Package 'demography'

July 22, 2025

Version 2.0

Title Forecasting Mortality, Fertility, Migration and Population Data

Description Functions for demographic analysis including lifetable calculations; Lee-Carter modelling; functional data analysis of mortality rates, fertility rates, net migration numbers; and stochastic population forecasting.

Depends R (>= 3.4), forecast (>= 8.5)

Imports ftsa (>= 4.8), rainbow, cobs, mgcv, strucchange, HMDHFDplus (>= 2.0.0)

LazyData yes

ByteCompile TRUE

URL https://pkg.robjhyndman.com/demography/, https://github.com/robjhyndman/demography

BugReports https://github.com/robjhyndman/demography/issues

License GPL (>= 3)

Encoding UTF-8

RoxygenNote 7.2.3

NeedsCompilation no

Author Rob Hyndman [aut, cre, cph] (ORCID: <https://orcid.org/0000-0002-2140-5352>), Heather Booth [ctb] (ORCID: <https://orcid.org/0000-0002-8356-0534>), Leonie Tickle [ctb] (ORCID: <https://orcid.org/0000-0002-6612-2401>), John Maindonald [ctb], Simon Wood [ctb], R Core Team [ctb]

Maintainer Rob Hyndman <Rob.Hyndman@monash.edu>

Repository CRAN

Date/Publication 2023-02-08 08:20:02 UTC

Contents

demography-package *Forecasting mortality and fertility data*

Description

Functions for demographic analysis including lifetable calculations, Lee-Carter modelling and functional data analysis of mortality rates.

Author(s)

Rob J Hyndman with contributions from Heather Booth, Leonie Tickle, John Maindonald, Simon Wood and the R Core Team.

Maintainer: <Rob.Hyndman@monash.edu>

---

as.data.frame.demogdata *Coerce a demogdata object to a data.frame object*

Description

Coerce a demogdata object to a data.frame object

Usage

```
## S3 method for class 'demogdata'
as.data.frame(x, ...)
```

Arguments

x	Object to be coerced to a data frame.
...	Other arguments not used

Value

A data.frame object.

Examples

```
# coerce demogdata object to data.frame ----
as.data.frame(fr.mort)
```

`aus.fert` *Australian fertility data*

Description

Age-specific fertility rates and female child-bearing population for Australia.

Format

Object of class `demogdata` containing the following components:

year Vector of years

age Vector of ages

rate List containing one matrix with one age group per row and one column per year.

pop Population data in same form as `rate`.

type Type of object. In this case, "fertility".

label Character string giving area from which data are taken. In this case, "Australia".

Details

Australian fertility rates and populations (1921-2002) for age groups (<20, 20-24, 25-29, 30-34, 35-39, 40-44, 45+). Data taken from v3.2b of the Australian Demographic Data Bank released 10 February 2005.

Author(s)

Rob J Hyndman

Source

The Australian Demographic Data Bank (courtesy of Len Smith).

Examples

```
plot(aus.fert)
```

`cm.spline` *Monotonic interpolating splines*

Description

Perform cubic spline monotonic interpolation of given data points, returning either a list of points obtained by the interpolation or a function performing the interpolation. The splines are constrained to be monotonically increasing (i.e., the slope is never negative).

Usage

```
cm.spline(x, y = NULL, n = 3 * length(x), xmin = min(x), xmax = max(x), ...)

cm.splinefun(x, y = NULL, ...)
```

Arguments

`x`, `y`	vectors giving the coordinates of the points to be interpolated. Alternatively a single plotting structure can be specified: see `xy.coords`.
`n`	interpolation takes place at n equally spaced points spanning the interval [`xmin`, `xmax`].
`xmin`	left-hand endpoint of the interpolation interval.
`xmax`	right-hand endpoint of the interpolation interval.
`...`	Other arguments are ignored.

Details

These are simply wrappers to the `splinefun` function family from the stats package.

Value

`cm.spline`	returns a list containing components `x` and `y` which give the ordinates where interpolation took place and the interpolated values.
`cm.splinefun`	returns a function which will perform cubic spline interpolation of the given data points. This is often more useful than `spline`.

Author(s)

Rob J Hyndman

References

Forsythe, G. E., Malcolm, M. A. and Moler, C. B. (1977) *Computer Methods for Mathematical Computations*. Hyman (1983) *SIAM J. Sci. Stat. Comput.* **4**(4):645-654. Dougherty, Edelman and Hyman 1989 *Mathematics of Computation*, **52**: 471-494.

Examples

```
x <- seq(0,4,l=20)
y <- sort(rnorm(20))
plot(x,y)
lines(spline(x, y, n = 201), col = 2) # Not necessarily monotonic
lines(cm.spline(x, y, n = 201), col = 3) # Monotonic
```

`coherentfdm` *Coherent functional demographic model for grouped data*

Description

Fits a coherent functional model to demographic data as described in Hyndman, Booth & Yasmeen (2012). If two of the series in `data` are named `male` and `female`, then it will use these two groups. Otherwise it will use all available groups.

Usage

```
coherentfdm(data, order1 = 6, order2 = 6, ...)
```

Arguments

`data`	demogdata object containing at least two groups.
`order1`	Number of basis functions to fit to the model for the geometric mean.
`order2`	Number of basis functions to fit to the models for each ratio.
`...`	Extra arguments passed to `fdm`.

Value

A list (of class `fdmpr`) consisting of two objects: `product` (an `fdm` object containing a del for the geometric mean of the data) and `ratio` (a list of `fdm` objects, being the models for the ratio of each series with the geometric mean).

Author(s)

Rob J Hyndman

References

Hyndman, R.J., Booth, H., and Yasmeen, F. (2012) Coherent mortality forecasting: the product-ratio method with functional time series models. *Demography*, to appear. https://robjhyndman.com/publications/coherentfdm/

See Also

`fdm`, `forecast.fdmpr`

Examples

```
fr.short <- extract.years(fr.sm,1950:2006)
fr.fit <- coherentfdm(fr.short)
summary(fr.fit)
plot(fr.fit$product, components=3)
```

`combine.demogdata` *Combine two demogdata objects into one demogdata object*

Description

Function to combine demogdata objects containing different years but the same age structure into one demogdata object. The standard use for this function will be combining historical data with forecasts. The objects must be of the same type.

Usage

```
combine.demogdata(obj1, obj2)
```

Arguments

`obj1`	First demogdata object (e.g., historical data).
`obj2`	Second demogdata object (e.g., forecasts).

Value

Object of class "demogdata" with the following components:

`year`	Vector of years
`age`	Vector of ages
`rate`	Matrix of rates with with one age group per row and one column per year.
`pop`	Matrix of populations in same form as `rate` and containing population numbers. This is only produced when both objects contain a `pop` component.
`type`	Type of object: "mortality", "fertility" or "migration".
`label`	Name of area from which the data are taken.

Author(s)

Rob J Hyndman

See Also

`demogdata`

Examples

```
fit <- fdm(fr.mort)
fcast <- forecast(fit, h=50)
france2 <- combine.demogdata(fr.mort,fcast)
plot(france2)
plot(life.expectancy(france2))
lines(rep(max(fr.mort$year)+0.5,2),c(0,100),lty=3)
```

`compare.demogdata` *Evaluation of demographic forecast accuracy*

Description

Computes mean forecast errors and mean square forecast errors for each age level. Computes integrated squared forecast errors and integrated absolute percentage forecast errors for each year.

Usage

```
compare.demogdata(
  data,
  forecast,
  series = names(forecast$rate)[1],
  ages = data$age,
  max.age = min(max(data$age), max(forecast$age)),
  years = data$year,
  interpolate = FALSE
)
```

Arguments

`data`	Demogdata object such as created using `read.demogdata` containing actual demographic rates.
`forecast`	Demogdata object such as created using `forecast.fdm` or `forecast.lca`.
`series`	Name of series to use. Default: the first matrix within `forecast$rate`.
`ages`	Ages to use for comparison. Default: all available ages.
`max.age`	Upper age to use for comparison.
`years`	Years to use in comparison. Default is to use all available years that are common between data and forecast.
`interpolate`	If TRUE, all zeros in data are replaced by interpolated estimates when computing the error measures on the log scale. Error measures on the original (rate) scale are unchanged.

Value

Object of class "errorfdm" with the following components:

`label`	Name of region from which data taken.
`age`	Ages from `data` object.
`year`	Years from `data` object.
`<error>`	Matrix of forecast errors on rates.
`<logerror>`	Matrix of forecast errors on log rates.
`mean.error`	Various measures of forecast accuracy averaged across years. Specifically ME=mean error, MSE=mean squared error, MPE=mean percentage error and MAPE=mean absolute percentage error.
`int.error`	Various measures of forecast accuracy integrated across ages. Specifically IE=integrated error, ISE=integrated squared error, IPE=integrated percentage error and IAPE=integrated absolute percentage error.
`life.expectancy`	If `data$type="mortality"`, function returns this component which is a matrix containing actual, forecast and actual-forecast for life expectancies.

Note that the error matrices have different names indicating if the series forecast was male, female or total.

Author(s)

Rob J Hyndman

See Also

forecast.fdm,plot.errorfdm

Examples

```
fr.test <- extract.years(fr.sm,years=1921:1980)
fr.fit <- fdm(fr.test,order=2)
fr.error <- compare.demogdata(fr.mort, forecast(fr.fit,20))
plot(fr.error)
par(mfrow=c(2,1))
plot(fr.error$age,fr.error$mean.error[,"ME"],
     type="l",xlab="Age",ylab="Mean Forecast Error")
plot(fr.error$int.error[,"ISE"],
     xlab="Year",ylab="Integrated Square Error")
```

demogdata *Create demogdata object from raw data matrices*

Description

Create demogdata object suitable for plotting using `plot.demogdata` and fitting an LC or BMS model using `lca` or an FDA model using `fdm`.

Usage

```
demogdata(data, pop, ages, years, type, label, name, lambda)
```

Arguments

`data`	Matrix of data: either mortality rates or fertility rates
`pop`	Matrix of population values of same dimension as data. These are population numbers as at 30 June of each year (i.e., the "exposures"). So, for example, the number of deaths is data*pop if data contains mortality rates.
`ages`	Vector of ages corresponding to rows of `data`.
`years`	Vector of years corresponding to columns of `data`.
`type`	Character string showing type of demographic series: either “mortality”, “fertility” or “migration”.
`label`	Character string of the name of area from which the data are taken.
`name`	Name of series: usually male, female or total.
`lambda`	Box-Cox transformation parameter.

Value

Object of class “demogdata” with the following components:

`year`	Vector of years
`age`	Vector of ages
`rate`	A list containing one or more rate matrices with one age group per row and one column per year.
`pop`	A list of the same form as `rate` but containing population numbers instead of demographic rates.
`type`	Type of object: “mortality”, “fertility” or “migration”.
`label`	label
`lambda`	lambda

Author(s)

Rob J Hyndman

See Also

read.demogdata

---

extract.ages *Extract some ages from a demogdata object*

Description

Creates subset of demogdata object.

Usage

```
extract.ages(data, ages, combine.upper = TRUE)
```

Arguments

data	Demogdata object such as created using read.demogdata or smooth.demogdata.
ages	Vector of ages to extract from data.
combine.upper	If TRUE, ages beyond the maximum of ages are combined into the upper age group.

Value

Demogdata object with same components as data but with a subset of ages.

Author(s)

Rob J Hyndman

Examples

```
france.teens <- extract.ages(fr.mort,13:19,FALSE)
plot(france.teens)
```

extract.years *Extract some years from a demogdata object*

Description

Creates subset of demogdata object.

Usage

```
extract.years(data, years)
```

Arguments

data	Demogdata object such as created using read.demogdata or smooth.demogdata.
years	Vector of years to extract from data.

Value

Demogdata object with same components as data but with a subset of years.

Author(s)

Rob J Hyndman

Examples

```
france.1918 <- extract.years(fr.mort,1918)
```

fdm *Functional demographic model*

Description

Fits a basis function model to demographic data. The function uses optimal orthonormal basis functions obtained from a principal components decomposition.

Usage

```
fdm(
  data,
  series = names(data$rate)[1],
  order = 6,
  ages = data$age,
  max.age = max(ages),
  method = c("classical", "M", "rapca"),
  lambda = 3,
```

```
  mean = TRUE,
  level = FALSE,
  transform = TRUE,
  ...
)
```

Arguments

data	demogdata object. Output from read.demogdata.
series	name of series within data holding rates (1x1).
order	Number of basis functions to fit.
ages	Ages to include in fit.
max.age	Maximum age to fit. Ages beyond this are collapsed into the upper age group.
method	Method to use for principal components decomposition. Possibilities are "M", "rapca" and "classical". See ftsm for details.
lambda	Tuning parameter for robustness when method="M".
mean	If TRUE, will estimate mean term in the model before computing basis terms. If FALSE, the mean term is assumed to be zero.
level	If TRUE, will include an additional (intercept) term that depends on the year but not on ages.
transform	If TRUE, the data are transformed with a Box-Cox transformation before the model is fitted.
...	Extra arguments passed to ftsm.

Value

Object of class "fdm" with the following components:

label	Name of country
age	Ages from data object.
year	Years from data object.
<series>	Matrix of demographic data as contained in data. It takes the name given by the series argument.
fitted	Matrix of fitted values.
residuals	Residuals (difference between observed and fitted).
basis	Matrix of basis functions evaluated at each age level (one column for each basis function). The first column is the fitted mean.
coeffs	Matrix of coefficients (one column for each coefficient series). The first column are all ones.
mean.se	Standard errors for the estimated mean function.
varprop	Proportion of variation explained by each basis function.
weights	Weight associated with each time period.
v	Measure of variation for each time period.
type	Data type (mortality, fertility, etc.)
y	The data stored as a functional time series object.

Author(s)

Rob J Hyndman

References

Hyndman, R.J., and Ullah, S. (2007) Robust forecasting of mortality and fertility rates: a functional data approach. *Computational Statistics & Data Analysis*, **51**, 4942-4956. https://robjhyndman.com/publications/funcfor/

See Also

ftsm, forecast.fdm

Examples

```
france.fit <- fdm(fr.mort)
summary(france.fit)
plot(france.fit)
plot(residuals(france.fit))
```

---

forecast.fdm *Forecast functional demographic model.*

Description

The coefficients from the fitted object are forecast using a univariate time series model. The forecast coefficients are then multiplied by the basis functions to obtain a forecast demographic rate curve.

Usage

```
## S3 method for class 'fdm'
forecast(
  object,
  h = 50,
  level = 80,
  jumpchoice = c("fit", "actual"),
  method = "arima",
  warnings = FALSE,
  ...
)
```

Arguments

object	Output from fdm.
h	Forecast horizon.
level	Confidence level for prediction intervals.
jumpchoice	If "actual", the forecasts are bias-adjusted by the difference between the fit and the last year of observed data. Otherwise, no adjustment is used.
method	Forecasting method to be used.
warnings	If TRUE, warnings arising from the forecast models for coefficients will be shown. Most of these can be ignored, so the default is warnings=FALSE.
...	Other arguments as for forecast.ftsm.

Value

Object of class fmforecast with the following components:

label	Name of region from which the data are taken.
age	Ages from lcaout object.
year	Years from lcaout object.
rate	List of matrices containing forecasts, lower bound and upper bound of prediction intervals. Point forecast matrix takes the same name as the series that has been forecast.
error	Matrix of one-step errors for historical data
fitted	Matrix of one-step forecasts for historical data
coeff	List of objects of type forecast containing the coefficients and their forecasts.
coeff.error	One-step errors for each of the coefficients.
var	List containing the various components of variance: model, error, mean, total and coeff.
model	Fitted model in obj.
type	Type of data: “mortality”, “fertility” or “migration”.

Author(s)

Rob J Hyndman

See Also

fdm, forecast.lca, forecast.ftsm.

Examples

```
france.fit <- fdm(fr.mort,order=2)
france.fcast <- forecast(france.fit,50)
plot(france.fcast)
models(france.fcast)
```

`forecast.fdmpr` *Forecast coherent functional demographic model.*

Description

The product and ratio models from `coherentfdm` are forecast, and the results combined to give forecasts for each group in the original data.

Usage

```
## S3 method for class 'fdmpr'
forecast(object, h = 50, level = 80, K = 100, drange = c(0, 0.5), ...)
```

Arguments

`object`	Output from `coherentfdm`.
`h`	Forecast horizon.
`level`	Confidence level for prediction intervals.
`K`	Maximum number of years to use in forecasting coefficients for ratio components.
`drange`	Range of fractional differencing parameter for the ratio coefficients.
`...`	Other arguments as for `forecast.fdm`.

Value

Object of class `fmforecast2` containing a list of objects each of class `fmforecast`. The forecasts for each group in the original data are given first. Then the forecasts from the product model, and finally a list of forecasts from each of the ratio models.

Author(s)

Rob J Hyndman

See Also

`coherentfdm`, `forecast.fdm`.

Examples

```
fr.short <- extract.years(fr.sm,1950:2006)
fr.fit <- coherentfdm(fr.short)
fr.fcast <- forecast(fr.fit)
plot(fr.fcast$male)
plot(fr.fcast$ratio$male, plot.type='component', components=3)
models(fr.fcast)
```

forecast.lca *Forecast demogdata data using Lee-Carter method.*

Description

The kt coefficients are forecast using a random walk with drift. The forecast coefficients are then multiplied by bx to obtain a forecast demographic rate curve.

Usage

```
## S3 method for class 'lca'
forecast(
  object,
  h = 50,
  se = c("innovdrift", "innovonly"),
  jumpchoice = c("fit", "actual"),
  level = 80,
  ...
)
```

Arguments

object	Output from lca.
h	Number of years ahead to forecast.
se	Method used for computation of standard error. Possibilities: "innovdrift" (innovations and drift) and "innovonly" (innovations only).
jumpchoice	Method used for computation of jumpchoice. Possibilities: "actual" (use actual rates from final year) and "fit" (use fitted rates).
level	Confidence level for prediction intervals.
...	Other arguments.

Value

Object of class fmforecast with the following components:

label	Region from which the data are taken.
age	Ages from object.
year	Years from object.
rate	List of matrices containing forecasts, lower bound and upper bound of prediction intervals. Point forecast matrix takes the same name as the series that has been forecast.
fitted	Matrix of one-step forecasts for historical data

Other components included are

e0	Forecasts of life expectancies (including lower and upper bounds)

kt.f	Forecasts of coefficients from the model.
type	Data type.
model	Details about the fitted model

Author(s)

Rob J Hyndman

See Also

lca, plot.fmforecast

Examples

```
france.lca <- lca(fr.mort, adjust="e0")
france.fcast <- forecast(france.lca, 50)
plot(france.fcast)
plot(france.fcast,'c')
```

---

fr.mort *French mortality data*

Description

Age-specific mortality rates and population for France.

Format

Object of class demogdata containing the following components:

year Vector of years

age Vector of ages

rate List of matrices containing rates with with one age group per row and one column per year. Matrices: total, female, male.

pop Population data in same form as rate.

type Type of object. In this case, "mortality".

label Character string giving area from which data are taken. In this case, "France".

Details

fr.mort contains French mortality rates and populations (1899-2005) for ages 0-110. Data taken from the Human Mortality Database on 20 February 2008. fr.sm contains a smoothed version of fr.mort obtained using the smooth.demogdata function.

Author(s)

Rob J Hyndman

Source

The Human Mortality Database (http://www.mortality.org).

Examples

```
plot(fr.mort,years=1950:1997)

plot(fr.mort,years=1990,type='p',pch=1)
lines(fr.sm,years=1990)
```

---

hmd *Read data from HMD and construct a mortality demogdata object*

Description

`hmd.mx` reads "Mx" (1x1) data from the Human Mortality Database (HMD https://www.mortality.org) and constructs a demogdata object suitable for plotting using `plot.demogdata` and fitting an LC or BMS model using `lca` or an FDA model using `fdm`. `hmd.pop` reads "Population" (1x1) data from the HMD and constructs a demogdata object suitable for plotting using `plot.demogdata`. `hmd.e0` reads life expectancy at birth from the HMD and returns the result as a `ts` object.

Usage

```
hmd.mx(country, username, password, label = country)

hmd.e0(country, username, password)

hmd.pop(country, username, password, label = country)
```

Arguments

`country`	Directory abbreviation from the HMD. For instance, Australia = "AUS".
`username`	HMD username (case-sensitive)
`password`	HMD password (case-sensitive)
`label`	Character string giving name of country from which the data are taken.

Details

In order to read the data, users are required to create their account via the HMD website (https://www.mortality.org), and obtain a valid username and password.

Value

hmd.mx returns an object of class demogdata with the following components:

year	Vector of years
age	Vector of ages
rate	A list containing one or more rate matrices with one age group per row and one column per year.
pop	A list of the same form as rate but containing population numbers instead of demographic rates.
type	Type of object: "mortality", "fertility" or "migration".
label	label

hmd.pop returns a similar object but without the rate component. hmd.e0 returns an object of class ts with columns male, female and total.

Author(s)

Rob J Hyndman

See Also

demogdata,read.demogdata,plot.demogdata, life.expectancy

Examples

```
## Not run:
norway <- hmd.mx("NOR", username, password, "Norway")
summary(norway)

## End(Not run)
```

---

isfe *Integrated Squared Forecast Error for models of various orders*

Description

Computes ISFE values for functional time series models of various orders.

Usage

```
isfe(...)

## S3 method for class 'demogdata'
isfe(
  data,
  series = names(data$rate)[1],
  max.order = N - 3,
```

```
  N = 10,
  h = 5:10,
  ages = data$age,
  max.age = max(ages),
  method = c("classical", "M", "rapca"),
 fmethod = c("arima", "ar", "arfima", "ets", "ets.na", "struct", "rwdrift", "rw"),
  lambda = 3,
  ...
)
```

Arguments

`...`	Additional arguments control the fitting procedure.
`data`	demogdata object.
`series`	name of series within data holding rates (1x1)
`max.order`	Maximum number of basis functions to fit.
`N`	Minimum number of functional observations to be used in fitting a model.
`h`	Forecast horizons over which to average.
`ages`	Ages to include in fit.
`max.age`	Maximum age to fit.
`method`	Method to use for principal components decomposition. Possibilities are "M", "rapca" and "classical".
`fmethod`	Method used for forecasting. Current possibilities are "ets", "arima", "ets.na", "struct", "rwdrift" and "rw".
`lambda`	Tuning parameter for robustness when `method="M"`.

Value

Numeric matrix with `(max.order+1)` rows and `length(h)` columns containing ISFE values for models of orders 0:max.order.

Author(s)

Rob J Hyndman

References

Hyndman, R.J., and Ullah, S. (2007) Robust forecasting of mortality and fertility rates: a functional data approach. *Computational Statistics & Data Analysis*, **51**, 4942-4956. https://robjhyndman.com/publications/funcfor/

See Also

`fdm`, `forecast.fdm`.

`lca` *Model mortality or fertility data using Lee-Carter approach*

Description

Lee-Carter model of mortality or fertility rates. `lca` produces a standard Lee-Carter model by default, although many other options are available. `bms` is a wrapper for `lca` and returns a model based on the Booth-Maindonald-Smith methodology.

Usage

```
lca(
  data,
  series = names(data$rate)[1],
  years = data$year,
  ages = data$age,
  max.age = 100,
  adjust = c("dt", "dxt", "e0", "none"),
  chooseperiod = FALSE,
  minperiod = 20,
  breakmethod = c("bai", "bms"),
  scale = FALSE,
  restype = c("logrates", "rates", "deaths"),
  interpolate = FALSE
)

bms(
  data,
  series = names(data$rate)[1],
  years = data$year,
  ages = data$age,
  max.age = 100,
  minperiod = 20,
  breakmethod = c("bms", "bai"),
  scale = FALSE,
  restype = c("logrates", "rates", "deaths"),
  interpolate = FALSE
)
```

Arguments

`data`	demogdata object of type "mortality" or "fertility". Output from read.demogdata.
`series`	name of series within data containing mortality or fertility values (1x1)
`years`	years to include in fit. Default: all available years.
`ages`	ages to include in fit. Default: all available ages up to `max.age`.

max.age	upper age to include in fit. Ages beyond this are collapsed into the upper age group.
adjust	method to use for adjustment of coefficients $k_t kt$. Possibilities are "dxt" (BMS method), "dt" (Lee-Carter method), "e0" (method based on life expectancy) and "none". Defaults are "dxt" for bms() and "dt" for lca().
chooseperiod	If TRUE, it will choose the best fitting period.
minperiod	Minimum number of years to include in fitting period if chooseperiod=TRUE.
breakmethod	method to use for identifying breakpoints if chooseperiod=TRUE. Possibilities are "bai" (Bai's method computed using breakpoints in the strucchange package) and "bms" (method based on mean deviance ratios described in BMS).
scale	If TRUE, it will rescale bx and kt so that kt has drift parameter = 1.
restype	method to use for calculating residuals. Possibilities are "logrates", "rates" and "deaths".
interpolate	If TRUE, it will estimate any zero mortality or fertility rates using the same age group from nearby years.

Details

All mortality or fertility data are assumed to be in matrices of mortality or fertility rates within data$rate. Each row is one age group (assumed to be single years). Each column is one year. The function produces a model for the series mortality or fertility rate matrix within data$rate. Forecasts from this model can be obtained using forecast.lca.

Value

Object of class "lca" with the following components:

label	Name of region
age	Ages from data object.
year	Years from data object.
<series>	Matrix of mortality or fertility data as contained in data. It takes the name given by the series argument.
ax	Average deathrates across fitting period
bx	First principal component in Lee-Carter model
kt	Coefficient of first principal component
residuals	Functional time series of residuals.
fitted	Functional time series containing estimated mortality or fertility rates from model
varprop	Proportion of variance explained by model.
y	The data stored as a functional time series object.
mdev	Mean deviance of total and base lack of fit, as described in Booth, Maindonald and Smith.

Author(s)

Heather Booth, Leonie Tickle, John Maindonald and Rob J Hyndman.

References

Booth, H., Maindonald, J., and Smith, L. (2002) Applying Lee-Carter under conditions of variable mortality decline. *Population Studies*, **56**, 325-336.

Lee, R.D., and Carter, L.R. (1992) Modeling and forecasting US mortality. *Journal of the American Statistical Association*, **87**, 659-671.

See Also

`forecast.lca`, `plot.lca`, `summary.lca`, `fdm`

Examples

```
## Not run:
france.LC1 <- lca(fr.mort, adjust="e0")
plot(france.LC1)
par(mfrow=c(1,2))
plot(fr.mort,years=1953:2002,ylim=c(-11,1))
plot(forecast(france.LC1,jumpchoice="actual"),ylim=c(-11,1))

france.bms <- bms(fr.mort, breakmethod="bai")
fcast.bms <- forecast(france.bms)
par(mfrow=c(1,1))
plot(fcast.bms$kt)

## End(Not run)
```

---

life.expectancy *Estimate life expectancy from mortality rates*

Description

All three functions estimate life expectancy from `lifetable`. The function `flife.expectancy` is primarily designed for forecast life expectancies and will optionally produce prediction intervals. Where appropriate, it will package the results as a forecast object which makes it much easier to product nice plots of forecast life expectancies. The `e0` function is a shorthand wrapper for `flife.expectancy` with `age=0`.

Usage

```
life.expectancy(
  data,
  series = names(data$rate)[1],
  years = data$year,
  type = c("period", "cohort"),
  age = min(data$age),
  max.age = min(100, max(data$age))
)
```

```
flife.expectancy(
  data,
  series = NULL,
  years = data$year,
  type = c("period", "cohort"),
  age,
  max.age = NULL,
  PI = FALSE,
  nsim = 500,
  ...
)

e0(
  data,
  series = NULL,
  years = data$year,
  type = c("period", "cohort"),
  max.age = NULL,
  PI = FALSE,
  nsim = 500,
  ...
)
```

Arguments

data	Demogdata object of type "mortality" such as obtained from read.demogdata, or an object of class fmforecast such as the output from forecast.fdm or forecast.lca, or an object of class fmforecast2 such as the output from forecast.fdmpr.
series	Name of mortality series to use. Default is the first demogdata series in data.
years	Vector indicating which years to use.
type	Either period or cohort.
age	Age at which life expectancy is to be calculated.
max.age	Maximum age for life table calculation.
PI	If TRUE, produce a prediction interval.
nsim	Number of simulations to use when computing a prediction interval.
...	Other arguments passed to simulate when producing prediction intervals.

Value

Time series of life expectancies (one per year), or a forecast object of life expectancies (one per year).

Author(s)

Rob J Hyndman

See Also

lifetable

Examples

```
plot(life.expectancy(fr.mort),ylab="Life expectancy")

france.LC <- lca(fr.mort,adjust="e0",years=1950:1997)
france.fcast <- forecast(france.LC,jumpchoice="actual")
france.e0.f <- life.expectancy(france.fcast)

france.fdm <- fdm(extract.years(fr.mort,years=1950:2006))
france.fcast <- forecast(france.fdm)
## Not run:
  e0.fcast <- e0(france.fcast,PI=TRUE,nsim=200)
  plot(e0.fcast)
## End(Not run)

life.expectancy(fr.mort,type='cohort',age=50)
```

---

lifetable *Construct lifetables from mortality rates*

Description

Computes period and cohort lifetables from mortality rates for multiple years.

Usage

```
lifetable(
  data,
  series = names(data$rate)[1],
  years = data$year,
  ages = data$age,
  max.age = min(100, max(data$age)),
  type = c("period", "cohort")
)
```

Arguments

data	Demogdata object such as obtained from read.demogdata, forecast.fdm or forecast.lca.
series	Name of series to use. Default is the first series in data[["rate"]].
years	Vector indicating which years to include in the tables.
ages	Vector indicating which ages to include in table.
max.age	Age for last row. Ages beyond this are combined.
type	Type of lifetable: period or cohort.

Details

For period lifetables, all years and all ages specified are included in the tables. For cohort lifetables, if `ages` takes a scalar value, then the cohorts are taken to be of that age in each year contained in `years`. But if `ages` is a vector of values, then the cohorts are taken to be of those ages in the first year contained in `years`.

For example, if `ages=0` then lifetables of the birth cohorts for all years in `years` are computed. On the other hand, if `ages=0:100` and `years=1950:2010`, then lifetables of each age cohort in 1950 are computed.

In all cases, $q_x = m_x/(1 + [(1 - a_x)m_x])$ as per Chiang (1984).

Warning: the code has only been tested for data based on single-year age groups.

Value

Object of class "lifetable" containing the following components:

`label`	Name of region from which data are taken.
`series`	Name of series
`age`	Ages for lifetable
`year`	Period years or cohort years
`mx`	Death rate at age x.
`qx`	The probability that an individual of exact age x will die before exact age x+1.
`lx`	Number of survivors to exact age x. The radix is 1.
`dx`	The number of deaths between exact ages x and x+1.
`Lx`	Number of years lived between exact age x and exact age x+1.
`Tx`	Number of years lived after exact age x.
`ex`	Remaining life expectancy at exact age x.

Note that the lifetables themselves are not returned, only their components. However, there is a print method that constructs (and returns) the lifetables from the above components.

Author(s)

Heather Booth, Leonie Tickle, Rob J Hyndman, John Maindonald and Timothy Miller

References

Chiang CL. (1984) *The life table and its applications*. Robert E Krieger Publishing Company: Malabar.

Keyfitz, N, and Caswell, H. (2005) *Applied mathematical demography*, Springer-Verlag: New York.

Preston, S.H., Heuveline, P., and Guillot, M. (2001) *Demography: measuring and modeling population processes*. Blackwell

See Also

`life.expectancy`

Examples

```
france.lt <- lifetable(fr.mort)
plot(france.lt)
lt1990 <- print(lifetable(fr.mort,year=1990))

france.LC <- lca(fr.mort)
france.fcast <- forecast(france.LC)
france.lt.f <- lifetable(france.fcast)
plot(france.lt.f)

# Birth cohort lifetables, 1900-1910
france.clt <- lifetable(fr.mort,type="cohort",age=0, years=1900:1910)

# Partial cohort lifetables for 1950
lifetable(fr.mort, years=1950)
```

---

mean.demogdata *Mean and median functions for data of class demogdata*

Description

Computes mean or median of demographic rates for each age level.

Usage

```
## S3 method for class 'demogdata'
mean(x, series = names(x$rate)[1], transform = TRUE, na.rm = TRUE, ...)

## S3 method for class 'demogdata'
median(
  x,
  na.rm = FALSE,
  series = names(x$rate)[1],
  transform = TRUE,
  method = c("hossjercroux", "coordinate"),
  ...
)
```

Arguments

x	Demogdata object such as created using read.demogdata or smooth.demogdata.
series	Name of demogdata series to plot..
transform	Should transform of data be taken first?
na.rm	a logical value indicating whether NA values should be stripped before the computation proceeds.
...	Other arguments.

method	Method for computing the median. Either "coordinate" for a coordinate-wise median, or "hossjercroux" for the L1-median using the Hossjer-Croux algorithm.

Value

A list containing x=ages and y=mean or median rates.

Author(s)

Rob J Hyndman

References

Hossjer, O., and Croux, C. (1995) Generalized univariate signed rank statistics for testing and estimating a multivariate location parameter. *Nonparametric Statistics*, **4**, 293-308.

Examples

```
plot(fr.mort)
lines(mean(fr.mort),lwd=2)
lines(median(fr.mort),lwd=2,col=2)
```

---

models *Show model information for the forecast coefficients in FDM models.*

Description

The models for the time series coefficients used in forecasting fdm models are shown.

Usage

```
models(object, ...)

## S3 method for class 'fmforecast'
models(object, select = 0, ...)

## S3 method for class 'fmforecast2'
models(object, ...)
```

Arguments

object	Output from forecast.fdm or forecast.fdmpr.
...	Other arguments.
select	Indexes of coefficients to display. If select=0, all coefficients are displayed.

Author(s)

Rob J Hyndman

See Also

forecast.fdm, forecast.fdmpr.

Examples

```
## Not run:
fr.short <- extract.years(fr.sm,1950:2006)
fr.fit <- fdm(fr.short,series="male")
fr.fcast <- forecast(fr.fit)
models(fr.fcast)

fr.fit <- coherentfdm(fr.short)
fr.fcast <- forecast(fr.fit)
models(fr.fcast,select=1:3)

## End(Not run)
```

---

netmigration *Calculate net migration from mortality and fertility data*

Description

Function to compute the net number of migrants in each year and for each age, based on the total population numbers, deaths and births in each year.

Usage

```
netmigration(mort, fert, startyearpop=mort, mfratio = 1.05)
```

Arguments

mort	Demogdata object of type "mortality"
fert	Demogdata object of type "fertility"
startyearpop	Demogdata object containing population data for first year of calculation.
mfratio	Male-female ratio to be used in simulating births.

Value

Object of class "demogdata" with the following components:

year	Vector of years
age	Vector of ages
rate	List containing matrices of net migration numbers (not "rates") with with one age group per row and one column per year. Names of matrices are the same as for mort$rate.
pop	List containing matrices of populations in same form as rate and containing population numbers.
type	Type of object. In this case, "migration".
label	label from mort$label

Author(s)

Rob J Hyndman

See Also

demogdata

Examples

```
## Not run:
require(addb)
aus.mig <- netmigration(australia,aus.fertility)
plot(aus.mig)
## End(Not run)
```

---

plot.demogdata *Plot age-specific demographic functions*

Description

If `plot.type="functions"`, then years are plotted using a rainbow palette so the earliest years are red, followed by orange, yellow, green, blue and indigo with the most recent years plotted in violet. If `plot.type="time"`, then each age is shown as a separate time series in a time plot.

Usage

```
## S3 method for class 'demogdata'
plot(
  x,
  series = ifelse(!is.null(x$rate), names(x$rate)[1], names(x$pop)[1]),
  datatype = ifelse(!is.null(x$rate), "rate", "pop"),
  years = x$year,
  ages = x$age,
  max.age = max(x$age),
  transform = (x$type == "mortality"),
  plot.type = c("functions", "time", "depth", "density"),
  type = "l",
  main = NULL,
  xlab,
  ylab,
  ...
)

## S3 method for class 'demogdata'
lines(
  x,
  series = ifelse(!is.null(x$rate), names(x$rate)[1], names(x$pop)[1]),
```

```
  datatype = ifelse(!is.null(x$rate), "rate", ""),
  years = x$year,
  ages = x$age,
  max.age = max(x$age),
  transform = (x$type == "mortality"),
  plot.type = c("functions", "time", "depth", "density"),
  ...
)

## S3 method for class 'demogdata'
points(..., pch = 1)
```

Arguments

x	Demogdata object such as created using read.demogdata or smooth.demogdata.
series	Name of series to plot. Default: the first matrix within datatype.
datatype	Name of demogdata object which contains series. Default "rate". Alternative: "pop".
years	Vector indicating which years to plot. Default: all available years.
ages	Vector indicating which ages to plot. Default: all available ages.
max.age	Maximum age to plot. Default: all available ages.
transform	Should a transformation of the data be plotted? Default is TRUE if the object contains mortality data and datatype="rate", and FALSE otherwise.
plot.type	Type of plot: either "functions" or "time".
type	What type of plot should be drawn. See plot for possible types.
main	Main title for the plot.
xlab	Label for x-axis.
ylab	Label for y-axis.
...	Other plotting parameters. In points.demogdata, all arguments are passed to lines.demogdata.
pch	Plotting character.

Value

None. Function produces a plot

Author(s)

Rob J Hyndman

Examples

```
plot(fr.mort)
par(mfrow=c(1,2))
plot(aus.fert,plot.type="time")
plot(aus.fert,plot.type="functions")
```

plot.errorfdm *Plot differences between actuals and estimates from fitted demographic model*

Description

Function produces a plot of errors from a fitted demographic model.

Usage

```
## S3 method for class 'errorfdm'
plot(x, transform = TRUE, ...)
```

Arguments

x	Object of class "errorfdm" generated by compare.demogdata.
transform	Plot errors on transformed scale or original scale?
...	Plotting parameters.

Author(s)

Rob J Hyndman

See Also

compare.demogdata

Examples

```
fr.fit <- lca(extract.years(fr.mort,years=1921:1980))
fr.error <- compare.demogdata(fr.mort, forecast(fr.fit,20))
plot(fr.error)
```

plot.fmforecast *Plot forecasts from a functional demographic modell*

Description

Type of plot depends on value of plot.type:

plot.type="function" produces a plot of the forecast functions;

plot.type="components" produces a plot of the basis functions and coefficients with forecasts and prediction intervals for each coefficient;

plot.type="variance" produces a plot of the variance components.

Usage

```
## S3 method for class 'fmforecast'
plot(
  x,
  plot.type = c("function", "component", "variance"),
  vcol = 1:4,
  mean.lab = "Mean",
  xlab2 = "Year",
  h = 1,
  ...
)

## S3 method for class 'lca'
plot(x, ...)
```

Arguments

x	Output from forecast.ftsm, forecast.fdm or lca.
plot.type	Type of plot. See details.
vcol	Colors to use if plot.type="variance".
mean.lab	Label for mean component.
xlab2	x-axis label for coefficient time series.
h	If plot.type="variance", h gives the forecast horizon for which the variance is plotted.
...	Other arguments are passed to plot.demogdata (if plot.type=="function"), plot (if plot.type=="variance") or plot.ftsf (if plot.type=="component").

Value

None. Function produces a plot

Author(s)

Rob J Hyndman

See Also

fdm, lca, forecast.fdm

Examples

```
france.fcast <- forecast(fdm(fr.mort))
plot(france.fcast)
plot(france.fcast,"c")
plot(france.fcast,"v")
```

`plot.lifetable` *Plot life expectancy from lifetable*

Description

plots life expectancy for each age and each year as functional time series.

Usage

```
## S3 method for class 'lifetable'
plot(
  x,
  years = x$year,
  main,
  xlab = "Age",
  ylab = "Expected number of years left",
  ...
)

## S3 method for class 'lifetable'
lines(x, years = x$year, ...)
```

Arguments

`x`	Output from `lifetable`.
`years`	Years to plot. Default: all available years.
`main`	Main title.
`xlab`	Label for x-axis.
`ylab`	Label for y-axis.
`...`	Additional arguments passed to `plot.fds`.

Author(s)

Rob J Hyndman

See Also

`life.expectancy`, `lifetable`.

Examples

```
france.lt <- lifetable(fr.mort)
plot(france.lt)

france.LC <- lca(fr.mort)
france.fcast <- forecast(france.LC)
france.lt.f <- lifetable(france.fcast)
plot(france.lt.f,years=2010)
```

pop.sim *Population simulation*

Description

Simulate future sample paths of a population using functional models for mortality, fertility and migration.

Usage

```
pop.sim(
  mort,
  fert = NULL,
  mig = NULL,
  firstyearpop,
  N = 100,
  mfratio = 1.05,
  bootstrap = FALSE
)
```

Arguments

mort	Forecasts of class fmforecast2 for mortality.
fert	Forecasts of class fmforecast for female fertility.
mig	Forecasts of class fmforecast2 for net migration.
firstyearpop	Population for first year of simulation.
N	Number of sample paths to simulate.
mfratio	Male-female ratio used in distributing births.
bootstrap	If TRUE, simulation uses resampled errors rather than normally distributed errors.

Value

A list of two arrays containing male and female future simulated population values. The arrays are of dimension (p,h,N) where p is the number of age groups, h is the forecast horizon and N is the number of simulated sample paths.

Author(s)

Rob J Hyndman

See Also

simulate.fmforecast, simulate.fmforecast2.

Examples

```
## Not run:
require(addb)
# Construct data objects
mort.sm <- smooth.demogdata(set.upperage(extract.years(australia,1950:2002),100))
fert.sm <- smooth.demogdata(extract.years(aus.fertility,1950:2002))
aus.mig <- netmigration(set.upperage(australia,100),aus.fertility,mfratio=1.0545)
# Fit models
mort.fit <- coherentfdm(mort.sm)
fert.fit <- fdm(fert.sm)
mig.fit <- coherentfdm(aus.mig)
# Produce forecasts
mort.fcast <- forecast(mort.fit)
fert.fcast <- forecast(fert.fit)
mig.fcast <- forecast(mig.fit)
# Simulate
aus.sim <- pop.sim(mort.fcast,fert.fcast,mig.fcast,australia)
## End(Not run)
```

`read.demogdata` *Read demographic data and construct demogdata object*

Description

Read data from text files and construct a demogdata object suitable for plotting using `plot.demogdata` and fitting an LC or BMS model using `lca` or an FDA model using `fdm`.

Usage

```
read.demogdata(
  file,
  popfile,
  type,
  label,
  max.mx = 10,
  skip = 2,
  popskip = skip,
  lambda,
  scale = 1
)
```

Arguments

`file`	Filename containing demographic rates.
`popfile`	Filename containing population numbers.
`type`	Character string showing type of demographic series: either “mortality”, “fertility” or “migration”.

label	Name of area from which the data are taken.
max.mx	Maximum allowable value for demographic rate. All values greater than max.mx will be set to max.mx.
skip	Number of lines to skip at the start of file.
popskip	Number of lines to skip at the start of popfile.
lambda	Box-Cox transformation parameter to be used in modelling and plotting. If missing, default values are 0 (for mortality), 0.4 (for fertility) and 1 (for migration).
scale	Number of people in the rate definition. scale=1 indicates the rates are per person; scale=1000 indicates the rates are per 1000 people.

Details

All data are assumed to be tab-delimited text files with the first column containing the year of observation and the second column containing the age level. All remaining columns are assumed to be demographic rates for sections of the population. The first row of the text file is assumed to contain the names of each column. Population data are assumed to have the same format but with population numbers in place of rates. The columns names in the two files should be identical. Note that this format is what is used by the Human Mortality Database http://www.mortality.org. If popfile contains the Exposures and file contains the Mx rates from the HMD, then everything will work seamlessly.

Value

Object of class "demogdata" with the following components:

year	Vector of years
age	Vector of ages
rate	A list containing one or more rate matrices with one age group per row and one column per year.
pop	A list of the same form as rate but containing population numbers instead of demographic rates.
type	Type of object: "mortality", "fertility" or "migration".
label	label

Author(s)

Rob J Hyndman

See Also

demogdata

Examples

```
  ## Not run:  norway <- read.demogdata("Mx_1x1.txt",
  "Exposures_1x1.txt", type="mortality", label="Norway")
## End(Not run)
```

`residuals.fdm` *Compute residuals and fitted values from functional demographic model or Lee-Carter model*

Description

After fitting a Lee-Carter model or functional demographic model, it is useful to inspect the residuals or plot the fitted values. These functions extract the relevant information from the fit object.

Usage

```
## S3 method for class 'fdm'
residuals(object, ...)

## S3 method for class 'fdm'
fitted(object, ...)

## S3 method for class 'lca'
fitted(object, ...)

## S3 method for class 'lca'
residuals(object, ...)
```

Arguments

`object`	Output from `fdm` or `lca`.
`...`	Other arguments.

Value

`residuals.fdm` and `residuals.lca` produce an object of class "fmres" containing the residuals from the model. `fitted.fdm` and `fitted.lca` produce an object of class "fts" containing the fitted values from the model.

Author(s)

Rob J Hyndman.

See Also

`fdm`, `lca`, `bms`

Examples

```
fit1 <- lca(fr.mort)
plot(residuals(fit1))
plot(fitted(fit1))
```

set.upperage *Combine the upperages of a demogdata object.*

Description

Computes demographic rates by combining age groups.

Usage

```
set.upperage(data, max.age)
```

Arguments

data	Demogdata object such as created using read.demogdata or smooth.demogdata.
max.age	Upper age group. Ages beyond this are combined into the upper age group.

Value

Demogdata object with same components as data but with a subset of ages.

Author(s)

Rob J Hyndman

Examples

```
france.short <- set.upperage(fr.mort, 85)
```

sex.ratio *Compute sex ratios from mortality rates*

Description

Calculates the Male/Female ratios from historical or forecasted mortality rates.

Usage

```
sex.ratio(data)
```

Arguments

data	Demogdata object of type "mortality" such as obtained from read.demogdata, or an object of class fmforecast such as the output from forecast.fdm or forecast.lca.

Value

Functional time series of sex ratios.

Author(s)

Rob J Hyndman

Examples

```
plot(sex.ratio(fr.mort),ylab="Sex ratios (M/F)")
```

`simulate.fmforecast` *Simulate future sample paths from functional demographic model forecasts.*

Description

This function will simulate future sample paths given forecasting models from a functional demographic model such as those obtained using `forecast.fdm` or `forecast.fdmpr`.

Usage

```
## S3 method for class 'fmforecast'
simulate(
  object,
  nsim = 100,
  seed = NULL,
  bootstrap = FALSE,
  adjust.modelvar = TRUE,
  ...
)

## S3 method for class 'fmforecast2'
simulate(object, ...)
```

Arguments

`object`	Object of class `fmforecast`. Typically, this is output from `forecast.fdm`.
`nsim`	Number of sample paths to simulate.
`seed`	Either NULL or an integer that will be used in a call to set.seed before simulating the time seriers. The default, NULL will not change the random generator state.
`bootstrap`	If TRUE, simulation uses resampled errors rather than normally distributed errors.
`adjust.modelvar`	If TRUE, will adjust the model variance by the ratio of the empirical and theoretical variances for one-step forecasts.
`...`	Other arguments passed to `simulate.fmforecast`.

Value

An array containing the future simulated values (in the case of a `fmforecast` object), or a list of arrays containing the future simulated values (in the case of a `fmforecast2` object).

Author(s)

Rob J Hyndman

See Also

`forecast.fdm`, `forecast.lca`, `forecast.ftsm`.

Examples

```
## Not run:
france.fit <- fdm(fr.mort,order=2)
france.fcast <- forecast(france.fit,50,method="ets")
france.sim <- simulate(france.fcast,nsim=100)

france.fit2 <- coherentfdm(fr.sm)
france.fcast2 <- forecast(france.fit2,50)
france.sim2 <- simulate(france.fcast2,nsim=100)
## End(Not run)
```

`smooth.demogdata` *Create smooth demogdata functions*

Description

Smooth demogdata data using one of four methods depending on the value of `method`

Usage

```
smooth.demogdata(
  data,
 method = switch(data$type, mortality = "mspline", fertility = "cspline", migration =
    "loess"),
  age.grid,
  power = switch(data$type, mortality = 0.4, fertility = 1, migration = 1),
  b = 65,
  k = 30,
  span = 0.2,
  lambda = 1e-10,
  interpolate = FALSE,
  weight = data$type != "migration",
  obs.var = "empirical"
)
```

Arguments

data	Demogdata object such as created using read.demogdata.
method	Method of smoothing. Possibilities: "mspline" (monotonic regression splines), "cspline" (concave regression splines), "spline" (unconstrained regression splines), "loess" (local quadratic using loess).
age.grid	Ages to use for smoothed curves. Default is single years over a slightly greater range than the unsmoothed data.
power	Power transformation for age variable before smoothing. Default is 0.4 for mortality data and 1 (no transformation) for fertility or migration data.
b	Lower age for monotonicity if method=="mspline". Above this, the smooth curve is assumed to be monotonically increasing.
k	Number of knots to use for penalized regression spline estimate. Ignored if method=="loess".
span	Span for loess smooth if method=="loess".
lambda	Penalty for constrained regression spline if method=="cspline".
interpolate	If interpolate==TRUE, a linear interpolation is used instead of smoothing.
weight	If TRUE, uses weighted smoothing.
obs.var	Method for computing observational variance. Possible values: "empirical" or "theoretical".

Details

The value of method determines the type of smoothing used.

method="mspline" Weighted penalized regression splines with a monotonicity constraint. The curves are monotonically increasing for age greater than b. Smoothness controlled by k. Methodology based on Wood (1994). Code calls gam for the basic computations.

method="cspline" Weighted regression B-splines with a concavity constraint. Smoothness controlled by lambda. Methodology based on He and Ng (1999). Code calls cobs for the basic computations.

method="spline" Unconstrained weighted penalized regression splines. Equivalent to "mspline" but with b=Inf.

method="loess" Weighted locally quadratic regression. Smoothness controlled by span. Code calls loess for the basic computations.

Value

Demogdata object identical to data except all rate matrices are replaced with smooth versions and pop matrices are replaced with disaggregated population estimates obtained using monotonic spline interpolation applied to the cumulative population data. Weight matrices are also added to the object showing the inverse variances of the estimated smooth curves.

Author(s)

Rob J Hyndman

Examples

```
france.sm <- smooth.demogdata(extract.years(fr.mort,1980:1997))
plot(france.sm)
plot(fr.mort,years=1980,type="p",pch=1)
lines(france.sm,years=1980,col=2)
```

---

summary.fdm *Summary for functional demographic model or Lee-Carter model*

Description

Summarizes a basis function model fitted to age-specific demographic rate data. It returns various measures of goodness-of-fit.

Usage

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

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

Arguments

object	Output from fdm or lca.
...	Other arguments.

Author(s)

Rob J Hyndman

See Also

fdm, lca, bms, compare.demogdata

Examples

```
fit1 <- lca(fr.mort)
fit2 <- bms(fr.mort,breakmethod="bai")
fit3 <- fdm(fr.mort)
summary(fit1)
summary(fit2)
summary(fit3)
```

tfr *Compute total fertility rate from fertility rates*

Description

Compute total fertility rates from age-specific fertility rates contained in a demogdata object.

Usage

```
tfr(data, PI = FALSE, nsim = 500, ...)
```

Arguments

data	Demogdata object of type "fertility" such as obtained from read.demogdata, forecast.fdm.
PI	If TRUE, produce a prediction interval.
nsim	Number of simulations to use when computing a prediction interval.
...	Other arguments passed to simulate when producing prediction intervals.

Value

If data are of class demogdata, the function returns a time series of fertility rates. If data are from forecast.fdm, the function returns an object of class forecast containing point forecasts and (optionally) prediction intervals.

Author(s)

Rob J Hyndman

See Also

fdm

Examples

```
plot(tfr(aus.fert))
ausfert.fcast <- forecast(fdm(aus.fert))
plot(tfr(ausfert.fcast,PI=TRUE,nsim=400))
```

update *Updating functional demographic models and coherent functional demographic models.*

Description

`update.fmforecast()` updates `fdm` forecasts. The argument `object` is the output from `forecast.fdm` which has been subsequently modified with new coefficient forecasts. These new forecasts are used when re-calculating the forecast of the mortality or fertility rates, or net migration numbers. `update.fmforecast2()` updates `fdmpr` forecasts. The argument `object` is the output from `forecast.fdmpr` which has been subsequently modified with new coefficient forecasts.

Usage

```
## S3 method for class 'fmforecast'
update(object, ...)

## S3 method for class 'fmforecast2'
update(object, ...)
```

Arguments

`object`	Output from either `fdm` or `coherentfdm`.
`...`	Extra arguments currently ignored.

Value

A list of the same class as `object`.

Author(s)

Rob J Hyndman.

See Also

`forecast.fdm`, `forecast.fdmpr`

Examples

```
## Not run:
france.fit <- fdm(fr.mort,order=2)
france.fcast <- forecast(france.fit,50)
# Replace first coefficient model with ARIMA(0,1,2)+drift
france.fcast$coeff[[2]] <- forecast(Arima(france.fit$coeff[,2],
                                    order=c(0,1,2), include.drift=TRUE), h=50, level=80)
france.fcast <- update(france.fcast)

fr.short <- extract.years(fr.sm,1950:2006)
```

```
fr.fit <- coherentfdm(fr.short)
fr.fcast <- forecast(fr.fit)
par(mfrow=c(1,2))
plot(fr.fcast$male)
# Replace first coefficient model in product component with a damped ETS model:
fr.fcast$product$coeff[[2]] <- forecast(ets(fr.fit$product$coeff[,2], damped=TRUE),
                                        h=50, level=80)
fr.fcast <- update(fr.fcast)
plot(fr.fcast$male)

## End(Not run)
```

Index