

# Package ‘diseasemapping’

February 7, 2024

**Type** Package

**Title** Modelling Spatial Variation in Disease Risk for Areal Data

**Version** 2.0.6

**Date** 2025-02-07

**Depends** R (>= 3.5.0)

**Imports** stats,  
utils,  
terra,  
methods

**Suggests** knitr

**Enhances** mapmisc (>= 1.9.0), INLA, mgcv, XML

**Additional\_repositories** <https://inla.r-inla-download.org/R/testing>

**Description** Formatting of population and case data, calculation of Standardized Incidence Ratios, and fitting the BYM model using 'INLA'. For details see Brown (2015) <[doi:10.18637/jss.v063.i12](https://doi.org/10.18637/jss.v063.i12)>.

**License** GPL

**VignetteBuilder** knitr

## R topics documented:

diseasemapping-package	2
bym-methods	3
cancerRates	5
casedata	6
formatCases	7
formatPopulation-methods	8
getBreaks	9
getRates	10
getSMR-methods	11
getStdRate	13
inla.models	14
kentucky	14
nbToInlaGraph	15
popdata	17
referencepop	17
<b>Index</b>	<b>18</b>

diseasemapping-package

*Disease Mapping*

---

**Description**

Functions for calculating observed and expected counts by region, and manipulating posterior samples from Bayesian models produced by glmmBUGS.

**Author(s)**

Patrick Brown

**Examples**

```
# creating SMR's
data('kentucky')
kentucky = terra::unwrap(kentucky)

kentucky2 = getSMR(kentucky, larynxRates, larynx,
regionCode="County")

if(require('mapmisc', quietly=TRUE)) {
  mycol = colourScale(kentucky2$SMR, breaks=9,
dec=-log10(0.5), style='equal', transform='sqrt')
  plot(kentucky2, col=mycol$plot)
  legendBreaks('topleft', mycol)
}

if(require("INLA", quietly=TRUE)) {
  if(requireNamespace('INLA')) {
    INLA::inla.setOption(num.threads=2)
    # not all versions of INLA support blas.num.threads
    try(INLA::inla.setOption(blas.num.threads=2), silent=TRUE)
  }

  kBYM = bym(observed ~ offset(logExpected) + poverty,
data= kentucky2,
prior = list(sd=c(0.1, 0.5), propSpatial=c(0.5, 0.5))
)

  kBYM$par$summary
}
```

**Description**

Uses inla to fit a Besag, York and Mollie disease mapping model

**Usage**

```
## S4 method for signature 'formula,ANY,ANY,missing'
bym(formula,data,adjMat,region.id,...)
## S4 method for signature 'formula,ANY,missing,missing'
bym(formula,data,adjMat,region.id,...)
## S4 method for signature 'formula,SpatVector,NULL,character'
bym(formula, data, adjMat, region.id, ...)
## S4 method for signature 'formula,SpatVector,missing,character'
bym(formula, data, adjMat, region.id, ...)
## S4 method for signature 'formula,SpatVector,matrix,character'
bym(formula,data,adjMat,region.id,...)
## S4 method for signature 'formula,data.frame,matrix,character'
bym(
  formula,data,adjMat,region.id,
  prior=list(sd=c(0.1,0.5),propSpatial=c(0.5,0.5)),
  family="poisson",formula.fitted=formula,...)
```

**Arguments**

formula	model formula, defaults to intercept-only model suitable for output from <a href="#">getSMR</a> if data is a <code>SpatialPolygonsDataFrame</code> .
data	The observations and covariates for the model, can be output from <a href="#">getSMR</a> .
adjMat	An object of class <code>nb</code> containing the adjacency matrix. If not supplied it will be computed from data, but is required if data is a <code>SpatialPolygonsDataFrame</code>
region.id	Variable in data giving identifiers for the spatial regions. If not supplied, row numbers will be used.
prior	named list of vectors specifying priors, see <a href="#">Details</a>
family	distribution of the observations, defaults to <code>poisson</code>
formula.fitted	formula to use to compute the fitted values, defaults to the model formula but may, for example, exclude individual-level covariates.
...	Additional arguments passed to <a href="#">inla</a> , such as <a href="#">control.inla</a>

**Details**

The Besag, York and Mollie model for Poisson distributed case counts is:

$$Y_i \sim \text{Poisson}(O_i \lambda_i)$$

$$\log(\mu_i) = X_i \beta + U_i$$

$$U_i \sim \text{BYM}(\sigma_1^2, \sigma_2^2)$$

- $Y_i$  is the response variable for region  $i$ , on the left side of the formula argument.
- $O_i$  is the 'baseline' expected count, which is specified in formula on the log scale with  $\log(O_i)$  an offset variable.
- $X_i$  are covariates, on the right side of formula
- $U_i$  is a spatial random effect, with a spatially structured variance parameter  $\sigma_1^2$  and a spatially independent variance  $\sigma_2^2$ .

The prior has elements named `sd` and `propSpatial`, which specifies `model="bym2"` should be used with penalized complexity priors. The `sd` element gives a prior for the marginal standard deviation  $\sigma_0 = \sqrt{\sigma_1^2 + \sigma_2^2}$ . This prior is approximately exponential, and `prior$sd = c(1, 0.01)` specifies a prior probability  $pr(\sigma_0 > 1) = 0.01$ . The `propSpatial` element gives the prior for the ratio  $\phi = \sigma_1/\sigma_0$ . Having `prior$propSpatial = c(0.5, 0.9)` implies  $pr(\phi < 0.5) = 0.9$ .

## Value

A list containing

<code>inla</code>	results from the call to <code>inla</code> . Two additional elements are added: <code>marginals.bym</code> for the marginal distributions of the spatial random effects, and <code>marginals.fitted.bym</code> for the marginals of the fitted values.
<code>data</code>	A data.frame or <code>SpatialPolygonsDataFrame</code> containing posterior means and quantiles of the spatial random effect and fitted values.
<code>parameters</code>	Prior and posterior distributions of the two covariance parameters, and a table summary with posterior quantiles of all model parameters.

## Author(s)

Patrick Brown

## See Also

[getSMR](#)

## Examples

```
data('kentucky')
kentucky = terra::unwrap(kentucky)

# get rid of under 10s
larynxRates = larynxRates[-grep("_{0|5}$",names(larynxRates))]

kentucky = getSMR(kentucky, larynxRates, larynx, regionCode="County")

if(requireNamespace('INLA')) {
  INLA::inla.setOption(num.threads=2)
  # not all versions of INLA support blas.num.threads
  try(INLA::inla.setOption(blas.num.threads=2), silent=TRUE)
}

kBYM <- try(
  bym(
    observed ~ offset(logExpected) + poverty, kentucky,
    prior= list(sd=c(0.1, 0.5), propSpatial=c(0.5, 0.5))
  ), silent=TRUE)
```

```

if(length(grep("parameters", names(kBYM)))) {
  kBYM$parameters$summary
}

if( require("mapmisc", quietly=TRUE) && length(grep("parameters", names(kBYM))) ) {
  thecol = colourScale(kBYM$data$fitted.exp, breaks=5, dec=1, style="equal")
  terra::plot(kBYM$data, col=thecol$plot)
  legendBreaks("topleft", thecol)
}

```

---

cancerRates	<i>Download cancer incidence rates from the International Agency for Research on Cancer (IARC)</i>
-------------	--

---

### Description

Rates by age and sex group are retrieved from <http://ci5.iarc.fr/CI5plus/ci5plus.htm>

### Usage

```
cancerRates(area = "canada", year=2000, sex=c("M", "F"), site="Lung")
```

### Arguments

area	Region to retrieve rates from,
year	year or sequence of years to retrieve data from, within the period 1978 to 2002
site	a vector of cancer sites, see details
sex	"M" or "F" for male or female rates only, c("M", "F") (the default) for both sexes.

### Details

area must be one of Canada, Norway, Latvia, Lithuania, Iceland, Finland, Estonia, Denmark, "Czech Republic", "Costa Rica", USA, Iowa, "New Mexico" or the Canadian provinces of Newfoundland, Prince Edward Island, Nova Scotia, Ontario, Manitoba, Saskatchewan, Alberta, and British Columbia. Alternately an integer specifying a registry code from <http://ci5.iarc.fr>.

site must be one or more of All Sites, Oral Cavity and Pharynx, Oesophagus. Stomach, Colon, Rectum and Anus, Liver, Gallbladder, Pancreas, Larynx, Lung, Bone, Melanoma of skin, Prostate (**Males only**), Testis (**Males only**), Breast (**Females only**), Cervix uteri (**Females only**), Corpus uteri (**Females only**), Ovary and other uterine adnexa (**Females only**), Kidney, Bladder, Eye, Brain and Central Nervous System, Thyroid, Non-Hodgkin Lymphoma, Hodgkin Lymphoma, Multiple myeloma, Leukaemia.

### Value

vector of cancer rates, by age and sex group

**Examples**

```
# won't work if offline or if the iarc web site is down
qcLungF=try(cancerRates(area="canada",
  year=2001:2002, site="lung", sex="F"), silent=TRUE)

if(length(grep("try-error", class(qcLungF)))) {
qcLungF = structure(c(0, 5e-06, 0, 0, 5e-06, 1e-05, 0, 3.4e-05, 9.6e-05,
0.000211, 0.000559, 0.001289, 0.002003, 0.002508, 0.002728, 0.003189,
0.002792, 0.001905), .Names = c("F_0", "F_5", "F_10", "F_15",
"F_20", "F_25", "F_30", "F_35", "F_40", "F_45", "F_50", "F_55",
"F_60", "F_65", "F_70", "F_75", "F_80", "F_85"), site = "Lung",
area = "Canada", year = "2001-2002")
}
qcLungF

data('popdata')
popdata = terra::unwrap(popdata)

qcLungExp = getSMR(popdata, qcLungF)

names(qcLungExp)

if(require('mapmisc', quietly=TRUE)) {

mycol = colourScale(qcLungExp$expected,
breaks=12, dec=0, style='quantile')
plot(popdata[1:400,])
plot(qcLungExp, col=mycol$plot, border='#00000040',add=TRUE)
legendBreaks('topright', mycol)
}
}
```

---

casedata

*Data set contains the number of cases information*


---

**Description**

Cases of Hepatitis Z in Ontario.

**Usage**

```
data(casedata)
```

**Format**

data frame

**Details**

This dataset refers to cases of Hepatitis Z in Ontario for the years 1916 to 1918, giving the number of cases in each census subdivision by age, sex and year. For reasons of privacy, any counts between 1 and 5 have been changed to 1.

## Examples

```
data(casedata)
head(casedata)
table(casedata$cases)
tapply(casedata$cases, casedata$age, sum)

## maybe str(casedata) ; plot(casedata) ...
```

---

formatCases

*Format the disease case data set*

---

## Description

The formatCases function formats the case data set. Changes other formats of age and sex group to three columns: age, ageNumeric and sex.

## Usage

```
formatCases(casedata, ageBreaks = NULL, years = NULL, aggregate.by = NULL)
```

## Arguments

casedata	disease cases data set, usually a data.frame which contains age and sex and number of cases.
ageBreaks	results from getBreaks function.
years	if it contains multiple years, define which years will be included in.
aggregate.by	if want to view the data set from a macro way, could aggregate the data set by age or sex or other variables.

## Details

After using formatCases function, the age columns will change to a "character" column contains ages in cut format, i.e [50,55), denotes age 50. The cut breaks can be found from the breaks of the population data set or defined by user. The original "age" column will be changed to "ageNumeric" columns as factors. The sex column will have two levels "M" and "F" as factors. If "aggregate.by" is not NULL, the number of cases will be summed up by the groups defined in aggregate.by argument.

## Value

formatCases function will return a data frame.

## Author(s)

Patrick Brown

**Examples**

```

data('casedata')
data('popdata')
head(casedata)
caseformat <- formatCases(casedata, ageBreaks = getBreaks(names(popdata)))
head(caseformat)
caseformatagg <- formatCases(casedata, ageBreaks = getBreaks(names(popdata)),
  aggregate.by=c("age", "sex"))
head(caseformatagg)

```

---

formatPopulation-methods

*Format a population data set*


---

**Description**

The formatCases function formats the population data set. Reshape the population data set to "long" format, add in 4 columns : GROUP, POPULATION, sex and age.

**Usage**

```

## S4 method for signature 'data.frame'
formatPopulation(
  popdata, aggregate.by = NULL, breaks = NULL, ...
)
## S4 method for signature 'SpatVector'
formatPopulation(
  popdata, aggregate.by = NULL, breaks = NULL, ...
)
## S4 method for signature 'list'
formatPopulation(
  popdata, aggregate.by = NULL, breaks = NULL,
  years=as.integer(names(popdata)), year.range=NULL,
  time="YEAR",
  personYears=TRUE,...
)

```

**Arguments**

popdata	population data set. It can be a data frame, list, database connection, or spatial polygon data frame
aggregate.by	if want to view the data set from a macro way, could aggregate the data set by age or sex or other variables
breaks	age breaks the user want to use. i.e breaks = c(10, 20, 30 ,40, 60, Inf).
time	the time variable, i.e years
personYears	convert populations to person-years
years	a vector with the year of each dataset
year.range	two dimensional vector with first and last year
...	additional arguments



**Details**

After using the `formatPopulation` function, it will return the population data set in the same class as the original data set. i.e if a spatial polygon data frame has been put into the `formatPopulation` function, it will return a spatial polygon data frame. If `aggregate.by` is not `NULL`, the number of cases will be sum up by the groups define in `aggregate.by`. The "Group" column contains information of sex and age groups, in the format of M.55, denotes male, year 55. The "POPULATION" column is a numeric column, denotes the size of population for the particular age and sex group. The "age" column will be a "character" column contains ages in a cut format. i.e [50,55), denotes age 50. The cut breaks will get from the breaks of population data set or define by user. The sex column will have two levels "M" and "F" as factors.

**Value**

A data frame or spatial object, matching the input.

**Note**

If `breaks` is not specified, the function will aggregate by "age" as default.

**Author(s)**

Patrick Brown

**Examples**

```
data('kentucky')
kentucky = terra::unwrap(kentucky)
head(terra::values(kentucky))
poptry <- formatPopulation(kentucky, breaks = c(seq(0, 80, by=10), Inf))
head(poptry)
poptryagg <- formatPopulation(kentucky,
  breaks = c(seq(0, 80, by=10), Inf),
  aggregate.by=c("sex", "age"))
head(poptryagg)
```

---

getBreaks

*Age Breaks*

---

**Description**

An internal function to return a list contains age breaks, ages in the population data set, sex in the population data set, and age sex groups will be used in the `formatPopulation` function.

**Usage**

```
getBreaks(colNames, breaks = NULL)
```

**Arguments**

<code>colNames</code>	names from the population data set
<code>breaks</code>	the age breaks, i.e <code>breaks = seq(0, 80, by= 10)</code>

**Value**

vector of ages

**Examples**

```
data('kentucky')
ageBreaks = getBreaks(names(kentucky), breaks=c(seq(0, 80, by=10), Inf))
ageBreaks
```

---

getRates	<i>Calculate the estimated coefficients of age and sex group from the glm model</i>
----------	---

---

**Description**

The getRates function calculates the estimated coefficient of the age and sex group from the case and population data set. It fits a glm model with Poisson distribution by default.

**Usage**

```
getRates(casedata, popdata, formula, family = 'poisson', minimumAge = 0,
         maximumAge = 100, S = c("M", "F"), years = NULL, year.range = NULL,
         case.years = grep("^year$", names(casedata), ignore.case = TRUE,
         value = TRUE), fit.numeric=NULL, breaks = NULL)
```

**Arguments**

casedata	A data frame of case data, with columns corresponding to variables in formula. Assumed to be one row per case, unless a column called y or cases or count is included, in which case this column gives the number of cases per row.
popdata	population data set
formula	the glm model you want to fit. ie. ~age*sex
family	the distribution to fit the model
minimumAge	the lower boundary of the age, default is 0
maximumAge	the higher boundary of the age, default is 100
S	vector of sexes to include in the analysis. Defaults to both "M" and "F"
years	a vector of census years
year.range	study period: a vector of two elements, starting dates and ending dates
case.years	variable name in the case data which contains time
fit.numeric	the variables which needed to be changed from factor to numeric
breaks	the age breaks

**Details**

It fits a glm model with Poisson or binomial distribution over case and population data sets. If there is no data set in some age and sex group, an NA will show there.

**Value**

A summary of the glm model contains set of estimated coefficients for different age and sex groups.

**Author(s)**

Patrick Brown

**Examples**

```
data('casedata')
data('popdata')
popdata = terra::unwrap(popdata)
therates = getRates(casedata, popdata, ~sex*age,
breaks=c(seq(0, 80, by=10), Inf))
therates
```

---

getSMR-methods

*Calculate the standardized mortality/morbidity ratios*

---

**Description**

Calculates the rate of observe value over expected value. It will also merge back the observed value, expected value and the ratio back to the population data set.

**Usage**

```
## S4 method for signature 'SpatVector,ANY,ANY,ANY,ANY'
getSMR(
  popdata, model, casedata, regionCode , regionCodeCases ,
  area.scale = 1, sex=c('m','f'),...
)
## S4 method for signature 'list,ANY,ANY,ANY,ANY'
getSMR(
  popdata, model, casedata, regionCode , regionCodeCases ,
  area.scale=1, sex=c('m','f'), ...
)
## S4 method for signature 'data.frame,ANY,missing,missing,missing'
getSMR(
  popdata, model, casedata, regionCode , regionCodeCases ,
  area.scale = 1, sex=c('m','f'),...
)

## S4 method for signature 'data.frame,ANY,data.frame,missing,missing'
getSMR(
  popdata, model, casedata, regionCode , regionCodeCases ,
  area.scale = 1, sex=c('m','f'),...
)
## S4 method for signature 'data.frame,ANY,data.frame,character,missing'
getSMR(
  popdata, model, casedata, regionCode , regionCodeCases ,
```

```

area.scale = 1, sex=c('m','f'),...
)
## S4 method for signature 'data.frame,ANY,missing,character,missing'
getSMR(
popdata, model, casedata, regionCode , regionCodeCases ,
area.scale = 1, sex=c('m','f'),...
)

## S4 method for signature 'data.frame,ANY,data.frame,character,character'
getSMR(
popdata, model, casedata, regionCode , regionCodeCases ,
area.scale = 1, sex=c('m','f'),...
)

```

### Arguments

popdata	the name of population data set
model	rates, either fitted model (usually a <code>glm</code> object), or a vector of rates.
casedata	the name of case data set
regionCode	the name of district area column in population data set
regionCodeCases	the name of district area column in case data set
area.scale	scale the unit of area. e.g $10^6$ : if your spatial coordinates are metres and you want intensity in cases per km <sup>2</sup>
sex	possible subsetting of cases and population, set <code>sex='f'</code> for females only.
...	additional arguments. When <code>popdata</code> is a <code>list</code> , arguments can be <code>personYears</code> (logical, convert rates to person years), <code>years</code> (a vector with the year of each dataset), or <code>year.range</code> (two dimensional vector with first and last year)

### Details

If `model` is numeric, it's assumed to be a vector of rates, with the names of the elements corresponding to columns of the population data set. Names do not need to match exactly (can have M in one set of names, male in another for instance).

Otherwise, `model` is passed to the `predict` function.

### Value

Returns a new population data set contains expected number of cases, observed number of cases and SMR. It has the same format as the population data set which put into the function.

### Examples

```

data(kentucky)
kentucky = terra::unwrap(kentucky)

kentucky2 = getSMR(kentucky, larynxRates, larynx,
regionCode="County")

terra::values(kentucky2)[1:10,grep("^F|^M", names(kentucky2), invert=TRUE)]

```

```

theBreaks = signif(seq(0, max(kentucky2$SMR, na.rm=TRUE), len=9),1)
theCol = heat.colors(length(theBreaks)-1)
terra::plot(kentucky2, col=theCol, breaks = theBreaks)
legend('left', fill=theCol, legend = paste(theBreaks[-length(theBreaks)], ' - ', theBreaks[-1]))

```

---

getStdRate	<i>Calculate the standardized rate</i>
------------	--

---

### Description

A function to calculate the standard rate according to the Canadian standard population data set from year 1991.

### Usage

```
getStdRate(relativeRate, model, referencePopulation, scale = 1e+05)
```

### Arguments

relativeRate	the relative cancer rate calculated by glmmBUGS of different sex and age group of people from ontario .
model	Model to standardize to, either glm model output or a vector of rates by age and sex group
referencePopulation	population to standardize to
scale	compute the expected rate per ‘scale’ number of people.

### Value

numeric value, incidence rate in reference population.

### Author(s)

Lutong Zhou

### Examples

```

data(kentucky)
kentucky = terra::unwrap(kentucky)

kentucky2 = getSMR(kentucky, larynxRates, larynx,
regionCode="County")

data(referencepop)
newpop <- getStdRate(kentucky2$SMR, larynxRates, referencepop, scale=100000)

newpop[1:10]

```

---

`inla.models`*Valid models in INLA*

---

**Description**

calls the function of the same name in INLA

**Usage**

```
inla.models()
```

**Value**

a list

**See Also**

<https://www.r-inla.org>

---

`kentucky`*Larynx cancer cases and population in Kentucky*

---

**Description**

Data set contains the information of population, by age, sex, and census subdivision.

**Usage**

```
data('kentucky')
```

**Format**

A `SpatialPolygonsDataFrame` of Kentucky boundaries and populations, case numbers for each county, and a vector of cancer rates by age and sex group.

**Details**

`larynx` is a `data.frame` of cancer case counts by county, obtained from <http://www.cancer-rates.info> and are for a single deliberately unspecified year.

`kentucky` contains country boundaries and populations.

`kentuckyTract` contains census tract boundaries and populations.

**Examples**

```

library('terra')
data('kentucky')
kentucky = terra::unwrap(kentucky)

head(larynx)
10^5*larynxRates[paste(c("M","F"), 50, sep="_")]

kentucky2 = getSMR(kentucky, larynxRates, larynx,
regionCode="County")

names(kentucky2)
length(kentucky2)

data('kentuckyTract')
kentuckyTract = unwrap(kentuckyTract)

length(kentuckyTract)

if(require('mapmisc', quietly=TRUE)) {
mycol = colourScale(kentucky2$SMR,
breaks=10, dec=-log10(0.5), style='quantile')
plot(kentucky2, col=mycol$plot, border='#00000040')
legendBreaks('topright', mycol)
} else {
terra::plot(kentucky2)
}

breaks = c(0,1,seq(2, ceiling(max(kentucky2$SMR,na.rm=TRUE)),by=2))
thecol = terrain.colors(length(breaks)-1)

plot(kentucky2, col = thecol[cut(kentucky2$SMR,
breaks,include.lowest=TRUE)] )

legend("topleft", pch=15, pt.cex=2.5, adj=c(0,15),
legend=rev(breaks), col=c(NA, rev(thecol)))

```

---

nbToInlaGraph

*Write a graph file for INLA*


---

**Description**

Writes a graph file from an adjacency matrix suitable for use with INLA.

**Usage**

```
nbToInlaGraph(adjMat, graphFile="graph.dat", region.id = attributes(adjMat)$region.id)
```

**Arguments**

adjMat	An object of class nb containing the adjacency matrix.
graphFile	name of file to save adjacency matrix to.
region.id	names of regions

**Details**

This function correctly handles regions which have zero neighbours.

**Value**

A vector of names and indices

**Author(s)**

Patrick Brown

**See Also**

[adjacent](#)

**Examples**

```
data('kentucky')
kentucky = terra::unwrap(kentucky)
# remove all the neighbours Ballard county
kSub = kentucky[-c(2,20,79),]

adjMat = terra::adjacent(kSub)
attributes(adjMat)$region.id = kSub$County

nFile = tempfile()
nbRes = nbToInlaGraph(adjMat, nFile)

# Ballard is region 3
nbRes['Ballard']
# note that Ballard has no neighbours
table(adjMat[, 'from']==3)

cat(readLines(nFile, n=5), sep='\n')

# there will be a warning about zero neighbours
junk = bym(poverty ~ 1, data=kSub, family='gaussian', num.threads=3)
```



---

popdata

*Ontario 2006 population by census subdivision*

---

**Description**

Data set contains the information of population, by age, sex, and census subdivision.

**Usage**

```
data(popdata)
```

**Format**

A SpatialPolygonsDataFrame object, which needs the sp package for full functionality.

**Details**

This data is from the 2006 Census of Canada offering by Statistics Canada web site, [www12.statcan.gc.ca/english/cen](http://www12.statcan.gc.ca/english/cen)

**Examples**

```
data('popdata')
popdata = terra::unwrap(popdata)
head(terra::values(popdata))

terra::plot(popdata)
```

---

referencepop

*Standard Canadian population data set from year 1991.*

---

**Description**

A data set contains population and age sex group from year 1991.

**Usage**

```
data(referencepop)
```

**Format**

Data frame with columns POPULATION, sex, and age for the Canada 1991 population.

**Details**

data frame with rows representing age-sex groups, first column giving proportion of Canada 1991 population in that group, and subsequent columns giving sex and start of age range for each group

**Examples**

```
data(referencepop)
head(referencepop)
sum(referencepop$POPULATION)
```

# Index

\* **Bayesian inference**  
bym-methods, 3

\* **datasets**  
casedata, 6  
kentucky, 14  
popdata, 17  
referencepop, 17

\* **disease mapping**  
bym-methods, 3

adjacent, 16

bym (bym-methods), 3

bym, formula, ANY, ANY, missing-method  
(bym-methods), 3

bym, formula, ANY, missing, missing-method  
(bym-methods), 3

bym, formula, data.frame, matrix, character-method  
(bym-methods), 3

bym, formula, SpatVector, matrix, character-method  
(bym-methods), 3

bym, formula, SpatVector, missing, character-method  
(bym-methods), 3

bym, formula, SpatVector, NULL, character-method  
(bym-methods), 3

bym-methods, 3

cancerRates, 5

casedata, 6

control.inla, 3

diseasemapping

(diseasemapping-package), 2

diseasemapping-package, 2

formatCases, 7

formatPopulation

(formatPopulation-methods), 8

formatPopulation, data.frame-method  
(formatPopulation-methods), 8

formatPopulation, list-method  
(formatPopulation-methods), 8

formatPopulation, SpatVector-method  
(formatPopulation-methods), 8

formatPopulation-methods, 8

getBreaks, 9

getRates, 10

getSMR, 3, 4

getSMR (getSMR-methods), 11

getSMR, data.frame, ANY, data.frame, character, character-method  
(getSMR-methods), 11

getSMR, data.frame, ANY, data.frame, character, missing-method  
(getSMR-methods), 11

getSMR, data.frame, ANY, data.frame, missing, missing-method  
(getSMR-methods), 11

getSMR, data.frame, ANY, missing, character, missing-method  
(getSMR-methods), 11

getSMR, data.frame, ANY, missing, missing, missing-method  
(getSMR-methods), 11

getSMR, list, ANY, ANY, ANY, ANY-method  
(getSMR-methods), 11

getSMR, SpatVector, ANY, ANY, ANY, ANY-method  
(getSMR-methods), 11

getSMR-methods, 11

getStdRate, 13

inla, 3, 4

inla.models, 14

kentucky, 14

kentuckyTract (kentucky), 14

larynx (kentucky), 14

larynxRates (kentucky), 14

nbToInlaGraph, 15

popdata, 17

referencepop, 17