Package 'bayesMig'

May 7, 2026

Version 1.0-0

Date 2024-11-08

Title Bayesian Projection of Migration

Depends R (>= 3.5.0), bayesTFR (>= 7.4.4)

Imports coda, truncnorm, wpp2019, data.table

Description Producing probabilistic projections of net migration rate for all countries of the world or for subnational units using a Bayesian hierarchical model by Azose an Raftery (2015) <doi:10.1007/s13524-015-0415-0>.

License GPL (>= 2)

URL http://bayespop.csss.washington.edu

RoxygenNote 7.3.2

Encoding UTF-8

NeedsCompilation no

Author Jon Azose [aut],
Hana Sevcikova [aut, cre],
Adrian Raftery [aut]

Maintainer Hana Sevcikova <hanas@uw.edu>

Repository CRAN

Date/Publication 2024-11-09 11:20:04 UTC

Contents

`bayesMig-package` *Bayesian Projection of Migration*

Description

Collection of functions for making probabilistic projections of net migration rate for all countries of the world, using a Bayesian hierarchical model (BHM) and the United Nations demographic time series. The model can be also applied to user-defined data for other locations, such as subnational units. Methodological details are provided in Azose & Raftery (2015). The projected rates can be used as input to population projections generated via the **bayesPop** package.

Details

The package is implemented in a similar way as the **bayesTFR** package and thus, many functions have their equivalents in **bayesTFR**. The main functions of the package are:

- `run.mig.mcmc`: Runs a Markov Chain Monte Carlo (MCMC) simulation. It results in posterior samples of the model parameters.
- `mig.predict`: Using the posterior parameter samples, trajectories of future net migration rates are generated for all countries or given locations.

The following functions can be used to analyze results:

- `mig.trajectories.plot`: Shows the posterior trajectories for a given location, including the median and given probability intervals.
- `mig.trajectories.table`: Shows a tabular form of the posterior trajectories for a given location.
- mig.map, mig.ggmap and mig.map.gvis: Show a world map of migration rates for a given projection or observed period, or for country-specific parameter estimates.
- `mig.partraces.plot` and `mig.partraces.cs.plot`: Plot the MCMC traces of country-independent parameters and country-specific parameters, respectively.
- `mig.pardensity.plot` and `mig.pardensity.cs.plot`: Plot the posterior density of the country-independent parameters and country-specific parameters, respectively.

- summary.bayesMig.mcmc.set: Summary method for the MCMC results.
- summary.bayesMig.prediction: Summary method for the prediction results.

For MCMC diagnostics, function mig.coda.list.mcmc creates an object of type "mcmc.list" that can be used with the **coda** package. Furthermore, function mig.diagnose analyzes the MCMCs using the Raftery diagnostics implemented in the **coda** package and gives information about parameters that did not converge.

Existing results can be accessed using the get.mig.mcmc (estimation) and get.mig.prediction (prediction) functions. Existing convergence diagnostics can be accessed using the get.mig.convergence and get.mig.convergence.all functions.

Historical data on migration rates are taken from the **wpp2019** (default), **wpp2024** , **wpp2022** or **wpp2017** package, depending on users settings. Alternatively, users can input their own data. These can be either 5-year or annual time series. An example file with historical annual US migration rates is included in the package. Its usage is shown in the Example of mig.predict.

Note

As this package has been designed for simulating migration on a national level, many functions use arguments and terminology related to countries. However, a "country" is to be interpreted as any location included in the simulation.

Author(s)

Jon Azose, Hana Sevcikova and Adrian Raftery

Maintainer: Hana Sevcikova hanas@uw.edu

References

Azose, J. J., & Raftery, A. E. (2015). Bayesian probabilistic projection of international migration. Demography, 52(5), 1627-1650. doi:10.1007/s1352401504150.

Azose, J.J., Ševčíková, H., Raftery, A.E. (2016): Probabilistic population projections with migration uncertainty. Proceedings of the National Academy of Sciences 113:6460–6465. doi:10.1073/pnas.1606119113.

See Also

Useful links:

- http://bayespop.csss.washington.edu

Examples

```
## Not run:
# Run a real simulation (can take long time)
sim.dir <- tempfile()
m <- run.mig.mcmc(nr.chains = 4, iter = 10000, thin = 10, output.dir = sim.dir,
        verbose.iter = 1000)

# Prediction for all countries
```

```
# (use argument save.as.ascii for passing predictions into bayesPop)
pred <- mig.predict(sim.dir = sim.dir, nr.traj = 1000, burnin = 1000)

# Explore results
summary(pred, country = "Germany")
mig.trajectories.plot(pred, country = "Germany", nr.traj = 50)

# Check convergence diagnostics
mig.diagnose(sim.dir, burnin = 4000, thin = 1)

unlink(sim.dir, recursive = TRUE)
# For annual projections on sub-national level, see ?mig.predict.

## End(Not run)
```

`convert.mig.trajectories` *Converting Trajectories of Migration Rates into ACSII Files*

Description

Converts trajectories of the net migration rates stored in a binary format into two CSV files.

Usage

```
convert.mig.trajectories(
  sim.dir = NULL,
  n = 1000,
  output.dir = NULL,
  verbose = FALSE
)
```

Arguments

`sim.dir`	Directory containing the prediction object. It should be the same as the `output.dir` argument in `mig.predict`.
`n`	Number of trajectories to be stored. It can be either a single number or the word "all" in which case all available trajectories are converted. If the number is smaller than the number of trajectories available in the prediction object, they are selected by equal spacing.
`output.dir`	Directory into which the resulting files will be stored. If it is `NULL`, the same directory is used as for the prediction.
`verbose`	Logical value. Switches log messages on and off.

Details

The function creates two files. First, "ascii_trajectories.csv" is a comma-separated table with the following columns:

"LocID": country code

"Period": prediction interval, e.g. 2015-2020

"Year": middle year of the prediction interval

"Trajectory": identifier of the trajectory

"mig": net migration rate

The second file is called "ascii_trajectories_wide.csv", also a comma-separated table and it contains the same information as above but in a wide format. I.e. the data for one country are ordered in columns, thus, there is one column per country. The country columns are ordered alphabetically.

If the prediction object has been adjusted via any of the adjustment functions, the exported trajectories are also adjusted.

Value

No return value.

Note

This function is automatically called from the `mig.predict` function, therefore in standard cases it will not be needed to call it directly. However, it can be useful for example, if different number of trajectories are to be converted, without having to re-run the prediction, or if the trajectories were adjusted.

See Also

`convert.tfr.trajectories`, `mig.write.projection.summary`, `get.mig.trajectories`

---

get.mig.convergence *Accessing Convergence Diagnostics Object*

Description

The function retrieves results of convergence diagnostics (created by `mig.diagnose`) from disk.

Usage

```
get.mig.convergence(sim.dir, thin = 225, burnin = 10000)

get.mig.convergence.all(sim.dir)
```

Arguments

sim.dir	Simulation directory used for computing the diagnostics.
thin	Thinning interval used with this diagnostics.
burnin	Burn-in used for computing the diagnostics.

Details

Function get.mig.convergence loads an object of class bayesMig.convergence for the specific thin and burnin used in mig.diagnose to generate this object. Function get.mig.convergence.all loads all bayesMig.convergence objects available in sim.dir.

Value

get.mig.convergence returns an object of class bayesMig.convergence;
get.mig.convergence.all returns a list of objects of class bayesMig.convergence.

See Also

mig.diagnose, summary.bayesMig.convergence

Examples

```
# Run a real simulation (can take long time)
sim.dir <- tempfile()
m <- run.mig.mcmc(nr.chains = 2, iter = 10000, thin = 10, output.dir = sim.dir)

# Run convergence diagnostics with different burning and thin
mig.diagnose(sim.dir, burnin = 1000, thin = 2)
mig.diagnose(sim.dir, burnin = 500, thin = 1)

diags <- get.mig.convergence.all(sim.dir)
for(i in 1:length(diags))
    print(summary(diags[[i]]))

unlink(sim.dir, recursive = TRUE)
```

---

get.mig.mcmc *Access MCMC results*

Description

Function get.mig.mcmc retrieves results of an MCMC simulation and creates an object of class bayesMig.mcmc.set. Function has.mig.mcmc checks the existence of such results.

Usage

```
get.mig.mcmc(
  sim.dir,
  chain.ids = NULL,
  low.memory = TRUE,
  burnin = 0,
  verbose = FALSE
)

has.mig.mcmc(sim.dir)
```

Arguments

sim.dir	Directory where simulation results are stored.
chain.ids	Chain identifiers in case only specific chains should be included in the resulting object. By default, all available chains are included.
low.memory	Logical. If FALSE full MCMC traces are loaded into memory.
burnin	Burn-in used for loading traces. Only relevant, if low.memory=FALSE.
verbose	Logical value. Switches log messages on and off.

Value

get.mig.mcmc returns an object of class bayesMig.mcmc.set.

has.mig.mcmc returns a logical value indicating if a migration simulation exists in the given directory.

See Also

run.mig.mcmc

Examples

```
# Toy simulation
sim.dir <- tempfile()
m <- run.mig.mcmc(nr.chains = 1, iter = 10, output.dir = sim.dir)

# can be later accessed via
m <- get.mig.mcmc(sim.dir)
summary(m)

has.mig.mcmc(sim.dir) # should be TRUE

unlink(sim.dir, recursive = TRUE)
```

`get.mig.parameter.traces` *Accessing MCMC Parameter Traces*

Description

Functions for accessing traces of the MCMC parameters, either country-independent or country-specific.

Usage

```
get.mig.parameter.traces(
  mcmc.list,
  par.names = NULL,
  burnin = 0,
  thinning.index = NULL,
  thin = NULL
)

get.mig.parameter.traces.cs(
  mcmc.list,
  country.obj,
  par.names = NULL,
  burnin = 0,
  thinning.index = NULL,
  thin = NULL
)
```

Arguments

`mcmc.list`	List of `bayesMig.mcmc` objects.
`par.names`	Names of country-independent parameters (in case of `get.mig.parameter.traces`) or country-specific parameters (in case of `get.mig.parameter.traces.cs`) to be included. By default all parameters are included, given either by `mig.parameter.names()` (for global parameters) or `mig.parameter.names.cs()` (for location-specific parameters).
`burnin`	Burn-in indicating how many iterations should be removed from the beginning of each chain.
`thinning.index`	Index of the traces for thinning. If it is `NULL`, `thin` is used. `thinning.index` does not include `burnin` and should be flattened over all chains. For example, if there are two MCMC chains of length 1000, `burnin=200` and we want an equidistantly spaced sample of length 400, then the value should be `thinning.index = seq(1, 1600, length = 400)`.
`thin`	An integer value for thinning. It is an alternative to `thinning.index`. The above example is equivalent to `thin=4`.
`country.obj`	Country object (see `get.country.object`).

Value

Both functions return a matrix with columns being the parameters and rows being the MCMC values, attached to one another in case of multiple chains. `get.mig.parameter.traces` returns country-independent parameters, `get.mig.parameter.traces.cs` returns country-specific parameters.

See Also

`mig.coda.list.mcmc` for another way of retrieving parameter traces; `mig.parameter.names` and `mig.parameter.names.cs` for parameter names.

Examples

```
# Toy simulation for US states
us.mig.file <- file.path(find.package("bayesMig"), "extdata", "USmigrates.txt")
sim.dir <- tempfile()
m <- run.mig.mcmc(nr.chains = 2, iter = 30, thin = 1, my.mig.file = us.mig.file,
        output.dir = sim.dir, present.year = 2017, annual = TRUE)
# obtain traces of hierarchical parameters
par.values <- get.mig.parameter.traces(m$mcmc.list, burnin = 5)
dim(par.values) # matrix 50 x 4
hist(par.values[, "mu_global"], main = "mu")

# obtain traces of location-specific traces for California
mig.parameter.names.cs() # allowed parameter names
par.values.cs <- get.mig.parameter.traces.cs(m$mcmc.list,
        country.obj = get.country.object("California", meta = m$meta),
        burnin = 5, par.names = "phi_c")
dim(par.values.cs) # matrix 50 x 1
hist(par.values.cs, main = colnames(par.values.cs))
unlink(sim.dir, recursive = TRUE)
```

---

get.mig.prediction *Access Prediction Object*

Description

Function `get.mig.prediction` retrieves results of a prediction and creates an object of class `bayesMig.prediction`. Function `has.mig.prediction` checks an existence of such results.

Usage

```
get.mig.prediction(mcmc = NULL, sim.dir = NULL, mcmc.dir = NULL)

has.mig.prediction(mcmc = NULL, sim.dir = NULL)
```

Arguments

mcmc	Object of class bayesMig.mcmc.set used to make the prediction. If it is NULL, the prediction is loaded from directory given by sim.dir.
sim.dir	Directory where the prediction is stored.
mcmc.dir	Optional argument to be used only in a special case when the mcmc object contained in the prediction object was estimated in different directory than in the one to which it points to (for example due to moving or renaming the original directory). The argument causes that the mcmc is redirected to the given directory. It can be set to NA if no loading of the mcmc object is desired.

Details

If mcmc is not NULL, the search directory is set to mcmc$meta$output.dir. This approach assumes that the prediction was stored in the same directory as the MCMC simulation, i.e. the output.dir argument of the mig.predict function was set to NULL. If it is not the case, the argument mcmc.dir should be used.

Value

Function get.mig.prediction returns an object of class bayesMig.prediction.

Function has.mig.prediction returns a logical indicating if a prediction exists.

---

get.mig.trajectories *Accessing Trajectories of Net Migration Rate*

Description

Function for accessing all future trajectories of the net migration rate from a prediction object in a form of an array.

Usage

```
get.mig.trajectories(mig.pred, country)
```

Arguments

mig.pred	Object of class bayesMig.prediction.
country	Name or numerical code of a country. It can also be given as ISO-2 or ISO-3 characters.

Details

The function loads projected trajectories of net migration rate for the given country from disk and returns it as a matrix.

Value

Array of size the number of projection periods (including the present year) times the number of trajectories.

See Also

bayesMig.prediction, get.mig.prediction, mig.trajectories.table

---

`mig.coda.list.mcmc` *Conversion to coda-formatted objects*

Description

The functions convert MCMC traces (simulated using `run.mig.mcmc`) into objects that can be used with the **coda** package.

Usage

```
mig.coda.list.mcmc(
  mcmc.list = NULL,
  country = NULL,
  chain.ids = NULL,
  sim.dir = NULL,
  par.names = NULL,
  par.names.cs = NULL,
  low.memory = FALSE,
  ...
)
```

Arguments

`mcmc.list`	A list of objects of class `bayesMig.mcmc`, or an object of class `bayesMig.mcmc.set` or `bayesMig.prediction`. If `NULL`, the MCMCs are loaded from `sim.dir`. Either `mcmc` or `sim.dir` must be given.
`country`	Location name or code. Used in connection with the `par.names.cs` argument (see below).
`chain.ids`	Vector of chain identifiers. By default, all chains available in the `mcmc.list` object are included.
`sim.dir`	Directory with the MCMC simulation results. Only used if `mcmc.list` is `NULL`.
`par.names`	Names of country-independent parameters to be included. Default names are those returned by the `mig.parameter.names` function, which includes all country-independent parameters in the BHM.
`par.names.cs`	Names of country-specific parameters to be included. The argument `country` is used to filter out traces that correspond to a specific location. If `country` is not given, traces of each parameter are given for all countries. Default names are those returned by `mig.parameter.names.cs()`, which includes all country-specific parameters in the BHM.

low.memory	Logical indicating if the function should run in a memory-efficient mode.
...	Additional arguments passed to the **coda**'s mcmc function, such as burnin and thin.

Value

Returns an object of class mcmc.list defined in the **coda** package.

---

mig.diagnose *MCMC convergence diagnostics*

Description

Runs convergence diagnostics of existing migration Markov chains using the raftery.diag function from the coda package.

Usage

```
mig.diagnose(
  sim.dir,
  thin = 80,
  burnin = 2000,
  express = FALSE,
  country.sampling.prop = NULL,
  keep.thin.mcmc = FALSE,
  verbose = TRUE
)

mig.raftery.diag(
  mcmc = NULL,
  sim.dir = NULL,
  burnin = 0,
  country = NULL,
  par.names = NULL,
  par.names.cs = NULL,
  country.sampling.prop = 1,
  verbose = TRUE,
  ...
)

estimate.a.hw(mcmc, burnin = 0, thin = NULL)
```

Arguments

sim.dir	Directory with MCMC simulation results.
thin	Thinning interval.
burnin	Number of iterations to discard from the beginning of the parameter traces.

express	Logical. If TRUE, the convergence diagnostic is run only on the country-independent parameters. If FALSE, the country-specific parameters are included in the diagnostics. The number of countries can be controlled by country.sampling.prop.
country.sampling.prop	Proportion of countries to include in the diagnostics. If it is NULL and express=FALSE, all countries are included. Setting a number between 0 and 1 will determine the proportion of countries to be randomly sampled. For long Markov chains, this argument may significantly influence the runtime of this function.
keep.thin.mcmc	Logical. If TRUE, the thinned traces used for computing the diagnostics are stored on disk.
verbose	Logical value. Switches log messages on and off.
mcmc	A bayesMig.mcmc or bayesMig.mcmc.set object. If not given, the object is loaded from the simulation directory given by sim.dir.
country	Name or code of a country. If it is given, only country-specific parameters parameters of that country are considered.
par.names	Names of country-independent parameters for which the Raftery diagnostics should be computed. By default all parameters are used.
par.names.cs	Names of country-specific parameters for which the Raftery diagnostics should be computed. By default all parameters are used.
...	Additional arguments passed to the mig.coda.list.mcmc function.

Details

The mig.diagnose function invokes the mig.raftery.diag function separately for country-independent parameters and for country-specific parameters. It results in two possible states: red, i.e. it did not converge, and green, i.e. it converged. The resulting object is stored in '{sim.dir}/diagnostics/bayesMig.convergence_ and can be accessed using the function get.mig.convergence.

Function has.mcmc.converged from the **bayesTFR** package can be used to check if the existing diagnostics converged.

For details on the mig.raftery.diag function, see tfr.raftery.diag.

The estimate.a.hw function estimates an optimal value for the a.half.width argument in run.mig.mcmc.

Value

mig.diagnose returns an object of class bayesMig.convergence containing summaries of the convergence check inputs and outputs. It has the same structure as bayesTFR.convergence. In addition it has an element a.hw.est which is the estimated value for the a.half.width argument in run.mig.mcmc.

See Also

tfr.raftery.diag, raftery.diag, get.mig.convergence

Examples

```
# See examples in ?bayesMig and ?get.mig.convergence
```

mig.map *World Map of Net Migration Rate*

Description

Generates a world map of the net migration rate for given quantile and time period, which can be either projection or estimation time period, using different techniques: `mig.map` and `mig.map.all` use **rworldmap**, `mig.ggmap` uses **ggplot2**, and `mig.map.gvis` creates an interactive map via **GoogleVis**. A map of country-specific model parameters is also supported.

Usage

```
mig.map(pred, ...)

mig.ggmap(pred, ...)

mig.map.gvis(pred, ...)

mig.map.all(
  pred,
  output.dir,
  output.type = "png",
  mig.range = NULL,
  nr.cats = 50,
  same.scale = TRUE,
  quantile = 0.5,
  file.prefix = "migwrldmap_",
  ...
)

get.mig.map.parameters(
  pred,
  mig.range = NULL,
  nr.cats = 50,
  same.scale = TRUE,
  quantile = 0.5,
  palette = "Blue-Red",
  ...
)
```

Arguments

`pred`	Object of class `bayesMig.prediction`. Note that location codes must correspond to the UN country codes in order to generate a world map.
`...`	In `mig.map`, ... are all arguments that can be passed to `tfr.map`, such as `quantile`, `year`, `projection.index`, `par.name`, `adjusted`, `device`, `main`, `device.args`, and `data.args`. In `mig.map.gvis`, ... are all arguments that can be passed to

	tfr.map.gvis. In e0.ggmap, ... are arguments that can be passed to tfr.ggmap. In addition, functions that use the **rworldmap** package accept arguments passed to the mapCountryData function of the **rworldmap** package.
output.dir	Directory into which resulting maps are stored.
output.type	Type of the resulting files. It can be "png", "pdf", "jpeg", "bmp", "tiff", or "postscript".
mig.range	Range of the migration rate to be displayed. It is of the form c(*mig.min*, *mig.max*). By default, the whole available range is considered. Note that countries with values outside of the given range will appear white.
nr.cats	Number of color categories.
same.scale	Logical controlling if maps for all years of this prediction object should be on the same color scale.
quantile	Quantile for which the map should be generated. It must be equal to one of the values in dimnames(pred$quantiles)[[2]], i.e. 0, 0.025, 0.05, 0.1, 0.2, 0.25, 0.3, 0.4, 0.5, 0.6, 0.7, 0.75, 0.8, 0.9, 0.95, 0.975, 1. Value 0.5 corresponds to the median.
file.prefix	Prefix for file names.
palette	Color palette to use.

Details

The functions only work for national simulations where location codes correspond to the countries' UN codes.

mig.map creates a single map for the given time period and quantile. mig.map.all generates a sequence of maps, namely one for each projection period. If the package **fields** is installed, a color bar legend at the botom of the map is created.

Function get.mig.map.parameters can be used in combination with mig.map. (Note that get.mig.map.parameters is called from inside of mig.map.all.) It sets breakpoints for the color scheme.

Function mig.ggmap is similar to mig.map, but used the **ggplot2** package in combination with the geom_sf function.

Function mig.map.gvis creates an interactive map using the **googleVis** package and opens it in an internet browser. It also generates a table of the mapped values that can be sorted by columns interactively in the browser.

By default, mig.map, mig.ggmap and mig.map.gvis produce maps of net migration rates. Alternatively, the functions can be used to plot country-specific MCMC parameters into a world map. They are given by the argument par.name. One can pass any value from mig.parameter.names.cs().

Value

get.mig.map.parameters returns a list with elements:

pred The bayesMig.prediction object used in the function.

quantile Value of the argument quantile.

catMethod If the argument same.scale is TRUE, this element contains breakpoints for categorization generated using the quantiles. Otherwise, it is NULL.

numCats Number of categories.

coulourPalette The color palette.

See Also

tfr.map

---

mig.median.set *Adjusting the Projection Medians*

Description

These functions are to be used by expert analysts. They allow to change the projection medians either to specific values, or shift the medians by a given constant or align one projection object with another.

Usage

```
mig.median.set(sim.dir, country, values, years = NULL, ...)

mig.median.shift(
  sim.dir,
  country,
  reset = FALSE,
  shift = 0,
  from = NULL,
  to = NULL
)

mig.median.reset(sim.dir, countries = NULL)

mig.align.predictions(
  sim.dir1,
  sim.dir2,
  country.codes = NULL,
  years = NULL,
  ...
)

mig.shift.prediction.to.wpp(sim.dir, ...)
```

Arguments

sim.dir	Directory containing the prediction object.
country	Name or numerical code of a country.
values	Vector of the new median values.

years	Numeric vector giving years for which to change the median. In mig.median.set it gives years which values correspond to. Ideally it should be of the same length as values. If it is NULL, values are set starting from the first prediction time period. If values correspond to consecutive years, only the first year might be given here. In mig.align.predictions it gives years for which the medians should be aligned.
...	Additional arguments passed to the underlying adjustment functions, such as verbose to show/hide the progress of the adjustment. For mig.shift.prediction.to.wpp it can be stat with values "median" (default) or "mean" to specify which statistics should be adjusted; wpp.year to adjust it to if it differs from the wpp year of the simulation.
reset	Logical. If TRUE medians in a range of from and to are reset to their original values.
shift	Constant by which the medians should be offset. It is not used if reset is TRUE.
from	Year from which the offset/reset should start. By default, it starts at the first prediction period.
to	Year until which the offset/reset should be done. By default, it is set to the last prediction period.
countries	Vector of country names or codes. If this argument is NULL (default), the reset is done for all countries.
sim.dir1	Directory with the bayesMig prediction object to be adjusted.
sim.dir2	Directory with the bayesMig prediction object used to align the medians from sim.dir1 to.
country.codes	Numerical codes of countries to adjust. By default all countries found in sim.dir2 are adjusted in sim.dir1.

Details

The function mig.median.set can be used to set the medians of the given country to specific values.

Function mig.median.shift can be used to offset the medians by a specific constant, or to reset the medians to their original values.

Function mig.median.reset resets medians of the given countries to the original values. By default it deletes adjustments for all countries.

Function mig.align.predictions shifts medians stored in sim.dir1 to match the medians in sim.dir1.

In all cases, if a median is modified, the corresponding offset is stored in the prediction object (element median.shift). All functions write the updated prediction object back to disk. All functions in the package that use trajectories and trajectory statistics use the median.shift values to offset the results correspondingly, i.e. trajectories are shifted the same way as the medians.

Function mig.shift.prediction.to.wpp shifts the projected medians or means (if stat is "mean"), so that they correspond to the values found in the migproj1dt or migproj5dt datasets of the **wpp** package that either corresponds to the package used for the simulation itself or is given by the wpp.year argument. Currently, the function only works for **wpp2024**. Note that regardless if it is an adjustment of the median or mean, the corresponding offset is always converted to a shift of the median.

Value

All functions return an updated object of class `bayesMig.prediction`.

`mig.parameter.names` *Accessing Parameter Names*

Description

Functions for accessing names of the MCMC parameters, either country-independent or country-specific.

Usage

```
mig.parameter.names()

mig.parameter.names.cs(country.code = NULL)
```

Arguments

`country.code`	Location code. If it is given, the country-specific parameter names contain the suffix '_country$X$' where X is the `country.code`.

Value

`mig.parameter.names` returns names of the world parameters.

`mig.parameter.names.cs` returns names of the country-specific parameters.

Examples

```
mig.parameter.names()

mig.parameter.names.cs()
```

`mig.pardensity.plot` *Plotting MCMC Parameter Density*

Description

Functions for plotting the density of the posterior distribution of the MCMC parameters from the migration model.

Usage

```
mig.pardensity.plot(
  mcmc.list = NULL,
  sim.dir = NULL,
  chain.ids = NULL,
  par.names = mig.parameter.names(),
  burnin = NULL,
  dev.ncol = 2,
  low.memory = TRUE,
  ...
)

mig.pardensity.cs.plot(
  country,
  mcmc.list = NULL,
  sim.dir = NULL,
  chain.ids = NULL,
  par.names = mig.parameter.names.cs(),
  burnin = NULL,
  dev.ncol = 3,
  low.memory = TRUE,
  ...
)
```

Arguments

mcmc.list	List of bayesMig.mcmc objects, or an object of class bayesMig.mcmc.set or of class bayesMig.prediction. If it is NULL, the values are loaded from sim.dir.
sim.dir	Directory with the MCMC simulation results. It is only used if mcmc.list is NULL.
chain.ids	List of MCMC identifiers to be plotted. If it is NULL, all chains found in mcmc.list or sim.dir are plotted.
par.names	Names of parameters for which density should be plotted. By default all country-independent parameters are plotted if used within mig.pardensity.plot, or country-specific parameters are plotted if used within mig.pardensity.cs.plot.
burnin	Number of iterations to be discarded from the beginning of each chain before computing the density.
dev.ncol	Number of column for the graphics device. If the number of parameters is smaller than dev.ncol, the number of columns is automatically decreased.
low.memory	Logical indicating if the processing should run in a low-memory mode. If it is FALSE, traces of all available parameters are loaded into memory. Otherwise, parameters are loaded as they are needed.
...	Further arguments passed to the density function.
country	Name or numerical code of a country. It can also be given as ISO-2 or ISO-3 characters.

Details

The functions plot the density of the posterior distribution either for country-independent parameters (`mig.pardensity.plot` or for country-specific parameters (`mig.pardensity.cs.plot`, one graph per parameter. One can restrict it to specific chains by setting the `chain.ids` argument and to specific parameters by setting the `par.names` argument.

If `mcmc.list` is an object of class `bayesMig.prediction` and if this object contains thinned traces, they are used instead of the full chains. In such a case, `burnin` and `chain.ids` cannot be modified - their value is set to the one used when the thinned traces were created, namely when running `mig.predict`. In a situation with long MCMC chains, this approach can significantly speed-up creation of the density plots.

Value

No return value.

`mig.partraces.plot` *Plotting MCMC Parameter Traces*

Description

Functions for plotting the MCMC parameter traces from the migration model.

Usage

```
mig.partraces.plot(
  mcmc.list = NULL,
  sim.dir = NULL,
  chain.ids = NULL,
  par.names = mig.parameter.names(),
  nr.points = NULL,
  dev.ncol = 2,
  ...
)

mig.partraces.cs.plot(
  country,
  mcmc.list = NULL,
  sim.dir = NULL,
  chain.ids = NULL,
  par.names = mig.parameter.names.cs(),
  nr.points = NULL,
  dev.ncol = 3,
  ...
)
```

Arguments

mcmc.list	List of bayesMig.mcmc objects, or an object of class bayesMig.mcmc.set or of class bayesMig.prediction. If it is NULL, the traces are loaded from sim.dir.
sim.dir	Directory with the MCMC simulation results. It is only used if mcmc.list is NULL.
chain.ids	List of MCMC identifiers to be plotted. If it is NULL, all chains found in mcmc.list or sim.dir are plotted.
par.names	Names of parameters for which traces should be plotted. By default all country-independent parameters are plotted if used within mig.partraces.plot, or country-specific parameters are plotted if used within mig.partraces.cs.plot.
nr.points	Number of points to be plotted. If NULL, all points are plotted, otherwise the traces are thinned evenly.
dev.ncol	Number of column for the graphics device. If the number of parameters is smaller than dev.ncol, the number of columns is automatically decreased.
...	Additional graphical parameters.
country	Name or numerical code of a country. It can also be given as ISO-2 or ISO-3 characters.

Details

The functions plot MCMC traces either for country-independent parameters (mig.partraces.plot or for country-specific parameters (mig.partraces.cs.plot, one graph per parameter. One can restrict it to specific chains by setting the chain.ids argument, and to specific parameters by setting the par.names argument.

Value

No return value.

---

mig.predict *Generate posterior trajectories of net migration rates*

Description

Using the posterior parameter samples simulated by run.mig.mcmc, generate posterior trajectories for the net migration rates for all countries of the world, or all locations included in the estimation. This code *does not* adjust trajectories to ensure that net migration counts over all countries sum to zero.

Usage

```
mig.predict(
  mcmc.set = NULL,
  end.year = 2100,
  sim.dir = NULL,
  replace.output = FALSE,
  start.year = NULL,
  nr.traj = NULL,
  thin = NULL,
  burnin = 20000,
  use.cummulative.threshold = FALSE,
  ignore.gcc.in.threshold = FALSE,
  fixed.thresholds = NULL,
  post.last.observed = c("obsdata", "alldata", "impute"),
  save.as.ascii = 0,
  output.dir = NULL,
  seed = NULL,
  verbose = TRUE,
  ...
)
```

Arguments

mcmc.set	Object of class bayesMig.mcmc.set corresponding to sampled parameter values for net migration model. If it is NULL, the object is loaded from the directory specified in sim.dir
end.year	End year of the prediction
sim.dir	Directory with MCMC simulation results. It should be the same as the output.dir argument in run.mig.mcmc
replace.output	Logical value. If TRUE, existing predictions in output.dir will be replaced by results of this run.
start.year	Start year of the prediction, i.e. the first predicted year. By default the prediction is started at the next time period after present.year set in the estimation step. If start.year is smaller than the default, the behavior is controlled by the post.last.observed argument: Either data post start.year is ignored (default) or the projection is set to the available data (post.last.observed = "a").
nr.traj	Number of trajectories to be generated. If NULL, the argument thin is taken to determine the number of trajectories. If both are NULL, the number of trajectories corresponds to the size of the parameter sample.
thin	Thinning interval used for determining the number of trajectories. Only relevant if nr.traj is NULL.
burnin	Number of iterations to be discarded from the beginning of the parameter traces.
use.cummulative.threshold	If TRUE historical cummulative thresholds are applied to avoid sampling rates that are too extreme. The thresholds are derived over prior rates of all locations.

As a time span for deriving the limits on projected rates, at each projected time point, six prior time periods are used in a 5-year simulation, corresponding to 30 years in an annual simulation. In a national simulation, prior rates of GCC countries (plus Western Sahara and Djibouti) are excluded when deriving thresholds for non-GCC countries. If this option is used in a non-country simulation, e.g. in a sub-national settings, set the `ignore.gcc.in.threshold` argument to `TRUE`.

`ignore.gcc.in.threshold`

If `use.cummulative.threshold` is `TRUE`, by default the GCC countries (plus Western Sahara and Djibouti) identified by numerical codes of the countries are excluded from computing the historical cummulative thresholds for non-GCC countries. If this argument is `TRUE`, this distinction is not made. It is important to set it to `TRUE` in a sub-national simulation to avoid any random overlaps of UN codes and user-defined codes.

`fixed.thresholds`

List with optional elements "lower" and "upper". Each of them is a list defining lower and upper bounds of the future migration rate for specific locations. The name of each item is the location code and the value is one number defining the corresponding threshold.

`post.last.observed`

If a user-specific data file was used during estimation and the data contained the "last.observed" column, this argument determines how to treat the time periods between the last observed point and the start year of the prediction, for locations where there is a gap between them, or if short-term predictions were included in the file. It is also relevant if `start.year` is set to a smaller value than `present.year` in the estimation. Possible values are:

- "obsdata" or "o" (default) uses any non-missing observed data provided in the data file during estimation, up to the time point defined by the argument `start.year` (excluding the start year itself).
- "alldata" or "a" would similarly use the provided data but would use all data, even if it goes beyond the start year. This allows to use short-term deterministic projections for locations where it is available.
- "impute" or "i" would ignore all data beyond the last observed data point and impute the missing time periods.

`save.as.ascii` Either a number determining how many trajectories should be converted into an ASCII file, or 'all' in which case all trajectories are converted. It should be set to 0 if no conversion is desired. If this argument is larger than zero, the resulting file can be used as input into population projection via **bayesPop**, see Details.

`output.dir` Directory into which the resulting prediction object and the trajectories are stored. If it is `NULL`, it is set to either `sim.dir`, or to `output.dir` of `mcmc.set$meta` if `mcmc.set` is given.

`seed` Seed of the random number generator. If `NULL` no seed is set. Can be used to generate reproducible projections.

`verbose` Logical value. Switches log messages on and off.

`...` Further arguments passed to the underlying functions.

Details

The trajectories of net migration rates for each location are generated using the model of Azose & Raftery (2015). Parameter samples simulated via `run.mig.mcmc` are used from all chains, from which the given `burnin` was discarded. They are evenly thinned to match `nr.traj` or using the `thin` argument. Such thinned parameter traces, collapsed into one chain, if they do not already exist, are stored on disk into the sub-directory '`thinned_mcmc_t_b`' where t is the value of `thin` and b the value of `burnin`.

The projection is run for all missing values before the present year, if any. Medians over the trajectories are used as imputed values and the trajectories are discarded. The process then continues by projecting the future values where all generated trajectories are kept.

A special case is when the argument `start.year` is given that is smaller than or equal to the present year. In such a case, imputed missing values before present year are treated as ordinary predictions (trajectories are kept). If `post.last.observed` is "a", all historical data between start year and present year are used as projections.

The resulting prediction object is saved into '`{output.dir}/predictions`'. Trajectories for all locations are saved into the same directory in a binary format, one file per location. At the end of the projection, if `save.as.ascii` is larger than 0, the function converts the given number of trajectories into a CSV file, called '`ascii_trajectories.csv`' also located in the '`predictions`' directory. The converted trajectories are selected by equal spacing. In addition to the converted trajectories, two summary files are created: one in a user-friendly format, the other using a UN-specific coding, as described in `mig.write.projection.summary`.

If it is desired to use these predictions as input to population projections in **bayesPop**, enter the full file path of the '`ascii_trajectories.csv`' file into the `inputs` argument of `bayesPop::pop.predict` as item `migtraj` and set the argument `mig.is.rate` appropriately.

Value

Object of class `bayesMig.prediction` which is a list with components containing details of the prediction. Key result component is an array of `quantiles` with dimensions (number of locations) x (number of computed quantiles) x (number of projected time points). First time point in the sequence is not a projection, but the last observed time period.

Other key result components include `traj.mean.sd`, a summary of means and standard deviations for each country at each time point. See `bayesTFR.prediction` for more detail.

References

Azose, J. J., & Raftery, A. E. (2015). Bayesian probabilistic projection of international migration. Demography, 52(5), 1627-1650. doi:10.1007/s1352401504150.

Azose, J.J., Ševčíková, H., Raftery, A.E. (2016): Probabilistic population projections with migration uncertainty. Proceedings of the National Academy of Sciences 113:6460–6465. doi:10.1073/pnas.1606119113.

Examples

```
# Toy simulation for US states
us.mig.file <- file.path(find.package("bayesMig"), "extdata", "USmigrates.txt")
sim.dir <- tempfile()
```

```
m <- run.mig.mcmc(nr.chains = 2, iter = 30, thin = 1, my.mig.file = us.mig.file,
        output.dir = sim.dir, present.year = 2017, annual = TRUE)

# Prediction
pred <- mig.predict(sim.dir = sim.dir, burnin = 5, end.year = 2050)
# here unrealistic results since this is a toy simulation
mig.trajectories.plot(pred, "Hawaii", pi = 80, ylim = c(-0.02, 0.02))
mig.trajectories.table(pred, "Hawaii")
summary(pred, "California")

# view locations included in the simulation
get.countries.table(pred)

unlink(sim.dir, recursive = TRUE)
# For projections on national level, see ?bayesMig.
```

`mig.trajectories.plot` *Output of posterior distribution of migration trajectories*

Description

The functions plot/tabulate the posterior distribution of trajectories of net migration rates for a given location, or for all locations, including their median and given probability intervals.

Usage

```
mig.trajectories.plot(
  mig.pred,
  country,
  pi = c(80, 95),
  nr.traj = 50,
  mark.estimation.points = FALSE,
  adjusted.only = TRUE,
  traj.index = NULL,
  show.mean = FALSE,
  show.median = TRUE,
  xlim = NULL,
  ylim = NULL,
  type = "b",
  xlab = "Year",
  ylab = "Migration rate",
  main = NULL,
  lwd = c(2, 2, 2, 2, 1),
  col = c("black", "green", "red", "red", "#00000020"),
  show.legend = TRUE,
  add = FALSE,
  scale = FALSE,
  ...
```

```
)

mig.trajectories.plot.all(
  mig.pred,
  output.dir = NULL,
  output.type = "png",
  verbose = FALSE,
  ...
)

mig.trajectories.table(mig.pred, country, pi = c(80, 95), ...)
```

Arguments

mig.pred	Prediction object of class bayesMig.prediction.
country	Name or numerical code of a location. If it is a country, it can also be given as ISO-2 or ISO-3 characters.
pi	Probability interval (as percentage) to be included in the output. It can be a single number or a vector.
nr.traj	Number of trajectories to be plotted. If NULL, all trajectories are plotted, otherwise they are thinned evenly.
mark.estimation.points	Logical. If TRUE, points that were not used in the estimation are shown in a lighter color.
adjusted.only	Logical. By default, if the projection median is adjusted using e.g. mig.median.set, the function plots the adjusted median. If this argument is FALSE the original (non-adjusted) median is plotted as well.
traj.index	Vector of trajectory indices to show. If not given, the trajectories are selected using equidistant spacing.
show.mean, show.median	Logical indicating if the mean or/and the median of the distribution should be shown.
xlim, ylim, type, xlab, ylab	Graphical parameters passed to the plot function.
main	Main title for the plot(s). In mig.trajectories.plot.all any occurrence of the string "XXX" is replaced by the name of the appropriate country.
lwd, col	Vector of five elements giving the line width and color for: 1. observed data, 2. imputed values, 3. median, 4. quantiles, 5. trajectories.
show.legend	Logical controlling whether a legend should be drawn.
add	Logical controlling whether the trajectories should be plotted into a new graphic device (FALSE) or into an existing device (TRUE). One can use this argument to plot trajectories from multiple countries into one graphics.
scale	Logical. If TRUE, values are scaled to be "per population", i.e. they are divided by pop.denom passed to run.mig.mcmc.

...	Additional graphical parameters. In addition, for `mig.trajectories.plot.all` any of the arguments of `tfr.trajectories.plot` can be passed here.
`output.dir`	Directory into which resulting plots are written. By default, the plots are saved into directory {sim.dir}/predictions/migTrajectories.
`output.type`	Type of the resulting plot files. Can be "png", "pdf", "jpeg", "bmp", "tiff", or "postscript".
`verbose`	Logical value. Switches log messages on and off.

Details

`mig.trajectories.plot` plots posterior distribution of trajectories of net migration rates for a given location. `mig.trajectories.table` gives the same output as a table. `mig.trajectories.plot.all` creates a set of graphs (one per location) that are stored in `output.dir`.

The median and given probability intervals are computed using all available trajectories. Thus, `nr.traj` does not influence those values - it is used only to control the number of trajectories in the graphs.

Value

No return value.

See Also

`mig.predict`, `summary.bayesMig.prediction`

Examples

```
# See example in ?mig.predict
```

---

mig.write.projection.summary

Writing Projection Summary Files

Description

The function creates two files containing projection summaries, such as the median, the lower and upper bound of the 80 and 90% probability intervals, respectively, and the constant variant. One file is in a user-friendly format, whereas the other is in a UN-specific format with internal coding of the time and the variants.

Usage

```
mig.write.projection.summary(pred, output.dir, ...)
```

Arguments

pred	Object of class bayesMig.prediction.
output.dir	Directory where output is written.
...	Additional arguments passed to the underlying functions. Here, argument precision can be set to determine the number of significant digits (default is 4).

Value

No return value.

See Also

write.projection.summary

---

run.mig.mcmc *Run Markov chain Monte Carlo for parameters of net migration rate model*

Description

Runs MCMCs for simulating the net migration rate of all countries of the world or for locations specified by users, using the Bayesian hierarchical model of Azose & Raftery (2015).

Usage

```
run.mig.mcmc(
  output.dir,
  nr.chains = 3,
  iter = 50000,
  thin = 1,
  replace.output = FALSE,
  annual = FALSE,
  start.year = 1950,
  present.year = 2020,
  wpp.year = 2019,
  my.mig.file = NULL,
  sigma.c.min = 1e-04,
  a.ini = NULL,
  a.half.width = NULL,
  mu.ini = NULL,
  exclude.from.world = NULL,
  pop.denom = 1,
  seed = NULL,
  parallel = FALSE,
  nr.nodes = nr.chains,
  buffer.size = 1000,
```

```
  verbose = TRUE,
  verbose.iter = 10,
  ...
)
```

Arguments

output.dir	A file path pointing to the directory in which to store results.
nr.chains	An integer number of independent Markov chains to run.
iter	The number of iterations to run per Markov chain.
thin	Thinning interval. A chain with 1000 iterations thinned by 20 will return a final count of 50 iterations.
replace.output	If the specified output directory already exists, should it be overwritten?
annual	If TRUE, the model assumes the underlying data is on annual time scale.
start.year	Start year for using historical data.
present.year	End year for using historical data.
wpp.year	Year for which WPP data is used if no user data is provided via my.mig.file. In such a case, the function loads a package called **wpp**x where x is the wpp.year and generates historical migration rates using the migration and pop datasets.
my.mig.file	File name containing user-specified historical time series of migration rates for all locations that should be included in the simulation. It should be a tab-separated file. For structure, see Details below.
sigma.c.min, a.ini, mu.ini	Settings for the parameters of the model (see Azose & Raftery 2015), such as minimum value and initial values. Initial values (*.ini) can be given as a vector of length nr.chains, giving one initial value per chain. By default the initial values are equidistantly spread between their respective ranges.
a.half.width	Half width for Metropolis proposals of the a parameter. This argument can greatly influence the convergence and it is dependent on the scale of the data. By default it is set to 0.01 for 5-year data defined as rate per population; to 0.03 for 5-year data defined as per 1000; to 0.3 for annual data per population; to 0.5 for annual data per 1000. If the default does not yield satisfactory results, use the function estimate.a.hw to estimate an appropriate value, based on an existing simulation. Also it is important to set the pop.denom argument correctly.
exclude.from.world	Vector of location codes that should be excluded from estimating the hyperparameters. These would be for example small locations or locations with unusual patters. Note that location-specific parameters are generated for all locations, regardless of this setting.
pop.denom	Denominator used to generate the input migration rates. It is used to derive an appropriate scaler for the priors and conditional distributions. Typically, this will be either 1 (default) if the rates are defined as per population, or 1000, if the rates are per 1000 population. Use this argument only if user-specified rates are supplied via the my.mig.file argument.

seed	Seed of the random number generator. If NULL no seed is set. It can be used to generate reproducible results.
parallel	Whether to run code in parallel.
nr.nodes	Relevant only if parallel is TRUE. It gives the number of nodes for running the simulation in parallel. By default it equals to the number of chains.
buffer.size	Buffer size (in number of iterations) for keeping data in the memory before flushing to disk.
verbose	Whether or not to print status updates to console window while the code is running.
verbose.iter	If verbose is TRUE, the number of iterations to wait between printing updates.
...	Additional parameters to be passed to the function performParallel, if parallel is TRUE.

Details

The function creates an object of class bayesMig.mcmc.meta and stores it in output.dir. It launches nr.chains MCMCs, either sequentially or in parallel. Parameter traces of each chain are stored as ASCII files in a subdirectory of output.dir, called mc$x$ where x is the identifier of that chain. There is one file per parameter, named after the parameter with the suffix ".txt". Location-specific parameters have the suffix _country$c$ where c is the location code. In addition to the trace files, each mc$x$ directory contains the object bayesMig.mcmc in binary format. All chain-specific files are written onto disk after the first, last and each i-th (thinned) iteration, where i is given by the argument buffer.size.

By default (if no data is passed via the my.mig.file argument), the function loads observed data (further denoted as WPP dataset), from the migration and pop datasets in the **wpp**x package where x is the wpp.year. Net migration rates are computed as migration(t) / (population(t_e) - migration(t)) where t_e means the end of time period t. For an annual simulation and wpp.year set to 2022, $t = t_e$ because the population in year t is considered at the end of the year. If wpp.year is smaller than 2022 and annual is TRUE the default dataset is interpolated from 5-year data.

The argument my.mig.file can be used to overwrite the default data. It should be a tab-separated file. If it is used, it should contain net migration rates for all locations to be used in the simulation, as no WPP data is used in such a case. The structure of the file has the same format as the migration dataset, but the values should be rates (instead of counts). Use the argument pop.denom to define the scale of the denominator in these rates, i.e. if the rates are to be interpreted as per population (default) or some other scale. Each row in the my.mig.file file corresponds to a location. It does not have to be necessarily a country - it can be for example a subnational unit. It must contain columns "country_code" or "code" (unique identifier of the location), "name", and columns representing 5-year time intervals (if annual is FALSE), e.g., "1995-2000", "2000-2005" etc., or single years (if annual is TRUE). An example dataset of annual net migration rates for US states is included in the package, see example below.

Optionally, the my.mig.file can contain columns called "first.observed" and/or "last.observed", containing for each location the year of the first and last observation, respectively. In such a case, any data before and after those time points will be ignored. Furthermore, the function mig.predict fills in the missing values after the last observation, using the median of the BHM procedure.

If there are countries or locations that should be excluded from influencing the hyperparameters, for example small countries or locations with unique migration patterns, their codes should be included

in the argument exclude.from.world. These locations will still get their parameters simulated and thus, will be included in a projection. Alternatively if my.mig.file is used, these locations can be determined using an additional column, called "include_code". Value 2 means the location is included in the BHM; value 1 means it's excluded but location-specific parameters are generated; value 0 means the location is ignored.

Value

An object of class bayesMig.mcmc.set which is a list with two components:

meta	An object of class bayesMig.mcmc.meta. It contains information that is common to all chains. Most items are the same as in bayesTFR.mcmc.meta. In addition, mig.rates is a matrix of the observed migration rates with NAs in spots that were not used for estimation. mig.rates.all is a similar matrix but contains all data, regardless if used for estimation or not. Item user.data is a logical indicating if the migration rates are given by the user (TRUE) or are taken from a **wpp** package (FALSE).
mcmc.list	A list of objects of class bayesMig.mcmc, one for each MCMC. Information stored here is specific to each MCMC chain, similarly to bayesTFR.mcmc.

References

Azose, J. J., & Raftery, A. E. (2015). Bayesian probabilistic projection of international migration. Demography, 52(5), 1627-1650.

See Also

get.mig.mcmc, summary.bayesMig.mcmc.set, mig.partraces.plot, mig.pardensity.plot, mig.predict

Examples

```
# Toy simulation for US states
us.mig.file <- file.path(find.package("bayesMig"), "extdata", "USmigrates.txt")
sim.dir <- tempfile()
m <- run.mig.mcmc(nr.chains = 3, iter = 100, thin = 1, my.mig.file = us.mig.file,
        annual = TRUE, output.dir = sim.dir)
summary(m)
summary(m, "Washington")

mig.partraces.plot(m)
mig.partraces.cs.plot("California", m)

# later one can access the object from disk
m <- get.mig.mcmc(sim.dir)

unlink(sim.dir, recursive = TRUE)
# For a country-level simulation, see example in ?bayesMig.
```

summary.bayesMig.convergence *Summary of Convergence Diagnostics*

Description

Summary of an object of class bayesMig.convergence created using the mig.diagnose function. It gives an overview about parameters that did not converge.

Usage

```
## S3 method for class 'bayesMig.convergence'
summary(object, expand = FALSE, ...)
```

Arguments

object	Object of class bayesMig.prediction.
expand	By default, the function does not show country-specific parameters for which there was no convergence (only country-independent parameters), if the status is 'red'. This argument can switch that option on.
...	Not used.

Value

List with items that summarize an object of class bayesMig.convergence.

summary.bayesMig.mcmc *Summary Statistics for Migration Markov Chain Monte Carlo*

Description

Summary of an object bayesMig.mcmc.set or bayesMig.mcmc, computed via run.mig.mcmc. It can be obtained either for all locations or for a specific location, and either for all parameters or for specific parameters. The function uses the summary.mcmc function of the **coda** package.

Usage

```
## S3 method for class 'bayesMig.mcmc'
summary(
  object,
  country = NULL,
  par.names = NULL,
  par.names.cs = NULL,
  thin = 1,
  burnin = 0,
```

```
  ...
)

## S3 method for class 'bayesMig.mcmc.set'
summary(
  object,
  country = NULL,
  chain.id = NULL,
  par.names = NULL,
  par.names.cs = NULL,
  meta.only = FALSE,
  thin = 1,
  burnin = 0,
  ...
)
```

Arguments

object	Object of class bayesMig.mcmc.set or bayesMig.mcmc.
country	Location name or code if a location-specific summary is desired. The code can be either numeric or (if locations are countries) ISO-2 or ISO-3 characters. By default, summary for all locations is generated.
par.names	Country independent parameters (hyperparameters) to be included in the summary. The default names are given by mig.parameter.names().
par.names.cs	Location-specific parameters to be included in the summary. The default names are given by mig.parameter.names.cs().
thin	Thinning interval. Only used if larger than the thin argument used in run.mig.mcmc.
burnin	Number of iterations to be discarded from the beginning of each chain before computing the summary.
...	Additional arguments passed to the summary.mcmc function of the **coda** package.
chain.id	Identifiers of MCMC chains. By default, all chains are considered.
meta.only	Logical. If it is TRUE, only meta information of the simulation is included.

Value

Return list with elements:

meta contains meta information about the object.

results contains result of summary.mcmc.

country.name optional; available if country is provided as argument.

Examples

```
# See example in ?run.mig.mcmc
```

summary.bayesMig.prediction

Summary of Prediction of Net Migration Rate

Description

Summary of an object of class bayesMig.prediction, created using the function mig.predict. The summary contains the mean, standard deviation and several commonly used quantiles of the simulated trajectories.

Usage

```
## S3 method for class 'bayesMig.prediction'
summary(object, country = NULL, compact = TRUE, ...)

## S3 method for class 'summary.bayesMig.prediction'
print(x, digits = 3, ...)
```

Arguments

object	Object of class bayesMig.prediction.
country	Location name or code if a location-specific summary is desired. The code can be either numeric or (if locations are countries) ISO-2 or ISO-3 characters. If it is NULL, only prediction meta info is included.
compact	Logical switching between a smaller and larger number of displayed quantiles.
...	A list of further arguments.
x	A result of the summary function.
digits	Minimal number of significant digits.

Value

summary returns a list with objects burnin, nr.traj, projection.years, country.name containing the MCMC burn-in, number of trajectories, projected years and name of the location, respectively. The projection results are stored in the item projections which is a matrix with rows being the years and columns being the mean and various quantiles.

Examples

```
# See example in ?mig.predict
```

Index