 # Create dataset of counts
 counts.all <- NULL
 for(i in 1:length(DemoData)){
 counts <- getCounts(DemoData[[i]][, c("clustid", "time", "age", "died",
                                       "region", "strata")],
          variables = 'died', by = c("age", "clustid", "region",
                                      "time", "strata"))
 counts <- counts %>% mutate(cluster = clustid, years = time, Y=died)
 counts$strata <- gsub(".*\\.","",counts$strata)
```

```
counts$survey <- names(DemoData)[i]
counts.all <- rbind(counts.all, counts)
}

# fit cluster-level model on the periods
periods <- levels(DemoData[[1]]$time)
fit <- smoothCluster(data = counts.all,
   Amat = DemoMap$Amat,
   time.model = "rw2",
   st.time.model = "rw1",
   strata.time.effect =  TRUE,
   survey.effect = TRUE,
   family = "betabinomial",
   year_label = c(periods, "15-19"))
summary(fit)
est <- getSmoothed(fit, nsim = 1000)

## End(Not run)
```

`ridgePlot` *Calculate and plot posterior densities of the projected estimates*

Description

The function `ridgePlot` replaces the previous function name `getSmoothedDensity` (before version 1.0.0).

Usage

```
ridgePlot(
  x = NULL,
  nsim = 1000,
  draws = NULL,
  year_plot = NULL,
  strata_plot = NULL,
  by.year = TRUE,
  ncol = 4,
  scale = 2,
  per1000 = FALSE,
  order = 0,
  direction = 1,
  results = NULL,
  save.density = FALSE,
  ...
)

getSmoothedDensity(
  x = NULL,
```

```
  nsim = 1000,
  draws = NULL,
  year_plot = NULL,
  strata_plot = NULL,
  by.year = TRUE,
  ncol = 4,
  scale = 2,
  per1000 = FALSE,
  order = 0,
  direction = 1,
  results = NULL,
  save.density = FALSE,
  ...
)
```

Arguments

x	output from smoothDirect for the smoothed direct estimates, or smoothCluster for the cluster-level estimates.
nsim	number of posterior draws to take. Only used for cluster-level models when draws is NULL. Otherwise the posterior draws in draws will be used instead without resampling.
draws	Output of getSmoothed with save.draws set to TRUE. This argument allows the previously sampled draws (by setting save.draws to be TRUE) be used in new aggregation tasks. This argument is only used for cluster-level models.
year_plot	A vector indicate which years to plot
strata_plot	Name of the strata to plot. If not specified, the overall is plotted.
by.year	logical indicator for whether the output uses years as facets.
ncol	number of columns in the output figure.
scale	numerical value controlling the height of the density plots.
per1000	logical indicator to multiply results by 1000.
order	order of regions when by.year is set to TRUE. Negative values indicate regions are ordered from high to low posterior medians from top to bottom. Positive values indicate from low to high. 0 indicate alphabetic orders.
direction	Direction of the color scheme. It can be either 1 (smaller values are darker) or -1 (higher values are darker). Default is set to 1.
results	output from ridgePlot returned object with save.density = TRUE. This argument can be specified to avoid calculating densities again when only the visualization changes.
save.density	Logical indicator of whether the densities will be returned with the ggplot object. If set to TRUE, the output will be a list consisting of (1) a data frame of computed densities and (2) a ggplot object of the plot.
...	additional configurations passed to inla.posterior.sample.

Value

ridge plot of the density, and if `save.density = TRUE`, also a data frame of the calculated densities

Author(s)

Zehang Richard Li

See Also

`plot.SUMMERproj`

Examples

```
## Not run:
years <- levels(DemoData[[1]]$time)

data <- getDirectList(births = DemoData,
years = years,
regionVar = "region", timeVar = "time",
clusterVar = "~clustid+id",
ageVar = "age", weightsVar = "weights",
geo.recode = NULL)
# obtain direct estimates
data_multi <- getDirectList(births = DemoData, years = years,
  regionVar = "region",  timeVar = "time", clusterVar = "~clustid+id",
  ageVar = "age", weightsVar = "weights", geo.recode = NULL)
data <- aggregateSurvey(data_multi)

#  national model
years.all <- c(years, "15-19")
fit1 <- smoothDirect(data = data, geo = NULL, Amat = NULL,
  year_label = years.all, year_range = c(1985, 2019),
  rw = 2, m = 5)
## Plot marginal posterior densities over time
ridgePlot(fit1, year_plot = years.all,
          ncol = 4, by.year = FALSE)

#  subnational model
fit2 <- smoothDirect(data = data, geo = DemoMap$geo, Amat = DemoMap$Amat,
  year_label = years.all, year_range = c(1985, 2019),
  rw = 2, m = 5, type.st = 1)

# Plot marginal posterior densities over time (regions are ordered alphabetically)
ridgePlot(fit2, year_plot = years.all, ncol = 4)

# Re-order the regions and save the density to avoid re-compute later
density <- ridgePlot(fit2, year_plot = years.all,
 ncol = 4, per1000 = TRUE, order = -1, save.density = TRUE)
density$g

# Show each region (instead of each year) in a panel
## Instead of recalculate the posteriors, we can use previously calculated densities as input
```

```
ridgePlot(results = density, year_plot = years.all,
ncol = 4, by.year=FALSE, per1000 = TRUE)

# Show more years
ridgePlot(results = density, year_plot = c(1990:2019),
ncol = 4, by.year=FALSE, per1000 = TRUE)



## End(Not run)
```

`rst` *Simulate spatial and temporal random effects*

Description

This function simulates spatial and temporal random effects with mean zero. The method is described in Algorithm 3.1 of Rue & Held 2015.

Usage

```
rst(
  n = 1,
  type = c("s", "t", "st")[1],
  type.s = "ICAR",
  type.t = c("RW1", "RW2")[2],
  Amat = NULL,
  n.t = NULL,
  scale.model = TRUE
)
```

Arguments

`n`	sample size
`type`	type of random effects: temporal (t), spatial (s), or spatial-temporal (st)
`type.s`	type of spatial random effect, currently only ICAR is available
`type.t`	type of temporal random effect, currently only RW1 and RW2 are available
`Amat`	adjacency matrix for the spatial regions
`n.t`	number of time points for the temporal random effect
`scale.model`	logical indicator of whether to scale the random effects to have unit generalized variance. See Sørbye 2013 for more details

Value

a matrix (for spatial or temporal) or a three-dimensional array (for spatial-temporal) of the random effects.

Author(s)

Zehang Richard Li

References

Rue, H., & Held, L. (2005). *Gaussian Markov random fields: theory and applications*. CRC press.

Sørbye, S. H. (2013). *Tutorial: Scaling IGMRF-models in R-INLA*. Department of Mathematics and Statistics, University of Tromsø.

Examples

```
## Not run:
data(DemoMap)
## Spatial random effects
out <- rst(n=10000, type = "s", Amat = DemoMap$Amat)
# To verify the mean under the conditional specification
mean(out[,1] - apply(out[,c(2,3,4)], 1, mean))
mean(out[,2] - apply(out[,c(1,3)], 1, mean))
mean(out[,3] - apply(out[,c(1,2,4)], 1, mean))
mean(out[,4] - apply(out[,c(1,3)], 1, mean))

## Temporal random effects (RW1)
out <- rst(n=1, type = "t", type.t = "RW1", n.t = 200, scale.model = FALSE)
par(mfrow = c(1,2))
plot(1:dim(out)[2], out, col = 1, type = "l", xlab = "Time", ylab = "Random effects")
# verify the first order difference is normally distributed
first_diff <- diff(as.numeric(out[1,]))
qqnorm(first_diff )
abline(c(0,1))

## Temporal random effects (RW2)
out <- rst(n=1, type = "t", type.t = "RW2", n.t = 200, scale.model = FALSE)
par(mfrow = c(1,2))
plot(1:dim(out)[2], out, col = 1, type = "l", xlab = "Time", ylab = "Random effects")
# verify the second order difference is normally distributed
first_diff <- diff(as.numeric(out[1,]))
second_diff <- diff(first_diff)
qqnorm(second_diff)
abline(c(0,1))

## Spatial-temporal random effects
out <- rst(n=1, type = "st", type.t = "RW2", Amat = DemoMap$Amat, n.t = 50)
dimnames(out)
par(mfrow = c(1,1))
plot(1:dim(out)[3], out[1,1,], col = 1,
 type = "l", ylim = range(out), xlab = "Time", ylab = "Random effects")
for(i in 2:4) lines(1:dim(out)[3], out[1,i,], col = i)
legend("bottomright", colnames(DemoMap$Amat), col = c(1:4), lty = rep(1,4))

## End(Not run)
```

rw.new *New random walk 1 and 2 models for m-year to period random effects*

Description

New random walk 1 and 2 models for m-year to period random effects

Usage

```
rw.new(
  cmd = c("graph", "Q", "mu", "initial", "log.norm.const", "log.prior", "quit"),
  theta = NULL
)
```

Arguments

cmd	list of model components
theta	log precision

rw.new.pc *New random walk 1 and 2 models for m-year to period random effects*

Description

New random walk 1 and 2 models for m-year to period random effects

Usage

```
rw.new.pc(
  cmd = c("graph", "Q", "mu", "initial", "log.norm.const", "log.prior", "quit"),
  theta = NULL
)
```

Arguments

cmd	list of model components
theta	log precision

simhyper *Simulate hyperpriors from an GMRF*

Description

Simulate hyperpriors from an GMRF

Usage

```
simhyper(
  R = 2,
  nsamp = 1e+05,
  nsamp.check = 5000,
  Amat = NULL,
  nperiod = 6,
  only.iid = TRUE
)
```

Arguments

R	Desired prior odds ratio. Default to 2, i.e., a 95% prior interval for the residual odds ratios lies in the interval (R, 1/R).
nsamp	Sample to simulate for scaling factor
nsamp.check	Sample to simulate for checking range
Amat	Adjacency matrix of the areas in the data.
nperiod	numerical value of how many time periods in the data
only.iid	Indicator for whether or not only IID hyperpriors are simulated

Author(s)

Zehang Richard Li, Laina Mercer

References

Wakefield, J. Multi-level modelling, the ecologic fallacy, and hybrid study designs. *International Journal of Epidemiology*, 2009, vol. 38 (pg. 330-336).

Examples

```
## Not run:
data(DemoMap)
mat <- DemoMap$Amat
priors <- simhyper(R = 2, nsamp = 1e+05, nsamp.check = 5000, Amat = mat)

## End(Not run)
```

`smoothCluster` *Cluster-level space-time smoothing models for mortality rates*

Description

The function `smoothCluster` replace the previous function name `fitINLA2` (before version 1.0.0).

Usage

```
smoothCluster(
  data,
  X = NULL,
  family = c("betabinomial", "binomial")[1],
  age.groups = c("0", "1-11", "12-23", "24-35", "36-47", "48-59"),
  age.n = c(1, 11, 12, 12, 12, 12),
  age.rw.group = c(1, 2, 3, 3, 3, 3),
  age.strata.fixed.group = c(1, 2, 3, 4, 5, 6),
  time.model = c("rw1", "rw2", "ar1")[2],
  st.time.model = NULL,
  Amat,
  bias.adj = NULL,
  bias.adj.by = NULL,
  formula = NULL,
  year_label,
  type.st = 4,
  survey.effect = FALSE,
  linear.trend = TRUE,
  common.trend = FALSE,
  strata.time.effect = FALSE,
  hyper = "pc",
  pc.u = 1,
  pc.alpha = 0.01,
  pc.u.phi = 0.5,
  pc.alpha.phi = 2/3,
  pc.u.cor = 0.7,
  pc.alpha.cor = 0.9,
  pc.st.u = NA,
  pc.st.alpha = NA,
  pc.st.slope.u = NA,
  pc.st.slope.alpha = NA,
  overdisp.mean = 0,
  overdisp.prec = 0.4,
  options = list(config = TRUE),
  control.inla = list(strategy = "adaptive", int.strategy = "auto"),
  verbose = FALSE,
  geo = NULL,
  rw = NULL,
```

```
  ar = NULL,
  st.rw = NULL,
  ...
)

fitINLA2(
  data,
  X = NULL,
  family = c("betabinomial", "binomial")[1],
  age.groups = c("0", "1-11", "12-23", "24-35", "36-47", "48-59"),
  age.n = c(1, 11, 12, 12, 12, 12),
  age.rw.group = c(1, 2, 3, 3, 3, 3),
  age.strata.fixed.group = c(1, 2, 3, 4, 5, 6),
  time.model = c("rw1", "rw2", "ar1")[2],
  st.time.model = NULL,
  Amat,
  bias.adj = NULL,
  bias.adj.by = NULL,
  formula = NULL,
  year_label,
  type.st = 4,
  survey.effect = FALSE,
  linear.trend = TRUE,
  common.trend = FALSE,
  strata.time.effect = FALSE,
  hyper = "pc",
  pc.u = 1,
  pc.alpha = 0.01,
  pc.u.phi = 0.5,
  pc.alpha.phi = 2/3,
  pc.u.cor = 0.7,
  pc.alpha.cor = 0.9,
  pc.st.u = NA,
  pc.st.alpha = NA,
  pc.st.slope.u = NA,
  pc.st.slope.alpha = NA,
  overdisp.mean = 0,
  overdisp.prec = 0.4,
  options = list(config = TRUE),
  control.inla = list(strategy = "adaptive", int.strategy = "auto"),
  verbose = FALSE,
  geo = NULL,
  rw = NULL,
  ar = NULL,
  st.rw = NULL,
  ...
)
```

Arguments

`data`	count data of person-months with the following columns • cluster: cluster ID • years: time period • region: region of the cluster • strata: stratum of the cluster • age: age group corresponding to the row • total: total number of person-month in this age group, stratum, cluster, and period • Y: total number of deaths in this age group, stratum, cluster, and period
`X`	Covariate matrix. It must contain either a column with name "region", or a column with name "years", or both. The covariates must not have missing values for all regions (if varying in space) and all time periods (if varying in time). The rest of the columns are treated as covariates in the mean model.
`family`	family of the model. This can be either binomial (with logistic normal prior), betabiniomial.
`age.groups`	a character vector of age groups in increasing order.
`age.n`	number of months in each age groups in the same order.
`age.rw.group`	vector indicating grouping of the ages groups. For example, if each age group is assigned a different random walk component, then set age.rw.group to c(1:length(age.groups)); if all age groups share the same random walk component, then set age.rw.group to a rep(1, length(age.groups)). The default for 6 age groups is c(1,2,3,3,3,3), which assigns a separate random walk to the first two groups and a common random walk for the rest of the age groups. The vector should contain values starting from 1.
`age.strata.fixed.group`	vector indicating grouping of the ages groups for different strata. The default is c(1:length(age.groups)), which correspond to each age group within each stratum receives a separate intercept. If several age groups are specified to be the same value in this vector, the stratum specific deviation from the baseline is assumed to be the same for these age groups. For example, if `age.strata.fixed.group = c(1,2,3,3,3,3)`, then the fixed effect part of the linear predictor consists of 6 overall age-specific intercepts and 3 set of strata effects (where a base stratum is chosen internally), for age groups 1, 2, and the rest respectively. For example, if each age group is assigned a different intercept, then set age.strata.fixed.group to c(1:length(age.groups)); if all age groups share the same intercept, then set age.strata.fixed.group to a rep(1, length(age.groups)). The default for 6 age groups is the former. It can also be set to be the same as `age.rw.group`. The vector should contain values starting from 1.
`time.model`	Model for the main temporal trend, can be rw1, rw2, ar1, or NULL (for spatial-only smoothing). Default to be rw2. For ar1 main effect, a linear slope is also added with time scaled to be between -0.5 to 0.5, i.e., the slope coefficient represents the total change between the first year and the last year in the projection period on the logit scale.

st.time.model — Temporal component model for the interaction term, can be rw1, rw2, or ar1. Default to be the same as time.model unless specified otherwise. The default does not include region-specific random slopes. They can be added to the interaction term by specifying pc.st.slope.u and pc.st.slope.alpha.

Amat — Adjacency matrix for the regions

bias.adj — the ratio of unadjusted mortality rates or age-group-specific hazards to the true rates or hazards. It needs to be a data frame that can be merged to thee outcome, i.e., with the same column names for time periods (for national adjustment), or time periods and region (for subnational adjustment). The column specifying the adjustment ratio should be named "ratio".

bias.adj.by — vector of the column names specifying how to merge the bias adjustment to the count data. For example, if bias adjustment factor is provided in bias.adj for each region and time, then bias.adj.by should be ‘c("region", "time")‘.

formula — INLA formula. See vignette for example of using customized formula.

year_label — string vector of year names

type.st — type for space-time interaction

survey.effect — logical indicator whether to include a survey iid random effect. If this is set to TRUE, there needs to be a column named ’survey’ in the input data frame. In prediction, this random effect term will be set to 0.

linear.trend — logical indicator whether a linear trend is added to the temporal main effect. If the temporal main effect is RW2, then it will be forced to FALSE. Default is TRUE.

common.trend — logical indicator whether all age groups and/or strata share the same linear trend in the temporal main effect.

strata.time.effect — logical indicator whether to include strata specific temporal trends.

hyper — Deprecated. which hyperpriors to use. Only supports PC prior ("pc").

pc.u — hyperparameter U for the PC prior on precisions.

pc.alpha — hyperparameter alpha for the PC prior on precisions.

pc.u.phi — hyperparameter U for the PC prior on the mixture probability phi in BYM2 model.

pc.alpha.phi — hyperparameter alpha for the PC prior on the mixture probability phi in BYM2 model.

pc.u.cor — hyperparameter U for the PC prior on the autocorrelation parameter in the AR prior, i.e. Prob(cor > pc.u.cor) = pc.alpha.cor.

pc.alpha.cor — hyperparameter alpha for the PC prior on the autocorrelation parameter in the AR prior.

pc.st.u — hyperparameter U for the PC prior on precisions for the interaction term.

pc.st.alpha — hyperparameter alpha for the PC prior on precisions for the interaction term.

pc.st.slope.u — hyperparameter U for the PC prior on precisions for the area-level random slope. If both pc.st.slope.u and pc.st.slope.alpha are not NA, an area-level random slope with iid prior will be added to the model. The parameterization of the random

slope is so that Prob(|beta| > pc.st.slope.u) = pc.st.slope.alpha, where time covariate is rescaled to be -0.5 to 0.5, so that the random slope can be interpreted as the total deviation from the main trend from the first year to the last year to be projected, on the logit scale.

`pc.st.slope.alpha`
: hyperparameter alpha for the PC prior on precisions for the area-level random slope. See above for the parameterization.

`overdisp.mean`
: hyperparameter for the betabinomial likelihood. Mean of the over-dispersion parameter on the logit scale.

`overdisp.prec`
: hyperparameter for the betabinomial likelihood. Precision of the over-dispersion parameter on the logit scale.

`options`
: list of options to be passed to control.compute() in the inla() function.

`control.inla`
: list of options to be passed to control.inla() in the inla() function. Default to the "adaptive" integration strategy.

`verbose`
: logical indicator to print out detailed inla() intermediate steps.

`geo`
: Deprecated. Spatial polygon file, legacy parameter from previous versions of the package.

`rw`
: Deprecated. Take values 0, 1 or 2, indicating the order of random walk. If rw = 0, the autoregressive process is used instead of the random walk in the main trend. See the description of the argument ar for details.

`ar`
: Deprecated. Order of the autoregressive component. If ar is specified to be positive integer, the random walk components will be replaced by AR(p) terms in the interaction part. The main temporal trend remains to be random walk of order rw unless rw = 0.

`st.rw`
: Deprecated. Take values 1 or 2, indicating the order of random walk for the interaction term. If not specified, it will take the same order as the argument rw in the main effect. Notice that this argument is only used if ar is set to 0.

`...`
: arguments to be passed to the inla() function call.

Value

INLA model fit using the provided formula, country summary data, and geographic data

Author(s)

Zehang Richard Li

See Also

`getDirect`

Examples

```
## Not run:
library(dplyr)
data(DemoData)
```

```
# Create dataset of counts
counts.all <- NULL
for(i in 1:length(DemoData)){
  counts <- getCounts(DemoData[[i]][, c("clustid", "time", "age", "died",
                                        "region", "strata")],
            variables = 'died', by = c("age", "clustid", "region",
                                       "time", "strata"))
  counts <- counts %>% mutate(cluster = clustid, years = time, Y=died)
  counts$strata <- gsub(".*\\.","",counts$strata)
  counts$survey <- names(DemoData)[i]
  counts.all <- rbind(counts.all, counts)
}

# fit cluster-level model on the periods
periods <- levels(DemoData[[1]]$time)
fit <- smoothCluster(data = counts.all,
     Amat = DemoMap$Amat,
     time.model = "rw2",
     st.time.model = "rw1",
     strata.time.effect =  TRUE,
     survey.effect = TRUE,
     family = "betabinomial",
     year_label = c(periods, "15-19"))
summary(fit)
est <- getSmoothed(fit, nsim = 1000)
plot(est$stratified, plot.CI=TRUE) + ggplot2::facet_wrap(~strata)

# fit cluster-level space-time model with covariate
# notice without projected covariates, we use periods up to 10-14 only
# construct a random covariate matrix for illustration
periods <- levels(DemoData[[1]]$time)
X <- expand.grid(years = periods,
       region = unique(counts.all$region))
X$X1 <- rnorm(dim(X)[1])
X$X2 <- rnorm(dim(X)[1])
fit.covariate <- smoothCluster(data = counts.all,
   X = X,
     Amat = DemoMap$Amat,
     time.model = "rw2",
     st.time.model = "rw1",
     strata.time.effect =  TRUE,
     survey.effect = TRUE,
     family = "betabinomial",
     year_label = c(periods))
est <- getSmoothed(fit.covariate, nsim = 1000)

# fit cluster-level model for one time point only
# i.e., space-only model
fit.sp <- smoothCluster(data = subset(counts.all, time == "10-14"),
     Amat = DemoMap$Amat,
     time.model = NULL,
     survey.effect = TRUE,
     family = "betabinomial")
```

```
summary(fit.sp)
est <- getSmoothed(fit.sp, nsim = 1000)
plot(est$stratified, plot.CI = TRUE) + ggplot2::facet_wrap(~strata)

# fit cluster-level model for one time point and covariate
# construct a random covariate matrix for illustration
X <- data.frame(region = unique(counts.all$region),
      X1 = c(1, 2, 2, 1),
      X2 = c(1, 1, 1, 2))
fit.sp.covariate <- smoothCluster(data = subset(counts.all, time == "10-14"),
     X = X,
     Amat = DemoMap$Amat,
     time.model = NULL,
     survey.effect = TRUE,
     family = "betabinomial")
summary(fit.sp.covariate)
est <- getSmoothed(fit.sp.covariate, nsim = 1000)

## End(Not run)
```

`smoothDirect` *Smoothed direct estimates for mortality rates*

Description

The function `smoothDirect` replaces the previous function name `fitINLA` (before version 1.0.0).

Usage

```
smoothDirect(
  data,
  Amat,
  formula = NULL,
  time.model = c("rw1", "rw2", "ar1")[2],
  st.time.model = NULL,
  year_label,
  year_range = c(1980, 2014),
  is.yearly = TRUE,
  m = 5,
  type.st = 1,
  survey.effect = FALSE,
  hyper = c("pc", "gamma")[1],
  pc.u = 1,
  pc.alpha = 0.01,
  pc.u.phi = 0.5,
  pc.alpha.phi = 2/3,
  pc.u.cor = 0.7,
  pc.alpha.cor = 0.9,
```

```
  pc.st.u = NA,
  pc.st.alpha = NA,
  control.compute = list(dic = TRUE, mlik = TRUE, cpo = TRUE, openmp.strategy =
    "default"),
  control.inla = list(strategy = "adaptive", int.strategy = "auto"),
  verbose = FALSE,
  geo = NULL,
  rw = NULL,
  ar = NULL,
  options = NULL
)

fitINLA(
  data,
  Amat,
  formula = NULL,
  time.model = c("rw1", "rw2", "ar1")[2],
  st.time.model = NULL,
  year_label,
  year_range = c(1980, 2014),
  is.yearly = TRUE,
  m = 5,
  type.st = 1,
  survey.effect = FALSE,
  hyper = c("pc", "gamma")[1],
  pc.u = 1,
  pc.alpha = 0.01,
  pc.u.phi = 0.5,
  pc.alpha.phi = 2/3,
  pc.u.cor = 0.7,
  pc.alpha.cor = 0.9,
  pc.st.u = NA,
  pc.st.alpha = NA,
  control.compute = list(dic = TRUE, mlik = TRUE, cpo = TRUE, openmp.strategy =
    "default"),
  control.inla = list(strategy = "adaptive", int.strategy = "auto"),
  verbose = FALSE,
  geo = NULL,
  rw = NULL,
  ar = NULL,
  options = NULL
)
```

Arguments

data	Combined dataset
Amat	Adjacency matrix for the regions
formula	INLA formula. See vignette for example of using customized formula.

time.model	Model for the main temporal trend, can be rw1, rw2, or ar1. ar1 is not implemented for yearly model with period data input. Default to be rw2. For ar1 main effect, a linear slope is also added with time scaled to be between -0.5 to 0.5, i.e., the slope coefficient represents the total change between the first year and the last year in the projection period on the logit scale.
st.time.model	Temporal component model for the interaction term, can be rw1, rw2, or ar1. ar1 is not implemented for yearly model with period data input. Default to be the same as time.model unless specified otherwise. For ar1 interaction model, region-specific random slopes can be added by specifying pc.st.slope.u and pc.st.slope.alpha.
year_label	string vector of year names
year_range	Entire range of the years (inclusive) defined in year_label.
is.yearly	Logical indicator for fitting yearly or period model.
m	Number of years in each period.
type.st	type for space-time interaction
survey.effect	logical indicator whether to include a survey iid random effect. If this is set to TRUE, there needs to be a column named 'survey' in the input data frame. In prediction, this random effect term will be set to 0. Notice this survey effect is implemented according to the Merter et al. (2015) model, and differently compared to the smoothCluster() function.
hyper	which hyperpriors to use. Default to be using the PC prior ("pc").
pc.u	hyperparameter U for the PC prior on precisions.
pc.alpha	hyperparameter alpha for the PC prior on precisions.
pc.u.phi	hyperparameter U for the PC prior on the mixture probability phi in BYM2 model.
pc.alpha.phi	hyperparameter alpha for the PC prior on the mixture probability phi in BYM2 model.
pc.u.cor	hyperparameter U for the PC prior on the autocorrelation parameter in the AR prior, i.e. Prob(cor > pc.u.cor) = pc.alpha.cor.
pc.alpha.cor	hyperparameter alpha for the PC prior on the autocorrelation parameter in the AR prior.
pc.st.u	hyperparameter U for the PC prior on precisions for the interaction term.
pc.st.alpha	hyperparameter alpha for the PC prior on precisions for the interaction term.
control.compute	list of options to be passed to control.compute() in the inla() function.
control.inla	list of options to be passed to control.inla() in the inla() function. Default to the "adaptive" integration strategy.
verbose	logical indicator to print out detailed inla() intermediate steps.
geo	Deprecated.
rw	Deprecated.
ar	Deprecated.
options	Deprecated.

Value

List of fitted object

Author(s)

Zehang Richard Li

References

Li, Z., Hsiao, Y., Godwin, J., Martin, B. D., Wakefield, J., Clark, S. J., & with support from the United Nations Inter-agency Group for Child Mortality Estimation and its technical advisory group. (2019). *Changes in the spatial distribution of the under-five mortality rate: Small-area analysis of 122 DHS surveys in 262 subregions of 35 countries in Africa.* PloS one, 14(1), e0210645.

Mercer, L. D., Wakefield, J., Pantazis, A., Lutambi, A. M., Masanja, H., & Clark, S. (2015). *Space-time smoothing of complex survey data: small area estimation for child mortality.* The annals of applied statistics, 9(4), 1889.

See Also

getDirect

Examples

```
## Not run:
  years <- levels(DemoData[[1]]$time)
  # obtain direct estimates
  data_multi <- getDirectList(births = DemoData, years = years,
  regionVar = "region",  timeVar = "time", clusterVar = "~clustid+id",
  ageVar = "age", weightsVar = "weights", geo.recode = NULL)
  data <- aggregateSurvey(data_multi)

  #  national model
  years.all <- c(years, "15-19")
  fit1 <- smoothDirect(data = data, Amat = NULL,
  year_label = years.all, year_range = c(1985, 2019),
  time.model = 'rw2', is.yearly=FALSE, m = 5, control.compute = list(config =TRUE))
  out1 <- getSmoothed(fit1)
  plot(out1)

  #  subnational model
  fit2 <- smoothDirect(data = data, Amat = DemoMap$Amat,
  year_label = years.all, year_range = c(1985, 2019),
  time.model = 'rw2',is.yearly=TRUE, m = 5, type.st = 4)
  out2 <- getSmoothed(fit2)
  plot(out2)

  #  subnational space-only model for one period
  fit3 <- smoothDirect(data = subset(data, years == "10-14"),
           time.model = NULL, Amat = DemoMap$Amat)
  source('projINLA.R')
  out3 <- getSmoothed(fit3)
```

```
  plot(out3, plot.CI = TRUE)

## End(Not run)
```

---

smoothSurvey *Fit space-time smoothing models for a generic outcome from complex surveys.*

Description

This function calculates the direct estimates by region and fit a simple spatial smoothing model to the direct estimates adjusting for survey design. Normal or binary variables are currently supported. For binary variables, the logit transformation is performed on the direct estimates of probabilities, and a Gaussian additive model is fitted on the logit scale using INLA.

Usage

```
smoothSurvey(
  data,
  geo = NULL,
  Amat = NULL,
  X = NULL,
  X.unit = NULL,
  responseType = c("binary", "gaussian")[1],
  responseVar,
  strataVar = "strata",
  weightVar = "weights",
  regionVar = "region",
  clusterVar = "~v001+v002",
  pc.u = 1,
  pc.alpha = 0.01,
  pc.u.phi = 0.5,
  pc.alpha.phi = 2/3,
  CI = 0.95,
  formula = NULL,
  timeVar = NULL,
  time.model = c("rw1", "rw2")[1],
  include_time_unstruct = FALSE,
  type.st = 1,
  direct.est = NULL,
  direct.est.var = NULL,
  is.unit.level = FALSE,
  is.agg = FALSE,
  strataVar.within = NULL,
  totalVar = NULL,
  weight.strata = NULL,
  nsim = 1000,
```

```
  save.draws = FALSE,
  ...
)

fitGeneric(
  data,
  geo = NULL,
  Amat = NULL,
  X = NULL,
  X.unit = NULL,
  responseType = c("binary", "gaussian")[1],
  responseVar,
  strataVar = "strata",
  weightVar = "weights",
  regionVar = "region",
  clusterVar = "~v001+v002",
  pc.u = 1,
  pc.alpha = 0.01,
  pc.u.phi = 0.5,
  pc.alpha.phi = 2/3,
  CI = 0.95,
  formula = NULL,
  timeVar = NULL,
  time.model = c("rw1", "rw2")[1],
  include_time_unstruct = FALSE,
  type.st = 1,
  direct.est = NULL,
  direct.est.var = NULL,
  is.unit.level = FALSE,
  is.agg = FALSE,
  strataVar.within = NULL,
  totalVar = NULL,
  weight.strata = NULL,
  nsim = 1000,
  save.draws = FALSE,
  ...
)
```

Arguments

data	The input data frame. The input data with column of the response variable (responseVar), region ID (regionVar), stratification within region (strataVar), and cluster ID (clusterVar). • For area-level model, the data frame consist of survey observations and corresponding survey weights (weightVar). • For unit-level model and is.agg = FALSE, the data frame should consist of aggregated counts by clusters (for binary responses), or any cluster-level response (for continuous response). For binary response (responseType =

'binary'), the beta-binomial model will be fitted for cluster-level counts. For continuous response (responseType = 'gaussian'), a Gaussian smoothing model will be fitted on the cluster-level response.

- For unit-level model and is.agg = TRUE, the data frame should be the same as in the area-level model. For binary response (responseType = 'binary'), the beta-binomial model will be fitted for cluster-level counts aggregated internally. For continuous response (responseType = 'gaussian'), the nested error model will be fitted on unit-level response.

geo	Deprecated argument from early versions.
Amat	Adjacency matrix for the regions. If set to NULL, the IID spatial effect will be used.
X	Areal covariates data frame. One of the column name needs to match the regionVar specified in the function call, in order to be linked to the data input. Currently only supporting time-invariant region-level covariates.
X.unit	Column names of unit-level covariates. When X.unit is specified, a nested error model will be fitted with unit-level IID noise, and area-level predictions are produced by plugging in the covariate specified in the X argument. When X is not specified, the empirical mean of each covariate will be used. This is only implemented for continuous response with the Gaussian likelihood model and unit-level model.
responseType	Type of the response variable, currently supports 'binary' (default with logit link function) or 'gaussian'.
responseVar	the response variable
strataVar	the strata variable used in the area-level model.
weightVar	the weights variable
regionVar	Variable name for region.
clusterVar	Variable name for cluster. For area-level model, this should be a formula for cluster in survey design object, e.g., '~clusterID + householdID'. For unit-level model, this should be the variable name for cluster unit.
pc.u	hyperparameter U for the PC prior on precisions.
pc.alpha	hyperparameter alpha for the PC prior on precisions.
pc.u.phi	hyperparameter U for the PC prior on the mixture probability phi in BYM2 model.
pc.alpha.phi	hyperparameter alpha for the PC prior on the mixture probability phi in BYM2 model.
CI	the desired posterior credible interval to calculate
formula	a string of user-specified random effects model to be used in the INLA call
timeVar	The variable indicating time period. If set to NULL then the temporal model and space-time interaction model are ignored.
time.model	the model for temporal trends and interactions. It can be either "rw1" or "rw2".
include_time_unstruct	Indicator whether to include the temporal unstructured effects (i.e., shocks) in the smoothed estimates from cluster-level model. The argument only applies

	to the unit-level models. Default is FALSE which excludes all unstructured temporal components. If set to TRUE all the unstructured temporal random effects will be included.
type.st	can take values 0 (no interaction), or 1 to 4, corresponding to the type I to IV space-time interaction.
direct.est	data frame of direct estimates, with column names of response and region specified by responseVar, regionVar, and timeVar. When direct.est is specified, it overwrites the data input.
direct.est.var	the column name corresponding to the variance of direct estimates.
is.unit.level	logical indicator of whether unit-level model is fitted instead of area-level model.
is.agg	logical indicator of whether the input is at the aggregated counts by cluster. Only used for unit-level model and binary response variable.
strataVar.within	the variable specifying within region stratification variable. This is only used for the unit-level model.
totalVar	the variable specifying total observations in counts. This is only used for the unit-level model when counts is specified.
weight.strata	a data frame with one column corresponding to regionVar, and columns specifying proportion of each strata for each region. This argument specifies the weights for strata-specific estimates. This is only used for the unit-level model.
nsim	number of posterior draws to take. This is only used for the unit-level model when weight.strata is provided.
save.draws	logical indicator of whether to save the full posterior draws.
...	additional arguments passed to svydesign function.

Details

The function smoothSurvey replaces the previous function name fitGeneric (before version 1.0.0).

Value

HT	Direct estimates
smooth	Smoothed direct estimates
fit	a fitted INLA object
CI	input argument
Amat	input argument
responseType	input argument
formula	INLA formula

Author(s)

Zehang Richard Li

See Also

getDirectList, smoothDirect

Examples

```
## Not run:
##
## 1. Area-level model with binary response
##

data(DemoData2)
data(DemoMap2)
fit0 <- smoothSurvey(data=DemoData2,
Amat=DemoMap2$Amat, responseType="binary",
responseVar="tobacco.use", strataVar="strata",
weightVar="weights", regionVar="region",
clusterVar = "~clustid+id", CI = 0.95)
summary(fit0)

# posterior draws can be returned with save.draws = TRUE
fit0.draws <- smoothSurvey(data=DemoData2,
Amat=DemoMap2$Amat, responseType="binary",
responseVar="tobacco.use", strataVar="strata",
weightVar="weights", regionVar="region",
clusterVar = "~clustid+id", CI = 0.95, save.draws = TRUE)
# notice the posterior draws are on the latent scale
head(fit0.draws$draws.est[, 1:10])

# Example with region-level covariates
 Xmat <- aggregate(age~region, data = DemoData2, FUN = mean)
 fit1 <- smoothSurvey(data=DemoData2,
  Amat=DemoMap2$Amat, responseType="binary",
  X = Xmat,
  responseVar="tobacco.use", strataVar="strata",
  weightVar="weights", regionVar="region",
  clusterVar = "~clustid+id", CI = 0.95)

# Example with using only direct estimates as input instead of the full data
direct <- fit0$HT[, c("region", "HT.est", "HT.var")]
fit2 <- smoothSurvey(data=NULL, direct.est = direct,
                   Amat=DemoMap2$Amat, regionVar="region",
                   responseVar="HT.est", direct.est.var = "HT.var",
                   responseType = "binary")
# Check it is the same as fit0
plot(fit2$smooth$mean, fit0$smooth$mean)

# Example with using only direct estimates as input,
#   and after transformation into a Gaussian smoothing model
# Notice: the output are on the same scale as the input
#   and in this case, the logit estimates.
direct.logit <- fit0$HT[, c("region", "HT.logit.est", "HT.logit.var")]
fit3 <- smoothSurvey(data=NULL, direct.est = direct.logit,
```

```
               Amat=DemoMap2$Amat, regionVar="region",
               responseVar="HT.logit.est", direct.est.var = "HT.logit.var",
               responseType = "gaussian")
# Check it is the same as fit0
plot(fit3$smooth$mean, fit0$smooth$logit.mean)

# Example with non-spatial smoothing using IID random effects
fit4 <- smoothSurvey(data=DemoData2, responseType="binary",
       responseVar="tobacco.use", strataVar="strata",
       weightVar="weights", regionVar="region",
       clusterVar = "~clustid+id", CI = 0.95)

# Using the formula argument, further customizations can be added to the
#  model fitted. For example, we can fit the Fay-Harriot model with
#  IID effect instead of the BYM2 random effect as follows.
#  The "region.struct" and "hyperpc1" are picked to match internal object
#  names. Other object names can be inspected from the source of smoothSurvey.
fit5 <- smoothSurvey(data=DemoData2,
       Amat=DemoMap2$Amat, responseType="binary",
       formula = "f(region.struct, model = 'iid', hyper = hyperpc1)",
       pc.u = 1, pc.alpha = 0.01,
       responseVar="tobacco.use", strataVar="strata",
       weightVar="weights", regionVar="region",
       clusterVar = "~clustid+id", CI = 0.95)
# Check it is the same as fit4, notice the region order may be different
regions <- fit5$smooth$region
plot(fit4$smooth[match(regions, fit4$smooth$region),]$logit.mean, fit5$smooth$logit.mean)

##
## 2. Unit-level model with binary response
##

# For unit-level models, we need to create stratification variable within regions
data <- DemoData2
data$urbanicity <- "rural"
data$urbanicity[grep("urban", data$strata)] <- "urban"

# Beta-binomial likelihood is used in this model
fit6 <- smoothSurvey(data=data,
  Amat=DemoMap2$Amat, responseType="binary",
  X = Xmat, is.unit.level = TRUE,
  responseVar="tobacco.use", strataVar.within = "urbanicity",
  regionVar="region", clusterVar = "clustid", CI = 0.95)

# We may use aggregated PSU-level counts as input as well
#    in the case of modeling a binary outcome
data.agg <- aggregate(tobacco.use~region + urbanicity + clustid,
                      data = data, FUN = sum)
data.agg.total <- aggregate(tobacco.use~region + urbanicity + clustid,
                      data = data, FUN = length)
colnames(data.agg.total)[4] <- "total"
data.agg <- merge(data.agg, data.agg.total)
head(data.agg)
```

```
fit7 <- smoothSurvey(data=data.agg,
  Amat=DemoMap2$Amat, responseType="binary",
  X = Xmat, is.unit.level = TRUE, is.agg = TRUE,
  responseVar = "tobacco.use", strataVar.within = "urbanicity",
  totalVar = "total", regionVar="region", clusterVar = "clustid", CI = 0.95)

# Check it is the same as fit6
plot(fit6$smooth$mean, fit7$smooth$mean)

##
## 3. Area-level model with continuous response
##

# The smoothing model is the same as area-level model with binary response
#  the continuous direct estimates are smoothed instead of
#  their logit-transformed versions for binary response.
fit8 <- smoothSurvey(data=DemoData2, Amat=DemoMap2$Amat,
       responseType="gaussian", responseVar="age", strataVar="strata",
       weightVar="weights", regionVar="region",
       pc.u.phi = 0.5, pc.alpha.phi = 0.5,
       clusterVar = "~clustid+id", CI = 0.95)

##
## 4. Unit-level model with continuous response
##    (or nested error models)

# The unit-level model assumes for each of the i-th unit,
#    Y_{i} ~ intercept + region_effect + IID_i
#    where IID_i is the error term specific to i-th unit

# When more than one level of cluster sampling is carried out,
#   they are ignored here. Only the input unit is considered.
#   So here we do not need to specify clusterVar any more.
fit9 <- smoothSurvey(data= data,
  Amat=DemoMap2$Amat, responseType="gaussian",
  is.unit.level = TRUE, responseVar="age", strataVar.within = NULL,
  regionVar="region", clusterVar = NULL, CI = 0.95)

# To compare, we may also model PSU-level responses. As an illustration,
data.median <- aggregate(age~region + urbanicity + clustid,
                      data = data, FUN = median)

fit10 <- smoothSurvey(data= data.median,
  Amat=DemoMap2$Amat, responseType="gaussian",
  is.unit.level = TRUE, responseVar="age", strataVar.within = NULL,
  regionVar="region", clusterVar = "clustid", CI = 0.95)


# To further incorporate within-area stratification

fit11 <- smoothSurvey(data = data,
  Amat = DemoMap2$Amat, responseType = "gaussian",
```

```
  is.unit.level = TRUE, responseVar="age", strataVar.within = "urbanicity",
  regionVar = "region", clusterVar = NULL, CI = 0.95)

# Notice the usual output is now stratified within each region
# The aggregated estimates require strata proportions for each region
# For illustration, we set strata population proportions below
prop <- data.frame(region = unique(data$region),
                           urban = 0.3,
                           rural = 0.7)
fit12 <- smoothSurvey(data=data,
  Amat=DemoMap2$Amat, responseType="gaussian",
  is.unit.level = TRUE, responseVar="age", strataVar.within = "urbanicity",
  regionVar="region", clusterVar = NULL, CI = 0.95,
  weight.strata = prop)

# aggregated outcome
head(fit12$smooth.overall)

# Compare aggregated outcome with direct aggregating posterior means.
# There could be small differences if only 1000 posterior draws are taken.
est.urb <- subset(fit11$smooth, strata == "urban")
est.rural <- subset(fit11$smooth, strata == "rural")
est.mean.post <- est.urb$mean * 0.3 + est.rural$mean * 0.7
plot(fit12$smooth.overall$mean, est.mean.post)


##
## 6. Unit-level model with continuous response and unit-level covariate
##

# For area-level prediction, area-level covariate mean needs to be
#   specified in X argument. And unit-level covariate names are specified
#   in X.unit argument.

set.seed(1)
sim <- data.frame(region = rep(c(1, 2, 3, 4), 1000),
                  X1 = rnorm(4000), X2 = rnorm(4000))
Xmean <- aggregate(.~region, data = sim, FUN = sum)
sim$Y <- rnorm(4000, mean = sim$X1 + 0.3 * sim$X2 + sim$region)
samp <- sim[sample(1:4000, 20), ]
fit.sim <- smoothSurvey(data=samp ,
                 X.unit = c("X1", "X2"),
                 X = Xmean, Amat=NULL, responseType="gaussian",
                 is.unit.level = TRUE, responseVar="Y", regionVar = "region",
                 pc.u = 1, pc.alpha = 0.01, CI = 0.95)


## End(Not run)
```

`st.new` *New Type I to IV space time interaction models for m-year to period random effects*

Description

New Type I to IV space time interaction models for m-year to period random effects

Usage

```
st.new(
  cmd = c("graph", "Q", "mu", "initial", "log.norm.const", "log.prior", "quit"),
  theta = NULL
)
```

Arguments

`cmd`	list of model components
`theta`	log precision

`st.new.pc` *New Type I to IV space time interaction models for m-year to period random effects*

Description

New Type I to IV space time interaction models for m-year to period random effects

Usage

```
st.new.pc(
  cmd = c("graph", "Q", "mu", "initial", "log.norm.const", "log.prior", "quit"),
  theta = NULL
)
```

Arguments

`cmd`	list of model components
`theta`	log precision

summary.SUMMERmodel *Summary method for the smoothing models.*

Description

This function is the summary method for class SUMMERmodel.

Usage

```
## S3 method for class 'SUMMERmodel'
summary(object, ...)
```

Arguments

object	output from smoothDirect or smoothCluster
...	not used

Author(s)

Zehang Li

See Also

summary.SUMMERmodel

Examples

```
## Not run:
  library(SUMMER)
  library(dplyr)
  data(DemoData)

  # Smooth Direct Model
  years <- levels(DemoData[[1]]$time)
  # obtain direct estimates
  data_multi <- getDirectList(births = DemoData, years = years,
  regionVar = "region",  timeVar = "time", clusterVar = "~clustid+id",
  ageVar = "age", weightsVar = "weights", geo.recode = NULL)
  data <- aggregateSurvey(data_multi)

  years.all <- c(years, "15-19")
  fit <- smoothDirect(data = data, Amat = NULL,
  year_label = years.all, year_range = c(1985, 2019),
  time.model = 'rw2', is.yearly=FALSE, m = 5)
  summary(fit)

  # Cluster-level Model
  counts.all <- NULL
  for(i in 1:length(DemoData)){
```

```
  counts <- getCounts(DemoData[[i]][, c("clustid", "time", "age", "died",
                                        "region", "strata")],
           variables = 'died', by = c("age", "clustid", "region",
                                      "time", "strata"))
  counts <- counts %>% mutate(cluster = clustid, years = time, Y=died)
  counts$strata <- gsub(".*\\.","",counts$strata)
  counts$survey <- names(DemoData)[i]
  counts.all <- rbind(counts.all, counts)
  }

  # fit cluster-level model on the periods
  periods <- levels(DemoData[[1]]$time)
  fit <- smoothCluster(data = counts.all,
     Amat = DemoMap$Amat,
     time.model = "rw2",
     st.time.model = "rw1",
     strata.time.effect =  TRUE,
     survey.effect = TRUE,
     family = "betabinomial",
     year_label = c(periods, "15-19"))
  summary(fit)

## End(Not run)
```

`summary.SUMMERmodel.svy` *Summary method for the smoothing model and output from* `smoothSurvey`.

Description

This function is the summary method for class `SUMMERmodel.svy`.

Usage

```
## S3 method for class 'SUMMERmodel.svy'
summary(object, ...)
```

Arguments

`object`	output from `smoothSurvey`
`...`	not used

Author(s)

Zehang Li

See Also

`summary.SUMMERmodel.svy`

Examples

```
## Not run:
data(DemoData2)
data(DemoMap2)
fit0 <- smoothSurvey(data=DemoData2,
Amat=DemoMap2$Amat, responseType="binary",
responseVar="tobacco.use", strataVar="strata",
weightVar="weights", regionVar="region",
clusterVar = "~clustid+id", CI = 0.95)
summary(fit0)

## End(Not run)
```

`tcpPlot` *Discrete-color maps based on the True Classification Probabilities*

Description

Discrete-color maps based on the True Classification Probabilities

Usage

```
tcpPlot(
  draws,
  geo,
  by.geo = NULL,
  year_plot = NULL,
  ncol = 4,
  per1000 = FALSE,
  thresholds = NULL,
  intervals = 3,
  size.title = 0.7,
  legend.label = NULL,
  border = "gray20",
  size = 0.5
)
```

Arguments

`draws`	a posterior draw object from `getSmoothed`
`geo`	SpatialPolygonsDataFrame object for the map
`by.geo`	variable name specifying region names in geo
`year_plot`	vector of year string vector to be plotted.
`ncol`	number of columns in the output figure.
`per1000`	logical indicator to multiply results by 1000.

thresholds	a vector of thresholds (on the mortality scale) defining the discrete color scale of the maps.
intervals	number of quantile intervals defining the discrete color scale of the maps. Required when thresholds are not specified.
size.title	a numerical value giving the amount by which the plot title should be magnified relative to the default.
legend.label	Label for the color legend.
border	color of the border
size	size of the border

Value

a list of True Classification Probability (TCP) tables, a list of individual spplot maps, and a gridded array of all maps.

Author(s)

Tracy Qi Dong, Zehang Richard Li

References

Tracy Qi Dong, and Jon Wakefield. (2020) *Modeling and presentation of vaccination coverage estimates using data from household surveys.* arXiv preprint arXiv:2004.03127.

Examples

```
## Not run:
library(dplyr)
data(DemoData)
# Create dataset of counts, unstratified
counts.all <- NULL
for(i in 1:length(DemoData)){
  counts <- getCounts(DemoData[[i]][, c("clustid", "time", "age", "died",
                                        "region")],
            variables = 'died', by = c("age", "clustid", "region",
                                       "time"))
  counts <- counts %>% mutate(cluster = clustid, years = time, Y=died)
  counts$strata <- NA
  counts$survey <- names(DemoData)[i]
  counts.all <- rbind(counts.all, counts)
}

# fit cluster-level model on the periods
periods <- levels(DemoData[[1]]$time)
fit <- smoothCluster(data = counts.all,
      Amat = DemoMap$Amat,
      time.model = "rw2",
      st.time.model = "rw1",
      strata.time.effect =  TRUE,
      survey.effect = TRUE,
```

```
      family = "betabinomial",
      year_label = c(periods, "15-19"))
est <- getSmoothed(fit, nsim = 1000, save.draws=TRUE)

tcp <- tcpPlot(est, DemoMap$geo, by.geo = "REGNAME", interval = 3, year_plot = periods)
tcp$g

## End(Not run)
```

Index