Package ‘bayesLife’

October 14, 2017

Type Package
Title Bayesian Projection of Life Expectancy
Version 3.2-0
Date 2017-10-13
Author Hana Sevcikova, Adrian Raftery, Jennifer Chunn
Maintainer Hana Sevcikova <hanas@uw.edu>
Description Making probabilistic projections of life expectancy for all countries of the world, using a Bayesian hierarchical model <doi:10.1007/s13524-012-0193-x>.
Depends bayesTFR (>= 5.0-4)
Imports wpp2017, hett, car, coda
Suggests wpp2015, wpp2012, wpp2010
License GPL (>= 2)
URL https://bayespop.csss.washington.edu
NeedsCompilation yes
Repository CRAN
Date/Publication 2017-10-14 07:07:47 UTC

R topics documented:

bayesLife-package ......................................................... 2
bayesLife.mcmc ........................................................... 4
bayesLife.mcmc.meta ..................................................... 6
convert.e0.trajectories ............................................... 7
e0.coda.list.mcmc ....................................................... 9
e0.diagnose ............................................................... 10
e0.dl.coverage ........................................................... 12
e0.DLcurve.plot ....................................................... 13
e0.gap.plot .............................................................. 15
e0.jmale.estimate ....................................................... 16
e0.jmale.predict ........................................................ 19
e0.joint.plot ............................................................ 21
Description

Collection of functions for making probabilistic projections of the life expectancy for all countries of the world, using a Bayesian hierarchical model and the United Nations demographic time series.

Details

Package: bayesLife
Type: Package
Version: 3.2-0
Date: 2017-10-13
License: GPL (>= 2)
URL: https://bayespop.csss.washington.edu

The projection follows a method developed by Chunn et al (2010). It uses historical data provided by the United Nations to simulate a posterior distribution of the life expectancy for all countries in the world simultaneously.

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 `bayesLife` package are:

- **run.e0.mcmc**: Runs a Markov Chain Monte Carlo (MCMC) simulation for one or more chains, possibly in parallel. It results in a posterior sample of the MCMC parameters. Existing simulation runs can be resumed using `continue.e0.mcmc`.
- **e0.predict**: Using the posterior parameter samples it derives posterior trajectories of the life expectancy for all countries.
- **e0.jmale.predict**: Given existing outputs of `e0.predict` for female life expectancy, this function estimates and predicts a joint male life expectancy as described in Raftery et al (2014).

A number of functions analyzing results are included in the package:

- **e0.trajectories.plot**: Shows the posterior trajectories for a given country, including their median and given probability intervals.
- **e0.trajectories.table**: Shows the posterior trajectories for a given country in a tabular form.
- **e0.map**: Shows a world map of life expectancy for a given projection period.
- **e0.DLcurve.plot**: Shows the posterior curves of the double logistic function used in the simulation, including their median and given probability intervals.
- **e0.partraces.plot** and **e0.partraces.cs.plot**: Plot the MCMC traces of country-independent parameters and country-specific parameters, respectively.
- **e0.pardensity.plot** and **e0.pardensity.cs.plot**: Plot the posterior density of the MCMCs for country-independent parameters and country-specific parameters, respectively.
- **summary.bayesLife.mcmc.set**: Summary function for the MCMC results.
- **summary.bayesLife.prediction**: Summary function for the prediction results.

For MCMC diagnostics, function `e0.coda.list.mcmc` creates an object of type “mcmc.list” that can be used with the `coda` package. Furthermore, function `e0.diagnose` analyzes the MCMCs using the Raftery diagnostics implemented in the `coda` package and gives information about parameters that did not converge. Function `e0.dl.coverage` computes a goodness of fit of the double logistic function.

Existing simulation results can be accessed using the `get.e0.mcmc` function. An existing prediction can be accessed via `get.e0.prediction`.

Historical data are taken from one of the packages `wpp2017` (default), `wpp2015`, `wpp2012` or `wpp2010`, depending on users settings.

**Note**

There is a directory `exMdata` shipped with the package which contains results from an example simulation, containing one chain with 60 iterations. The Example section below shows how these results were created. These data are used in Example sections throughout the manual. The user can either reproduce the data in her/his local directory, or use the ones from the package.

**Author(s)**

Hana Sevcikova, Adrian Raftery, Jennifer Chunn

Maintainer: Hana Sevcikova <hanas@uw.edu>
References


See Also

bayesTFR

Examples

```r
## Not run:
sim.dir <- tempfile()
m <- run.e0.mcmc(sex='F', nr.chains=1, iter=60, seed=1, thin=1, output.dir=sim.dir, verbose=TRUE)
pred <- e0.predict(m, burnin=30, verbose=TRUE)
summary(pred, country="Canada")
unlink(sim.dir, recursive=TRUE)
## End(Not run)
```

bayesLife.mcmc  

**MCMC Simulation Object**

Description

MCMC simulation object bayesLife.mcmc containing information about one MCMC chain. A set of such objects belonging to the same simulation together with a bayesLife.mcmc.meta object constitute a bayesLife.mcmc.set object.

Details

An object bayesLife.mcmc points to a place on disk (element output.dir) where MCMC results from all iterations are stored. They can be retrieved to the memory using get.e0.mcmc(...).

The object is in standard cases not to be manipulated by itself, but rather as part of a bayesLife.mcmc.set object.
Value

A `bayesLife.mcmc` object contains parameters of the Bayesian hierarchical model, more specifically, their initial values (all names with the suffix `.ini`) and values from the last iteration. These are:

Triangle/Triangle.ini, lambda/lambda.ini - world parameters, containing four values each. They correspond to model parameters $\Delta_1, \ldots, \Delta_4$ and $\lambda_1, \ldots, \lambda_4$, respectively.
k/k.ini, z/z.ini, omega/omega.ini, lambda.k/lambda.k.ini, lambda.z/lambda.z.ini - world parameters, containing one value each. They correspond to model parameters $k, z, \omega, \lambda_k$, and $\lambda_z$, respectively.

Triangle.c - country-specific parameter $\Delta^c_1, \ldots, \Delta^c_4$ with four values for each country, i.e. an $4 \times C$ matrix where $C$ is the number of countries.
k.c, z.c - country-specific parameters $k^c$ and $z^c$ (1d arrays of length $C$).

Furthermore, the object contains components:

- `iter` Total number of iterations the simulation was started with.
- `finished.iter` Number of iterations that were finished. Results from the last finished iteration are stored in the parameters above.
- `length` Length of the MCMC stored on disk. It differs from `finished.iter` only if `thin` is larger than one.
- `thin` Thinning interval used when simulating the MCMCs.
- `id` Identifier of this chain.
- `output.dir` Subdirectory (relative to `output.dir` in the `bayesLife.mcmc.meta` object) where results of this chain are stored.
- `traces` This is a placeholder for keeping whole parameter traces in the memory. If the processing operates in a low memory mode, it will be 0. It can be filled in using the function `get.e0.mcmc(...)`, `1ow.memory=FALSE`). In such a case, traces is a $I \times J$ array where $I$ is the MCMC length and $J$ is the number of parameters.
- `traces.burnin` Burnin used to retrieve the traces, i.e. how many stored iterations are missing from the beginning in the traces array comparing to the ‘raw’ traces on the disk.
- `rng.state` State of the random number generator at the end of the last finished iteration.
- `meta` Object of class `bayesLife.mcmc.meta` used for simulation of this chain.

Author(s)

Hana Sevcikova

See Also

`run.e0.mcmc`, `get.e0.mcmc`, `bayesLife.mcmc.set`, `bayesLife.mcmc.meta`

Examples

```r
sim.dir <- file.path(find.package("bayesLife"), "ex-data", "bayesLife.output")
# loads traces from one chain
m <- get.e0.mcmc(sim.dir, low.memory=FALSE, burnin=35, chain.ids=1)
```
# should have 13 rows, since 60 iterations in total minus 35 burnin with thin=2
dim(e0.mcmc(m, 1)$traces)
summary(m)

bayesLife.mcmc.meta  MCMC Simulation Meta Object

Description
Simulation meta object bayesLife.mcmc.meta used by all chains of the same MCMC simulation. It contains information that is common to all chains. It is a part of a bayesLife.mcmc.set object.

Details
The object is in standard cases not to be manipulated by itself, but rather as part of a bayesLife.mcmc.set object.

Value
A bayesLife.mcmc.meta object contains components sex, nr.chains, start.year, present.year, wpp.year, buffer.size, my.e0.file, a, delta, tau, Triangle.ini, k.ini, z.ini, omega.ini, lambda.ini, lambda.k.ini, lambda.z.ini, Triangle.ini.low, Triangle.ini.up, k.ini.low, k.ini.up, z.ini.low, z.ini.up, lambda.ini.low, lambda.ini.up, lambda.k.ini.low, lambda.k.ini.up, lambda.z.ini.low, lambda.z.ini.up, omega.ini.low, omega.ini.up, Triangle.c.ini.norm, k.c.ini.norm, z.c.ini.norm, Triangle.c.prior.low, Triangle.c.prior.up, k.c.prior.low, k.c.prior.up, z.c.prior.low, z.c.prior.up, Triangle.c.width, k.c.width, z.c.width, nu, dl.p1, dl.p2, sumTriangle.lim, auto.conf. Their meaning and value are the same as the input arguments to the run.e0.mcmc function. Furthermore, it contains components:

e0.matrix.all  A $q \times n$ matrix with the United Nations life expectancy estimates. $q$ is number of years, $n$ is number of countries (see nr.countries below). The first $n_e$ columns correspond to countries included in the MCMC estimation (see nr.countries.estimation below), where $n_e \leq n$. The order of the countries corresponds to the order of countries in the element regions, see below.
e0.matrix  Like e0.matrix.all, but it has NA values for years where no historical data is available.
d.ct  A difference e0 matrix of size $(q - 1) \times n$. Each element $x_{t,i} =$ difference $y_{t+1,i} - y_{t,i}$ where $y$ denotes elements of e0.matrix.
loessSD  The loess matrix of d.ct.

nr.countries  Number of countries included in the e0 matrices.
nr.countries.estimation  Number of countries included in the MCMC estimation. It must be smaller or equal to nr.countries.

c.t.index  A list with one element per country. For each country, it contains the index within e0.matrix where the observed historical data are not NA, i.e. it points to the data used in the estimation.
regions

List of arrays of length nr.countries. These are:
name - Region name for each country.
code - Region code for each country.
area.name - Area name for each country.
area.code - Area code for each country.
country.name - Array of country names.
country.code - Array of country codes.
Any country indices in the bayesLife.mcmc.meta object are derived from this component.

output.dir

Directory for storing simulation output.

suppl.data

If supplemental data were used in the simulation (i.e. start year was set prior to 1950), this is a list containing information about the additional data. It has the following components of the same form as described above, but related only to the additional data: e0.matrix, regions, Tc.index, nr.countries, d.ct, loessSD. In addition, it has the vectors
index.from.all.countries - a vector of size nr.countries (all countries) where each element is an index to the supplemental data, i.e. a match from regions$country_code to suppl.data$regions$country_code.
index.to.all.countries - a vector of size suppl.data$nr.countries (additional countries) where each element is an index to all data, i.e. a match from suppl.data$regions$country_code to regions$country_code.

Author(s)

Hana Sevcikova

See Also

run.e0.mcmc, get.e0.mcmc

Examples

```r
sim.dir <- file.path(find.package("bayesLife"), "ex-data", "bayesLife.output")
m <- get.e0.mcmc(sim.dir)
summary(m, meta.only = TRUE)
names(m$meta)
```
Usage

convert.e0.trajectories(dir = file.path(getwd(), "bayesLife.output"),
                        n = 1000, output.dir = NULL, verbose = FALSE)

Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>dir</td>
<td>Directory containing the prediction object. It should correspond to the output.dir argument of the e0.predict function.</td>
</tr>
<tr>
<td>n</td>
<td>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.</td>
</tr>
<tr>
<td>output.dir</td>
<td>Directory in which the resulting files will be stored. If NULL the same directory is used as for the prediction. Otherwise, if the directory contains joint predictions for both sexes, the outputs are stored into subdirectories ‘f’ and ‘m’.</td>
</tr>
<tr>
<td>verbose</td>
<td>Logical switching log messages on and off.</td>
</tr>
</tbody>
</table>

Details

The function creates two files per sex. One is called “ascii_trajectories.csv”, it 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
- “e0”: life expectancy

The second file is called “ascii_trajectories_wide.csv”, it is also a comma-separated table and it contains the same information as above but in a `transposed` 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 n is smaller than the total number of trajectories, the trajectories are selected using equal spacing.

Note

This function is automatically called from the e0.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.

Author(s)

Hana Sevcikova

See Also

write.e0.projection.summary, e0.predict
Examples

```r
# Not run:
sim.dir <- file.path(find.package("bayesLife"), "ex-data", "bayesLife.output")
pred.dir <- file.path(getwd(), "exampleLpred")

# stores 10 trajectories out of 26 (2x(60-35)/2 + 1) into
# exampleLpred/predictions/ascii_trajectories.csv
e0.predict(sim.dir=sim.dir, output.dir=pred.dir,
  burnin=35, save.as.ascii=10, verbose=TRUE)

# stores all 26 trajectories into the current directory
convert.e0.trajectories(dir=pred.dir, n="all", output.dir=".", verbose=TRUE)

# Note: If the output.dir argument in e0.predict is omitted,
# call convert.e0.trajectories with dir=sim.dir

# End(Not run)
```

e0.coda.list.mcmc  Conversion to coda's Objects

Description

The functions convert MCMC traces (simulated using run.e0.mcmc) into objects that can be used with the coda package.

Usage

```r
e0.coda.list.mcmc(mcmc.list = NULL, country = NULL, chain.ids = NULL,
  sim.dir = file.path(getwd(), "bayesLife.output"),
  par.names = e0.parameter.names(),
  par.names.cs = e0.parameter.names.cs(), low.memory = FALSE, ...)

e0.coda(mcmc, country = NULL, par.names = e0.parameter.names(),
  par.names.cs = e0.parameter.names.cs(), ...)
```

Arguments

- **mcmc.list**: List of bayesLife.mcmc objects, or an object of class bayesLife.mcmc.set or bayesLife.prediction. If it is NULL, the MCMCs are loaded from sim.dir. Either mcmc.list or sim.dir must be given.
- **mcmc**: Object of class bayesLife.mcmc.
- **country**: Country name or code. It is 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.
Directory with the MCMC simulation results. Only used if mcmc.list is NULL.

Names of country-independent parameters to be included.

Names of country-specific parameters to be included. The argument country is used to filter out traces that correspond to a specific country. If country is not given, for each parameter, traces for all countries are included.

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.

The function e0.coda.list.mcmc returns an object of class "mcmc.list". The function coda.mcmc returns an object of class "mcmc", both defined in the coda package.

Hana Sevcikova

See Also
e0.partraces.plot for plotting the MCMC traces and summary.bayesLife.mcmc.set.

Examples

```r
sim.dir <- file.path(find.package("bayesLife"), "ex-data", "bayesLife.output")
coda.list <- e0.coda.list.mcmc(sim.dir=sim.dir, country="France", burnin=30)
summary(coda.list)
```

Description

Function e0.diagnose runs convergence diagnostics of existing MCMCs, using the raftery.diag function from the coda package.

Usage

```r
e0.diagnose(sim.dir, thin=225, burnin = 10000, express = FALSE,
             country.sampling.prop = NULL, keep.thin.mcmc=FALSE, verbose = TRUE)
```
Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>sim.dir</td>
<td>Directory with the MCMC simulation results.</td>
</tr>
<tr>
<td>thin</td>
<td>Thinning interval.</td>
</tr>
<tr>
<td>burnin</td>
<td>Number of iterations to be discarded from the beginning of the parameter traces.</td>
</tr>
<tr>
<td>express</td>
<td>Logical. If TRUE, the convergence diagnostics 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.</td>
</tr>
<tr>
<td>country.sampling.prop</td>
<td>Proportion of countries that are included in the diagnostics. If it is NULL and express=FALSE, all countries are included. Setting here a number between 0 and 1, one can limit the number of countries which are then randomly sampled. Note that for long MCMCs, this argument may significantly influence the runtime of this function.</td>
</tr>
<tr>
<td>keep.thin.mcmc</td>
<td>Logical. If TRUE the thinned traces used for computing the diagnostics are stored on disk (see create.thinned.e0.mcmc).</td>
</tr>
<tr>
<td>verbose</td>
<td>Logical switching log messages on and off.</td>
</tr>
</tbody>
</table>

Details

The function invokes the e0.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/bayesLife.convergence_{thin}_{burnin}.rda' and can be accessed using the function get.e0.convergence.

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

Value

e0.diagnose returns an object of class bayesLife.convergence with components:

<table>
<thead>
<tr>
<th>Component</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>result</td>
<td>Table containing all not-converged parameters. Its columns include ‘Total iterations needed’ and ‘Remaining iterations’.</td>
</tr>
<tr>
<td>lresult.country.independent</td>
<td>Number of rows in result that correspond to country-independent parameters. These rows are grouped at the beginning of the table.</td>
</tr>
<tr>
<td>country.independent</td>
<td>Result of e0.raftery.diag processed on country-independent parameters.</td>
</tr>
<tr>
<td>country.specific</td>
<td>Result of e0.raftery.diag processed on country-specific parameters.</td>
</tr>
<tr>
<td>iter.necessary</td>
<td>Number of additional iterations suggested in order to achieve convergence.</td>
</tr>
<tr>
<td>iter.total</td>
<td>Total number of iterations of the original unthinned set of chains.</td>
</tr>
<tr>
<td>use.nr.traj</td>
<td>Suggestion for number of trajectories in generating predictions.</td>
</tr>
<tr>
<td>burnin</td>
<td>Burnin used.</td>
</tr>
<tr>
<td>thin</td>
<td>Thinning interval used.</td>
</tr>
</tbody>
</table>
e0.dl.coverage

<table>
<thead>
<tr>
<th>status</th>
<th>Vector of character strings containing the result status. Possible values: ‘green’, ‘red’.</th>
</tr>
</thead>
<tbody>
<tr>
<td>mcmc.set</td>
<td>Object of class bayesLife.mcmc.set that corresponds to the original set of MCMCs on which the diagnostics was run.</td>
</tr>
<tr>
<td>thin.mcmc</td>
<td>If keep.thin.mcmc is TRUE, it is an object of class bayesLife.mcmc.set that corresponds to the thinned mcmc set on which the diagnostics was run, otherwise NULL.</td>
</tr>
<tr>
<td>express</td>
<td>Value of the input argument express.</td>
</tr>
<tr>
<td>nr.countries</td>
<td>Vector with elements used - number of countries used in this diagnostics, and total - number of countries that this mcmc.set object was estimated on.</td>
</tr>
</tbody>
</table>

Author(s)

Hana Sevcikova, Adrian Raftery

See Also

e0.raftery.diag, raftery.diag, summary.bayesLife.convergence, get.e0.convergence, create.thinned.e0.mcmc

e0.dl.coverage  Goodness of Fit of the Double Logistic Function

Description

The function computes coverage, i.e. the ratio of observed data fitted within the given probability intervals of the predictive posterior distribution of the double logistic function, as well as the root mean square error of the simulation.

Usage

e0.dl.coverage(sim.dir, pi = c(80, 90, 95), burnin = 10000, verbose = TRUE)

Arguments

<table>
<thead>
<tr>
<th>sim.dir</th>
<th>Directory with the MCMC simulation results. If a prediction and its corresponding thinned mcmcs are available in the simulation directory, those are taken for assessing the goodness of fit.</th>
</tr>
</thead>
<tbody>
<tr>
<td>pi</td>
<td>Probability interval. It can be a single number or an array.</td>
</tr>
<tr>
<td>burnin</td>
<td>Burnin. Only relevant if sim.dir does not contain thinned chains.</td>
</tr>
<tr>
<td>verbose</td>
<td>Logical switching log messages on and off.</td>
</tr>
</tbody>
</table>

Value

List with the same components as tfr.dl.coverage.
Note
To see the fit visually per country, use `e0.DLcurve.plot(..., predictive.distr=TRUE,...).

Author(s)
Hana Sevcikova

See Also
`e0.DLcurve.plot`

Examples
```r
## Not run:
sim.dir <- file.path(find.package("bayesLife"), "ex-data", "bayesLife.output")
e0 <- get.e0.mcmc(sim.dir)
# Note that this simulation is a toy example and thus has not converged.
gof <- e0.dl.coverage(sim.dir)
gof$country.coverage
e0.DLcurve.plot(e0, country="608", predictive.distr=TRUE, pi=c(80, 90, 95))
## End(Not run)
```

Description
The functions plot the posterior distribution of the double logistic function used in the simulation, including their median and given probability intervals.

Usage
```r
e0.DLcurve.plot(mcmc.list, country, burnin = NULL, pi = 80,
e0.lim = NULL, nr.curves = 20, predictive.distr = FALSE, ylim = NULL,
xlab = "e(0)", ylab = "5-year gains", main = NULL, show.legend=TRUE,
col=c('black', 'red', '#00000020'), ...) 
e0.DLcurve.plot.all(mcmc.list = NULL, sim.dir = NULL,
output.dir = file.path(getwd(), "DLcurves"),
output.type = "png", burnin = NULL, verbose = FALSE, ...)
e0.parDL.plot(mcmc.set, country = NULL, burnin = NULL, lty = 2,
ann = TRUE, ...)
e0.world.dlcurves(x, mcmc.list, burnin=NULL, ...)
e0.country.dlcurves(x, mcmc.list, country, burnin=NULL, ...)
```
Arguments

- **mcmc.list**: List of `bayesLife.mcmc` objects, an object of class `bayesLife.mcmc.set` or of class `bayesLife.prediction`. In case of `e0.DLcurve.plot` or `e0.DLcurve.plot.all` if `mcmc.list` is `NULL`, it is loaded from `sim.dir`.
- **mcmc.set**: Object of class `bayesLife.mcmc.set` or `bayesLife.prediction`.
- **country**: Name or numerical code of a country.
- **burnin**: Number of iterations to be discarded from the beginning of parameter traces.
- **pi**: Probability interval. It can be a single number or an array.
- **e0.lim**: It can be a tuple of the minimum and maximum life expectancy to be shown in the plot. If `NULL`, it takes the minimum of observed data and 40, and the maximum of observed data and 90.
- **nr.curves**: Number of curves to be plotted. If `NULL`, all curves are plotted.
- **predictive.distr**: Logical. If `TRUE`, an error term is added to each trajectory.
- **ylim, xlab, ylab, main, lty**: Graphical parameters passed to the `plot` function.
- **show.legend**: Logical determining if the legend should be shown.
- **col**: Vector of colors in this order: 1. observed data points, 2. quantiles, 3. trajectories
- **...**: Additional graphical parameters. In addition, any arguments from `e0.DLcurve.plot` except `country` can be passed to `e0.DLcurve.plot.all`.
- **sim.dir**: Directory with the simulation results. Only relevant, if `mcmc.list` is `NULL`.
- **output.dir**: Directory into which resulting graphs are stored.
- **output.type**: Type of the resulting files. It can be “png”, “pdf”, “jpeg”, “bmp”, “tiff”, or “postscript”.
- **verbose**: Logical switching log messages on and off.
- **x**: `e0` values for which the double logistic should be computed.
- **ann**: Logical if parameters should be annotated.

Details

`e0.DLcurve.plot` plots double logistic curves for the given country. `e0.DLcurve.plot.all` creates such plots for all countries and stores them in `output.dir`. Parameters passed to the double logistic function are either thinned traces created by the `e0.predict` function (if `mcmc.list` is an object of class `bayesLife.prediction`), or they are selected by equal spacing from the MCMC traces. In the former case, `burnin` is set automatically; in the latter case, `burnin` defaults to 0 since such object has already been “burned”. If `nr.curves` is smaller than 2000, the median and probability intervals are computed on a sample of 2000 equally spaced data points, otherwise on all plotted curves.

Function `e0.parDL.plot` draws the means of the DL parameters as vertical and horizontal lines. The lines are added to the current graphical device and annotated if `ann` is `TRUE`. If `country` is `NULL`, the mean of world parameters are drawn.

Function `e0.world.dlcurves` returns the DL curves of the hierarchical distribution. Function `e0.country.dlcurves` returns DL curves for a given country. If `mcmc.list` is a prediction object, `burnin` should not be given, as such object has already been “burned”.

**Value**

e0.world.dlcurves and e0.country.dlcurves return a matrix of size $N \times M$ where $N$ is the number of trajectories and $M$ is the number of values of $x$.

**Author(s)**

Hana Sevcikova

**Examples**

```r
# Not run:
sim.dir <- file.path(find.package("bayesLife"), "ex-data", "bayesLife.output")
mcmc.set <- get.e0.mcmc(sim.dir=sim.dir)
e0.DLcurve.plot(mcmc.set, country="Japan", burnin=40)
e0.parDL.plot(mcmc.set, "Japan")

# add the median of the hierarchical DL curves
x <- seq(40, 90, length=100)
world <- e0.world.dlcurves(x, mcmc.set, burnin=40)
qw <- apply(world, 2, median)
lines(x, qw, col='blue')

# End(Not run)
```

---

**Description**

The functions plot the posterior distribution of the gaps between female and male life expectancy, modeled and predicted using a model described in Lalic (2011). This can be done for one country (e0.gap.plot) or for all countries (e0.gap.plot.all).

**Usage**

e0.gap.plot(e0.pred, country, e0.pred2 = NULL, pi = c(80, 95),
            nr.traj = 0, xlim = NULL, ylim = NULL, type = "b",
            xlab = "Year", ylab = "Gap in life expectancy", main = NULL,
            show.legend = TRUE, ...)

e0.gap.plot.all(e0.pred, output.dir = file.path(getwd(), "e0gaps"),
               output.type = "png", verbose = FALSE, ...)

**Arguments**

- `e0.pred`: Object of class `bayesLife.prediction` containing female projections. If `e0.pred2` is not given, then this object must contain the ‘joint.male’ component, generated using the `e0.jmale.predict` function.
country  
---
Name or numerical code of a country.

e0.pred2  
---
Object of class `bayesLife.prediction` containing male projections. By default the male projections is taken from the joint female-male projections in `e0.pred`, see above.

pi  
---
Probability interval. It can be a single number or an array.

nr.traj  
---
Number of trajectories to be plotted.

xlim, ylim, type, xlab, ylab, main  
---
Graphical parameters passed to the `plot` function.

show.legend  
---
Logical controlling whether the legend should be drawn.

output.dir  
---
Directory into which resulting graphs are stored.

output.type  
---
Type of the resulting files. It can be “png”, “pdf”, “jpeg”, “bmp”, “tiff”, or “postscript”.

verbose  
---
Logical switching log messages on and off.

...  
---
Additional graphical parameters. In addition, for `e0.gap.plot.all`, ... contains any of the arguments of `e0.gap.plot`.

---

**Author(s)**

Hana Sevcikova

**References**


**See Also**

- `e0.joint.plot`
- `e0.jmale.estimate`
- `e0.jmale.predict`
- `get.e0.jmale.prediction`

**Examples**

```r
# See example for e0.jmale.predict
```

---

**Description**

The function estimates the joint female-male model of life expectancy, as described in Raftery et al. (2014, 2012) and Lalic (2011). It consist of two equations with t-distributed errors, see Details below.
Usage

e0.jmale.estimate(mcmc.set, countries.index = NULL, estDof.eq1 = TRUE, start.eq1 = list(dof = 2), max.e0.eq1 = 83, estDof.eq2 = TRUE, start.eq2 = list(dof = 2), constant.gap.eq2 = TRUE, my.e0.file = NULL, my.locations.file = NULL, verbose = FALSE)

Arguments

mcmc.set Object of class bayesLife.mcmc.set containing estimation results of female life expectancy.
countries.index Index of countries (within the mcmc.set object) to be included in the estimation. By default, all countries included in the estimation of mcmc.set are used.
estDof.eq1, estDof.eq2 Logical, controlling whether the degrees of freedom of the first and second equation, respectively, should be estimated. If it is FALSE the degrees of freedom are set using the arguments start.eq1 and start.eq2, respectively.
start.eq1, start.eq2 Argument start of the tlm function of the hett package, which is applied to the first and second equation of the model, respectively. It is a list of possibly four named components, ("beta", "lambda", "dof", "omega"), for the location, scale, degrees of freedom parameters and random scale effects respectively. If estDof.eq1 (estDof.eq2) is FALSE, the "dof" component must be given.
max.e0.eq1 Maximum female life expectancy of records included in the estimation of the first equation (parameter M in Details below).
constant.gap.eq2 Logical. If TRUE the coefficient of the second equation ($\gamma_1$) is set to one and the standard deviation is estimated under the assumption of normal distribution.
my.e0.file File name containing user-specified male time series for one or more countries. The function replaces the corresponding country data from the WPP dataset by values in this file. Only columns are replaced that match column names of the WPP dataset.
my.locations.file File name containing user-specified locations if different from the default UNlocations dataset. It should be the same file as passed to run.e0.mcmc for female life expectancy.
verbose Logical switching log messages on and off. If TRUE summary results from the tlm function of both equations are shown.

Details

The joint female-male life expectancy model is a model for estimating gaps $G$ between female and male life expectancy. It consists of two parts, see Equation (1) in Raftery et al. (2012):

1. If $l_{c,t} \leq M$, then

$$G_{c,t} = \beta_0 + \beta_1 l_{c,1953} + \beta_2 G_{c,t-1} + \beta_3 l_{c,t} + \beta_4 (l_{c,t} - 75)_+ + \epsilon_{c,t}$$
where $\epsilon_{c,t}$ is iid $t(\mu = 0, \sigma_1^2, \nu_1)$.

2. If $l_{c,t} > M$, then

$$G_{c,t} = \gamma_1 G_{c,t-1} + \epsilon_{c,t}$$

where $\epsilon_{c,t}$ is iid $t(\mu = 0, \sigma_2^2, \nu_2)$.

Here, $t$ is the time and $c$ is the country index. $G_{c,t}$ is the gap for country $c$ at time $t$ and $l_{c,t}$ is the female life expectancy for country $c$ at time $t$. $M$ can be set in the `maxNePNeq1` argument.

Using the `tlm` function of the `hett` package, the function estimates the coefficients $\beta_i$ ($i = 1, \ldots, 4$) and $\gamma_1$, as well as parameters $\sigma_j$ ($j = 1, 2$) and optionally the degrees of freedom $\nu_j$ ($j = 1, 2$). If `constantNgapNeqR` is TRUE, $\gamma_1$ is set to 1 and $\epsilon_{c,t}$ is iid $N(\mu = 0, \sigma_2^2)$.

The `mcmc.set` object should be a `bayesLife.mcmc.set` object obtained from a simulation of a female life expectancy. Note that since only the observed data and no MCMC results are used in this estimation, the `mcmc.set` object can be obtained from a toy simulation such as in the example below. The function extracts observed data from this object and treats them as $l_{c,t}$. For the male historical time series, the function takes the male WPP dataset from the same `wpp` package as the female data (`e0M`) and possibly partly replaces it by any user-specified data given in `my.e0.file`.

**Value**

List with the components, `eq1` and `eqR`, each containing estimation results from the first and second equation, respectively. These are:

- **coefficients**: Estimated coefficients $\beta_i$.
- **sigma**: Parameter $\sigma_j$.
- **dof**: Degrees of freedom $\nu_j$. If `estDof.eq1` (or `estDof.eq2`) is TRUE this parameter is estimated, otherwise it is set to the value of `start.eq1$dof` (or `start.eq2$dof`).

**Author(s)**

Hana Sevcikova

**References**


**See Also**

`e0.jmale.predict`
### Examples

```
## Not run:
sim.dir <- file.path(find.package("bayesLife"), "ex-data", "bayesLife.output")
m <- get.e0.mcmc(sim.dir)
fit <- e0.jmale.estimate(m, verbose=TRUE)

## End(Not run)
```

---

**e0.jmale.predict**  
*Prediction of the Joint Female-Male Model*

---

### Description

Prediction of the joint female-male model of life expectancy, as described in Raftery et al. (2014, 2012) and Lalic (2011).

### Usage

```
e0.jmale.predict(e0.pred, estimates = NULL, gap.lim = c(0, 18),
                 max.e0.eq1.pred = 86, my.e0.file = NULL, my.locations.file = NULL,
                 save.as.ascii = 1000, verbose = TRUE, ...
```

### Arguments

- **e0.pred** Object of class `bayesLife.prediction` containing female projections of life expectancy.
- **estimates** List of the same structure as returned by `e0.jmale.estimate`, containing the model estimation results. If it is not given, the `e0.jmale.estimate` function is invoked.
- **gap.lim** Vector of length two giving the minimum and maximum bounds for the female-male life expectancy gaps.
- **max.e0.eq1.pred** Maximum female life expectancy for male projections handled by the first equation (parameter $M$ in Equation (1) in Raftery et al. (2012)).
- **my.e0.file** File name containing user-specified male time series for one or more countries. The function replaces the corresponding country data from the WPP dataset by values in this file. Only columns are replaced that match column names of the WPP dataset.
- **my.locations.file** File name containing user-specified locations if different from the default `UNlocations` dataset. It should be the same file as passed to `run.e0.mcmc` for female life expectancy.
- **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.
- **verbose** Logical switching log messages on and off.
- **...** Further arguments passed to `e0.jmale.estimate`.  

---
Details

If no estimates are given, the function invokes an estimation by calling `e0.jmale.estimate`. Using those estimates, the male life expectancy is projected forward in time (as a function of a female-male gap), using the female predictions from `e0.pred`. The initial male data point is extracted from the male WPP dataset (`e0.m`) and possibly partly replaced by any user-specified data given in `my.e0.file`.

The resulting trajectory files are stored in a subdirectory of the female prediction directory, called ‘joint_male’. Furthermore, an object of class `bayesLife.prediction` is created and added to `e0.pred` as a component called `joint.male`.

The predicted gaps can be viewed using the `e0.gap.plot` function.

Value

Updated `e0.pred` object where a new component was added, called `joint.male`. It is also an object of class `bayesLife.prediction` and it contains results from this prediction.

Author(s)

Hana Sevcikova

References


See Also

`e0.jmale.estimate, get.e0.jmale.prediction, e0.predict, e0.gap.plot`

Examples

```r
## Not run:
sim.dir <- tempfile()
m <- run.e0.mcmc(sex='F', nr.chains=1, iter=30, thin=1, output.dir=sim.dir)
pred <- e0.predict(m, burnin=15, verbose=FALSE, save.as.ascii=0)
both.pred <- e0.jmale.predict(pred)
e0.traj.plot(both.pred, 'Guatemala') # Female
e0.traj.plot(get.e0.jmale.prediction(both.pred), 'Guatemala') # Male

# Marginal distribution of the sex-specific projections
e0.traj.plot(both.pred, 'Guatemala', both.sexes=TRUE, pi=80)

# Plotting the gaps
e0.gap.plot(both.pred, 'Guatemala')
```
# Joint distribution of the sex-specific projections
e0.joint.plot(both.pred, 'Guatemala', pi=80, years=c(2013, 2043, 2093))

unlink(sim.dir, recursive=TRUE)

## End(Not run)

---

## e0.joint.plot

### Joint Posterior Distribution of Female and Male Life Expectancy

**Description**

The functions plot the joint posterior distribution of female and male life expectancy, modeled and predicted using the joint model described in Lalic (2011). This can be done for one country (e0.joint.plot) or for all countries (e0.joint.plot.all).

**Usage**

```r
e0.joint.plot(e0.pred, country, pi = 95, years, nr.points = 500,
              obs.pch = 17, obs.cex=1, xlim = NULL,
              ylim = NULL, xlab = "Female life expectancy", ylab = "Male life expectancy",
              main = NULL, col = NULL, show.legend = TRUE, add = FALSE, ...)
e0.joint.plot.all(e0.pred, output.dir = file.path(getwd(), "e0joint"),
                 output.type = "png", verbose = FALSE, ...)
```

**Arguments**

- `e0.pred`: Object of class `bayesLife.prediction` containing female projections.
- `country`: Name or numerical code of a country.
- `pi`: Probability interval. It can be a single number or an array.
- `years`: Array of future years for which to plot the distribution.
- `nr.points`: Number of points shown in the plot for each year.
- `obs.pch`, `obs.cex`: Graphical parameters used for displaying observed data or data without variation.
- `xlim`, `ylim`, `xlab`, `ylab`, `main`: Graphical parameters passed to the `plot` function.
- `col`: Array of colors, one for each year.
- `show.legend`: Logical controlling whether the legend should be drawn.
- `add`: Logical controlling whether the distribution should be plotted into a new graphic device (FALSE) or into an existing device (TRUE).
- `output.dir`: Directory into which resulting graphs are stored.
output.type: Type of the resulting files. It can be “png”, “pdf”, “jpeg”, “bmp”, “tiff”, or “postscript”.

verbose: Logical switching log messages on and off.

... Additional graphical parameters passed to the `lines` function drawing ellipses. In addition, for `e0.joint.plot.all`, ... contains any of the arguments of `e0.joint.plot`.

Author(s)

Hana Sevcikova, Adrian Raftery

References


See Also

e0.gap.plot, e0.trajectories.plot, e0.jmale.predict

Examples

# See example for e0.jmale.predict

e0.map: World Map of the Life Expectancy

Description

Generates a world map of the life expectancy for given quantile and projection or estimation period.

Usage

e0.map(pred, ...)
e0.map.all(pred, output.dir, output.type = "png", e0.range = NULL, nr.cats = 50, same.scale = TRUE, quantile = 0.5, file.prefix = "e0wrldmap_", ...)

gte0.map.parameters(pred, e0.range = NULL, nr.cats = 50, same.scale = TRUE, quantile = 0.5, ...)
e0.map.gvis(pred, ...)
Arguments

pred Object of class `bayesLife.prediction`.
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”.
e0.range Range of the life expectancy to be displayed. It is of the form \(c(e0.min, e0.max)\). By default, the whole 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.
... In `e0.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 `e0.map.gvis`, ... are all arguments that can be passed to `tfr.map.gvis`. In addition, the first three functions accept arguments passed to the `mapCountryData` function of the `rworldmap` package.

Details

`e0.map` creates a single map for the given time period and quantile. `e0.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 bottom of the map is created.

Function `get.e0.map.parameters` can be used in combination with `e0.map`. (Note that `get.e0.map.parameters` is called from inside of `e0.map.all`.) It sets breakpoints for the color scheme using quantiles of a fitted gamma distribution.

Function `e0.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, both `e0.map` and `e0.map.gvis` produce maps of life expectancy. 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 `e0.parameter.names.cs.extended()`.

Value

get.e0.map.parameters returns a list with elements:

pred The object of class `bayesLife.prediction` used in the function.
quantile Value of the argument `quantile`. 
**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.

**Usage**

```r
e0.median.set(sim.dir, country, values, years = NULL, joint.male = FALSE)
e0.median.shift(sim.dir, country, reset = FALSE, shift = 0, from = NULL, to = NULL, joint.male = FALSE)
e0.median.reset(sim.dir, countries, joint.male = FALSE)
```

**Examples**

```r
## Not run:
sim.dir <- file.path(find.package("bayesLife"), "ex-data", "bayesLife.output")
pred <- get.e0.prediction(sim.dir=sim.dir)
# Uses heat colors and seven categories by default
e0.map(pred)
# Uses more colors with more suitable categorization
params <- get.e0.map.parameters(pred)
do.call('e0.map', params)
# Another projection year on the same scale
do.call('e0.map', c(list(year=2043), params))
```

**Author(s)**

Hana Sevcikova, Adrian Raftery

**See Also**

`tfr.map`
Arguments

- **sim.dir**: Directory containing the prediction object.
- **country**: Name or numerical code of a country.
- **countries**: Vector of country names or codes.
- **values**: Array of the new median values.
- **years**: Numeric vector giving 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 period. If values correspond to consecutive years, only the first year might be given here. A year \( t \) represents a prediction period \([t_i, t_{i+1}]\) if \( t_i < t \leq t_{i+1} \).
- **joint.male**: Logical. If `TRUE`, the function is applied to a male prediction that was generated using the joint female-male model implemented in the function `e0.jmale.predict`.
- **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.

Details

The function `e0.median.set` can be used to set the medians of the given country to specific values. Function `e0.median.shift` can be used to offset the medians by a specific constant, or to reset the medians to their original BHM values. Function `e0.median.reset` resets medians of the given countries to the original values. In all cases, if a median is modified, the corresponding offset is stored in the prediction object (element `median.shift`). All three 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.

Value

All three functions return an updated object of class `bayesLife.prediction`.

Author(s)

Hana Sevcikova
e0.parameter.names  Accessing Parameter Names

Description

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

Usage

e0.parameter.names()
e0.parameter.names.cs()
e0.parameter.names.extended()
e0.parameter.names.cs.extended(country.code = NULL)

Arguments

country.code  Country code. If it is given, the country-specific parameter names contain the suffix '_c\text{x}' where x is the country.code.

Value

e0.parameter.names returns names of the world parameters.
e0.parameter.names.cs returns names of the country-specific parameters.
e0.parameter.names.extended returns names of all world parameters in their extended format. I.e. parameters 'Triangle' and 'lambda' have the suffix '_1', '_2', '_3', and '_4'.
e0.parameter.names.cs.extended returns names of all country-specific parameters in their extended format. I.e. parameters 'Triangle.c' and 'lambda.c' are in their extended format with the suffix '_1', '_2' and '_3'.

Author(s)

Hana Sevcikova

Examples

e0.parameter.names()
e0.parameter.names.extended()
e0.parameter.names.cs()
e0.parameter.names.cs.extended()
Plotting MCMC Parameter Density

Description

Functions for plotting density of the posterior distribution of the MCMC parameters.

Usage

```r
e0.pardensity.plot(mcmc.list = NULL,
    sim.dir = file.path(getwd(), "bayesLife.output"),
    chain.ids = NULL, par.names = e0.parameter.names(),
    burnin = NULL, dev.ncol = 5, low.memory = TRUE, ...)
e0.pardensity.cs.plot(country, mcmc.list = NULL,
    sim.dir = file.path(getwd(), "bayesLife.output"),
    chain.ids = NULL, par.names = e0.parameter.names.cs(),
    burnin = NULL, dev.ncol = 3, low.memory = TRUE, ...)
```

Arguments

- **country**: Name or numerical code of a country.
- **mcmc.list**: List of `bayesLife.mcmc` objects, or an object of class `bayesLife.mcmc.set` or of class `bayesLife.prediction`. If it is NULL, the parameter 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 `e0.pardensity.plot`, or all country-specific parameters are plotted if used within `e0.pardensity.cs.plot`.
- **burnin**: Number of iterations to be discarded from the beginning of each chain.
- **dev.ncol**: Number of columns 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 memory-efficient mode.
- ... Further arguments passed to the `density` function.

Details

The functions plot the density of the posterior distribution either for country-independent parameters (`e0.pardensity.plot`) or for country-specific parameters (`e0.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 `bayesLife.prediction`, thinned traces 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 `e0.predict`.

Author(s)

Hana Sevcikova

See Also

`e0.partraces.plot`

Examples

```r
## Not run:
sim.dir <- file.path(find.package("bayesLife"), "ex-data", "bayesLife.output")
e0.pardensity.plot(sim.dir=sim.dir, burnin=10)
e0.pardensity.cs.plot(country="Ireland", sim.dir=sim.dir, burnin=10)

## End(Not run)
```

---

**e0.partraces.plot**  
Plotting MCMC Parameter Traces

Description

Functions for plotting the MCMC parameter traces.

Usage

```r
e0.partraces.plot(mcmc.list = NULL,  
                 sim.dir = file.path(getwd(), "bayesLife.output"),  
                 chain.ids = NULL, par.names = e0.parameter.names(),  
                 nr.points = NULL, dev.ncol = 5, low.memory = TRUE, ...)
e0.partraces.cs.plot(country, mcmc.list = NULL,  
                     sim.dir = file.path(getwd(), "bayesLife.output"),  
                     chain.ids = NULL, par.names = e0.parameter.names.cs(),  
                     nr.points = NULL, dev.ncol = 3, low.memory = TRUE, ...)
```

Arguments

- `country`  
  Name or numerical code of a country.
- `mcmc.list`  
  List of `bayesLife.mcmc` objects, or an object of class `bayesLife.mcmc.set` or of class `bayesLife.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`.
null
parNnames 
null
nrNpoints 
null
devNncol 
null
lowNmemory 
null
NNN

Details
The functions plot MCMC traces either for country-independent parameters (e0.partraces.plot) or for country-specific parameters (e0.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.

Author(s)
Hana Sevcikova

See Also
e0.coda.list.mcmc and get.e0.parameter.traces for retrieving the raw values of traces.

Examples
## Not run:
sim.dir <- file.path(find.package("bayesLife"), "ex-data", "bayesLife.output")
e0.partraces.plot(sim.dir=sim.dir)
e0.partraces.cs.plot(country="Ireland", sim.dir=sim.dir)

## End(Not run)
Usage

e0.predict(mcmc.set = NULL, end.year = 2100,
  sim.dir = file.path(getwd(), "bayesLife.output"), replace.output = FALSE,
  predict.jmale = TRUE, nr.traj = NULL, thin = NULL, burnin = 10000,
  use.diagnostics = FALSE, save.as.ascii = 1000, start.year = NULL,
  output.dir = NULL, low.memory = TRUE, seed = NULL, verbose = TRUE, ...)

Arguments

mcmc.set Object of class bayesLife.mcmc.set. If it is NULL, the object is loaded from
the directory given by sim.dir.
end.year End year of the prediction.
sim.dir Directory with the MCMC simulation results. It should equal to the output.dir
argument in run.e0.mcmc.
replace.output Logical. If TRUE, existing predictions in output.dir will be replaced by results
of this run.
predict.jmale Logical controlling if a joint female-male prediction should be performed. This
is done only if the underlying mcmcs in sim.dir correspond to a female sim-
ulation. In such a case the e0.jmale.predict is invoked. Arguments to this
function can be passed in ....
.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 minimum of the size of the parameter sample and 2000.
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.diagnostics Logical determining if an existing convergence diagnostics should be used for
choosing the values of thin and burnin. In such a case, arguments nr.traj,
thin and burnin are ignored. The 'best' values are chosen from results of
running the e0.diagnose function. Only diagnostics can be used that suggest a
convergence of the underlying MCMCs. If there are more than one such objects,
the one is chosen whose recommendation for the number of trajectories is larger
and closest to 2000.
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.
start.year This argument should be only used if the start year of the prediction is before or
at the present year of the MCMC run (see Details below). By default the predic-
tion starts in the next time period after the present year (passed to run.e0.mcmc).
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.
low.memory Logical indicating if the prediction should run in a low-memory mode. If it is FALSE, the whole traces of all parameters, including the burnin, are loaded into memory. Otherwise, burnins are discarded and parameters are loaded as they are needed and are not kept in the memory.

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

verbose Logical switching log messages on and off.

... Additional arguments passed to the e0.jmale.predict function.

Details

The trajectories are generated using the double logistic function (Chunn et al. 2010). Parameter samples simulated via run.e0.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 (see create.thinned.e0.mcmc).

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 or equal the present year. In such a case, imputed missing values before present year are treated as ordinary predictions (trajectories are kept). 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 countries are saved into the same directory in a binary format, one file per country. 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 of a UN-specific format. They are selected by equal spacing (see function convert.e0.trajectories for more details on the conversion). In addition, two summary files are created: one in a user-friendly format, the other using a UN-specific coding of the variants and time (see write.e0.projection.summary for more details).

Value

Object of class bayeslife.prediction which is a list containing components:

quantiles A $n \times q \times p$ array of quantile values computed on the trajectories. $n$ is the number of countries, $q$ is the number of quantile bounds and $p$ is the number of projections.

traj.mean.sd A $n \times 2 \times p$ array holding the mean of all trajectories in the first column and the standard deviation in the second column. $n$ and $p$ are the number of countries and number of projections, respectively.

nr.traj Number of trajectories.

e0.matrix.reconstructed Matrix containing imputed e0 values on spots where the original e0 matrix has missing values, i.e. between the last observed data point and the present year.

output.directory Directory where trajectories corresponding to this prediction are stored.
nr.projections: Number of projections.
burnin: Burnin used for this prediction.
end.year: The end year of this prediction.
mcmc.set: Object of class bayesLife.mcmc.set used for this prediction, i.e. the burned, thinned, and collapsed MCMC chain.
joint.male: If `e0.jmale.predict` was invoked, this is an object of class bayesLife.prediction containing male projections. In addition to the components above, it contains elements `fit` (estimation results from `e0.jmale.estimate`) and `meta.changes` (components of `bayesLife.mcmc.meta` that differ from the female meta component).

Author(s)

Hana Sevcikova, using code from Jennifer Chunn

References


See Also

`run.e0.mcmc`, `e0.jmale.predict`, `create.thinned.e0.mcmc`, `convert.e0.trajectories`, `get.e0.prediction`, `summary.bayesLife.prediction`

Examples

```r
# Not run:
m <- run.e0.mcmc(nr.chains=1, iter=50, thin=1, verbose=TRUE)
pred <- e0.predict(m, burnin=25, verbose=TRUE)
summary(pred, country="Portugal")

# End(Not run)
```

---

**e0.predict.extra**

*Generating Posterior Trajectories of the Life Expectancy for Specific Countries or Regions*

**Description**

Using the posterior parameter samples the function generates posterior trajectories of the life expectancy for given countries or regions. It is intended to be used after running `run.e0.mcmc.extra`, but it can be also used for purposes of testing specific settings on one or a few countries.
**Usage**

```r
e0.predict.extra(sim.dir = file.path(getwd(), 'bayesLife.output'),
                  prediction.dir = sim.dir, countries = NULL,
                  save.as.ascii = 1000, verbose = TRUE, ...)
```

**Arguments**

- `sim.dir`: Directory with the MCMC simulation results.
- `prediction.dir`: Directory where the prediction object and the trajectories are stored.
- `countries`: Vector of country codes for which the prediction should be made. If it is `NULL`, the prediction is run for all countries that are included in the MCMC object but for which no prediction was generated.
- `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 conversions is desired. Note that the conversion is done on all countries.
- `verbose`: Logical switching log messages on and off.
- `...`: Additional arguments passed to a joint female-male prediction.

**Details**

In order to use this function, a prediction object must exist, i.e. the function `e0.predict` must have been processed prior to using this function.

Trajectories for given countries or regions are generated and stored in binary format along with other countries (in `prediction.dir`). The existing prediction object is updated and stored in the same directory. If `save.as.ascii` is larger than zero, trajectories of ALL countries are converted to an ascii format.

If the prediction object contains joint male projections, these are also created for the given countries.

**Value**

Updated object of class `bayesLife.prediction`.

**Author(s)**

Hana Sevcikova

**See Also**

`e0.predict`, `run.e0.mcmc.extra`
Description

The function computes the Raftery diagnostics for each parameter in the same way as `tfr.raftery.diag` of the `bayesTFR` package.

Usage

```r
e0.raftery.diag(mcmc = NULL, sim.dir = file.path(getwd(), "bayesLife.output"),
               burnin = 0, country = NULL, par.names = e0.parameter.names(),
               par.names.cs = e0.parameter.names.cs(),
               country.sampling.prop = 1, verbose = TRUE, ...)
```

Arguments

- `mcmc`: A `bayesLife.mcmc` or `bayesLife.mcmc.set` object.
- `sim.dir`: Directory with the MCMC simulation results. Only used if `mcmc` is NULL.
- `burnin`: Burnin.
- `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.
- `par.names.cs`: Names of country-specific parameters for which the Raftery diagnostics should be computed.
- `country.sampling.prop`: Proportion of countries that are included in the diagnostics. It should be between 0 and 1. If it is smaller than 1, the countries are randomly sampled. It is only relevant if `par.names.cs` is not NULL.
- `verbose`: Logical switching log messages on and off.
- `...`: Additional arguments passed to the `e0.coda.list.mcmc` function.

Details

See `tfr.raftery.diag` for details. This function is called from `e0.diagnose`.

Author(s)

Hana Sevcikova, Adrian Raftery

See Also

`tfr.raftery.diag, raftery.diag, e0.diagnose`
Posterior Distribution of Trajectories of Life Expectancy

Description

The functions plot/tabulate the posterior distribution of trajectories of the life expectancy for a given country, or for all countries, including their median and given probability intervals.

Usage

```r
e0.traj.plot(e0.pred, country, pi = c(80, 95), both.sexes = FALSE,
              nr.traj = NULL, adjusted.only = TRUE, typical.trajectory = FALSE,
              xlim = NULL, ylim = NULL, type = "b",
              xlab = "Year", ylab = "Life expectancy at birth", main = NULL,
              lwd = c(2, 2, 2, 2, 1), col = c("black", "green", "red", "red", "#00000020"),
              col2 = c("gray39", "greenyellow", "hotpink", "hotpink", "#00000020"),
              show.legend = TRUE, add = FALSE, ...)
```

```r
e0.traj.plot.all(e0.pred,
                 output.dir = file.path(getwd(), "e0 trajectories"),
                 output.type = "png", verbose = FALSE, ...)
```

```r
e0.traj.table(e0.pred, country, pi = c(80, 95),
              both.sexes = FALSE, ...)
```

Arguments

- **e0.pred**: Object of class `bayesLife.prediction`.
- **country**: Name or numerical code of a country.
- **pi**: Probability interval. It can be a single number or an array. If `both.sexes` is TRUE the default is 95.
- **both.sexes**: Logical or the character "A". If TRUE the distribution of both sexes is plotted into one graphics (or tabulated), provided the `e0.pred` is a female prediction and contains a joint male prediction as a result of running the function `e0.jmale.predict`. For "A" it plots/tabulates the distribution of the average life expectancy over both sexes.
- **nr.traj**: Number of trajectories to be plotted. If NULL, all trajectories are plotted, otherwise they are thinned evenly. If `both.sexes` is TRUE the default is zero.
- **adjusted.only**: Logical. By default, if the projection median is adjusted using e.g. `e0.median.set`, the function plots the adjusted median. If `adjusted.only`=FALSE the original (non-adjusted) median is plotted as well.
- **typical.trajectory**: Logical. If TRUE one trajectory is shown for which the median absolute deviation from the median e0 projection is the median among all the trajectories.
xlim, ylim, type, xlab, ylab, main
Graphical parameters passed to the plot function.

lwd, col, col2  Vector of five elements giving the line width and color for: 1. observed data, 2. imputed missing data, 3. median, 4. quantiles, 5. trajectories. col2 is only used if both.sexes is TRUE. In such a case, col2 is used for female lines and col is used for male lines, which in this case defaults to c('black', 'green', 'darkgreen', 'darkgreen', 'gray').

show.legend  Logical controlling whether the 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.

...  Additional graphical parameters. In addition, for e0_trajectories.plot.all, ... contains any of the arguments of e0_trajectories.plot, and for e0_trajectories.table, ... contains the pi and country arguments.

output.dir  Directory into which resulting graphs are stored.

output.type  Type of the resulting files. It can be “png”, “pdf”, “jpeg”, “bmp”, “tiff”, or “postscript”.

verbose  Logical switching log messages on and off.

Details
e0_trajectories.plot plots posterior distribution of trajectories of life expectancy for a given country. e0_trajectories.table gives the same output in a tabular format.
e0_trajectories.plot.all creates a set of such graphs (one per country) 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 plotted.

Author(s)
Hana Sevcikova

See Also
bayesLife.prediction

Examples

```r
# Not run:
sim.dir <- file.path(find.package("bayesLife"), "ex-data", "bayesLife.output")
pred <- get.e0.prediction(sim.dir)
e0_trajectories.table(pred, country="Japan", pi=c(80, 95))
e0_trajectories.plot(pred, country="Japan", pi=c(80, 95))

# plot multiple countries into one plot
e0_trajectories.plot(pred, "Japan", col=rep("green", 5), nr_traj=0, pi=c(80), show.legend=FALSE, main="")
e0_trajectories.plot(pred, "United States of America", col=rep("blue", 5),
```
Accessing a Convergence Object

Description

The functions load objects of class `bayesLife.convergence` from disk that were created using the function `e0.diagnose`.

Usage

```r
get.e0.convergence(sim.dir = file.path(getwd(), "bayesLife.output"), thin = 225, burnin = 10000)
get.e0.convergence.all(sim.dir = file.path(getwd(), "bayesLife.output"))
```

Arguments

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

Details

Function `get.e0.convergence` loads an object of class `bayesLife.convergence` for the specific `thin` and `burnin`. Function `get.e0.convergence.all` loads all `bayesLife.convergence` objects available in `sim.dir`.

Value

- `get.e0.convergence` returns an object of class `bayesLife.convergence`;
- `get.e0.convergence.all` returns a list of objects of class `bayesLife.convergence`.

Author(s)

Hana Sevcikova

See Also

`e0.diagnose`, `summary.bayesLife.convergence`. 
Description

The function `get.e0.mcmc` retrieves results of an MCMC simulation and creates an object of class `bayesLife.mcmc.set`. Function `has.e0.mcmc` checks the existence of such results. Function `e0.mcmc` extracts a single chain and `e0.mcmc.list` extracts several or all chains from the simulation results.

Usage

```r
get.e0.mcmc(sim.dir = file.path(getwd(), "bayesLife.output"),
    chain.ids = NULL, low.memory = TRUE, burnin = 0, verbose = FALSE)

has.e0.mcmc(sim.dir)

e0.mcmc(mcmc.set, chain.id = 1)

e0.mcmc.list(mcmc.set, chain.ids= NULL)
```

Arguments

- `sim.dir`: Directory where the 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`: If FALSE full MCMC traces are loaded into memory.
- `burnin`: Burnin used for loading traces. Only relevant, if `low.memory` is FALSE.
- `verbose`: Logical switching log messages on and off.
- `chain.id`: Chain identifier.
- `mcmc.set`: Object of class `bayesLife.mcmc.set`.

Value

- `get.e0.mcmc` returns an object of class `bayesLife.mcmc.set`.
- `has.e0.mcmc` returns a logical value.
- `e0.mcmc` returns an object of class `bayesLife.mcmc`, and `e0.mcmc.list` returns a list of `bayesLife.mcmc` objects.

Author(s)

Hana Sevcikova

See Also

- `bayesLife.mcmc.set`
**Examples**

```r
sim.dir <- file.path(find.package("bayesLife"), "ex-data", "bayesLife.output")
m <- get.e0.mcmc(sim.dir)
summary(m)

# summary of the world parameters for the single chains
for(mc in e0.mcmc.list(m)) print(summary(mc, par.names.cs=NULL))
```

---

**get.e0.parameter.traces**

*Accessing MCMC Parameter Traces*

**Description**

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

**Usage**

```r
get.e0.parameter.traces(mcmc.list, par.names = e0.parameter.names(),
                         burnin = 0, thinning.index = NULL, thin = NULL)
get.e0.parameter.traces.cs(mcmc.list, country.obj,
                          par.names = e0.parameter.names.cs(),
                          burnin = 0, thinning.index = NULL, thin = NULL)
```

**Arguments**

- `mcmc.list`: List of `bayesLife.mcmc` objects.
- `country.obj`: Country object list (see `get.country.object`).
- `par.names`: Names of country-independent parameters (in case of `get.e0.parameter.traces`) or country-specific parameters (in case of `get.e0.parameter.traces.cs`) to be included.
- `burnin`: Burnin 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 a sample of length 400, then the value should be `thinning.index=seq(1,1600, length=400)`.
- `thin`: Alternative to `thinning.index`. The above example is equivalent to `thin=4`.

**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.e0.parameter.traces` returns country-independent parameters, `get.e0.parameter.traces.cs` returns country-specific parameters.
Function `get.e0.prediction` retrieves results of a prediction and creates an object of class `bayesLife.prediction`. Function `has.e0.prediction` checks an existence of such results. Analogously, functions `get.e0.jmale.prediction` and `has.e0.jmale.prediction` retrieve and check an existence of male predictions from a given female prediction object.

**Usage**
```
get.e0.prediction(mcmc = NULL, sim.dir = NULL, joint.male = FALSE, mcmc.dir = NULL)
has.e0.prediction(mcmc = NULL, sim.dir = NULL)
get.e0.jmale.prediction(e0.pred)
has.e0.jmale.prediction(e0.pred)
```

**Arguments**
- **mcmc**: Object of class `bayesLife.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. It should correspond to the value of the `output.dir` argument used in the `e0.predict` function. Only relevant if `mcmc` is `NULL`.

**Examples**
```
## Not run:
sim.dir <- file.path(find.package("bayesLife"), "ex-data", "bayesLife.output")
m <- get.e0.mcmc(sim.dir)
e0.values <- get.e0.parameter.traces(m$mcmc.list, burnin=10, par.names="z")
hist(e0.values, main=colnames(e0.values))

e0.values.cs <- get.e0.parameter.traces.cs(m$mcmc.list,
    get.country.object("Canada", meta=m$meta),
    burnin=10, par.names="z.c")
hist(e0.values.cs, main=colnames(e0.values.cs))
## End(Not run)
```
Function for accessing trajectories of the life expectancy.

**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 e0.predict function was set to NULL. If it is not the case, the argument mcmc.dir should be used.

Function get.e0.jmale.predict extracts male projections from the e0.pred objects (which should be a female prediction object), if the male prediction was generated using the e0.jmale.predict function. has.e0.jmale.predict checks if such male prediction was generated.

**Value**

Functions has.e0.prediction and has.e0.jmale. prediction return a logical indicating if a prediction exists.

Functions get.e0.prediction and get.e0.jmale.predict return an object of class bayesLife.prediction.

**Author(s)**

Hana Sevcikova

**See Also**

bayesLife.prediction, e0.predict, summary.bayesLife.prediction, e0.jmale.predict

**Examples**

```r
sim.dir <- file.path(find.package("bayesLife"), "ex-data", "bayesLife.output")
pred <- get.e0.prediction(sim.dir=sim.dir)
summary(pred, country="Canada")
```

---

**Description**

Function for accessing trajectories of the life expectancy.
Usage

get.e0.trajectories(e0.pred, country)

Arguments

e0.pred Object of class bayesLife.prediction.
country Name or numerical code of a country.

Details

The function loads trajectories of life expectancy 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. The row names correspond to the mid-years of the prediction periods.

Author(s)

Hana Sevcikova

See Also

bayesLife.prediction, get.e0.prediction, e0.trajectories.table

Examples

sim.dir <- file.path(find.package("bayesLife"), "ex-data", "bayesLife.output")
pred <- get.e0.prediction(sim.dir=sim.dir)
get.e0.trajectories(pred, "Germany")
get.thinned.e0.mcmc

Arguments

mcmc.set Object of class bayesLife.mcmc.set.
thin, burnin Thinning interval and burnin used for creating or identifying the thinned object.
output.dir Directory for storing the thinned object. By default it is stored into the same
directory as mcmc.set.
verbose Logical switching log messages on and off.

Details

The function create.thinned.e0.mcmc is called from e0.predict and thus, the resulting object
contains exactly the same MCMCs used for generating projections.

The thinning is done as follows: The given burnin is removed from the beginning of each chain
in the original MCMC set. Then each chain is thinned by thin using equal spacing and all chains
are collapsed into one single chain per parameter. They are stored in output.dir under the name
‘thinned_mcmc_t_b’ where t is the value of thin and b the value of burnin.

Value

Both functions return an object of class bayesLife.mcmc.set. get.thinned.e0.mcmc returns
NULL if such object does not exist.

Author(s)

Hana Sevcikova

See Also

bayesLife.mcmc.set, e0.predict

Examples

## Not run:
sim.dir <- tempfile()
m <- run.e0.mcmc(nr.chains=2, iter=60, thin=2, output.dir=sim.dir, verbose=TRUE)
e0.predict(m, burnin=30) # creates thinned MCMCs
mb <- get.thinned.e0.mcmc(m, thin=2, burnin=30)
summary(mb, meta.only=TRUE) # length 30 = 2chains x (60-30)iters./2thin
unlink(sim.dir, recursive=TRUE)

## End(Not run)
**Inclusion Codes**

**Description**

Data sets containing codes that determine which countries are to be included into a simulation or/and projections.

**Usage**

```r
data(include_RP17)
data(include_RP15)
data(include_RP12)
data(include_RP10)
```

**Format**

Data frames containing one record per country or region. It has the following variables:

- `country`: Name of country or region. Not used.
- `include_code`: Entries for which `include_code=2` are included in MCMC simulations (i.e. estimation of the model parameters). Entries for which `include_code` is 1 or 2 are included in the prediction.

**Details**

In a simulation, an `include_*` dataset is selected that corresponds to the given `wpp.year` passed to the function `run.e0.mcmc`. It is merged with a `e0` dataset from the corresponding `wpp` package using the `country_code` column. Thus, the country entries in this dataset should correspond to entries in the `e0` dataset.

The package contains also a dataset called `my.e0_template` (in ‘extdata’ directory) which is a template for user-specified `e0` time series. It has the same structure as the `e0` dataset, except that most of the columns are optional. The only required column is `country_code` (see description of the argument `my.e0.file` in `run.e0.mcmc`).

**Note**

In all three datasets, countries affected by AIDS are not included in the estimation, i.e. the `include_code` is set to 3.

**Source**

Data provided by the United Nations Population Division.
Examples

data(include_2017)
head(include_2017)
# select AIDS countries
subset(include_2017, include_code == 3)

run.e0.mcmc

Running Markov Chain Monte Carlo for Parameters of Life Expectancy

Description

Runs (or continues running) MCMCs for simulating the life expectancy for all countries of the world, using a Bayesian hierarchical model.

Usage

run.e0.mcmc(sex = c("Female", "Male"), nr.chains = 3, iter = 160000,
           output.dir = file.path(getwd(), "bayesLife.output"),
           thin = 10, replace.output = FALSE,
           start.year = 1873, present.year = 2015, wpp.year = 2017,
           my.e0.file = NULL, my.locations.file = NULL, buffer.size = 100,
           a = c(13.215, 41.070, 9.235, 17.605, 2.84, 0.385),
           delta = c(3.844, 4.035, 11.538, 5.639, 0.901, 0.4),
           tau = c(15.5976503, 23.650006, 14.5056919,
                   14.718598, 3.4514285, 0.5667531),
           Triangle.ini = list(NULL, NULL, NULL, NULL), k.ini = NULL,
           z.ini = NULL, lambda.ini = list(NULL, NULL, NULL, NULL),
           lambda.k.ini = NULL, lambda.z.ini = NULL, omega.ini = NULL,
           Triangle.ini.low = c(10, 30, 0.1, 10), Triangle.ini.up = c(30, 50, 10, 30),
           k.ini.low = 3, k.ini.up = 5, z.ini.low = 1e-04, z.ini.up = 0.653,
           lambda.ini.low = c(0.01, 0.01, 0.01, 0.01),
           lambda.ini.up = c(0.1, 0.1, 0.1, 0.1),
           lambda.k.ini.low = 0.3, lambda.k.ini.up = 1,
           lambda.z.ini.low = 1, lambda.z.ini.up = 40,
           omega.ini.low = 0.1, omega.ini.up = 5,
           Triangle.prior.low = c(0, 0, -20, 0), Triangle.prior.up = c(100, 100, 50, 100),
           k.prior.low = 0, k.prior.up = 10, z.prior.low = 0, z.prior.up = 0.653,
           Triangle.c.ini.norm = list(round(Triangle.ini.low + (Triangle.ini.up - Triangle.ini.low)/2),
                                       Triangle.ini.low + (Triangle.ini.up - Triangle.ini.low)/2),
           k.c.ini.norm = c(round(k.ini.low + (k.ini.up - k.ini.low)/2), 2),
           z.c.ini.norm = c(round(z.ini.low + (z.ini.up - z.ini.low)/2), 2),
           Triangle.c.prior.norm = c(0, 0, -20, 0), Triangle.c.prior.up = c(100, 100, 50, 100),
           k.c.prior.norm = c(0, 0, -20, 0), k.c.prior.up = 10, z.c.prior.norm = c(0, 0, -20, 0),
           z.c.prior.up = 0.653,
           country.overwrites = NULL, nu = 4, dl.p1 = 9, dl.p2 = 9,
           sumTriangle.lim = c(30, 86), constant.variance = FALSE,
           outliers = c(-5, 10), seed = NULL,
parallel = FALSE, nr.nodes = nr.chains, compression.type = 'None',
auto.conf = list(max.loops=5, iter=160000, iter.incr=20000,
     nr.chains=3, thin=225, burnin=10000),
verbose = FALSE, verbose.iter = 100, ...)

continue.e0.mcmc(iter, chain.ids = NULL,
output.dir = file.path(getwd(), "bayesLife.output"),
parallel = FALSE, nr.nodes = NULL, auto.conf = NULL,
verbose = FALSE, verbose.iter = 10, ...)

**Arguments**

- **sex**: Sex for which to run the simulation.
- **nr.chains**: Number of MCMC chains to run.
- **iter**: Number of iterations to run in each chain. In addition to a single value, it can have the value ‘auto’ in which case the function runs for the number of iterations given in the auto.conf list (see below), then checks if the MCMCs converged (using the auto.conf settings). If it did not converge, the procedure is repeated until convergence is reached or the number of repetition exceeded auto.conf$maxloops.
- **output.dir**: Directory which the simulation output should be written into.
- **thin**: Thinning interval between consecutive observations to be stored on disk.
- **replace.output**: If TRUE, existing outputs in output.dir will be replaced by results of this simulation.
- **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. The functions loads a package called wpp where \(x\) is the wpp.year and uses the ePJ datasets.
- **my.e0.file**: File name containing user-specified e0 time series for one or more countries. See Details below.
- **my.locations.file**: File name containing user-specified locations. See Details below.
- **buffer.size**: Buffer size (in number of [thinned] iterations) for keeping data in the memory. The smaller the buffer.size the more often will the process access the hard disk and thus, the slower the run. On the other hand, the smaller the buffer.size the less data will be lost in case of failure.
- **a**: A vector of the \(a_1, \ldots, a_6\) parameters, which are the prior means of the world-level parameters \((\Delta_1, \ldots, \Delta_4, k, z)\).
- **delta**: A vector of the \(\delta_1, \ldots, \delta_6\) parameters, which are the prior standard deviations of the world-level parameters \((\Delta_1, \ldots, \Delta_4, k, z)\).
- **tau**: A vector of the \(\tau_1, \ldots, \tau_6\) parameters, which is the square root rate of the prior Gamma distribution of the world-level parameters \((\lambda_1, \ldots, \lambda_4, \lambda_k, \lambda_z)\).
- **Triangle.ini**: List (of length four) of initial values for \(\Delta_1, \ldots, \Delta_4\). Each list item should be of the length nr.chains. If a list item is NULL, the initial values are equally spaced between Triangle.ini.low and Triangle.ini.up. By default, if there is just one chain, the value is the middle point of the interval.
\texttt{k.ini, z.ini}  An array of length \texttt{nr.chains} of initial values for \(k(z)\). By default, the initial values are equally spaced between \texttt{k.ini.low} and \texttt{k.ini.up} (\texttt{z.ini.low} and \texttt{z.ini.up}). If \texttt{nr.chains}=1 and \texttt{k.ini} (\texttt{z.ini}) is NULL, the initial value is the middle point of the interval.

\texttt{lambda.ini}  List (of length four) of initial values for \(\lambda_1, \ldots, \lambda_4\). Each list item should be of the length \texttt{nr.chains}. If a list item is NULL, the initial values are equally spaced between \texttt{lambda.ini.low} and \texttt{lambda.ini.up}. By default, if there is just one chain, the value is the middle point of the interval.

\texttt{lambda.k.ini, lambda.z.ini}  An array of length \texttt{nr.chains} of initial values for \(\lambda_k(\lambda_z)\). By default, the initial values are equally spaced between \texttt{lambda.k.ini.low} and \texttt{lambda.k.ini.up} (\texttt{lambda.z.ini.low} and \texttt{lambda.z.ini.up}). If \texttt{nr.chains}=1 and \texttt{lambda.k.ini} (\texttt{lambda.z.ini}) is NULL, the initial value is the middle point of the interval.

\texttt{omega.ini}  An array of length \texttt{nr.chains} of initial values for \(\omega\). By default, the initial values are equally spaced between \texttt{omega.ini.low} and \texttt{omega.ini.up}. If \texttt{nr.chains}=1 and \texttt{omega.ini} is NULL, the initial value is the middle point of the interval.

\texttt{Triangle.ini.low, Triangle.ini.up}  A vector of length four. They are the lower and upper bounds for initial values of \(\Delta_1, \ldots, \Delta_4\). An \(i\)-th item is only used if \texttt{Triangle.ini[[i]]} is NULL.

\texttt{k.ini.low, k.ini.up}  Single value giving the lower and upper bounds for initial values of \(k\). It is only used if \texttt{k.ini} is NULL.

\texttt{z.ini.low, z.ini.up}  Single value giving the lower and upper bounds for initial values of \(z\). It is only used if \texttt{z.ini} is NULL. Regarding defaults, see Note below.

\texttt{lambda.ini.low, lambda.ini.up}  A vector of length four. They are the lower and upper bounds for initial values of \(\lambda_1, \ldots, \lambda_4\). An \(i\)-th item is only used if \texttt{lambda.ini[[i]]} is NULL.

\texttt{lambda.k.ini.low, lambda.k.ini.up}  Single value giving the lower and upper bounds for initial values of \(\lambda_k\). It is only used if \texttt{lambda.k.ini} is NULL.

\texttt{lambda.z.ini.low, lambda.z.ini.up}  Single value giving the lower and upper bounds for initial values of \(\lambda_z\). It is only used if \texttt{lambda.z.ini} is NULL.

\texttt{omega.ini.low, omega.ini.up}  Single value giving the lower and upper bounds for initial values of \(\omega\). It is only used if \texttt{omega.ini} is NULL.

\texttt{Triangle.prior.low, Triangle.prior.up}  A vector of length four. They are the lower and upper bounds for the prior (truncated normal) distribution of \(\Delta_1, \ldots, \Delta_4\).

\texttt{k.prior.low, k.prior.up}  Single value giving the lower and upper bounds for the prior (truncated normal) distribution of \(k\).

\texttt{z.prior.low, z.prior.up}  Single value giving the lower and upper bounds for the prior (truncated normal) distribution of \(z\). Regarding defaults, see Note below.
Triangle.c.ini.norm
A list with two elements, each of them being a vector of size four. The first and second element in the list corresponds to the means and standard deviation, respectively, for the initial values of the country-specific parameters $\Delta^c_1, \ldots, \Delta^c_4$ which are drawn from a truncated normal distribution with bounds defined by Triangle.c.prior.low and Triangle.c.prior.up.

k.c.ini.norm, z.c.ini.norm
A vector of length two. The first and second element corresponds to the means and standard deviation, respectively, for the initial values of the country-specific parameters $k^c (z^c)$ which are drawn from a normal distribution truncated between k.c.prior.low and k.c.prior.up (z.c.prior.low and z.c.prior.up).

Triangle.c.prior.low, Triangle.c.prior.up
A vector of length four. They are the lower and upper bounds for the prior (truncated normal) distribution of country-specific $\Delta^c_1, \ldots, \Delta^c_4$.

k.c.prior.low, k.c.prior.up
Single value giving the lower and upper bounds for the prior (truncated normal) distribution of country-specific $k^c$.

z.c.prior.low, z.c.prior.up
Single value giving the lower and upper bounds for the prior (truncated normal) distribution of country-specific $z^c$. Regarding defaults, see Note below.

country.overwrites
This argument allows to overwrite some of the prior parameters for specific countries. If it is not NULL it should be a data frame with an obligatory column ‘country_code’. Each row then corresponds to one country. Other columns can be ‘k.c.prior.low’, ‘k.c.prior.up’, ‘z.c.prior.low’, ‘z.c.prior.up’, ‘Triangle_1.c.prior.low’ and ‘Triangle_4.c.prior.up’ where $x$ can be an integer from 1 to 4.

nu
The shape parameter of the Gamma distributions of all $\lambda$ parameters is $\nu/2$.

dl.p1, dl.p2
Values of the parameters $p_1$ and $p_2$ of the double logistic function.

sumTriangle.lim
Lower and upper limits for the sum of the $\Delta_i$ parameters. MCMC proposals that are outside of this limit are rejected. It is applied to both, the world parameters as well as the country specific parameters.

constant.variance
Logical indicating if the model should be estimated using constant variance.

outliers
Ranges for determining outliers in the historical data. If outliers = $(x, y)$ then any increase in life expectancy smaller than $x$ or larger than $y$ is considered as an outlier and removed from the estimation.

seed
Seed of the random number generator. If NULL no seed is set. It can be used to generate reproducible results.

parallel
Logical determining if the simulation should run multiple chains in parallel. If it is TRUE, the package snowFT is required.

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.

compression.type
auto.conf List containing a configuration for an ‘automatic’ run (see description of argument iter). Item iter gives the number of iterations in the first chunk of the MCMC simulation; item iter.incr gives the number of iterations in the following chunks; nr.chains gives the number of chains in all chunks of the MCMC simulation; items thin and burnin are used in the convergence diagnostics following each chunk; max.loops controls the maximum number of chunks. All items must be integer values. This argument is only used if the function argument iter is set to ‘auto’.

verbose Logical switching log messages on and off.

verbose.iter Integer determining how often (in number of iterations) log messages are outputted during the estimation.

... Additional parameters to be passed to the function performParallel, if parallel is TRUE.

chain.ids Array of chain identifiers that should be resumed. If it is NULL, all existing chains in output.dir are resumed.

Details

The function run.e0.mcmc creates an object of class bayesLife.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 (possibly compressed) ASCII files in a subdirectory of output.dir, called mcx where x is the identifier of that chain. There is one file per parameter, named after the parameter with the suffix “.txt”, possibly followed by a compression suffix if compression.type is given. Country-specific parameters have the suffix _countryc where c is the country code. In addition to the trace files, each mcx directory contains the object bayesLife.mcmc in binary format. All chain-specific files are written into disk after the first, last and each buffer.size-th (thinned) iteration.

Using the function continue.e0.mcmc one can continue simulating an existing MCMCs by iter iterations for either all or selected chains.

The function loads observed data (further denoted as WPP dataset), depending on the specified sex, from the e0F (e0M) and e0F_supplemental (e0M_supplemental) datasets in a wppx package where x is the wpp.year. It is then merged with the include dataset that corresponds to the same wpp.year. The argument my.e0.file can be used to overwrite those default data. Such a file can include a subset of countries contained in the WPP dataset, as well as a set of new countries. In the former case, the function replaces the corresponding country data from the WPP dataset with values in this file. Only columns are replaced that match column names of the WPP dataset, and in addition, columns ‘last.observed’ and ‘include_code’ are used, if present. Countries are merged with WPP using the column ‘country_code’. In addition, in order the countries to be included in the simulation, in both cases (whether they are included in the WPP dataset or not), they must be contained in the table of locations (UNlocations). In addition, their corresponding ‘include_code’ must be set to 2. If the column ‘include_code’ is present in my.e0.file, its value overwrites the default include code, unless is -1.

The default UN table of locations mentioned above can be overwritten/extended by using a file passed as the my.locations.file argument. Such a file must have the same structure as the UNlocations dataset. Entries in this file will overwrite corresponding entries in UNlocations matched by the column ‘country_code’. If there is no such entry in the default dataset, it will be appended. This option of appending new locations is especially useful in cases when my.e0.file
contains new countries/regions that are not included in UNlocations. In such a case, one must provide a my.locations.file with a definition of those countries/regions.

For simulation of the hyperparameters of the Bayesian hierarchical model, all countries are used that are included in the WPP dataset, possibly complemented by the my.e0.file, that have include_code equal to 2. The hyperparameters are used to simulate country-specific parameters, which is done for all countries with include_code equal 1 or 2. The following values of include_code in my.e0.file are recognized: -1 (do not overwrite the default include code), 0 (ignore), 1 (include in prediction but not estimation), 2 (include in both, estimation and prediction). Thus, the set of countries included in the estimation and prediction can be fully specified by the user.

Optionally, my.e0.file can contain a column called last.observed containing the year of the last observation for each country. In such a case, the code would ignore any data after that time point. Furthermore, the function e0.predict fills in the missing values using the median of the BHM procedure (stored in e0.matrix.reconstructed of the bayesLife.prediction object). For last.observed values that are below a middle year of a time interval \([t_i, t_{i+1}]\) (computed as \(t_i + 3\)) the last valid data point is the time interval \([t_{i-1}, t_i]\), whereas for values larger equal a middle year, the data point in \([t_i, t_{i+1}]\) is valid.

The package contains a dataset called ‘my_e0_template’ (in ‘extdata’ directory) which is a template for user-specified my.e0.file.

Value

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

- meta. An object of class bayesLife.mcmc.meta.
- mcmc.list. A list of objects of class bayesLife.mcmc, one for each MCMC.

Note

Parameter \(z\) determines the asymptote in gains in life expectancy. The following text gives an explanation for the choice of upper limits on \(z\)-related parameters:

The pace of improvement and the asymptotic limit in future gains in female life expectancy vary for each projected trajectory, but ultimately is informed and constrained by the finding that the rate of increase of maximum female life expectancy over the past 150 year has been highly linear (2a, 2b) (i.e., about 2.4 years per decade), albeit at slightly lower pace once the leading countries started to exceed 75 years of female life expectancy at birth in the 1960s (3) (about 2.26 years of gains per decade). By assuming that the asymptotic average rate of increase in life expectancy is nonnegative, life expectancy is assumed to continually increase (on average), and no limit is imposed to life expectancy in the foreseeable future. The increase in maximum female life span among countries with highest life expectancy and reliable data on very old age provide further guidance on future rate of progress which has also been increasingly linear at least since the 1970s (4a-4c) (about 1.25 years per decade for countries like Sweden and Norway), and is used to inform the asymptotic average rate of increase in female life expectancy used in the 2012 WPP Revision. To set the posterior median to an annual gain of 0.125 year (or 5-year gain of 0.625 in this context) the upper bound value of 0.653 is used for the world prior \((z)\) and country-specific prior \((z_c)\) as default values in the estimation of the double-logistic parameters.
Author(s)

Hana Sevcikova, Patrick Gerland contributed to the documentation.

References


See Also

get.e0.mcmc, summary.bayesLife.mcmc.set.

Examples

```r
## Not run:
m <- run.e0.mcmc(nr.chains=1, iter=5, thin=1, verbose=TRUE)
summary(m)
m <- continue.e0.mcmc(iter=5, verbose=TRUE)
summary(m)
## End(Not run)
```

run.e0.mcmc.extra Run MCMC for Extra Countries, Areas or Regions

Description

Run MCMC for extra countries, areas or regions. It uses the posterior distribution of model hyper-parameters from an existing simulation to generate country-specific parameters.
Usage

```r
run.e0.mcmc.extra(sim.dir = file.path(getwd(), "bayesLife.output"),
  countries = NULL, my.e0.file = NULL,
  iter = NULL, thin = 1, burnin = 2000, country.overwrites = NULL,
  parallel = FALSE, nr.nodes = NULL, my.locations.file = NULL,
  verbose = FALSE, verbose.iter = 100, ...)
```

Arguments

- **sim.dir**: Directory with an existing simulation.
- **countries**: Vector of country codes. These include codes of areas and regions (see column `country_code` in `UNlocations`).
- **my.e0.file**: File name containing user-specified time series of life expectancy for countries for which the simulation should run (see Details below).
- **iter**: Number of iterations to be used for sampling from the posterior distribution of the hyperparameters. By default, the number of (possibly thinned) iterations used in the existing simulation is taken.
- **thin**: Thinning interval for sampling from the posterior distribution of the hyperparameters.
- **burnin**: Number of iterations discarded before sampling from the posterior distribution of the hyperparameters.
- **country.overwrites**: This argument allows to overwrite some of the prior parameters for specific countries. If it is not `null` it should be a data frame with an obligatory column `country_code` in `UNlocations`.
- **parallel**: Logical determining if the simulation should run multiple chains 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 contained in the existing simulation.
- **my.locations.file**: File name containing user-specified locations. See Details below.
- **verbose**: Logical switching log messages on and off.
- **verbose.iter**: Integer determining how often (in number of iterations) log messages are outputted during the estimation.
- **...**: Additional parameters to be passed to the function `performParallel`, if `parallel` is `TRUE`.

Details

The function can be used to make predictions for countries, areas or regions (further denoted as ‘countries’) that were not included in the MCMC estimation (invoked by `run.e0.mcmc`). It creates
MCMC traces for country-specific parameters. The purpose of this function is to have country-specific parameters available in order to be able to generate projections for additional countries or their aggregations, without having to re-run the often time-expensive MCMC simulation.

The set of countries to be considered by this function can be given either by their codes, using the argument `countries`, in which case the countries must be included in the UN WPP e0 dataset. Or, it can be given by a user-specific file, using the argument `my.e0.file`. The function considers a union of both arguments. The function will ignore all countries that were used in the existing MCMC simulation for estimating the hyperparameters. Countries that already own country-specific parameters (e.g. because they were included in `my.e0.file` passed to `run.e0.mcmc`) get their parameters recomputed. Note that all countries must be included in the `unlocations` dataset, but unlike in `run.e0.mcmc`, their `include_code` is ignored. As in the case of `run.e0.mcmc`, the default dataset of locations `UNlocations` can be overwritten using a file of the same structure as `UNlocations` passed via the `my.locations.file` argument. This file should be especially used, if e0 is simulated for new locations that are not included in `UNlocations`.

Value

An object of class `bayesLife.mcmc.set`.

Note

If there is an existing projection for the directory `sim.dir`, use `e0.predict.extra` to obtain projections for the extra countries used in this function.

Author(s)

Hana Sevcikova

See Also

`run.e0.mcmc`, `e0.predict.extra`

Examples

```r
## Not run:
m <- run.e0.mcmc(nr.chains=1, iter=20, thin=1, verbose=TRUE)
m <- run.e0.mcmc.extra(countries=c(908,924), burnin=10, verbose=TRUE)
summary(m, country=924)
pred <- e0.predict(burnin=10, verbose=TRUE)
summary(pred, country=908)
## End(Not run)
```
Summary of a Life Expectancy Convergence Object

Description

Summary of an object of class `bayesLife.convergence` created using the `e0.diagnose` function. It gives an overview about parameters that did not converge.

Usage

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

Arguments

- `object`: Object of class `bayesLife.convergence`.
- `expand`: By default, the function does not show parameters for each country for which there was no convergence, if the status is ‘red’. This argument can switch that option on.
- `...`: Not used.

Author(s)

Hana Sevcikova

See Also

e0.diagnose

---

Summary Statistics for Life Expectancy MCMCs

Description

Summary of an object `bayesLife.mcmc.set` or `bayesLife.mcmc`, computed via `run.e0.mcmc`. It can be obtained either for all countries or for a specific country, and either for all parameters or for specific parameters. The function uses the `summary.mcmc` function of the `coda` package.
Usage

```r
## S3 method for class 'bayesLife.mcmc.set'
summary(object, country = NULL, chain.id = NULL,
         par.names = e0.parameter.names(),
         par.names.cs = e0.parameter.names.cs(),
         meta.only = FALSE, thin = 1, burnin = 0, ...)

## S3 method for class 'bayesLife.mcmc'
summary(object, country = NULL,
         par.names = e0.parameter.names(),
         par.names.cs = e0.parameter.names.cs(), ...)
```

Arguments

- `object`: Object of class `bayesLife.mcmc.set` or `bayesLife.mcmc`
- `country`: Country name or code if a country-specific summary is desired.
- `chain.id`: Identifiers of MCMC chains. By default, all chains are considered.
- `par.names`: Country independent parameters to be included in the summary.
- `par.names.cs`: Country-specific parameters to be included in the summary.
- `meta.only`: Logical. If it is `TRUE`, only meta information of the simulation is included.
- `thin`: Thinning interval. Only used if larger than the `thin` argument used in `run.e0.mcmc`.
- `burnin`: Number of iterations to be discarded from the beginning of each chain before computing the summary.
- `...`: For `summary.bayesLife.mcmc`, arguments `thin` and `burnin` can be passed here. In addition, both functions accept arguments passed to the `summary.mcmc` function of the `coda` package.

Author(s)

Hana Sevcikova

See Also

- `bayesLife.mcmc.set, summary.mcmc`

Examples

```r
sim.dir <- file.path(find.package("bayesLife"), "ex-data", "bayesLife.output")
m <- get.e0.mcmc(sim.dir)
summary(m, country="Czechia", burnin=20)
```
**summary.bayesLife.prediction**

*Summary of a Prediction of the Life Expectancy*

**Description**

Country-specific summary of an object of class `bayesLife.prediction`, created using the function `e0.predict`. The summary contains the mean, standard deviation and several commonly used quantiles of the simulated trajectories.

**Usage**

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

**Arguments**

- `object` Object of class `bayesLife.prediction`.
- `country` Country name or code.
- `compact` Logical switching between a smaller and larger number of displayed quantiles.
- `...` Not used.

**Author(s)**

Hana Sevcikova

**See Also**

`bayesLife.prediction`

**Examples**

```r
sim.dir <- file.path(find.package("bayesLife"), "ex-data", "bayesLife.output")
pred <- get.e0.predict(sim.dir=sim.dir)
summary(pred, country="Iceland")
```
write.e0.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

write.e0.projection.summary(dir = file.path(getwd(), "bayeslife.output"),
output.dir = NULL, revision = NULL, adjusted = FALSE)

Arguments

dir Directory containing the prediction object. It should correspond to the output.dir argument of the e0.predict function.
output.dir Directory in which the resulting file will be stored. If NULL the same directory is used as for the prediction.
revision UN revision number. If NULL it is determined from the corresponding WPP year: WPP 2008 corresponds to revision 13, every subsequent WPP increases the revision number by one. Used as a constant in the second file only.
adjusted Logical. By default the function writes summary using the original BHM projections. If the projection medians are adjusted (using e.g. e0.median.set), setting this argument to TRUE causes writing the adjusted projections.

Details

The first file that the function creates is called 'projection_summary_user_friendly.csv', it is a comma-separated table with the following columns:

- "country_name": country name
- "country_code": country code
- "variant": name of the variant, such as "median", "lower 80", "upper 80", "lower 95", "upper 95", "constant"
- period1: e.g. "2010-2015": life expectancy for the first time period
- period2: e.g. "2015-2020": life expectancy for the second time period
- ... further columns with life expectancy projections

The second file, called 'projection_summary.csv', also comma-separated table, contains the same information as above in a UN-specific format:

- "RevID": revision number, passed to the function as an argument;
• “VarID”: variant identifier, extracted from the `UN_variants` dataset in the `bayesTFR` package;
• “LocID”: country code;
• “TimeID”: time identifier, extracted from the `UN_time` dataset in the `bayesTFR` package;
• “e0”: the life expectancy for this variant, location and time period.

If the simulation directory contains joint male predictions, summary files for those are created as well. In such a case, if `output.dir` is given, separate subdirectories for female and male are created.

**Note**

This function is automatically called from the `e0.predict` and `e0.jmale.predict` functions, therefore in standard cases it will not be needed to call it directly.

**Author(s)**

Hana Sevcikova

**See Also**

`convert.e0.trajectories`, `e0.predict`
Index

*Topic IO
  convert.e0.trajectories, 7
  write.e0.projection.summary, 57
*Topic classes
  bayesLife.mcmc, 4
  bayesLife.mcmc.meta, 6
*Topic datasets
  include, 44
*Topic distribution
  e0.predict, 29
  e0.predict.extra, 32
  run.e0.mcmc, 45
  run.e0.mcmc.extra, 51
*Topic hplot
  e0.DLcurve.plot, 13
  e0.gap.plot, 15
  e0.joint.plot, 21
  e0.map, 22
  e0.pardensity.plot, 27
  e0.partraces.plot, 28
  e0.trajectories.plot, 35
*Topic htest
  e0.diagnose, 10
  e0.dl.coverage, 12
*Topic manip
  convert.e0.trajectories, 7
  e0.coda.list.mcmc, 9
  e0.median.set, 24
  e0.parameter.names, 26
  get.e0.convergence, 37
  get.e0.parameter.traces, 39
  get.e0.prediction, 40
  get.e0.trajectories, 41
  get.thinned.e0.mcmc, 42
*Topic models
  e0.jmale.estimate, 16
  e0.jmale.predict, 19
*Topic multivariate
  e0.predict, 29
  e0.predict.extra, 32
  run.e0.mcmc, 45
  run.e0.mcmc.extra, 51
*Topic package
  bayesLife-package, 2
*Topic print
  summary.bayesLife.convergence, 54
  summary.bayesLife.prediction, 56
*Topic programming
  get.e0.mcmc, 38
  get.thinned.e0.mcmc, 42
*Topic univar
  summary.bayesLife.mcmc.set, 54
  summary.bayesLife.prediction, 56

bayesLife (bayesLife-package), 2
bayesLife-package, 2
bayesLife.convergence, 37, 54
bayesLife.convergence (e0.diagnose), 10
bayesLife.mcmc, 4, 9, 14, 27, 38, 39, 49, 50,
  54, 55
bayesLife.mcmc.meta, 4, 5, 6, 32, 49, 50
bayesLife.mcmc.set, 4–6, 9, 12, 14, 17, 18,
  27, 30, 32, 38, 40, 43, 53–55
bayesLife.mcmc.set (run.e0.mcmc), 45
bayesLife.prediction, 9, 14–16, 19–21, 23,
  25, 27, 28, 33, 35, 36, 40–42, 50, 56
bayesLife.prediction (e0.predict), 29
bayesTFR, 4
coda.mcmc.bayesLife.mcmc
  (e0.coda.list.mcmc), 9
  continue.e0.mcmc, 3
  continue.e0.mcmc (run.e0.mcmc), 45
  convert.e0.trajectories, 7, 31, 32, 58
  create.thinned.e0.mcmc, 11, 12, 31, 32
  create.thinned.e0.mcmc
    (get.thinned.e0.mcmc), 42
  e0, 44, 53
e0.coda.list.mcmc, 3, 9, 29, 34, 40
e0.country.dlcures (e0.DLcurve.plot), 13
e0.diagnose, 3, 10, 30, 34, 37, 54
e0.dl.coverage, 3, 12
e0.DLcurve.plot, 3, 13, 13
e0.gap.plot, 15, 20, 22
e0.jmale.estimate, 16, 16, 19, 20, 32
e0.jmale.predict, 3, 15, 16, 18, 19, 22, 25, 30–32, 35, 41, 58
e0.joint.plot, 16, 21
e0.map, 3, 22
e0.mcmc (getNe0.mcmc), 38
e0.median.reset (e0.median.set), 24
e0.median.set, 24, 35, 57
e0.median.shift (e0.median.set), 24
e0.parameter.names, 26
e0.parameter.names.cs.extended, 23
e0.pardensity.cs.plot, 3
e0.pardensity.cs.plot (e0.pardensity.plot), 27
e0.pardensity.plot, 3, 27
e0.parDL.plot(e0.DLcurve.plot), 13
e0.partraces.cs.plot, 3
e0.partraces.cs.plot (e0.partraces.plot), 28
e0.partraces.plot, 3, 10, 28, 28
e0.predict, 3, 8, 14, 20, 28, 29, 33, 40, 41, 43, 50, 56–58
e0.predict.extra, 32, 53
e0.raftery.diag, 12, 12, 34
e0.trajecories.plot, 3, 22, 35
e0.trajecories.plot, 3, 42
e0.trajecories.table, 3, 35
e0.trajecories.table (e0.trajecories.plot), 35
e0.world.dlcures (e0.DLcurve.plot), 13
e0F_supplemental, 49
0M, 18, 20, 49
0M_supplemental, 49
get.country.object, 39
get.e0.convergence, 11, 12, 37
get.e0.jmale.prediction, 16, 20
get.e0.jmale.predict
(get.e0.predict), 40
get.e0.map.parameters(e0.map), 22
gte0.mcmc, 3–5, 7, 38, 51
gte0.parameter.traces, 29, 39
gte0.prediction, 3, 32, 40, 42
gte0.trajecories, 41
gte0.thinned.e0.mcmc, 42
has.e0.jmale.prediction
(get.e0.predict), 40
has.e0.mcmc (get.e0.mcmc), 38
has.e0.predict (get.e0.predict), 40
has.mcmc.converged, 11
include, 44, 49
include_2010 (include), 44
include_2012 (include), 44
include_2015 (include), 44
include_2017 (include), 44
lines, 22
mapCountryData, 23
mcmc, 10
performParallel, 49, 52
plot, 21
print.summary.bayesLife.mcmc.set
(summary.bayesLife.mcmc.set), 54
print.summary.bayesLife.prediction
(summary.bayesLife.prediction), 56
raftery.diag, 12, 34
run.e0.mcmc, 3, 5–7, 9, 17, 19, 29–32, 44, 45, 52–55
run.e0.mcmc.extra, 32, 33, 51
snowF, 48
summary.bayesLife.convergence, 12, 37, 54
summary.bayesLife.mcmc
(summary.bayesLife.mcmc.set), 54
summary.bayesLife.mcmc.set, 3, 10, 51, 54
summary.bayesLife.prediction, 3, 32, 41, 56
summary.mcmc, 54, 55
tfr.dl.coverage, 12
tfr.map, 23, 24
tfr.map.gvis, 23
INDEX

tfr.raftery.diag, 34

tlm, 17, 18

UN_time, 58
UN_variants, 58
UNlocations, 17, 19, 49, 50, 52, 53

write.e0.projection.summary, 8, 31, 57