Package 'bayesTFR'

April 15, 2017

Version 6.0-0

Date 2017-4-14

Title Bayesian Fertility Projection

Author Hana Sevcikova (hanas@uw.edu), Leontine Alkema (alkema@nus.edu.sg), Adrian Raftery (raftery@uw.edu), Bailey Fosdick (bfosdick@uw.edu), Patrick Gerland (gerland@un.org)

Maintainer Hana Sevcikova <hanas@uw.edu>

Depends R (>= 2.14.2)

Imports mvtnorm, MASS, coda, wpp2015, graphics, grDevices, stats, utils

Suggests rworldmap, snowFT, googleVis, rgdal, rworldxtra, sp, wpp2012, wpp2010

Description Making probabilistic projections of total fertility rate for all countries of the world, using a Bayesian hierarchical model.

License GPL (>= 2)

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

NeedsCompilation yes

Repository CRAN

Date/Publication 2017-04-15 05:15:17 UTC

R topics documented:

`bayesTFR-package` *Bayesian Fertility Projection*

Description

Collection of functions for making probabilistic projections of total fertility rate (TFR) for all countries of the world, using a Bayesian hierarchical model (BHM) and the United Nations demographic time series. Functions for subnational projections are also available.

Details

Package:	bayesTFR
Version:	6.0-0
Date:	2017-4-14
License:	GPL (>= 2)
URL:	https://bayespop.csss.washington.edu

The projection follows a method developed by Alkema et al. (2011) and Raftery et al (2013). It uses historical data provided by the United Nations to simulate a posterior distribution of total fertility rates for all countries in the world simultaneously.

The estimation is split into two parts:

1. BHM for fertility in a transition phase (Phase II), as described in Alkema et al. (2011),
2. BHM for fertility in a post-transition phase (Phase III), as described in Raftery et al (2013).

The second part is optional and can be replaced by a simple AR(1) process.

The main functions of the package are:

- run.tfr.mcmc: Evokes running a Markov Chain Monte Carlo (MCMC) simulation for TFR in Phase II using 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.tfr.mcmc.
- run.tfr3.mcmc: Starts MCMCs for TFR in Phase III. Existing simulation runs can be resumed using continue.tfr3.mcmc.
- tfr.predict: Using the posterior parameter samples it derives posterior trajectories of the total fertility rate for all countries.
- run.tfr.mcmc.extra: Runs MCMC for extra countries or regions, i.e. for countries not included in the Bayesian hierarchical model. It can be also used for aggregations.
- tfr.predict.extra: Generates predictions for extra countries or aggregated regions.

The order of the functions above roughly corresponds to a typical workflow when using the package: 1. run a Phase II MCMC simulation, 2. run a Phase III MCMC simulation (optional but recommended), 3. generate predictions, 4. analyze results (using the functions below). If there are countries that were not included in steps 1.-3., or if there are aggregated regions for which a prediction is desired, one proceeds with the two functions at the bottom of the list above, followed by the analyzing functions below.

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

- tfr.trajectories.plot: Shows the posterior trajectories for a given country, including their median and given probability intervals.
- tfr.trajectories.table: Shows the posterior trajectories for a given country in a tabular form.
- tfr.map: Shows a TFR world map for a given projection period.
- DLcurve.plot: Shows the posterior curves of the double logistic function used in the simulation of PhaseII, including their median and given probability intervals.

- tfr.partraces.plot and tfr.partraces.cs.plot: Plot the Phase II MCMC traces of country-independent parameters and country-specific parameters, respectively. tfr3.partraces.plot and tfr3.partraces.cs.plot do the same for Phase III MCMCs.
- tfr.pardensity.plot and tfr.pardensity.cs.plot: Plot the posterior density of the Phase II MCMCs for country-independent parameters and country-specific parameters, respectively. tfr3.pardensity.plot and tfr3.pardensity.cs.plot do the same for Phase III MCMCs.
- summary.bayesTFR.mcmc.set: Summary function for the MCMC results.
- summary.bayesTFR.prediction: Summary function for the prediction results.

For MCMC diagnostics, functions coda.list.mcmc and coda.list.mcmc3 create an object of type "mcmc.list" that can be used with the **coda** package. Furthermore, function tfr.diagnose and tfr3.diagnose analyze the MCMCs using the Raftery diagnostics implemented in the **coda** package and gives information about parameters that did not converge.

Existing simulation results can be accessed using the get.tfr.mcmc (Phase II) and get.tfr3.mcmc (Phase III) functions. An existing prediction can be accessed via get.tfr.prediction. Existing convergence diagnostics can be accessed using the get.tfr.convergence, get.tfr.convergence.all, get.tfr3.convergence and get.tfr3.convergence.all functions.

The historical national TFR data are taken from one of the packages **wpp2015** (default), **wpp2012** or **wpp2010**, depending on users settings.

Subnational TFR projections can be generated using tfr.predict.subnat. In this case, historical data must be provided by the user. Existing projections can be accessed from disk via get.regtfr.prediction.

Note

There is a directory `ex-data` 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 <hanas@uw.edu>, Leontine Alkema <alkema@nus.edu.sg>, Adrian Raftery <raftery@uw.edu>, Bailey Fosdick <bfosdick@uw.edu>, Patrick Gerland (gerland@un.org)

Maintainer: Hana Sevcikova <hanas@uw.edu>

References

Hana Sevcikova, Leontine Alkema, Adrian E. Raftery (2011). bayesTFR: An R Package for Probabilistic Projections of the Total Fertility Rate. Journal of Statistical Software, 43(1), 1-29. http://www.jstatsoft.org/v43/i01/.

L. Alkema, A. E. Raftery, P. Gerland, S. J. Clark, F. Pelletier, Buettner, T., Heilig, G.K. (2011). Probabilistic Projections of the Total Fertility Rate for All Countries. Demography, Vol. 48, 815-839.

Raftery, A.E., Alkema, L. and Gerland, P. (2014). Bayesian Population Projections for the United Nations. Statistical Science, Vol. 29, 58-68.

Examples

```
## Not run:
# This command produces output data such as in the directory ex-data
sim.dir <- tempfile()
# Phase II MCMCs
m <- run.tfr.mcmc(nr.chains=1, iter=60, output.dir=sim.dir, seed=1, verbose=TRUE)
# Phase III MCMCs (not included in the package)
m3 <- run.tfr3.mcmc(sim.dir=sim.dir, nr.chains=2, iter=100, thin=1, seed=1, verbose=TRUE)
# Prediction
pred <- tfr.predict(m, burnin=30, burnin3=50, verbose=TRUE)
summary(pred, country='Ghana')
unlink(sim.dir, recursive=TRUE)

## End(Not run)
```

---

bayesTFR.mcmc *MCMC Simulation Object*

Description

MCMC simulation object `bayesTFR.mcmc` containing information about one MCMC chain, either from Phase II or Phase III simulation. A set of such objects belonging to the same simulation together with a `bayesTFR.mcmc.meta` object constitute a `bayesTFR.mcmc.set` object.

Details

An object `bayesTFR.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.tfr.mcmc(...)` (Phase II) or `get.tfr3.mcmc(...)` (Phase III), and `tfr.mcmc(...)`.

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

Value

A `bayesTFR.mcmc` object contains parameters of the Bayesian hierarchical model, more specifically, their values from the last iteration. If it is a **Phase II** object these parameters are:
`psi, chi, a_sd, b_sd, const_sd, S_sd, sigma0, mean_eps_tau, sd_eps_tau, Triangle4` - non-country specific parameters, containing one value each.
`alpha, delta` - non-country specific parameters, containing three values each.
`U_c, d_c, Triangle_c4` - country-specific parameters (1d array).
`gamma_ci` - country-specific parameter with three values for each country, i.e. an $n \times 3$ matrix where n is the number of countries.
Phase III MCMC objects contain single-value parameters `mu`, `rho`, `sigma.mu`, `sigma.rho`, `sigma.eps` and n-size vectors `mu.c` and `rho.c`.
Furthermore, the object (independent of Phase) 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 `bayesTFR.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.tfr.mcmc(..., low.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 interation.
`compression.type`	Type of compression of the underlying files.
`meta`	Object of class `bayesTFR.mcmc.meta` used for simulation of this chain.

Author(s)

Hana Sevcikova

See Also

`run.tfr.mcmc`, `get.tfr.mcmc`, `run.tfr3.mcmc`, `get.tfr3.mcmc`, `bayesTFR.mcmc.set`, `bayesTFR.mcmc.meta`

Examples

```
sim.dir <- file.path(find.package("bayesTFR"), "ex-data", "bayesTFR.output")
# loads traces from one chain
m <- get.tfr.mcmc(sim.dir, low.memory=FALSE, burnin=35, chain.ids=1)
# should have 25 rows, since 60 iterations in total minus 35 burnin
dim(tfr.mcmc(m, 1)$traces)
summary(m, chain.id=1)
```

---

bayesTFR.mcmc.meta *MCMC Simulation Meta Object*

Description

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

Details

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

Value

A `bayesTFR.mcmc.meta` object contains various components that correspond to the input arguments of the `run.tfr.mcmc` and `run.tfr3.mcmc` functions. Furthermore, it contains components:

`nr.chains`	Number of MCMC chains.
`phase`	Value 2 or 3, depending which Phase the object belongs to.
`output.dir`	Directory for storing simulation output.

Value - Phase II

Furthermore, Phase II meta objects contain components:

tfr_matrix_all A $q \times n$ matrix with the United Nations TFR estimates. q is number of years (see `T_end` below), 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 $n_e <= n$.

tfr_matrix_observed Like `tfr_matrix_all`, but it has `NA` values for years where no historical data is available (i.e. after the last observed time period).

tfr_matrix Like `tfr_matrix_observed`, but it has `NA` values before and after country"s fertility transition.

nr_countries Number of countries included in the tfr matrices.

nr_countries_estimation Number of countries included in the MCMC estimation. It must be smaller or equal to `nr_countries`.

tau_c Estimated start year of the fertility decline for each country (as a row index within the tfr matrices). -1 means that the decline started before `start.year`.

id_Tistau Index of countries for which `present.year` is equal to `tau_c`.

id_DL Index of countries for which the projection is made using the double logistic function, i.e. high fertility countries.

id_early Index of countries with early decline, i.e. countries for which `tau_c=-1`.

id_notearly Index of countries with not early decline.

lambda_c Start period of the recovery phase for each country (as an index of the `tfr_matrix`).

start_c Maximum of `tau_c` and 1 for each country. Thus, it is a row index of the `tfr_matrix` where the fertility decline starts.

proposal_cov_gammas_cii Proposal covariance matrices of γ_{ci} for each country.

T_end Number of years for which United Nations historical data are available (i.e. number of rows of `tfr_matrix`).

T_end_c Like `T_end` but country specific.

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 `bayesTFR.mcmc.meta` object are derived from this component.

Value - Phase III

Phase III meta objects contain additional components:

id_phase3 Indices of countries included in the Phase III estimation. It is relative to the order of countries in the `region` object in the parent meta object.

nr.countries Number of countries included in the estimation.

parent Link to the Phase II meta object.

Author(s)

Hana Sevcikova, Leontine Alkema

See Also

`run.tfr.mcmc`, `get.tfr.mcmc`, `run.tfr3.mcmc`, `get.tfr3.mcmc`

Examples

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

---

coda.list.mcmc *Convertion to coda's Objects*

Description

The functions convert MCMC traces (simulated using `run.tfr.mcmc` and `run.tfr3.mcmc`) into objects that can be used with the **coda** package.

Usage

```
coda.list.mcmc(mcmc = NULL, country = NULL, chain.ids = NULL,
    sim.dir = file.path(getwd(), "bayesTFR.output"),
    par.names = tfr.parameter.names(),
    par.names.cs = tfr.parameter.names.cs(),
    rm.const.pars = FALSE, burnin = 0,
    low.memory = FALSE, ...)

coda.list.mcmc3(mcmc = NULL, country = NULL, chain.ids = NULL,
    sim.dir = file.path(getwd(), "bayesTFR.output"),
    par.names = tfr3.parameter.names(),
    par.names.cs = tfr3.parameter.names.cs(),
    burnin = 0, low.memory = FALSE, ...)

## S3 method for class 'bayesTFR.mcmc'
coda.mcmc(mcmc, country = NULL,
    par.names = NULL, par.names.cs = NULL,
    burnin = 0, thin = 1, ...)
```

Arguments

mcmc	In coda.mcmc, it is an object of class bayesTFR.mcmc. In coda.list.mcmc and coda.list.mcmc3, it is either a list of bayesTFR.mcmc objects, or an object of class bayesTFR.mcmc.set or in case of coda.list.mcmc it can be bayesTFR.prediction. If it is NULL, the MCMCs are loaded from sim.dir. Either mcmc or sim.dir must be given.
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.
sim.dir	Directory with the MCMC simulation results. Only used if mcmc.list is NULL.
par.names	Names of country-independent parameters to be included. In coda.mcmc the default names are tfr.parameter.names() if the mcmc object is an MCMC of phase II or tfr3.parameter.names() if the MCMC is of phase III.
par.names.cs	Names of country-specific parameters to be included. The argument country is used to filter out traces that correspond to a specific country. If country is not given, for each parameter, traces for all countries are included. In coda.mcmc the default names are tfr.parameter.names.cs() if the mcmc object is an MCMC of phase II or tfr3.parameter.names.cs() if the MCMC is of phase III.
rm.const.pars	Logical indicating if parameters with constant values should be removed.
burnin	Burnin indicating how many iterations should be removed from the beginning of each chain.
low.memory	Logical indicating if the function should run in a memory-efficient mode.
thin	Thinning interval.
...	Additional arguments passed to the **coda**'s mcmc function.

Details

Function `coda.list.mcmc` is for accessing all chains of phase II MCMCs; Function `coda.list.mcmc3` is for accessing all chains of phase III MCMCs.

Value

The function `coda.list.mcmc` and `coda.list.mcmc3` return an object of class "mcmc.list". The function `coda.mcmc` returns an object of class "mcmc", both defined in the **coda** package.

Author(s)

Hana Sevcikova

---

convert.tfr.trajectories

Converting TFR Trajectories into ACSII Files

Description

Converts TFR trajectories stored in a binary format into two CSV files of a UN-specific format.

Usage

```
convert.tfr.trajectories(dir = file.path(getwd(), 'bayesTFR.output'),
    n = 1000, output.dir = NULL, verbose = FALSE)
```

Arguments

`dir`	Directory containing the prediction object. It should correspond to the `output.dir` argument of the `tfr.predict` function.
`n`	Number of trajectories to be stored. It can be either a single number or the word "all" in which case all trajectories are stored.
`output.dir`	Directory in which the resulting files will be stored. If `NULL` the same directory is used as for the prediction.
`verbose`	Logical switching log messages on and off.

Details

The function creates two files. 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

- "TF"total fertility rate

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 `tfr.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.projection.summary`, `tfr.predict`

Examples

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

# stores 10 trajectories out of 35 (1x(60-25)) into
# exampleTFRpred/predictions/ascii_trajectories.csv
tfr.predict(sim.dir=sim.dir, output.dir=pred.dir, use.tfr3=FALSE,
            burnin=25, save.as.ascii=10, verbose=TRUE)

# stores all 35 trajectories into the current directory
convert.tfr.trajectories(dir=pred.dir, n="all", output.dir=".", verbose=TRUE)

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

## End(Not run)
```

---

country.names *Accessing Country Names*

Description

The function returns country names for countries given either by their codes or by index.

Usage

```
country.names(meta, countries = NULL, index = FALSE)
```

Arguments

meta	Object of class bayesTFR.mcmc.meta, bayesTFR.mcmc.set, bayesTFR.mcmc, or bayesTFR.prediction.
countries	Vector of country codes or indices. If it is not given, names of all countries are returned.
index	Logical indicating if the argument countries is an index.

Details

The function considers countries that are included in the simulations and predictions. If the argument countries is not given, all countries are returned in the same order as they are stored in the meta object.

Value

Vector of country names.

Author(s)

Hana Sevcikova

See Also

get.countries.table, get.country.object

Examples

```
sim.dir <- file.path(find.package("bayesTFR"), "ex-data", "bayesTFR.output")
m <- get.tfr.mcmc(sim.dir)
country.names(m)
# these two calls should give the same answer
country.names(m, c(800, 120))
country.names(m, c(15, 20), index=TRUE)
```

DLcurve.plot *Plotting Posterior Distribution of the Double Logistic Function*

Description

The functions for plotting and retrieving the posterior distribution of the double logistic function used in the simulation of Phase II. Plots include the median and given probability intervals of the distribution.

Usage

```
DLcurve.plot(mcmc.list, country, burnin = NULL, pi = 80, tfr.max = 10,
    nr.curves = NULL, predictive.distr = FALSE, ylim = NULL,
    xlab = 'TFR (reversed)', ylab = 'TFR decrement', main = NULL,
    show.legend = TRUE, col=c('black', 'red', "#00000020"), ...)

DLcurve.plot.all(mcmc.list = NULL, sim.dir = NULL,
    output.dir = file.path(getwd(), 'DLcurves'),
    output.type = "png", burnin = NULL, verbose = FALSE, ...)

tfr.world.dlcurves(x, mcmc.list, burnin=NULL, countryUc=NULL, ...)

tfr.country.dlcurves(x, mcmc.list, country, burnin=NULL, ...)
```

Arguments

mcmc.list	List of bayesTFR.mcmc objects, an object of class bayesTFR.mcmc.set or of class bayesTFR.prediction. In case of DLcurve.plot.all if it si NULL, it is loaded from sim.dir.
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.
tfr.max	Maximum TFR to be shown in the plot.
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	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
...	For the plotting functions, there are additional graphical parameters. For DLcurve.plot.all, ... contains also arguments pi, tfr.max and nr.curves. For the tfr.*.dlcurves functions, these are arguments passed to the underlying functions (predictive.distr and return.sigma for obtaining a sample of the standard deviation of the error term).
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	TFR values for which the double logistic should be computed.
countryUc	Country to use the parameter U_c from (start of the fertility transition). If it is not given, the middle point of the prior distribution is used.

Details

DLcurve.plot plots double logistic curves for the given country. DLcurve.plot.all creates such plots for all countries and stores them in output.dir. Parameters inputting the double logistic function are either thinned traces created by the tfr.predict function (if mcmc.list is an object of class bayesTFR.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 tfr.world.dlcurves returns the DL curves of the hierarchical distribution, conditioned on the starting point of the fertility transition in a given country (given by the countryUc argument). Function tfr.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

tfr.world.dlcurves and tfr.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. If the argument return.sigma is set to TRUE, the return value is a list with the first element being the DL values and the second element being a matrix of the standard deviation of the DL error term sigma_eps.

Author(s)

Hana Sevcikova, Leontine Alkema

Examples

```
## Not run:
sim.dir <- file.path(find.package("bayesTFR"), "ex-data", "bayesTFR.output")
mcmc.set <- get.tfr.mcmc(sim.dir=sim.dir)
DLcurve.plot(country="Burkina Faso", mcmc.set, burnin=15)

# add the median of the hierarchical DL curves
x <- seq(0, 10, length=100)
world <- tfr.world.dlcurves(x, mcmc.set, burnin=15, countryUc="Burkina Faso")
qw <- apply(world, 2, median)
lines(x, qw, col='blue')

## End(Not run)
```

---

get.country.object *Accessing Country Information*

Description

Function get.country.object returns an object containing country name, code and index. Functions get.countries.table return a data frame containing codes and names of all countries.

Usage

```
get.country.object(country, meta = NULL, country.table = NULL, index = FALSE)
## S3 method for class 'bayesTFR.mcmc.set'
get.countries.table(object, ...)
## S3 method for class 'bayesTFR.prediction'
get.countries.table(object, ...)
```

Arguments

country	Country name, code or index. If it is an index, the argument index must be set to TRUE.
meta	Object of class bayesTFR.mcmc.meta. If it is not given, the argument country.table must be given.
country.table	A table containing columns "name" and "code" from which the country info can be extracted. Only relevant, if meta is NULL.
index	Logical determining if the argument country is an index.
object	Object of class bayesTFR.mcmc.set or bayesTFR.prediction.
...	Not used.

Details

Given partial information about a country (i.e. having either name or code or index), the function get.country.object returns an object containing all three pieces of information. Only countries are considered that are included in the simulations and predictions. Country index is an internal index used in various components of a bayesTFR.mcmc.meta object.

Value

Function get.country.object returns a list with components:

name	Country name
code	Country code
index	Country index

Function get.countries.table return a data frame with columns code and name.

Author(s)

Hana Sevcikova

See Also

country.names

Examples

```
sim.dir <- file.path(find.package("bayesTFR"), "ex-data", "bayesTFR.output")
m <- get.tfr.mcmc(sim.dir)
# all four calls should give the same answer
get.country.object('China', m$meta)
get.country.object(156, m$meta)
get.country.object(56, m$meta, index=TRUE)
get.country.object(156, NULL, country.table=get.countries.table(m))
```

`get.cov.gammas` *Covariance Matrices of Gamma Parameters*

Description

From a given MCMC, obtain a covariance matrix of the γ_{ci} $(i = 1, 2, 3)$ parameters for each country c.

Usage

```
get.cov.gammas(mcmc.set = NULL, sim.dir = NULL, burnin = 200, chain.id = 1)
```

Arguments

`mcmc.set`	Object of class `bayesTFR.mcmc.set`. If it is `NULL`, the `sim.dir` is used to load existing simulation results.
`sim.dir`	Directory with existing MCMC simulation results. It is only used if `mcmc.set` is `NULL`.
`burnin`	Number of burn-in iterations to be discarded from the begining of the chain.
`chain.id`	Identifier of the MCMC to be used. By default the first chain is used.

Details

In order to speed-up MCMC convergence, the package contains default values of gamma covariance that were obtained from a previous run (they can be loaded using `data(proposal_cov_gammas_cii)`). However, this function allows to obtain new values and overwrite the default values by passing the resulting object to the `run.tfr.mcmc` function as the `proposal_cov_gammas` argument.

Value

A list with components:

`values`	An array of size nr_countries $\times$ 3 $\times$ 3 containing values of the covariance matrix of γ_{ci} $(i = 1, 2, 3)$ for each country c.
`country_codes`	A vector of size nr_countries. A covariance matrix `values[i,,]` corresponds to a country with the code `country_codes[i]`.

Author(s)

Leontine Alkema, Hana Sevcikova

See Also

run.tfr.mcmc

Examples

```
## Not run:
sim.dir <- file.path(find.package("bayesTFR"), "ex-data", "bayesTFR.output")
cov.gammas <- get.cov.gammas(sim.dir=sim.dir, burnin=0)
m <- run.tfr.mcmc(nr.chains=1, iter=10, proposal_cov_gammas=cov.gammas, verbose=TRUE)

## End(Not run)
```

---

get.regtfr.prediction *Accessing Subnational Prediction Objects*

Description

Retrieve subnational (regional) prediction results produced by tfr.predict.subnat, either for one country or for all available countries.

Usage

```
get.regtfr.prediction(sim.dir, country = NULL)
```

Arguments

sim.dir	Simulation directory of the subnational predictions. It corresponds to the argument output.dir in tfr.predict.subnat.
country	Numerical country code. If it is not given, all available subnational predictions are retrieved.

Details

Predictions for country x are assumed to be stored in "sim.dir/subnat/cx".

Value

If argument country is given, the function returns an object of class bayesTFR.prediction. If it is NULL, it returns a list of such objects. Names of the list items are the country codes.

See Also

tfr.predict.subnat

Examples

```
# Subnational example data
my.subtfr.file <- file.path(find.package("bayesTFR"), 'extdata', 'subnational_tfr_template.txt')
subtfr <- read.delim(my.subtfr.file, check.names=FALSE, stringsAsFactor=FALSE)
countries <- unique(subtfr[, c("country_code", "country")])

# Directory with national projections (contains 30 trajectories for each country)
nat.dir <- file.path(find.package("bayesTFR"), "ex-data", "bayesTFR.output")

# Subnational projections for all three countries ()
subnat.dir <- tempfile()
tfr.predict.subnat(countries$country_code, my.tfr.file=my.subtfr.file,
    sim.dir=nat.dir, output.dir=subnat.dir, start.year=2013)

# Retrieve results for all countries
preds <- get.regtfr.prediction(subnat.dir)
names(preds)

# View tables of subregions for each country
for(i in 1:nrow(countries)) {
  cat("\n\n", countries$country[i], "\n")
  print(get.countries.table(preds[[as.character(countries$country_code[i])]]))
}
# Quantiles for individual subregions
tfr.trajectories.table(preds[["218"]], "Bolivar")

# Retrieve results for one country
pred <- get.regtfr.prediction(subnat.dir, 218)
tfr.trajectories.plot(pred, "Loja")

# cleanup
unlink(subnat.dir)

# See more examples in ?tfr.predict.subnat
```

---

get.tfr.convergence *Accessing a Convergence Object*

Description

The function loads objects of class bayesTFR.convergence from disk.

Usage

```
get.tfr.convergence(sim.dir = file.path(getwd(), "bayesTFR.output"),
    thin=80, burnin = 2000)

get.tfr.convergence.all(sim.dir = file.path(getwd(), "bayesTFR.output"))
```

```
get.tfr3.convergence(sim.dir = file.path(getwd(), "bayesTFR.output"),
    thin=60, burnin = 10000)

get.tfr3.convergence.all(sim.dir = file.path(getwd(), "bayesTFR.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.tfr.convergence loads an object of class bayesTFR.convergence for the specific thin and burnin generated for Phase II MCMCs. Function get.tfr.convergence.all loads all Phase II bayesTFR.convergence objects available for sim.dir. Functions get.tfr3.convergence and get.tfr3.convergence.all do the same thing for Phase III MCMCs.

Value

get.tfr.convergence and get.tfr3.convergence return an object of class bayesTFR.convergence; get.tfr.convergence.all and get.tfr3.convergence.all return a list of objects of class bayesTFR.convergence.

Author(s)

Hana Sevcikova

See Also

bayesTFR.convergence, summary.bayesTFR.convergence.

---

get.tfr.mcmc *Accessing MCMC Results*

Description

The function get.tfr.mcmc retrieves results of an MCMC simulation of Phase II and creates an object of class bayesTFR.mcmc.set. Function has.tfr.mcmc checks the existence of such results. Functions get.tfr3.mcmc and has.tfr3.mcmc do the same for Phase III MCMCs. Function tfr.mcmc extracts a single chain and tfr.mcmc.list extracts several or all chains from the simulation results.

Usage

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

has.tfr.mcmc(sim.dir)

get.tfr3.mcmc(sim.dir = file.path(getwd(), "bayesTFR.output"), ...)

has.tfr3.mcmc(sim.dir)

tfr.mcmc(mcmc.set, chain.id)

tfr.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=FALSE.
verbose	Logical switching log messages on and off.
chain.id	Chain identifier.
mcmc.set	Object of class bayesTFR.mcmc.set.
...	Arguments passed to get.tfr.mcmc.

Details

Function get.tfr.mcmc is an accessor of results generated using run.tfr.mcmc and continue.tfr.mcmc. Function get.tfr3.mcmc can be used to access results generated using run.tfr3.mcmc and continue.tfr3.mcmc.

Value

get.tfr.mcmc and get.tfr3.mcmc return an object of class bayesTFR.mcmc.set. has.tfr.mcmc and has.tfr3.mcmc return a logical value. tfr.mcmc returns an object of class bayesTFR.mcmc, and tfr.mcmc.list returns a list of bayesTFR.mcmc objects.

Author(s)

Hana Sevcikova

See Also

bayesTFR.mcmc.set

Examples

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

# summary of the single chains
for(mc in tfr.mcmc.list(m)) print(summary(mc))
```

`get.tfr.parameter.traces`

Accessing MCMC Parameter Traces

Description

Functions for accessing traces of the MCMC parameters, either country-independent or country-specific. Functions `get.tfr.parameter.traces` and `get.tfr.parameter.traces.cs` access Phase II MCMCs; Functions `get.tfr3.parameter.traces` and `get.tfr3.parameter.traces.cs` access Phase III MCMCs.

Usage

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

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

get.tfr3.parameter.traces(mcmc.list, par.names = tfr3.parameter.names(), ...)

get.tfr3.parameter.traces.cs(mcmc.list, country.obj,
    par.names=tfr3.parameter.names.cs(), ...)
```

Arguments

`mcmc.list`	List of `bayesTFR.mcmc` objects.
`country.obj`	Country object list (see `get.country.object`).
`par.names`	Names of country-independent parameters (in case of `get.tfr.parameter.traces`) or country-specific parameters (in case of `get.tfr.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`. 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. In the above example it would be thin=4.
...	Arguments passed to underlying functions (i.e. to get.tfr.parameter.traces or get.tfr.parameter.traces.cs).

Value

All 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.tfr.parameter.traces and get.tfr3.parameter.traces return country-independent parameters, get.tfr.parameter.traces.cs and get.tfr3.parameter.traces.cs return country-specific parameters.

Author(s)

Hana Sevcikova

See Also

coda.list.mcmc for another way of retrieving parameter traces.

Examples

```
sim.dir <- file.path(find.package("bayesTFR"), "ex-data", "bayesTFR.output")
m <- get.tfr.mcmc(sim.dir)
tfr.values <- get.tfr.parameter.traces(m$mcmc.list, burnin=10, par.names="sigma0")
## Not run:
hist(tfr.values, main=colnames(tfr.values))

## End(Not run)
tfr.values.cs <- get.tfr.parameter.traces.cs(m$mcmc.list,
                    get.country.object("Canada", meta=m$meta),
                    burnin=10, par.names="Triangle_c4")
## Not run:
hist(tfr.values.cs, main=colnames(tfr.values.cs))

## End(Not run)
```

---

get.tfr.prediction *Accessing a Prediction Object*

Description

Function get.tfr.prediction retrieves results of a prediction and creates an object of class bayesTFR.prediction. Function has.tfr.prediction checks an existence of such results.

Usage

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

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

Arguments

mcmc	Object of class bayesTFR.mcmc.set used to make the prediction. It must correspond to a Phase II MCMC. 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 tfr.predict function. Only relevant if mcmc is NULL.
mcmc.dir	Optional argument to be used only in a special case when the mcmc object contained in the prediction object was estimated in different directory than in the one to which it points to (for example due to moving or renaming the original directory). The argument causes that the mcmc is redirected to the given directory. It can be set to NA if no loading of the mcmc object is desired.

Details

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

Value

Function has.tfr.prediction returns a logical indicating if a prediction exists for the given mcmc.

Function get.tfr.prediction returns an object of class bayesTFR.prediction.

Author(s)

Hana Sevcikova

See Also

bayesTFR.prediction, tfr.predict, summary.bayesTFR.prediction

Examples

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

---

get.tfr.trajectories *Accessing TFR Trajectories*

Description

Function for accessing TFR trajectories.

Usage

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

Arguments

tfr.pred	Object of class bayesTFR.prediction.
country	Name or numerical code of a country.

Details

The function loads TFR trajectories for the given country from disk, offsets it if needed (see tfr.median.shift) and returns it as a matrix.

Value

Array of size 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

bayesTFR.prediction, get.tfr.prediction, tfr.trajectories.table, tfr.median.shift

Examples

```
sim.dir <- file.path(find.package("bayesTFR"), "ex-data", "bayesTFR.output")
pred <- get.tfr.prediction(sim.dir=sim.dir)
get.tfr.trajectories(pred, "Germany")
```

---

get.thinned.tfr.mcmc *Creating and Accessing Thinned MCMCs*

Description

The function get.thinned.tfr.mcmc accesses a thinned and burned version of the given Phase II MCMC set. create.thinned.tfr.mcmc creates such a set.

Usage

```
get.thinned.tfr.mcmc(mcmc.set, thin = 1, burnin = 0)

create.thinned.tfr.mcmc(mcmc.set, thin = 1, burnin = 0,
    output.dir = NULL, verbose = TRUE)
```

Arguments

mcmc.set	Object of class bayesTFR.mcmc.set of Phase II.
thin, burnin	Thinning interval and burnin used for creating or identifying the thinned object.
output.dir	Output directory. It is only used if the output goes to a non-standard directory.
verbose	Logical switching log messages on and off.

Details

The function create.thinned.tfr.mcmc is called from tfr.predict and thus, the resulting object contains exactly the same MCMCs used for generating projections. In addition, it can be also called from tfr.diagnose if desired, so that the projection process can re-use such a set that lead to a convergence.

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 the main simulation directory 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 bayesTFR.mcmc.set. get.thinned.tfr.mcmc returns NULL if such object does not exist.

Author(s)

Hana Sevcikova

See Also

bayesTFR.mcmc.set, tfr.predict, tfr.diagnose

Examples

```
## Not run:
sim.dir <- tempfile()
m <- run.tfr.mcmc(nr.chains=2, iter=30, seed=1, output.dir=sim.dir, verbose=TRUE)
tfr.predict(m, burnin=15, use.tfr3=FALSE) # creates thinned MCMCs
mb <- get.thinned.tfr.mcmc(m, thin=1, burnin=15)
summary(mb, meta.only=TRUE) # length 30 = 2chains x (30-15)iters.
unlink(sim.dir, recursive=TRUE)

## End(Not run)
```

`get.total.iterations` *Total Number of Iterations*

Description

Function `get.total.iterations` gives the total number of iterations of MCMCs summed over chains whith burnin being subtracted from each chain. Function `get.stored.mcmc.length` gives the total length of the MCMCs stored on disk minus those iterations that correspond to burnin. Result of the latter will be different from the former only if the MCMCs were run with value of `thin` larger than one.

Usage

```
get.total.iterations(mcmc.list, burnin = 0)

get.stored.mcmc.length(mcmc.list, burnin = 0)
```

Arguments

`mcmc.list`	List of `bayesTFR.mcmc` objects.
`burnin`	Number of iterations to be subtracted from each chain.

Value

A single number.

Author(s)

Hana Sevcikova

See Also

`bayesTFR.mcmc`

Examples

```
sim.dir <- file.path(find.package("bayesTFR"), "ex-data", "bayesTFR.output")
mcmc.set <- get.tfr.mcmc(sim.dir=sim.dir)
get.total.iterations(mcmc.set$mcmc.list) # 60=1chain x 60iters
get.total.iterations(mcmc.set$mcmc.list, burnin=20) # 40=1x(60-20)

## Not run:
sim.dir <- tempfile()
m <- run.tfr.mcmc(iter=10, nr.chains=2, output.dir=sim.dir, thin=5, verbose=TRUE)
get.total.iterations(m$mcmc.list) # 20=2x10
get.stored.mcmc.length(m$mcmc.list) # 6=2x3
unlink(sim.dir, recursive=TRUE)

## End(Not run)
```

include *Inclusion Codes*

Description

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

Usage

```
data(include_2015)
data(include_2012)
data(include_2010)
```

Format

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

`country` Name of country or region. Not used.

`country_code` Numerical Location Code (3-digit codes following ISO 3166-1 numeric standard) - see http://en.wikipedia.org/wiki/ISO_3166-1_numeric.

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.tfr.mcmc`. It is merged with a `tfr` dataset from the corresponding wpp package using the `country_code` column. Thus, the country entries in this dataset should correspond to entries in the `tfr` dataset.

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

Source

Data provided by the United Nations Population Division.

Examples

```
data(include_2015)
head(include_2015)
```

`run.tfr.mcmc` *Running Markov Chain Monte Carlo for Parameters of Total Fertility Rate in Phase II*

Description

Runs (or continues running) MCMCs for simulating the total fertility rate of all countries of the world (phase II), using a Bayesian hierarchical model.

Usage

```
run.tfr.mcmc(nr.chains = 3, iter = 62000,
    output.dir = file.path(getwd(), "bayesTFR.output"),
    thin = 1, replace.output = FALSE,
    start.year = 1950, present.year = 2015, wpp.year = 2015,
    my.tfr.file = NULL, my.locations.file = NULL, buffer.size = 100,
    U.c.low = 5.5, U.up = 8.8, U.width = 3,
    mean.eps.tau0 = -0.25, sd.eps.tau0 = 0.4, nu.tau0 = 2,
    Triangle_c4.low = 1, Triangle_c4.up = 2.5,
    Triangle_c4.trans.width = 2,
    Triangle4.0 = 0.3, delta4.0 = 0.8, nu4 = 2,
    S.low = 3.5, S.up = 6.5, S.width = 0.5,
    a.low = 0, a.up = 0.2, a.width = 0.02,
    b.low = a.low, b.up = a.up, b.width = 0.05,
    sigma0.low = 0.01, sigma0.up = 0.6, sigma0.width = 0.1,
    sigma0.min = 0.001,
    const.low = 0.8, const.up = 2, const.width = 0.3,
    d.low = 0.05, d.up = 0.5, d.trans.width = 1,
    chi0 = -1.5, psi0 = 0.6, nu.psi0 = 2,
    alpha0.p = c(-1, 0.5, 1.5), delta0 = 1, nu.delta0 = 2,
    dl.p1 = 9, dl.p2 = 9,
    S.ini = NULL, a.ini = NULL, b.ini = NULL, sigma0.ini = NULL,
    Triangle_c4.ini = NULL, const.ini = NULL, gamma.ini = 1,
    proposal_cov_gammas = NULL,
    seed = NULL, parallel = FALSE, nr.nodes = nr.chains,
    save.all.parameters = FALSE, compression.type = 'None',
    auto.conf = list(max.loops=5, iter=62000, iter.incr=10000,
        nr.chains=3, thin=80, burnin=2000),
    verbose = FALSE, verbose.iter = 10, ...)

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

Arguments

`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$max.loops`.

`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**x where x is the `wpp.year` and uses the `tfr*` datasets.

`my.tfr.file` File name containing user-specified TFR 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 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.

`U.c.low, U.up` Lower and upper bound of the parameter U_c, the start level of the fertility transition. The lower bound is set for each country as the maximum of `U.c.low` and the minimum of historical TFR for that country.

`U.width` Width for slice sampling used when updating the U_c parameter.

`mean.eps.tau0, sd.eps.tau0`
Mean and standard deviation of the prior distribution of m_τ which is the mean of the distortion terms $\epsilon_{c,\tau}$ in start periods τ_c.

`nu.tau0` The shape parameter of the prior Gamma distribution of $1/s_\tau^2$ is `nu.tau0/2`. s_τ is standard deviation of the distortion terms in start periods τ_c.

`Triangle_c4.low, Triangle_c4.up`
Lower and upper bound of the Δ_{c4} parameter.

`Triangle_c4.trans.width`
Width for slice sampling used when updating the logit-transformed Δ_{c4} parameter.

`Triangle4.0, delta4.0`
Mean and standard deviation of the prior distribution of the Δ_4 parameter which is the hierarchical mean of the logit-transformed Δ_{c4}.

`nu4` The shape parameter of the prior Gamma distribution of $1/\delta_4^2$ is `nu4/2`. δ_4 is standard deviation of the logit-transformed Δ_{c4}.

`S.low, S.up` Lower and upper bound of the uniform prior distribution of the S parameter which is the TFR at which the distortion has maximum variance.

`S.width` Width for slice sampling used when updating the S parameter.

`a.low, a.up` Lower and upper bound of the uniform prior distribution of the a parameter which is a coefficient for linear decrease of the TFR for TFR larger than S.

`a.width` Width for slice sampling used when updating the a parameter.

`b.low, b.up` Lower and upper bound of the uniform prior distribution of the b parameter which is a coefficient for linear decrease of the TFR for TFR smaller than S.

`b.width` Width for slice sampling used when updating the b parameter.

`sigma0.low, sigma0.up` Lower and upper bound of the uniform prior distribution of the σ_0 parameter. σ_0^2 is the maximum variance of the distortions at TFR equals S.

`sigma0.width` Width for slice sampling used when updating the σ_0 parameter.

`sigma0.min` Minimum value that σ_0 can take.

`const.low, const.up` Lower and upper bound of the uniform prior distribution of the c parameter which is the multiplier of standard deviation of the distortions before 1975.

`const.width` Width for slice sampling used when updating the c parameter.

`d.low, d.up` Lower and upper bound of the parameter d_c, the maximum annual decrement for country c. (Note that in Alkema et al. this parameter is a five-years decrement.)

`d.trans.width` Width for slice sampling used when updating the logit-transformed d_c parameter.

`chi0, psi0` Prior mean and standard deviation of the χ parameter which is the hierarchical mean of logit-transformed maximum decline parameter d_c.

`nu.psi0` The shape parameter of the prior Gamma distribution of $1/\psi^2$ is `nu.psi0`/2. ψ is the standard devation of logit-transformed maximum decline parameter d_c.

`alpha0.p` Vector of prior means of the α_i parameters, $i = 1, 2, 3$. α_i is the hierarchical mean of γ_{ci}.

`delta0` Prior standard deviation of the α_i parameters. It is a single value, i.e. the same standard deviation is used for all i.

`nu.delta0` The shape parameter of the prior Gamma distribution of $1/\delta_i^2$ is `nu.delta0`/2. δ_i is the standard deviation of γ_{ci}.

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

`S.ini` Initial value for the S parameter. It can be a single value or an array of the size `nr.chains`. By default, if `nr.chains` is 1, it is the middle point of the interval [`S.low, S.up`]. Otherwise, it is equally spaced distributed between `S.low` and `S.up`.

`a.ini` Initial value for the a parameter. It can be a single value or an array of the size `nr.chains`. By default, if `nr.chains` is 1, it is the middle point of the interval [`a.low, a.up`]. Otherwise, it is equally spaced distributed between `a.low` and `a.up`.

`b.ini` Initial value for the b parameter. It can be a single value or an array of the size `nr.chains`. By default, if `nr.chains` is 1, it is the middle point of the interval [`b.low, b.up`]. Otherwise, it is equally spaced distributed between `b.low` and `b.up`.

`sigma0.ini` Initial value for the σ_0 parameter. It can be a single value or an array of the size `nr.chains`. By default, if `nr.chains` is 1, it is the middle point of the interval [`sigma0.low, sigma0.up`]. Otherwise, it is equally spaced distributed between `sigma0.low` and `sigma0.up`.

`Triangle_c4.ini`
Initial value for the Δ_{c4} parameter. It can be a single value or an array of the size `nr.chains`. By default, if `nr.chains` is 1, it is the middle point of the interval [`Triangle_c4.low, Triangle_c4.up`]. Otherwise, it is equally spaced distributed between `Triangle_c4.low` and `Triangle_c4.up`.

`const.ini` Initial value for the c parameter. It can be a single value or an array of the size `nr.chains`. By default, if `nr.chains` is 1, it is the middle point of the interval [`const.low, const.up`]. Otherwise, it is equally spaced distributed between `const.low` and `const.up`.

`gamma.ini` Initial value for the γ_c parameter. The same value is used for all countries.

`proposal_cov_gammas`
Proposal for the gamma covariance matrices for each country. It should be a list with two values: `values` and `country_codes`. The structure corresponds to the object returned by the function `get.cov.gammas`. By default the covariance matrices are obtained using `data(proposal_cov_gammas_cii)`. This argument overwrite the defaults for countries contained the argument.

`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. In such a case further arguments can be passed, see description of ....

`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.

`save.all.parameters`
If `TRUE`, the distortion terms $\epsilon_{c,t}$ for all t are stored on disk, otherwise not.

`compression.type`
One of 'None', 'gz', 'xz', 'bz', determining type of a compression of the MCMC files. Important: Do not use this option for a long MCMC simulation as this tends to cause very long run times due to slow reading!

`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. For example cltype which is 'SOCK' by default but can be set to 'MPI'.
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.tfr.mcmc creates an object of class bayesTFR.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 mc$x$ 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 (U, d, γ) have the suffix _cy where y is the country code. In addition to the trace files, each mc$x$ directory contains the object bayesTFR.mcmc in binary format. All chain-specific files are written into disk after the first, last and each buffer.size-th iteration.

Using the function continue.tfr.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) from the tfr and tfr_supplemental datasets in a **wpp**x 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.tfr.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 by 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.tfr.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.tfr.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.tfr.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.tfr.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 user-specific.

Optionally, my.tfr.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 tfr.predict fills in the missing values using the median of the

BHM procedure (stored in tfr_matrix_reconstructed of the bayesTFR.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_tfr_template' (in 'extdata' directory) which is a template for user-specified my.tfr.file.

Value

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

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

Author(s)

Hana Sevcikova, Leontine Alkema

References

L. Alkema, A. E. Raftery, P. Gerland, S. J. Clark, F. Pelletier, Buettner, T., Heilig, G.K. (2011). Probabilistic Projections of the Total Fertility Rate for All Countries. Demography, Vol. 48, 815-839.

See Also

get.tfr.mcmc, summary.bayesTFR.mcmc.set.

Examples

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

## End(Not run)
```

---

run.tfr.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 hyperparameters from an existing simulation to generate country-specific parameters.

Usage

```
run.tfr.mcmc.extra(sim.dir = file.path(getwd(), "bayesTFR.output"),
    countries = NULL, my.tfr.file = NULL,
    iter = NULL, thin = 1, burnin = 2000,
    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.tfr.file	File name containing user-specified TFR time series 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 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. It is also used when computing proposal of gamma covariance matrices (see get.cov.gammas).
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.tfr.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 tfr dataset. Or, it can be given by a user-specific TFR file, using the argument my.tfr.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.tfr.file` passed to `run.tfr.mcmc`) get their parameters recomputed. Note that all countries should be included in the `UNlocations` dataset, but unlike in `run.tfr.mcmc`, their `include_code` is ignored. As in the case of `run.tfr.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 TFR is simulated for new locations that are not included in `UNlocations`.

Value

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

Note

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

Author(s)

Hana Sevcikova, Leontine Alkema

See Also

`run.tfr.mcmc`, `tfr.predict.extra`

Examples

```
## Not run:
sim.dir <- tempfile()
m <- run.tfr.mcmc(nr.chains=1, iter=20, output.dir=sim.dir, verbose=TRUE)
m <- run.tfr.mcmc.extra(sim.dir=sim.dir, countries=c(908,924), burnin=10, verbose=TRUE)
summary(m, country=924)
pred <- tfr.predict(m, burnin=10, use.tfr3=FALSE, verbose=TRUE)
summary(pred, country=908)
unlink(sim.dir, recursive=TRUE)

## End(Not run)
```

`run.tfr3.mcmc` *Running Markov Chain Monte Carlo for Parameters of Total Fertility Rate in Phase III*

Description

Runs (or continues running) MCMCs for simulating phase III total fertility rate, using a Bayesian hierarchical version of an AR(1) model.

Usage

```
run.tfr3.mcmc(sim.dir, nr.chains = 3, iter = 50000, thin = 10,
    replace.output = FALSE, my.tfr.file = NULL, buffer.size = 100,
    use.extra.countries = FALSE,
   mu.prior.range = c(0, 2.1), rho.prior.range = c(0, 1 - .Machine$double.xmin),
   sigma.mu.prior.range = c(1e-05, 0.318), sigma.rho.prior.range = c(1e-05, 0.289),
    sigma.eps.prior.range = c(1e-05, 0.5),
    mu.ini = NULL, mu.ini.range = mu.prior.range,
    rho.ini = NULL, rho.ini.range = rho.prior.range,
    sigma.mu.ini = NULL, sigma.mu.ini.range = sigma.mu.prior.range,
    sigma.rho.ini = NULL, sigma.rho.ini.range = sigma.rho.prior.range,
    sigma.eps.ini = NULL, sigma.eps.ini.range = sigma.eps.prior.range,
   seed = NULL, parallel = FALSE, nr.nodes = nr.chains, compression.type = "None",
   auto.conf = list(max.loops = 5, iter = 50000, iter.incr = 20000, nr.chains = 3,
                    thin = 60, burnin = 10000),
    verbose = FALSE, verbose.iter = 1000, ...)

continue.tfr3.mcmc(sim.dir, iter, chain.ids=NULL,
    parallel = FALSE, nr.nodes = NULL, auto.conf = NULL,
    verbose=FALSE, verbose.iter = 1000, ...)
```

Arguments

`sim.dir`	Directory with an existing simulation of phase II TFR (see `run.tfr.mcmc`).
`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$max.loops`.
`thin`	Thinning interval between consecutive observations to be stored on disk.
`replace.output`	If `TRUE`, previously stored results of a phase III simulation will be overwritten.
`my.tfr.file`	File name containing user-specified TFR time series for one or more countries. See description of this argument in `run.tfr.mcmc`.
`buffer.size`	Buffer size (in number of iterations) for keeping data in the memory.
`use.extra.countries`	By default, only countries are used in the MCMCs that were assigned for estimation (i.e. their 'include_code' is 2 in the include) dataset and are in phase III at present time (argument `present.year` in `run.tfr.mcmc`). If this argument is `TRUE`, countries that were added using `run.tfr.mcmc.extra` and are in phase III are also included.
`mu.prior.range, rho.prior.range, sigma.mu.prior.range, sigma.rho.prior.range, sigma.eps.prior.range`	Min and max for the prior (uniform) distribution of these paraemters.
`mu.ini, rho.ini, sigma.mu.ini, sigma.rho.ini, sigma.eps.ini`	Initial value(s) of the parameters. It can be a single value or an array of the size `nr.chains`. By default, if `nr.chains` is 1, it is the middle point of the

corresponding range. Otherwise, it is uniformly randomly distributed within the range.

`mu.ini.range, rho.ini.range, sigma.mu.ini.range, sigma.rho.ini.range, sigma.eps.ini.range`
Min and max for the initial values.

`seed` Seed of the random number generator. If `NULL` no seed is set.

`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.

`compression.type`
One of 'None', 'gz', 'xz', 'bz', determining type of a compression of the MCMC files. Important: Do not use this option for a long MCMC simulation as this tends to cause very long run times due to slow reading!

`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) 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 are resumed.

Details

The MCMCs are stored in `sim.dir` in a subdirectory called "phaseIII". It has exactly the same structure as phase II MCMCs described in `run.tfr.mcmc`.

Value

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

`meta` An object of class `bayesTFR.mcmc.meta`.

`mcmc.list` A list of objects of class `bayesTFR.mcmc`, one for each MCMC.

Author(s)

Hana Sevcikova

References

Raftery, A.E., Alkema, L. and Gerland, P. (2014). Bayesian Population Projections for the United Nations. Statistical Science, Vol. 29, 58-68.

See Also

run.tfr.mcmc, get.tfr3.mcmc

Examples

```
## Not run:
sim.dir <- tempfile()
# Runs Phase II MCMCs (must be run before Phase III)
m <- run.tfr.mcmc(nr.chains=1, iter=5, output.dir=sim.dir, verbose=TRUE)
# Runs Phase III MCMCs
m3 <- run.tfr3.mcmc(sim.dir=sim.dir, nr.chains=2, iter=50, thin=1, verbose=TRUE)
m3 <- continue.tfr3.mcmc(sim.dir=sim.dir, iter=10, verbose=TRUE)
summary(m3, burnin=10)
unlink(sim.dir, recursive=TRUE)

## End(Not run)
```

---

summary.bayesTFR.convergence

Summary of a TFR Convergence Object

Description

Summary of an object of class bayesTFR.convergence created using the tfr.diagnose or tfr3.diagnose functions. It gives an overview about parameters that did not converge.

Usage

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

Arguments

object	Object of class bayesTFR.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

tfr.diagnose, tfr3.diagnose

---

summary.bayesTFR.mcmc.set

Summary Statistics for TFR Markov Chain Monte Carlo Chains

Description

Summary of an object bayesTFR.mcmc.set or bayesTFR.mcmc, computed via run.tfr.mcmc or run.tfr3.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

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

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

Arguments

object	Object of class bayesTFR.mcmc.set or bayesTFR.mcmc.
country	Country name or code if a country-specific summary is desired. By default, summary for all countries is generated.
chain.id	Identifiers of MCMC chains. By default, all chains are considered.
par.names	Country independent parameters to be included in the summary. If the underlying object is an MCMC of phase II, the default names are given by tfr.parameter.names(), if it is phase III the names are tfr3.parameter.names().
par.names.cs	Country-specific parameters to be included in the summary. If the underlying object is an MCMC of phase II, the default names are given by tfr.parameter.names.cs(), if it is phase III the names are tfr3.parameter.names.cs().
meta.only	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.tfr.mcmc or run.tfr3.mcmc.
burnin	Number of iterations to be discarded from the beginning of each chain before computing the summary.
...	Additional arguments passed to the summary.mcmc function of the **coda** package.

Author(s)

Hana Sevcikova

See Also

bayesTFR.mcmc.set, summary.mcmc

Examples

```
sim.dir <- file.path(find.package("bayesTFR"), "ex-data", "bayesTFR.output")
m <- get.tfr.mcmc(sim.dir)
summary(m, country="Czech Republic", burnin=15)
```

---

summary.bayesTFR.prediction

Summary of a Prediction of the Total Fertility Rate

Description

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

Usage

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

Arguments

object	Object of class bayesTFR.prediction.
country	Country name or code. If it is NULL, only prediction parameters are included.
compact	Logical switching between a smaller and larger number of displayed quantiles.
...	A list of further arguments.

Author(s)

Hana Sevcikova

See Also

bayesTFR.prediction

Examples

```
## Not run:
sim.dir <- file.path(find.package("bayesTFR"), "ex-data", "bayesTFR.output")
pred <- tfr.predict(sim.dir=sim.dir,
                    output.dir=file.path(getwd(), "exampleTFRpred"),
                    use.tfr3=FALSE, burnin=15, verbose=TRUE)
# If the above function was run previously, do
# pred <- get.tfr.prediction(sim.dir=file.path(getwd(), "exampleTFRpred"))

summary(pred, country="Ireland")

## End(Not run)
```

---

tfr.diagnose *Convergence Diagnostics of TFR Markov Chain Monte Carlo*

Description

Functions `tfr.diagnose` and `tfr3.diagnose` run convergence diagnostics of existing TFR MCMCs for phase II and phase III, respectively, using the `raftery.diag` function from the **coda** package. `has.mcmc.converged` checks if the existing diagnostics converged.

Usage

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

tfr3.diagnose(sim.dir, thin = 60, burnin = 10000, express = TRUE,
    country.sampling.prop = NULL, verbose = TRUE, ...)

has.mcmc.converged(diag)
```

Arguments

`sim.dir`	Directory with the MCMC simulation results.
`thin`	Thinning interval.
`burnin`	Number of iterations to be discarded from the beginning of the parameter traces.
`express`	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`.
`country.sampling.prop`	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.

keep.thin.mcmc	Logical. If TRUE the thinned traces used for computing the diagnostics are stored on disk (see create.thinned.tfr.mcmc). It is only available for phase II MCMCs.
verbose	Logical switching log messages on and off.
diag	Object of class bayesTFR.convergence.
...	Not used.

Details

The diagnose functions invoke the tfr.raftery.diag (or tfr3.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 from tfr.diagnose is stored in '{sim.dir}/diagnostics/bayesTFR.convergence_{thin}_{burnin}.rda' and can be accessed using the function get.tfr.convergence. Function tfr3.diagnose stores its result into '{sim.dir}/phaseIII/diagnostics/bayesTFR.convergence_{thin}_{burnin}.rda' which can be accessed via get.tfr3.convergence.

Value

has.mcmc.converged returns a logical value determining if there is convergence or not.

tfr.diagnose and tfr3.diagnose return an object of class bayesTFR.convergence with components:

result	Table containing all not-converged parameters. Its columns include 'Total iterations needed' and 'Remaining iterations'.
lresult.country.independent	Number of rows in result that correspond to country-independent paramters. These rows are groupped at the beginning of the table.
country.independent	Result of tfr.raftery.diag processed on country-independent parameters.
country.specific	Result of tfr.raftery.diag processed on country-specific parameters.
iter.needed	Number of additional iterations suggested in order to achieve convergence.
iter.total	Total number of iterations of the original unthinned set of chains.
use.nr.traj	Suggestion for number of trajectories in generating predictions.
burnin	Burnin used.
thin	Thinning interval used.
status	Vector of character strings containing the result status. Possible values: 'green', 'red'.
mcmc.set	Object of class bayesTFR.mcmc.set that corresponds to the original set of MCMCs on which the diagnostics was run.
thin.mcmc	If keep.thin.mcmc is TRUE, it is an object of class bayesTFR.mcmc.set that corresponds to the thinned mcmc set on which the diagnostics was run, otherwise NULL.

express	Value of the input argument express.
nr.countries	Vector with elements used - number of countries used in this diagnostics, and total - number of countries that this mcmc.set object was estimated on.

Author(s)

Hana Sevcikova, Leontine Alkema, Adrian Raftery

See Also

tfr.raftery.diag, raftery.diag, summary.bayesTFR.convergence, get.tfr.convergence, create.thinned.tfr.mcmc

---

tfr.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 and mean absolute error of the simulation.

Usage

```
tfr.dl.coverage(sim.dir, pi = c(80, 90, 95), burnin = 2000, verbose = TRUE)
```

Arguments

sim.dir	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.
pi	Probability interval. It can be a single number or an array.
burnin	Burnin. Only relevant if sim.dir does not contain thinned chains.
verbose	Logical switching log messages on and off.

Value

List with the following components:

total.coverage	Vector of the coverage, one element per probability interval. For each pi, it is the ratio of the number of observed data points that fall within the probability interval of the posterior distribution over the total number of data points, i.e. TFR for all countries and historical time periods.
time.coverage	Matrix corresponding to the coverage computed per time period. (Rows correspond to probability intervals, columns correspond to time.) It is derived like total.coverage except that both, the nominator and denominator, contain only data points belonging to the corresponding time period.

`country.coverage`
: Matrix corresponding to the coverage computed per country. (Rows correspond to probability intervals, columns correspond to countries.) It is derived like `total.coverage` except that both, the nominator and denominator, contain only data points belonging to the corresponding country.

`total.rmse`
: Root mean square error as $\sqrt{(1/n \sum (x-m)^2)}$ where x are observed data points, m is the mean of the posterior distribution and n is the number of data points. Here the sum is taken over all countries and historical time periods.

`time.rmse`
: Like `total.rmse` except that each time period is considered separately.

`country.rmse`
: Like `total.rmse` except that each country is considered separately.

`total.mae`
: Mean absolute error as $1/n \sum |x-m|$ where x are observed data points, m is the median of the posterior distribution and n is the number of data points. Here the sum is taken over all countries and historical time periods.

`time.mae`
: Like `total.mae` except that each time period is considered separately.

`country.mae`
: Like `total.mae` except that each country is considered separately.

`pred.cdf`
: $T \times C$ matrix (with T being the number of time periods and C being the number of countries), containing the predictive CDF of the observation, i.e. the quantile of each data point within the predictive posterior distribution.

`n`
: 0-1 $T \times C$ matrix indicating if the corresponding data point was included in the goodness of fit computation. Zeros indicate missing historical values.

Note

To see the fit visually per country, use `DLcurve.plot(..., predictive.distr=TRUE,...)`.

Author(s)

Hana Sevcikova

See Also

`DLcurve.plot`

Examples

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

## End(Not run)
```

tfr.map *TFR World Map*

Description

Generates a world map of the total fertility rate for given projection period and quantile. In addition, country specific Phase II MCMC parameters can be projected into the world map.

Usage

```
tfr.map(pred, quantile = 0.5,
    year = NULL, par.name = NULL, adjusted = FALSE,
    projection.index = 1, device = "dev.new", main = NULL,
    resolution=c("coarse","low","less islands","li","high"),
    device.args = NULL, data.args = NULL, ...)

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

get.tfr.map.parameters(pred, tfr.range = NULL,
    nr.cats = 50, same.scale = TRUE, quantile = 0.5, ...)

tfr.map.gvis(pred, year = NULL, quantile = 0.5, pi = 80,
    par.name = NULL, adjusted = FALSE, ...)
```

Arguments

pred	Object of class bayesTFR.prediction.
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.
year	Year to be plotted. It can be a year within a projection period or a year within an estimation period. In the latter case, the observed data are plotted. If not given, projection.index determines the projection year.
par.name	Name of a country-specific parameter to be plotted. If NULL, the TFR is plotted. Allowed values are any of those returned by tfr.parameter.names.cs.extended() and 'lambda' (see Details).
adjusted	Logical indicating if the measure to be plotted is based on adjusted TFRs.
projection.index	Index of the projection to be displayed. It is only relevant if year is NULL. projection.index=1 means the present year, projection.index=2 means the first projection period after present year, etc..

device	Device for displaying the map. It is passed to the mapDevice function of the **rworldmap** package. If it is equal to 'dev.cur', the current device is used. Otherwise, it can be 'dev.new', 'png', 'pdf' etc.
main	Title for the map. If it is NULL, a default title is constructed from the projection year and quantile.
resolution	Map resolution as implemented in getMap. High resolution requires the **rworldxtra** package.
device.args	List of additional arguments to be passed to the mapDevice function of the **rworldmap** package.
data.args	List of additional arguments to be passed to the underlying data retrieving function.
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".
tfr.range	Range of the total fertility rate to be displayed. It is of the form c(*tfr.min*, *tfr.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 projection years of this prediction object should be on the same color scale.
file.prefix	Prefix for file names.
...	Arguments passed to the mapCountryData function of the **rworldmap** package. In case of tfr.map.gvis these are passed to the underlying data retrieving function (the same as data.args).
pi	Probability interval to be shown when a country is selected in an interactive map. The corresponding quantiles must be available (see argument quantile above).

Details

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

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

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

By default, both tfr.map and tfr.map.gvis produce maps of TFRs. Alternatively, the functions can be used to plot country-specific Phase II MCMC parameters into a world map. They are given by the argument par.name. In addition to the MCMC parameters, if par.name='lambda', the period of the end of TFR decline (i.e. start of Phase III) is computed for each country and projected into the map. In such a case, we recommend to adjust the color scale in tfr.map e.g. using the arguments catMethod='pretty' and numCats=20 (see mapCountryData).

Value

`get.tfr.map.parameters` returns a list with elements:

`pred`	The object of class `bayesTFR.prediction` used in the function.
`quantile`	Value of the argument `quantile`.
`catMethod`	If the argument `same.scale` is `TRUE`, this element contains breakpoints for categorization. It is generated from a fitted gamma distribution. Otherwise, it is `NULL`.
`numCats`	Number of categories.
`coulourPalette`	Subset of the rainbow palette, starting from dark blue and ending at red.
`...`	Additional arguments passed to the function.

Author(s)

Hana Sevcikova, Patrick Gerland, Adrian Raftery

Examples

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

# Using Google Vizualization tool
tfr.map.gvis(pred)

## End(Not run)
```

`tfr.median.set` *Editing Medians of the Projection*

Description

These functions are to be used by expert analysts. They allow to change the projection medians either to specific values or shift the medians by a given constant, or by a specific adjusting procedure.

Usage

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

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

tfr.median.adjust(sim.dir, countries, factor1 = 2/3, factor2 = 1/3, forceAR1 = FALSE)

tfr.median.reset(sim.dir, countries)
```

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 $t_i < t \leq t_{i+1}$.
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.
factor1, factor2	Adjusting factors for the first and second projection period, respectively (see below).
forceAR1	Logical. If TRUE, the given countries are forced to enter Phase III (i.e. the AR(1) process) in the first projection period.

Details

The function tfr.median.set can be used to set the medians of the given country to specific values. Function tfr.median.shift can be used to offset the medians by a specific constant, or to reset the medians to their original BHM values. Function tfr.median.adjust runs the prediction procedure for the given countries with an additional decrement in the model in the first two projection periods. In the first projection period it is computed as factor1*S where S is a difference between observed decrement and the expected decrement (by the double logistic function) in the last observed period. In the second projection period, in the formula factor1 is replaced by factor2. If forceAR1 is set to TRUE, we recommend to set factor1 and factor2 to 0. The function then calls tfr.median.set in order to store the new median for each country. Function tfr.median.reset resets medians of the given countries to the original values.

In all four functions, if a median is modified, the corresponding offset is stored in the prediction object (element `median.shift`) and the updated prediction object is written 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 `bayesTFR.prediction`.

Author(s)

Hana Sevcikova, Leontine Alkema

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

Description

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

Usage

```
tfr.parameter.names(trans = NULL)
tfr.parameter.names.cs(country.code = NULL, trans = NULL, back.trans = TRUE)
tfr.parameter.names.extended()
tfr.parameter.names.cs.extended(country.code = NULL)

tfr3.parameter.names()
tfr3.parameter.names.cs(country.code = NULL)
```

Arguments

`trans`	It can be `NULL` or logical. If `TRUE`, names of the transformable parameters (i.e. 'alpha' in case of country-independent parameters, or 'gamma' in case of country-specific parameters) are replaced by the names of the transformed parameters (i.e. 'alphat', or 'gammat'). If `trans=FALSE`, there is no such replacement. If `trans=NULL`, all parameter names, including the transformable parameters are returned.
`country.code`	Country code. If it is given, the country-specific parameter names contain the postfix '_cx' where x is the `country.code`.
`back.trans`	Logical indicating if back-transformable parameter names (i.e. 'Triangle_c1', ..., 'Triangle_c3') should be returned.

Value

`tfr.parameter.names` returns names of the country-independent Phase II parameters.
`tfr.parameter.names.cs` returns names of the country-specific Phase II parameters.
`tfr.parameter.names.extended` returns names of all country-independent Phase II parameters, including the transformed parameters. Parameters 'alpha', 'delta', 'alphat', and'deltat' are in their extended format with the postfix '_1', '_2' and '_3'.
`tfr.parameter.names.cs.extended` returns names of all country-specific Phase II parameters, including the transformed parameters. Parameters 'gamma' and'gammat' are in their extended format with the postfix '_1', '_2' and '_3'.
`tfr3.parameter.names` returns names of the country-independent Phase III parameters.
`tfr3.parameter.names.cs` returns names of the country-specific Phase III parameters.

Author(s)

Hana Sevcikova

Examples

```
tfr.parameter.names()
tfr.parameter.names.extended()
tfr.parameter.names.cs()
tfr.parameter.names.cs.extended()
tfr3.parameter.names()
tfr3.parameter.names.cs()
```

---

tfr.pardensity.plot *Plotting MCMC Parameter Density*

Description

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

Usage

```
tfr.pardensity.plot(mcmc.list = NULL,
    sim.dir = file.path(getwd(), "bayesTFR.output"),
    chain.ids = NULL, par.names = tfr.parameter.names(trans = TRUE),
    burnin = NULL, dev.ncol=5, low.memory = TRUE, ...)

tfr.pardensity.cs.plot(country, mcmc.list=NULL,
    sim.dir=file.path(getwd(), "bayesTFR.output"),
    chain.ids=NULL, par.names=tfr.parameter.names.cs(trans=TRUE),
    burnin=NULL, dev.ncol=3, low.memory=TRUE, ...)

tfr3.pardensity.plot(mcmc.list = NULL,
    sim.dir = file.path(getwd(), "bayesTFR.output"),
```

```
    chain.ids = NULL, par.names = tfr3.parameter.names(),
    burnin = NULL, dev.ncol=3, low.memory = TRUE, ...)

tfr3.pardensity.cs.plot(country, mcmc.list=NULL,
    sim.dir=file.path(getwd(), "bayesTFR.output"),
    chain.ids=NULL, par.names=tfr3.parameter.names.cs(),
    burnin=NULL, dev.ncol=2, low.memory=TRUE, ...)
```

Arguments

country	Name or numerical code of a country.
mcmc.list	List of bayesTFR.mcmc objects, or an object of class bayesTFR.mcmc.set or of class bayesTFR.prediction (allowed only for Phase II MCMCs). 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 (possibly transformed) country-independent parameters are plotted if used within tfr.pardensity.plot and tfr3.pardensity.plot, or country-specific parameters are plotted if used within tfr.pardensity.cs.plot and tfr3.pardensity.cs.plot.
burnin	Number of iterations to be discarded from the beginning of each chain.
dev.ncol	Number of column for the graphics device. If the number of parameters is smaller than dev.ncol, the number of columns is automatically decreased.
low.memory	Logical indicating if the processing should run in a low-memory mode. If it is FALSE, traces of all available parameters are loaded into memory. Otherwise, parameters are loaded as they are needed and are not kept in the memory.
...	Further arguments passed to the density function.

Details

The functions plot the density of the posterior distribution either for country-independent parameters (tfr.pardensity.plot for phase II MCMCs and tfr3.pardensity.plot for phase III MCMCs) or for country-specific parameters (tfr.pardensity.cs.plot for phase II and tfr3.pardensity.cs.plot for phase III), 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 bayesTFR.prediction (which is allowed in tfr.pardensity.plot and tfr.pardensity.cs.plot only) and if this object contains thinned traces, they are used instead of the full chains. In such a case, burnin and chain.ids cannot be modified - their value is set to the one used when the thinned traces were created, namely when running tfr.predict. In a situation with long MCMC chains, this approach can significantly speed-up creation of the density plots.

Author(s)

Hana Sevcikova

See Also

tfr.partraces.plot

Examples

```
## Not run:
sim.dir <- file.path(find.package("bayesTFR"), "ex-data", "bayesTFR.output")
tfr.pardensity.plot(sim.dir=sim.dir)
tfr.pardensity.cs.plot(country="Ireland", sim.dir=sim.dir, bw=0.2)

## End(Not run)
```

---

tfr.partraces.plot *Plotting MCMC Parameter Traces*

Description

Functions for plotting the MCMC parameter traces.

Usage

```
tfr.partraces.plot(mcmc.list = NULL,
    sim.dir = file.path(getwd(), "bayesTFR.output"), chain.ids = NULL,
    par.names = tfr.parameter.names(trans = TRUE),
    nr.points = NULL, dev.ncol=5, low.memory = TRUE, ...)

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

tfr3.partraces.plot(mcmc.list = NULL,
    sim.dir = file.path(getwd(), "bayesTFR.output"), chain.ids = NULL,
    par.names = tfr3.parameter.names(),
    nr.points = NULL, dev.ncol=3, low.memory = TRUE, ...)

tfr3.partraces.cs.plot(country, mcmc.list = NULL,
    sim.dir = file.path(getwd(), "bayesTFR.output"), chain.ids = NULL,
    par.names = tfr3.parameter.names.cs(),
    nr.points = NULL, dev.ncol=2, low.memory = TRUE, ...)
```

Arguments

country	Name or numerical code of a country.
mcmc.list	List of bayesTFR.mcmc objects, or an object of class bayesTFR.mcmc.set or of class bayesTFR.prediction (allowed only for Phase II MCMCs). If it is NULL, the traces are loaded from sim.dir.
sim.dir	Directory with the MCMC simulation results. It is only used if mcmc.list is NULL.
chain.ids	List of MCMC identifiers to be plotted. If it is NULL, all chains found in mcmc.list or sim.dir are plotted.
par.names	Names of parameters for which traces should be plotted. By default all (possibly transformed) country-independent parameters are plotted if used within tfr.partraces.plot and tfr3.partraces.plot, or country-specific parameters are plotted if used within tfr.partraces.cs.plot and tfr3.partraces.cs.plot.
nr.points	Number of points to be plotted. If NULL, all points are plotted, otherwise the traces are thinned evenly.
dev.ncol	Number of column for the graphics device. If the number of parameters is smaller than dev.ncol, the number of columns is automatically decreased.
low.memory	Logical indicating if the processing should run in a low-memory mode. If it is FALSE, traces of all available parameters are loaded into memory. Otherwise, parameters are loaded as they are needed and are not kept in the memory.
...	Additional graphical arguments.

Details

The functions plot MCMC traces either for country-independent parameters (tfr.partraces.plot for phase II MCMCs and tfr3.partraces.plot for phase III MCMCs) or for country-specific parameters (tfr.partraces.cs.plot for phase II MCMCs and tfr3.partraces.cs.plot for phase III MCMCs), 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

coda.list.mcmc for retrieving raw values of the traces.

Examples

```
## Not run:
sim.dir <- file.path(find.package("bayesTFR"), "ex-data", "bayesTFR.output")
tfr.partraces.plot(sim.dir=sim.dir)
tfr.partraces.cs.plot(country="Netherlands", sim.dir=sim.dir)

## End(Not run)
```

tfr.predict *Generating Posterior Trajectories of the Total Fertility Rate*

Description

Using the posterior parameter samples simulated by run.tfr.mcmc (and possibly run.tfr3.mcmc) the function generates posterior trajectories for the total fertility rate for all countries of the world.

Usage

```
tfr.predict(mcmc.set = NULL, end.year = 2100,
    sim.dir = file.path(getwd(), "bayesTFR.output"),
    replace.output = FALSE, start.year = NULL,
    nr.traj = NULL, thin = NULL, burnin = 2000,
    use.diagnostics = FALSE, use.tfr3 = TRUE, burnin3 = 10000,
    mu = 2.1, rho = 0.8859, sigmaAR1 = 0.1016, min.tfr = 0.5,
    use.correlation = FALSE, save.as.ascii = 1000, output.dir = NULL,
    low.memory = TRUE, seed = NULL, verbose = TRUE, ...)
```

Arguments

mcmc.set	Object of class bayesTFR.mcmc.set corresponding Phase II MCMCs. 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.tfr.mcmc.
replace.output	Logical. If TRUE, existing predictions in output.dir will be replaced by results of this run.
start.year	Start year of the prediction. By default the prediction is started at the next time period after present.year set in the estimation step. If start.year is smaller than the default, projections for countries and time periods that have data available after start.year are set to those data.
nr.traj	Number of trajectories to be generated. If NULL, the argument thin is taken to determine the number of trajectories. If both are NULL, the number of trajectories corresponds to the size of the parameter sample.
thin	Thinning interval used for determining the number of trajectories. Only relevant, if nr.traj is NULL.
burnin	Number of iterations to be discarded from the beginning of the parameter traces.
use.diagnostics	Logical determining if an existing convergence diagnostics for phase II MCMCs 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 tfr.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.

use.tfr3	Logical determining if phase III should be predicted via MCMCs (simulated via run.tfr3.mcmc) or a classic AR(1) process. If TRUE but no phase III MCMCs were simulated, a warning is given and the prediction switches automatically to a classic AR(1) process.
burnin3	Burnin used for phase III MCMCs. Only relevant if use.tfr3 is TRUE.
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.
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.
mu	Long-term mean μ in the AR(1) projection model. Only used if use.tfr3 is FALSE.
rho	Autoregressive parameter ρ in AR(1) projection model. If it is NULL it is estimated from the data. Only used if use.tfr3 is FALSE.
sigmaAR1	Standard deviation s of AR(1) distortion terms in short-term projections. If it is NULL it is estimated from the data. It can be a single value or a vector giving the standard deviations for single projections. If the vector is shorter than the number of projections simulated via the AR(1) process, the last value is repeated for the remaining projections. In case of a single value (default), the standard deviation is kept constant over all AR(1) projections. Only used if use.tfr3 is FALSE.
min.tfr	Smallest TFR value allowed.
use.correlation	Logical. If TRUE the model errors are sampled jointly for all countries (Fosdick and Raftery, 2014).
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.
...	Further arguments passed to the underlying functions.

Details

The trajectories are generated using a distribution of country-specific decline curves (Alkema et al 2011) and either a classic AR(1) process or a country-specific AR(1) process (Raftery et al 2013). Phase II parameter samples simulated using run.tfr.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.tfr.mcmc).

If Phase III is projected using a BHM (i.e. if use.tfr3 is TRUE), parameter samples simulated via run.tfr3.mcmc are used from which burnin (given by burnin3) is discarded and the chains are

evenly thinned in a way that the total size corresponds to the final size of the Phase II parameter samples. Countries for which there are no simulated country-specific Phase III parameters (e.g. because their TFR is still in Phase II or it is an aggregated region) use samples of the "world" AR(1) parameters.

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.

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.tfr.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.projection.summary` for more details).

Value

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

`quantiles` A $n \times q \times p$ array of quantile values computed on all 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.

`trf_matrix_reconstructed` Matrix containing imputed TFR values on spots where the original TFR 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, thin, burnin3, thin3` Burnin and thin used for this prediction for Phase II and Phase III, respectively.

`end.year` The end year of this prediction.

`mu, rho, sigma_t, sigmaAR1` Parameters of the AR(1) process. `sigma_t` is a vector of actual values of the standard deviation s used for each projection.

`min.tfr` Input value of minimum threshold for TFR.

`na.index` Index of trajectories for which at least one country got `NA` values.

`mcmc.set` Object of class `bayesTFR.mcmc.set` used for this prediction, i.e. the burned, thinned, and collapsed MCMC chain.

Author(s)

Hana Sevcikova, Leontine Alkema, Bailey Fosdick

References

L. Alkema, A. E. Raftery, P. Gerland, S. J. Clark, F. Pelletier, Buettner, T., Heilig, G.K. (2011). Probabilistic Projections of the Total Fertility Rate for All Countries. Demography, Vol. 48, 815-839.

Raftery, A.E., Alkema, L. and Gerland, P. (2014). Bayesian Population Projections for the United Nations. Statistical Science, Vol. 29, 58-68.

Fosdick, B., Raftery, A.E. (2014). Regional Probabilistic Fertility Forecasting by Modeling Between-Country Correlations. Demographic Research, Vol. 30, 1011-1034.

See Also

run.tfr.mcmc, run.tfr3.mcmc, create.thinned.tfr.mcmc, convert.tfr.trajectories, write.projection.summary
get.tfr.prediction, summary.bayesTFR.prediction

Examples

```
## Not run:
sim.dir <- tempfile()
m <- run.tfr.mcmc(nr.chains=1, iter=10, output.dir=sim.dir, verbose=TRUE)
m3 <- run.tfr3.mcmc(sim.dir=sim.dir, nr.chains=2, iter=40, thin=1, verbose=TRUE)
pred <- tfr.predict(m, burnin=0, burnin3=10, verbose=TRUE)
summary(pred, country="Iceland")
unlink(sim.dir, recursive=TRUE)

## End(Not run)
```

---

tfr.predict.extra — *Generating Posterior Trajectories of the Total Fertility Rate for Specific Countries or Regions*

Description

Using the posterior parameter samples the function generates posterior trajectories of the total fertility rate for given countries or regions. It is intended to be used after running run.tfr.mcmc.extra, but it can be also used for purposes of testing specific settings on one or a few countries.

Usage

```
tfr.predict.extra(sim.dir = file.path(getwd(), 'bayesTFR.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 converions is desired. Note that the convertion is done on all countries.
verbose	Logical switching log messages on and off.

Details

In order to use this function, a prediction object must exist, i.e. the function tfr.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.

Value

Updated object of class bayesTFR.prediction.

Author(s)

Hana Sevcikova

See Also

tfr.predict

---

tfr.predict.subnat *Generating Posterior Trajectories of Subnational TFR*

Description

Generates posterior trajectories of the total fertility rate for subregions of given countries, using the Scale-AR(1) method.

Usage

```
tfr.predict.subnat(countries, my.tfr.file,
                   sim.dir = file.path(getwd(), "bayesTFR.output"),
                   end.year = 2100, start.year = NULL, output.dir = NULL,
                   nr.traj = NULL, seed = NULL, min.tfr = 0.5,
                   ar.pars = c(mu = 1, rho = 0.92464, sigma = 0.04522),
                   save.as.ascii = 0, verbose = TRUE)
```

Arguments

countries	Vector of numerical country codes or country names.
my.tfr.file	Tab-separated ASCII file containing the subnational TFR data. See Details for more information on its format.
sim.dir	Simulation directory with the national projections generated using tfr.predict.
end.year	End year of the projections.
start.year	Start year of the projections. By default, projections start at the same time point as the national projections.
output.dir	Directory into which the resulting prediction objects and the trajectories are stored. See below for details.
nr.traj	Number of trajectories to be generated. If NULL, the number of trajectories in the national projections is used.
seed	Seed of the random number generator. If NULL no seed is set. It can be used to generate reproducible projections.
min.tfr	Lower bound on TFR.
ar.pars	List containing the parameter estimates of the AR(1) process. It must have elements called mu, rho and sigma.
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. By default no conversion is performed.
verbose	Logical switching log messages on and off.

Details

The function implements the methodology described in Sevcikova et al (2017). Given a set of national bayesTFR projections, it applies the Scale-AR(1) model to each national trajectory and each subregion of given countries which yield subnational TFR projections.

The file on subnational data passed in my.tfr.file has to have a column "country_code" with numerical values corresponding to countries given in the argument countries, and column "reg_code" giving the numerical identifier of each subregion. Column "name" should be used for subregion name, and column "country" for country name. An optional column "include_code" can be used to eliminate entries from processing. Entries with values of 1 or 2 will be included, all others will be ignored. Column "last.observed" can be used to define which time period contains the last observed data point (given as integer, e.g. year in the middle of the time period). Remaining columns define the time periods, e.g. "2000-2005", "2005-2010". The package contains an example of such dataset, see Example below.

Argument output.dir gives a location on disk where results of the function should be stored. If it is NULL (default), results are stored in the same directory as the national projections. In both cases a subdirectory called "subnat" is created in which each country has its own subfolder with the country code in its name. Each such subfolder contains the same type of outputs as in the national case generated using tfr.predict, most importantly a directory "predictions" with trajectories for each region.

Value

A list of objects of class bayesTFR.prediction. The name of each element includes its country code. Not all elements of the class bayesTFR.prediction are available. For example, no mcmc.set is attached to these objects. Thus, not all functions that work with bayesTFR.prediction can be applied to these results.

Note

Even though the resulting object contains subnational results, the names of its elements are the same as in the national case. This allows to apply the same functions on both objects (subnational and national). However, it means that sometimes the meaning of the elements or function arguments does not match the subnational context. For example, various functions expect the argument country. When a subnational object is passed to such a function, country means a subregion.

Author(s)

Hana Sevcikova

References

Hana Sevcikova, Adrian E. Raftery, Patrick Gerland (2017). Probabilistic Projection of Subnational Total Fertility Rates. arXiv:1701.01787, https://arxiv.org/abs/1701.01787.

See Also

get.regtfr.prediction, tfr.predict

Examples

```
# View the example data
my.subtfr.file <- file.path(find.package("bayesTFR"), 'extdata', 'subnational_tfr_template.txt')
subtfr <- read.delim(my.subtfr.file, check.names=FALSE)
head(subtfr)

# Directory with national projections (contains 30 trajectories for each country)
nat.dir <- file.path(find.package("bayesTFR"), "ex-data", "bayesTFR.output")

# Subnational projections for Australia and Canada
subnat.dir <- tempfile()
preds <- tfr.predict.subnat(c(36, 124), my.tfr.file=my.subtfr.file,
    sim.dir=nat.dir, output.dir=subnat.dir, start.year=2013)
names(preds)
get.countries.table(preds[["36"]])
summary(preds[["36"]], "Queensland")
tfr.trajectories.plot(preds[["36"]], "Queensland")

# plot subnational and national TFR in one plot
nat.pred <- get.tfr.prediction(nat.dir)
tfr.trajectories.plot(preds[["36"]], 186, pi=80, half.child.variant=FALSE)
tfr.trajectories.plot(nat.pred, "Australia", half.child.variant=FALSE,
      add=TRUE, col=rep("darkgreen", 5), nr.traj=0, show.legend=FALSE)
```

```
legend("topright", c("regional TFR", "national TFR"), col=c("red", "darkgreen"),
  lty=1, bty='n')

# Retrieve trajectories
trajs.Alberta <- get.tfr.trajectories(preds[["124"]], "Alberta")
summary(t(trajs.Alberta))

# cleanup
unlink(subnat.dir)

# See more examples in ?get.regtfr.prediction
```

`tfr.raftery.diag` *Raftery Diagnostics for Parameters of the Total Fertility Rate*

Description

The functions compute the Raftery diagnostics for each parameter of MCMCs of phase II (`tfr.raftery.diag`) and phase III (`tfr3.raftery.diag`), taking median over all chains.

Usage

```
tfr.raftery.diag(mcmc = NULL,
    sim.dir = file.path(getwd(), "bayesTFR.output"),
    burnin = 0, country = NULL,
    par.names = tfr.parameter.names(trans = TRUE),
    par.names.cs = tfr.parameter.names.cs(trans = TRUE, back.trans = FALSE),
    country.sampling.prop = 1, verbose=TRUE, ...)

tfr3.raftery.diag(mcmc = NULL,
    sim.dir = file.path(getwd(), "bayesTFR.output"),
    burnin = 0, country = NULL,
    par.names = tfr3.parameter.names(),
    par.names.cs = tfr3.parameter.names.cs(),
    country.sampling.prop = 1, verbose=TRUE, ...)
```

Arguments

`mcmc`	A `bayesTFR.mcmc` or `bayesTFR.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, country-specific parameters are reduced to parameters of that country.
`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 `coda.list.mcmc` function.

Details

The Raftery diagnostics is computed for each parameter, using **coda**'s `raftery.diag` with r=0.0125, q=0.025 and q=0.975. Values of N and burnin are taken as the median over chains. For each country-specific parameter, the maximum over all included countries of such medians is taken.

Value

List with the components:

`Nmedian`
: 2-d array of N values (processed as described in Details) with two rows: first corresponding to q=0.025, second corresponding to q=0.975. Each column corresponds to one parameter.

`burnin`
: 2-d array of the same structure as `Nmedian`, containing the burnin values (processed as described in Details).

`not.converged.parameters`
: List with two elements, each of which is a data frame containing columns "parameter.name", "chain.id", and "N". These are parameters for which the computed value of Raftery diagnostics N is larger than the total number of finished iterations summed over all chains. The first element of the list corresponds to q=0.025, second corresponds to q=0.975.

`not.converged.inchain.parameters`
: List of the same structure as `not.converged.parameters`. The parameters included are those for which the computed value of Raftery diagnostics N is larger than the number of finished iterations in the corresponding chain.

`N.country.indep`
: Data frame containing columns "parameter.name", "chain.id", "N0.025", and "N0.975". Each row gives N computed with the two different q for each country-independent parameter and chain.

`N.country.spec`
: The same as `N.country.indep`, but here the country-specific parameters are considered.

`Nmedian.country.spec`
: 2-d array of N values for country-specific parameters containing medians over chains.

`thin.ind`
: List with elements '0.025', '0.975' and `median`. The first two elements are matrices with one row per chain and one column per parameter. They contain values of `thin` that makes the MCMC independent, for q=0.025 and q=0.975, respectively. The `median` element is of the same structure as `Nmedian`, containing medians ove rows in the two matrices in this list.

`nr.countries`
: Vector with elements `used` (number of countries used in this diagnostics) and `total` (number of countries that this `mcmc` object was estimated on).

Author(s)

Hana Sevcikova, Adrian Raftery

See Also

raftery.diag

---

tfr.trajectories.plot *Output of Posterior Distribution of TFR Trajectories*

Description

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

Usage

```
tfr.trajectories.plot(tfr.pred, country, pi = c(80, 95),
    half.child.variant = TRUE, nr.traj = NULL,
    adjusted.only = TRUE, typical.trajectory = FALSE,
    mark.estimation.points = FALSE,
    xlim = NULL, ylim = NULL, type = 'b', xlab = 'Year', ylab = 'TFR',
    main = NULL, lwd = c(2, 2, 2, 2, 2, 1),
    col=c('black', 'green', 'red', 'red', 'blue', '#00000020'),
    show.legend = TRUE, add = FALSE, ...)

tfr.trajectories.plot.all(tfr.pred,
    output.dir = file.path(getwd(), 'TFRtrajectories'),
    output.type = "png", main = NULL, verbose = FALSE, ...)

tfr.trajectories.table(tfr.pred, country, pi = c(80, 95),
    half.child.variant = TRUE)
```

Arguments

tfr.pred	Object of class bayesTFR.prediction.
country	Name or numerical code of a country.
pi	Probability interval. It can be a single number or an array.
half.child.variant	If TRUE the United Nations variant of "+/-0.5 child" (relative to the median) is shown.
nr.traj	Number of trajectories to be plotted. If NULL, all trajectories are plotted, otherwise they are thinned evenly.
adjusted.only	Logical. By default, if the projection median is adjusted using e.g. tfr.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 that has the smallest distance to the median.

`mark.estimation.points`
: Logical. If `TRUE`, points that were not used in the estimation (phase I) are shown in lighter color than points in phase II and III.

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

`lwd, col`
: Vector of six elements giving the line width and color for: 1. observed data, 2. imputed missing data, 3. median, 4. quantiles, 5. half-child variant, 6. trajectories.

`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 `tfr.trajectories.plot.all`, `...` contains any of the arguments of `tfr.trajectories.plot`.

`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".

`main`
: Main title for the plot(s). In `tfr.trajectories.plot.all` any occurence of the string "XXX" is replaced by the country name.

`verbose`
: Logical switching log messages on and off.

Details

`tfr.trajectories.plot` plots posterior distribution of TFR trajectories for a given country. `tfr.trajectories.table` gives the same output as a table. `tfr.trajectories.plot.all` creates a set of 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, Leontine Alkema

See Also

`bayesTFR.prediction`

Examples

```
## Not run:
sim.dir <- file.path(find.package("bayesTFR"), "ex-data", "bayesTFR.output")
pred <- get.tfr.prediction(sim.dir)
tfr.trajectories.plot(pred, country="Burkina Faso", pi=c(80, 95))
```

```
tfr.trajectories.table(pred, country="Burkina Faso", pi=c(80, 95))

## End(Not run)
```

`UN_time` *Dataset with UN-specific Time Coding*

Description

Dataset used by the UN for coding time. It is an TAB-separated ASCII file called "UN_time.txt".

Usage

```
data(UN_time)
```

Format

A data frame with 1034 observations on the following 4 variables.

`TimeID` Time identifier.

`TLabel` Label of the time, with minimum values of `1950` and `1950-1955`, and maximum values of `2399`, `2400` and `2400-2405`.

`TDate` Equal to `TLabel` if it is a single year, or the starting year of `TLabel`, if it is an interval.

`Tinterval` Length of the time interval, or zero, if it is a single year.

Details

For 5-year period data, fertility rates are defined from 1 July of year (t) to 1 July of year (t+5), with 1 January of year (t+3) as exact mid-date. This means for example that data for 2000-2005, refer to the period between 2000.5 and 2005.5, with 2003.0 as exact mid-point.

Source

Data provided by the United Nations Population Division

Examples

```
data(UN_time)
str(UN_time)
```

UN_variants *Dataset with UN-specific Coding of Variants*

Description

Dataset used by the UN for coding variants. It also includes variants for the lower and upper bounds of the 80 and 95% probability intervals, respectively, resulting from the Bayesian hierarchical model. The dataset is stored in a TAB-separated ASCII file called "UN_variants.txt".

Usage

```
data(UN_variants)
```

Format

A data frame with 23 observations on the following 5 variables.

`RevID` Revision identifier.

`VarID` Identifier of the variant.

`Vshort` Short name of the variant.

`VName` Full name of the variant.

`VariantDomain` Domain of the variant.

Source

Data provided by the United Nations Population Division

Examples

```
data(UN_variants)
str(UN_variants)
```

write.projection.summary *Writing Projection Summary Files*

Description

The function creates two files containing projection summaries, such as the median, the lower and upper bound of the 80 and 90% probability intervals, respectively, the +/- 0.5 child variant 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. In addition, a file containing some of the model parameters is created.

Usage

```
write.projection.summary(dir = file.path(getwd(), "bayesTFR.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 tfr.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 WPP 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. tfr.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' (or 'projection_summary_user_friendly_adjusted.csv' if adjusted=TRUE), 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", "-0.5child", "+0.5child", "constant"
- period1: e.g. "2005-2010": TFR for the first time period
- period2: e.g. "2010-2015": TFR for the second time period
- ... further columns with TFR projections

The second file, called 'projection_summary.csv' (or 'projection_summary_adjusted.csv' if adjusted=TRUE), 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
- "LocID": country code
- "TimeID": time identifier, extracted from the UN_time dataset
- "TFR": the total fertility rate for this variant, location and time period

The third comma-separated file, called 'projection_summary_parameters.csv' contains the following columns:

- "country_name": country name
- "country_code": country code

- "TF_time_start_decline": start period of TFR decline
- "TF_max": TFR at the onset of the fertitility transition (median of the U_c parameter)
- "TF_max_decrement": maximum decrement of TFR decline (median of the d_c parameter)
- "TF_end_level": median of the end level of the fertility transition (Δ_{c4} parameter)
- "TF_end_level_low": 2.5 percentile of the Δ_{c4} distribution
- "TF_end_level_high": 97.5 percentile of the Δ_{c4} distribution
- "TF_time_end_decline": period of the end decline, measured using the prediction median

Note that this file is not created if `adjusted=TRUE`.

Note

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

Author(s)

Hana Sevcikova

See Also

`convert.tfr.trajectories`, `tfr.predict`

Index