Package 'Bchron'

November 21, 2020

Type Package

Title Radiocarbon Dating, Age-Depth Modelling, Relative Sea Level Rate Estimation, and Non-Parametric Phase Modelling

Version 4.7.3

Date 2020-11-20

URL http://andrewcparnell.github.io/Bchron/

BugReports https://github.com/andrewcparnell/Bchron/issues

Depends R (>= 3.4.0),

Imports utils, MASS, coda, mclust, ggplot2, ggridges, magrittr, purrr, ggforce, dplyr, scales, stringr

Encoding UTF-8

Author Andrew Parnell

Maintainer Andrew Parnell <Andrew.Parnell@mu.ie>

Description Enables quick calibration of radiocarbon dates under various calibration curves (including user generated ones); age-depth modelling as per the algorithm of Haslett and Parnell (2008) <DOI:10.1111/j.1467-9876.2008.00623.x>; Relative sea level rate estimation incorporating time uncertainty in polynomial regression models (Parnell and Gehrels 2015) <DOI:10.1002/9781118452547.ch32>; non-parametric phase modelling via Gaussian mixtures as a means to determine the activity of a site (and as an alternative to the Oxcal function SUM; currently unpublished), and reverse calibration of dates from calibrated into un-calibrated years (also unpublished).

License GPL (>= 2)

Suggests knitr, testthat, rmarkdown, covr

VignetteBuilder knitr

NeedsCompilation yes

RoxygenNote 7.1.1

Language en-US

Repository CRAN

Date/Publication 2020-11-21 16:40:02 UTC

R topics documented:

`Bchron` *Bchron: Radiocarbon dating, age-depth modelling, relative sea level rate estimation, and non-parametric phase modelling*

Description

This package enables quick calibration of radiocarbon dates under various calibration curves (including user generated ones); Age-depth modelling as per the algorithm of Haslett and Parnell (2008); Relative sea level rate estimation incorporating time uncertainty in polynomial regression models; and non-parametric phase modelling via Gaussian mixtures as a means to determine the activity of a site (and as an alternative to the Oxcal function SUM)

Bchron functions

The most important functions are `BchronCalibrate` to calibrate radiocarbon (and non-radiocarbon) dates, `Bchronology` for the age-depth model of Haslett and Parnell (2008), `BchronRSL` to get rate estimates for relative sea level data, `BchronDensity` and `BchronDensityFast` for non-parametric phase modelling of age data. See the help files for these functions for examples. See the vignette for more complete documentation

`BchronCalibrate` *Fast radiocarbon calibration*

Description

A fast function for calibrating large numbers of radiocarbon dates involving multiple calibration curves

Usage

```
BchronCalibrate(
  ages,
  ageSds,
  calCurves = rep("intcal20", length(ages)),
  ids = NULL,
  positions = NULL,
  pathToCalCurves = system.file("data", package = "Bchron"),
  eps = 1e-05,
  dfs = rep(100, length(ages))
)
```

Arguments

ages	A vector of ages (most likely 14C)
ageSds	A vector of 1-sigma values for the ages given above
calCurves	A vector of values containing either intcal20, shcal20, marine20, or normal (older calibration curves are supposed such as intcal13). Should be the same length the number of ages supplied. Non-standard calibration curves can be used provided they are supplied in the same format as those previously mentioned and are placed in the same directory. Normal indicates a normally-distributed (non-14C) age.
ids	ID names for each age
positions	Position values (e.g. depths) for each age
pathToCalCurves	File path to where the calibration curves are located. Defaults to the system directory where the 3 standard calibration curves are stored.
eps	Cut-off point for density calculation. A value of eps>0 removes ages from the output which have negligible probability density
dfs	Degrees-of-freedom values for the t-distribution associated with the calibration calculation. A large value indicates Gaussian distributions assumed for the 14C ages

Details

This function provides a direct numerical integration strategy for computing calibrated radiocarbon ages. The steps for each 14C age are approximately as follows: 1) Create a grid of ages covering the range of the calibration curve 2) Calculate the probability of each age according to the 14C age, the standard deviation supplied and the calibration curve 3) Normalise the probabilities so that they sum to 1 4) Remove any probabilities that are less than the value given for eps Multiple calibration curves can be specified so that each 14C age can have a different curve. For ages that are not 14C, use the 'normal' calibration curve which treats the ages as normally distributed with given standard deviation

Value

A list of lists where each element corresponds to a single age. Each element contains:

ages	The original age supplied
ageSds	The original age standard deviation supplied
positions	The position of the age (usually the depth)
calCurves	The calibration curve used for that age
ageGrid	A grid of age values over which the density was created
densities	A vector of probability values indicating the probability value for each element in ageGrid
ageLab	The label given to the age variable
positionLab	The label given to the position variable

See Also

Bchronology, BchronRSL, BchronDensity, BchronDensityFast, createCalCurve

Examples

```
# Calibrate a single age
ages1 = BchronCalibrate(ages=11553,ageSds=230,calCurves='intcal20',ids='Date-1')
summary(ages1)
plot(ages1)

# Calibrate multiple ages with different calibration curves
ages2 = BchronCalibrate(ages=c(3445,11553,7456),ageSds=c(50,230,110),
                        calCurves=c('intcal20','intcal20','shcal20'))
summary(ages2)
plot(ages2)

# Calibrate multiple ages with multiple calibration curves and including depth
ages3 = BchronCalibrate(ages=c(3445,11553),ageSds=c(50,230),positions=c(100,150),
                        calCurves=c('intcal20','normal'))
summary(ages3)
plot(ages3,withPositions=TRUE)
```

BchronDensity *Non-parametric phase model*

Description

This function runs a non-parametric phase model on 14C and non-14C ages via Gaussian Mixture density estimation

Usage

```
BchronDensity(
  ages,
  ageSds,
  calCurves,
  pathToCalCurves = system.file("data", package = "Bchron"),
  dfs = rep(100, length(ages)),
  numMix = 50,
  iterations = 10000,
  burn = 2000,
  thin = 8,
  updateAges = FALSE,
  store_density = TRUE
)
```

Arguments

ages	A vector of ages (most likely 14C)
ageSds	A vector of 1-sigma values for the ages given above
calCurves	A vector of values containing either intcal20, shcal20, marine20, or normal (older calibration curves such as intcal13 are also supported). Should be the same length the number of ages supplied. Non-standard calibration curves can be used provided they are supplied in the same format as those previously mentioned and are placed in the same directory. Normal indicates a normally-distributed (non-14C) age.
pathToCalCurves	File path to where the calibration curves are located. Defaults to the system directory where the 3 standard calibration curves are stored
dfs	Degrees-of-freedom values for the t-distribution associated with the calibration calculation. A large value indicates Gaussian distributions assumed for the 14C ages
numMix	The number of mixture components in the phase model. Might need to be increased if the data set is large and the phase behaviour is very complex
iterations	The number of iterations to run for
burn	The number of starting iterations to discard
thin	The step size of iterations to keep
updateAges	Whether or not to update ages as part of the MCMC run. Default is FALSE. Changing this to TRUE will improve performance but will fit a slightly invalid model
store_density	Whether or not to store the density and age grid. Useful for plotting the output in other packages

Details

This model places a Gaussian mixture prior distribution on the calibrated ages and so estimates the density of the overall set of radiocarbon ages. It is designed to be a probabilistic version of the Oxcal SUM command which takes calibrated ages and sums the probability distributions with the aim of estimating activity through age as a proxy.

Value

An object of class BchronDensityRun with the following elements:

- thetaThe posterior samples of the restricted ages
- pPosterior samples of the mixture proportions
- muValues of the means of each Gaussian mixture
- calAgesThe calibrated ages from BchronCalibrate
- GThe number of mixture components. Equal to numMix
- age_gridA grid of ages used for the final density estimate
- densityThe density estimate based on the above age grid

See Also

Bchronology, BchronRSL, BchronDensityFast for a faster approximate version of this function

Examples

```
# Read in some data from Sluggan Moss
data(Sluggan)

# Run the model
SlugDens = BchronDensity(ages=Sluggan$ages,ageSds=Sluggan$ageSds,
                         calCurves=Sluggan$calCurves)

# plot it
plot(SlugDens)
```

BchronDensityFast *Non-parametric phase model (faster version)*

Description

This function runs a non-parametric phase model on 14C and non-14C ages via Gaussian Mixture density estimation through the mclust package

Usage

```
BchronDensityFast(
  ages,
  ageSds,
  calCurves,
  pathToCalCurves = system.file("data", package = "Bchron"),
  dfs = rep(100, length(ages)),
  samples = 2000,
  G = 30
)
```

Arguments

ages	A vector of ages (most likely 14C)
ageSds	A vector of 1-sigma values for the ages given above
calCurves	A vector of values containing either intcal20, shcal20, marine20, or normal (older calibration curves such as intcal13 are also supported). Should be the same length the number of ages supplied. Non-standard calibration curves can be used provided they are supplied in the same format as those previously mentioned and are placed in the same directory. Normal indicates a normally-distributed (non-14C) age.

`pathToCalCurves`
: File path to where the calibration curves are located. Defaults to the system directory where the 3 standard calibration curves are stored.

`dfs`
: Degrees-of-freedom values for the t-distribution associated with the calibration calculation. A large value indicates Gaussian distributions assumed for the 14C ages

`samples`
: Number of samples of calibrated dates required

`G`
: Number of Gaussian mixture components

Details

This is a faster approximate version of `BchronDensity` that uses the `densityMclust` function to compute the Gaussian mixtures for a set of calibrated ages. The method is an approximation as it does not fit a fully Bayesian model as `BchronDensity` does. It is designed to be a probabilistic version of the Oxcal SUM command which takes calibrated ages and sums the probability distributions with the aim of estimating activity through age as a proxy.

Value

An object of class `BchronDensityRunFast` with the following components:

`out`
: The output from the run of `densityMclust` with the given number of mixture components

`calAges`
: The calibrated ages from the `BchronDensity` function

See Also

`Bchronology`, `BchronCalibrate`, `BchronRSL`, `BchronDensity` for a slower exact version of this function

Examples

```
# Read in some data from Sluggan Moss
data(Sluggan)

# Run the model
SlugDensFast = BchronDensityFast(ages=Sluggan$ages,ageSds=Sluggan$ageSds,
                                 calCurves=Sluggan$calCurves)

# plot it
plot(SlugDensFast)
```

`Bchronology` *Runs the Compound Poisson-Gamma chronology model of Haslett and Parnell (2008)*

Description

Fits a non-parametric chronology model to age/position data according to the Compound Poisson-Gamma model defined by Haslett and Parnell (2008) <DOI:10.1111/j.1467-9876.2008.00623.x>. This version uses a slightly modified Markov chain Monte Carlo fitting algorithm which aims to converge quicker and requires fewer iterations. It also a slightly modified procedure for identifying outliers

Usage

```
Bchronology(
  ages,
  ageSds,
  positions,
  positionThicknesses = rep(0, length(ages)),
  calCurves = rep("intcal20", length(ages)),
  ids = NULL,
  outlierProbs = rep(0.01, length(ages)),
  predictPositions = seq(min(positions), max(positions), length = 100),
  pathToCalCurves = system.file("data", package = "Bchron"),
  jitterPositions = FALSE,
  iterations = 10000,
  burn = 2000,
  thin = 8,
  extractDate = 1950 - as.numeric(format(Sys.time(), "%Y")),
  maxExtrap = 1000,
  thetaMhSd = 0.5,
  muMhSd = 0.1,
  psiMhSd = 0.1,
  ageScaleVal = 1000,
  positionNormalise = TRUE
)
```

Arguments

`ages`	A vector of ages (most likely 14C)
`ageSds`	A vector of 1-sigma values for the ages given above
`positions`	Position values (e.g. depths) for each age
`positionThicknesses`	Thickness values for each of the positions. The thickness value should be the full thickness value of the slice. By default set to zero.

`calCurves` A vector of values containing either 'intcal20', 'shcal20', 'marine20', or 'normal' (older calibration curves are also supported, e.g. intcal13). Should be the same length the number of ages supplied. Non-standard calibration curves can be used provided they are supplied in the same format as those previously mentioned and are placed in the same directory, or created via `createCalCurve`. Normal indicates a normally-distributed (non-14C) age.

`ids` ID names for each age

`outlierProbs` A vector of prior outlier probabilities, one for each age. Defaults to 0.01

`predictPositions`
A vector of positions (e.g. depths) at which predicted age values are required. Defaults to a sequence of length 100 from the top position to the bottom position

`pathToCalCurves`
File path to where the calibration curves are located. Defaults to the system directory where the 3 standard calibration curves are stored.

`jitterPositions`
Whether to jigger the positions at startup or not. Default is FALSE but if there are lots of dates at similar depths this may resolve some initialisation problems

`iterations` The number of iterations to run the procedure for

`burn` The number of starting iterations to discard

`thin` The step size for every iteration to keep beyond the burnin

`extractDate` The top age of the core. Used for extrapolation purposes so that no extrapolated ages go beyond the top age of the core. Defaults to the current year

`maxExtrap` The maximum number of extrapolations to perform before giving up and setting the predicted ages to NA. Useful for when large amounts of extrapolation are required, i.e. some of the predictPositions are a long way from the dated positions

`thetaMhSd` The Metropolis-Hastings standard deviation for the age parameters

`muMhSd` The Metropolis-Hastings standard deviation for the Compound Poisson-Gamma mean

`psiMhSd` The Metropolis-Hastings standard deviation for the Compound Poisson-Gamma scale

`ageScaleVal` A scale value for the ages. `Bchronology` works best when the ages are scaled to be approximately between 0 and 100. The default value is thus 1000 for ages given in years.

`positionNormalise`
Whether to normalise the position values. `Bchronology` works best when the positions are normalise to be between 0 and 1 The default value is `TRUE`

Details

The `Bchronology` function fits a compound Poisson-Gamma distribution to the increments between the dated levels. This involves a stochastic linear interpolation step where the age gaps are Gamma distributed, and the position gaps are Exponential. Radiocarbon and non-radiocarbon dates (including outliers) are updated within the function also by MCMC.

Value

A list of class BchronologyRun which include elements:

theta	The posterior estimated values of the ages
phi	The posterior estimated outlier values (1=outlier, 2=not outlier). The means of this parameter give the posterior estimated outlier probabilities
mu	The posterior values of the Compound Poisson-Gamma mean
psi	The posterior values of the Compound Poisson-Gamma scale
thetaPredict	The posterior estimated ages for each of the values in predictPosition
predictPositions	The positions at which estimated ages were required
calAges	The calibrated ages as output from BchronCalibrate
inputVals	All of the input values to the Bchronology run

References

Haslett, J., and Parnell, A. C. (2008). A simple monotone process with application to radiocarbon-dated depth chronologies. Journal of the Royal Statistical Society, Series C, 57, 399-418. DOI:10.1111/j.1467-9876.2008.00623.x Parnell, A. C., Haslett, J., Allen, J. R. M., Buck, C. E., and Huntley, B. (2008). A flexible approach to assessing synchroneity of past events using Bayesian reconstructions of sedimentation history. Quaternary Science Reviews, 27(19-20), 1872-1885. DOI:10.1016/j.quascirev.2008.07.009

See Also

BchronCalibrate, BchronRSL, BchronDensity, BchronDensityFast

Examples

```
# Data from Glendalough
data(Glendalough)

# Run in Bchronology - all but first age uses intcal20
GlenOut = Bchronology(ages=Glendalough$ages,ageSds=Glendalough$ageSds,
                      calCurves=Glendalough$calCurves,positions=Glendalough$position,
                      positionThicknesses=Glendalough$thickness,ids=Glendalough$id,
                      predictPositions=seq(0,1500,by=10))

# Summarise it a few different ways
summary(GlenOut) # Default is for quantiles of ages at predictPosition values
summary(GlenOut, type='convergence') # Check model convergence
summary(GlenOut, type='outliers') # Look at outlier probabilities

# Predict for some new positions
predictAges = predict(GlenOut, newPositions = c(150,725,1500), newPositionThicknesses=c(5,0,20))

# Plot the output
plot(GlenOut,main="Glendalough",xlab='Age (cal years BP)',ylab='Depth (cm)',las=1)
```

BchronRSL *Relative sea level rate (RSL) estimation*

Description

Relative sea level rate (RSL) estimation

Usage

```
BchronRSL(
  BchronologyRun,
  RSLmean,
  RSLsd,
  degree = 1,
  iterations = 10000,
  burn = 2000,
  thin = 8
)
```

Arguments

BchronologyRun	Output from a run of Bchronology
RSLmean	A vector of RSL mean estimates of the same length as the number of predictPositions given to the Bchronology function
RSLsd	A vector RSL standard deviations of the same length as the number of predictPositions given to the Bchronology function
degree	The degree of the polynomial regression: linear=1 (default), quadratic=2, etc. Supports up to degree 5, though this will depend on the data given
iterations	The number of MCMC iterations to run
burn	The number of starting iterations to discard
thin	The step size of iterations to discard

Details

This function fits an errors-in-variables regression model to relative sea level (RSL) data. An errors-in-variables regression model allows for uncertainty in the explanatory variable, here the age of sea level data point. The algorithm is more fully defined in the reference below

Value

An object of class BchronRSLRun with elements itemize

References

Andrew C. Parnell and W. Roland Gehrels (2013) 'Using chronological models in late holocene sea level reconstructions from salt marsh sediments' In: I. Shennan, B.P. Horton, and A.J. Long (eds). Handbook of Sea Level Research. Chichester: Wiley

See Also

BchronCalibrate, Bchronology, BchronDensity, BchronDensityFast

Examples

```
# Load in data
data(TestChronData)
data(TestRSLData)

# Run through Bchronology
RSLrun = Bchronology(ages=TestChronData$ages,
                     ageSds=TestChronData$ageSds,
                     positions=TestChronData$position,
                     positionThicknesses=TestChronData$thickness,
                     ids=TestChronData$id,
                     calCurves=TestChronData$calCurves,
                     predictPositions=TestRSLData$Depth,
                     jitterPositions = TRUE)

# Now run through BchronRSL
RSLrun2 = BchronRSL(RSLrun,RSLmean=TestRSLData$RSL,RSLsd=TestRSLData$Sigma,degree=3)

# Summarise it
summary(RSLrun2)

# Plot it
plot(RSLrun2)
```

---

choosePositions *Compute positions to date next which result in maximal decrease of chronological uncertainty*

Description

This function finds, for a given current chronology, created via Bchronology, which positions (depths) to date next If N = 1 it just finds the position with the biggest uncertainty If N>1 it puts a date at the N = 1 position and re-runs Bchronology with the extra psuedo date. It uses the unCalibrate function with the un-calibrated age estimated at the median of the chronology and the sd as specified via the newSds argument. Other arguments specify the new thicknesses, calibration curves, and outlier probabilities for newly inserted psuedo-dates.

Usage

```
choosePositions(
  bchrRun,
  N = 1,
  newSds = 30,
```

```
  newThicknesses = 0,
  positions = bchrRun$predictPositions,
  newCalCurve = "intcal20",
  newOutlierProb = 0.05,
  level = 0.5,
  plot = TRUE,
  count = 1,
  linesAt = NULL
)
```

Arguments

bchrRun	A run of the current chronology as output from Bchronology
N	The number of new positions required
newSds	The new standard deviations of the psuedo-added dates
newThicknesses	The new thicknesses of the psuedo-added dates
positions	The positions allowed to estimate the new positions to date. Defaults to the value of predictPositions from the Bchronology run
newCalCurve	The new calibration curve of the psuedo-added dates
newOutlierProb	The new outlier probabilities of the psuedo-added dates
level	The confidence level required for minimising the uncertainty. Defaults to 50%. (Note: this will be estimated more robustly than the 95% level)
plot	Whether to plot the chronologies as they are produced
count	Counter function (not for use other than by the function itself)
linesAt	Horizontal line positions (not for use other than by the function itself)

Value

Some plots and the positions to date next

See Also

Bchronology for the main function to create chronologies, unCalibrate for the ability to invert calendar dates for a given calibration curve.

Examples

```
data(Glendalough)
GlenOut = Bchronology(ages=Glendalough$ages,
                      ageSds=Glendalough$ageSds,
                      calCurves=Glendalough$calCurves,
                      positions=Glendalough$position,
                      positionThicknesses=Glendalough$thickness,
                      ids=Glendalough$id,
                      predictPositions=seq(0,1500,by=10))
```

```
# Find out which two positions (depths) to date if we have room for two more dates
# Here going to choose 3 new positions to date
newPositions = choosePositions(GlenOut, N = 3)
print(newPositions)

# Suppose you are only interested in dating the new depths at 500, 600, or 700 cm
newPositions2 = choosePositions(GlenOut, N = 2,
                                positions = seq(500, 700, by = 10))
print(newPositions2)
```

`coreInfluence` *Find the influence of dates in a pair of Bchronology runs across the core*

Description

This function takes as input two `Bchronology` runs and compares the uncertainty intervals. It does this by computing the mean uncertainty across the core (`type = 'mean'`) at a specified percentile level (e.g. 95%) and subsequently reporting the reduction/increase in uncertainty between the two runs. Both cores must have the same set of depths/positions at regular intervals.

Usage

```
coreInfluence(
  bchrRun1,
  bchrRun2,
  percentile = 0.95,
  type = c("plot", "summary", "max"),
  ageTolerance = 500,
  ...
)
```

Arguments

`bchrRun1`	The output of a run of the `Bchronology` function
`bchrRun2`	The output of another run of the `Bchronology` function, possibly with different dates. Note this must have the same value of `predictPositions` as `bchrRun1`
`percentile`	The value of the percentile to compare the uncertainties. Default is 95%
`type`	if `plot` will return a plot of the difference in uncertainties at the specified percentile level. If `summary` will return text output of the reduction in uncertainty at each position. If `max` will return the position of the maximum decrease in uncertainty and a list of all the positions where the reduction in uncertainty exceeds the value of `ageTolerance`
`ageTolerance`	A value in years for which to report the positions at which the reduction in uncertainty exceeds this value.
`...`	Additional arguments to plot

Details

For example, if the ageTolerance value is 500 years, then coreInfluence will return all of the positions at which the uncertainty reduction is bigger than 500.

Value

Depending on type will outputs some text and plots providing the influence values for the cores in question.

See Also

Bchronology, choosePositions, dateInfluence for finding the influence of removing a single date from a core

Examples

```
data(Glendalough)
# Start with a run that remove two dates
GlenOut1 = Bchronology(ages=Glendalough$ages[-c(3:4)],
                       ageSds=Glendalough$ageSds[-c(3:4)],
                       calCurves=Glendalough$calCurves[-c(3:4)],
                       positions=Glendalough$position[-c(3:4)],
                       positionThicknesses=Glendalough$thickness[-c(3:4)],
                       ids=Glendalough$id[-c(3:4)],
                       predictPositions=seq(0,1500,by=10))
GlenOut2 = Bchronology(ages=Glendalough$ages,
                       ageSds=Glendalough$ageSds,
                       calCurves=Glendalough$calCurves,
                       positions=Glendalough$position,
                       positionThicknesses=Glendalough$thickness,
                       ids=Glendalough$id,
                       predictPositions=seq(0,1500,by=10))

# Now compare their influence
coreInfluence(GlenOut1,
              GlenOut2,
              type = c('max', 'plot'),
              xlab = 'Age (cal years BP)',
              ylab = 'Depth (cm)',
              main = 'Chronology difference at 95% for
              Glendalough removing two dates',
              las = 1)
```

---

createCalCurve *Create a new calibration curve*

Description

A function for creating a new calibration curve not already available in Bchron

Usage

```
createCalCurve(
  name,
  calAges,
  uncalAges,
  oneSigma = rep(0, length(calAges)),
  pathToCalCurves = getwd(),
  createFile = TRUE
)
```

Arguments

name	The name of the new calibration curve
calAges	A vector of the calendar/calibrated ages in years before present
uncalAges	A vector of values of uncalibrated ages in appropriate units (e.g. 14C years BP)
oneSigma	The one sigma (one standard deviation) values in uncalibrated units. If left blank it assumes these are all zero
pathToCalCurves	The path to the calibration curves. Will write by default to the working directory
createFile	whether to write out the new file or not. Only turned off for testing purposes

Details

All calibration curves are stored by Bchron in the standard R gzipped text format. You can find the location of the calibration curves by typing `system.file('data',package='Bchron')`. Any created calibration curve will be converted to this format. However R packages are not allowed to write to this directory so it is up to the user to put the resulting calibration curve file in the appropriate directory. It can then be used as in the examples below. However note that re-installing Bchron will likely over-write previously created calibration curves so you should make sure to store the code used to create it. As a short-cut to copying it by hand you can instead use the `file.copy` command in the example below.

See Also

BchronCalibrate

Examples

```
## Not run:
# Load in the calibration curve with:
intcal09 = read.table(system.file('extdata/intcal09.14c', package = 'Bchron'), sep=',')
# Run createCalCurve
createCalCurve(name='intcal09',calAges=intcal09[,1],
uncalAges=intcal09[,2],oneSigma=intcal09[,3])
```

```
# Copy the file to the right place
file.copy(from = 'intcal09.rda',
          to = system.file('data',package='Bchron'),
                      overwrite = TRUE) # Only need this if you've run it more than once

# Calibrate the ages under two calibration curves
age_09 = BchronCalibrate(ages=15500, ageSds=150,
                         calCurves = 'intcal09',ids='My Date',
                         pathToCalCurves = getwd())
age_20 = BchronCalibrate(ages=15500,ageSds=150,calCurves = 'intcal20')

# Finally plot the difference
library(ggplot2)
plot(age_09) +
geom_line(data = as.data.frame(age_20$Date1),
         aes(x = ageGrid, y = densities), col = 'red') +
     ggtitle('Intcal09 vs Intcal20')

## End(Not run)
```

`dateInfluence` *Find the influence of the dates in a Bchronology run*

Description

This function takes as input a `Bchronology` run and allows the user to estimate a value of 'influence' for either a particular date (by name or number), for all dates in a core (`whichDate = 'all'`), or for all internal dates (`whichDate = 'internal'`). It measures the influence by either the Kullback-Leibler divergence (`KL`), the absolute mean difference (`absMeanDiff`), or the absolute median difference (`absMedianDiff`).

Usage

```
dateInfluence(
  bchrRun,
  whichDate = "all",
  measure = c("KL", "absMeanDiff", "absMedianDiff")
)
```

Arguments

`bchrRun`	The output of a run of the `Bchronology` function
`whichDate`	The chosen date to remove. Either `'all'` which removes each date in turn, or `'internal'` which removes all but the top/bottom dates, or the date number (in the order same order as in argument 1), or the name of the date from the Bchronology run output file.
`measure`	Either `'KL'` for Kullback Leibler divergence (recommended); or `'absMeanDiff'` or `'absMedianDiff'` for distances in years from the mean/median age respectively

Details

The KL measure is preferred as it takes account of the full probability distributions but it lacks a simple interpretation. The best way to use it is with whichDate = 'all': the largest value corresponds to the most influential date in the chronology. For simpler interpretation use measure = 'absMeanDiff' or measure = 'absMedianDiff' as for these the influence is measured in years.

When the predictPositions from the original Bchronology run do not include those of the date(s) being left out then the function uses the closest position and reports a warning.

Value

Outputs some text providing the influence values for the date(s) in question. If given an assignment value also return a list containing all the probability distributions.

See Also

Bchronology, summary.BchronologyRun, coreInfluence, choosePositions

Examples

```
data(Glendalough)
GlenOut = Bchronology(ages=Glendalough$ages,
                      ageSds=Glendalough$ageSds,
                      calCurves=Glendalough$calCurves,
                      positions=Glendalough$position,
                      positionThicknesses=Glendalough$thickness,
                      ids=Glendalough$id,
                      predictPositions=seq(0,1500,by=10))
dateInfluence(GlenOut, whichDate = 4, measure = 'absMeanDiff')
```

Glendalough *Glendalough data*

Description

Chronology data for Glendalough data set

Usage

```
data(Glendalough)
```

Format

A data frame with 6 observations on the following 6 variables:

`id` ID of each age

`ages` Age in (14C) years BP

`ageSds` Age standard deviations

`position` Depths in cm

`thickness` Thicknesses in cm

`calCurves` Calibration curve for each age

Details

This Glendalough data can be used with `Bchronology` or `BchronDensity`

Source

Haslett, J., Whiley, M., Bhattacharya, S., Mitchell, F. J. G., Allen, J. R. M., Huntley, B., \& Salter-Townshend, M. (2006). Bayesian palaeoclimate reconstruction. Journal of the Royal Statistical Society, Series A, 169, 395-438.

---

hdr *Calculate highest density regions for Bchron calibrated ages*

Description

A function for computing highest density regions (HDRs)

Usage

```
hdr(date, prob = 0.95)
```

Arguments

`date`	A calibrated Bchron date, via e.g. `BchronCalibrate`
`prob`	The desired probability interval, in the range(0, 1)

Details

The output of this function is a list of contiguous ranges which cover the probability interval requested. A highest density region might have multiple such ranges if the calibrated date is multi-modal. These differ from credible intervals, which are always contiguous but will not be a good representation of a multi-modal probability distribution.

Value

A list where each element is one of the contiguous sets making up the HDR

See Also

BchronCalibrate

Examples

```
# Calibrate multiple ages and summarise them
ages = BchronCalibrate(ages=11553,ageSds=230,
                       calCurves='intcal20')
# Get samples
hdr(ages$Date1)
```

---

intcal13 *Northern hemisphere 2013 calibration curve*

Description

Northern hemisphere 2013 calibration curve

Usage

```
data(intcal13)
```

Format

A data frame with 5141 observations on 5 variables.

Details

For full details and reference see http://intcal.org/blurb.html. For usage details see BchronCalibrate

---

intcal20 *Northern hemisphere 2020 calibration curve*

Description

Northern hemisphere 2020 calibration curve

Usage

```
data(intcal20)
```

Format

A data frame with 9501 observations on 5 variables.

Details

For full details and reference see http://intcal.org/blurb.html. For usage details see BchronCalibrate

marine13 *Marine 2013 calibration curve*

Description

Marine 2013 calibration curve

Usage

```
data(marine13)
```

Format

A data frame with 4801 observations on 5 variables

Details

For full details and reference see http://intcal.org/blurb.html. For usage details see BchronCalibrate

marine20 *Marine 2020 calibration curve*

Description

Marine 2020 calibration curve

Usage

```
data(marine20)
```

Format

A data frame with 5501 observations on 5 variables

Details

For full details and reference see http://intcal.org/blurb.html. For usage details see BchronCalibrate

normal *Data for dummy calibration of normally distributed ages*

Description

Data for dummy calibration of normally distributed ages

Usage

```
data(normal)
```

Format

A data frame with 2 observations on 3 variables.

Details

This is dummy data so that BchronCalibrate can calibrate normally distributed dates.

plot.BchronCalibratedDates *Plot calibrated dates from a BchronCalibrate run*

Description

Plots calibrated radiocarbon dates from a BchronCalibrate run. Has options to plot on a position (usually depth) scale if supplied with the original run

Usage

```
## S3 method for class 'BchronCalibratedDates'
plot(
  x,
  date = NULL,
  withPositions = ifelse(length(x) > 1 & !is.null(x[[1]]$positions) & !includeCal,
    TRUE, FALSE),
  includeCal = FALSE,
  dateHeight = 100,
  dateLabels = TRUE,
  fillCol = "darkslategray",
  withHDR = TRUE,
  ageScale = c("bp", "bc", "b2k"),
  scaleReverse = TRUE,
  pathToCalCurves = system.file("data", package = "Bchron"),
  ...
)
```

Arguments

x	Output from BchronCalibrate
date	Either numbers or date names to plot (only used if multiple dates have been calibrated)
withPositions	Whether to plot with positions (i.e. using the position values as the y axis). By default TRUE if x has more than one date and contains positions
includeCal	Whether to plot the date alongside the calibration curve and the normally distributed uncalibrated date.
dateHeight	The height of the dates in the plot in the same units as the position values. Only relevant if withPositions=TRUE.
dateLabels	Whether to add the names of the dates to the left of them. Default TRUE
fillCol	A colour to fill the date densities when withPositions is TRUE, or HDR ranges when it is FALSE
withHDR	Whether to plot the 95% highest density region values
ageScale	Either bp for years before present, bc for years BC/AD (BC will be negative), b2k for years before 2000. Others not supported (yet).
scaleReverse	Whether to reverse the x-axis scale. Defaults to TRUE which works best for dates presented in e.g. years BP
pathToCalCurves	The Bchron path to calibration curves. Defaults to the package location might need to be set to another folder if user defined calibration curves are being used
...	Other arguments to plot (currently ignored)

Details

These plots are intended to be pretty basic and used simply for quick information. Users are encouraged to learn the R plotting features to produce publication quality graphics

See Also

BchronCalibrate, Bchronology, BchronRSL, BchronDensity, BchronDensityFast

---

plot.BchronDensityRun *Plot output from* BchronDensity

Description

Plot output from BchronDensity

Usage

```
## S3 method for class 'BchronDensityRun'
plot(
  x,
  plotDates = TRUE,
  plotRawSum = FALSE,
  plotPhase = TRUE,
  phaseProb = 0.95,
  ...
)
```

Arguments

x	Output from BchronDensity
plotDates	Whether to plot the individual calibrated dates
plotRawSum	Whether to plot the raw sum of the probability distributions
plotPhase	Whether to plot the phase values
phaseProb	The probability value for the phase identification
...	Other graphical commands. See par

See Also

See BchronDensity for examples, also Bchronology, BchronRSL, and BchronDensityFast for a faster approximate version of this function

---

plot.BchronDensityRunFast

Plot run from BchronDensityFast

Description

Plots output from BchronDensityFast

Usage

```
## S3 method for class 'BchronDensityRunFast'
plot(x, plotDates = TRUE, plotSum = FALSE, ...)
```

Arguments

x	Output from BchronDensityFast
plotDates	Whether to include individual age pdfs (default TRUE)
plotSum	Whether to include sum of age pdfs (default FALSE)
...	Other graphical parameters, see par

Details

Creates a basic plot of output for a run of BchronDensityFast

See Also

Examples in BchronDensityFast, and see BchronDensity, for a slower, more accurate version of this function

---

`plot.BchronologyRun` *Plot output from Bchronology*

Description

Plots output from a run of Bchronology

Usage

```
## S3 method for class 'BchronologyRun'
plot(
  x,
  dateHeight = 100,
  dateLabels = TRUE,
  dateCol = "darkslategray",
  chronCol = "deepskyblue4",
  chronTransparency = 0.75,
  alpha = 0.95,
  nudgeX = 0,
  nudgeY = 0,
  expandX = if (dateLabels) {     c(0.1, 0) } else {     c(0, 0) },
  expandY = c(0.05, 0),
  ageScale = c("bp", "bc", "b2k"),
  scaleReverse = TRUE,
  ...
)
```

Arguments

`x`	The object created by Bchronology
`dateHeight`	The height of the dates in the plot (on the same scale as the positions)
`dateLabels`	Whether to label the dates on the vertical axis (default TRUE)
`dateCol`	The colour of the date labels
`chronCol`	The colour of the chronology uncertainty ribbon to be plotted
`chronTransparency`	The amount of transparency for the chronology ribbon
`alpha`	The credible interval of the chronology run to be plotted. Defaults to 95 percent

nudgeX	The amount to move the date labels in the x direction. Can be negative. See geom_text for details
nudgeY	The amount to move the date labels in the y direction. Can be negative. See geom_text for details
expandX	The amount to expand the horizontal axis in case part are missed off the plot. See expand_limits for details
expandY	The amount to expand the vertical axis in case part are missed off the plot. See expand_limits for details
ageScale	Either bp for years before present, bc for years BC/AD (BC will be negative), b2k for years before 2000. Others not supported (yet).
scaleReverse	Whether to reverse the x-axis scale. Defaults to TRUE which works best for dates presented in e.g. years BP
...	Other arguments to plot (currently ignored)

Details

Creates a simple plot of the chronology output. The height of the date densities in the plots can be manipulated via the dateHeight argument which is represented in the same units as the positions/depths provided. More detailed plots can be created by manipulating the Bchronology object as required.

See Also

For examples see Bchronology. Also BchronCalibrate, BchronRSL, BchronDensity, BchronDensityFast

---

plot.BchronRSLRun *Plot output from BchronRSL*

Description

Plot output from the BchronRSL function

Usage

```
## S3 method for class 'BchronRSLRun'
plot(
  x,
  type = c("RSL", "rate", "accel"),
  alpha = 0.95,
  ellipseCol = "darkslategray",
  lineCol = "deepskyblue4",
  ...
)
```

Arguments

x	An object created by BchronRSL
type	One of RSL, rate, or accel. If RSL produces a plot of RSL estimates from the model. If rate, produces rate estimates. If accel produces acceleration estimates.
alpha	confidence level used for plotting ellipses
ellipseCol	The colour of the ellipse used for plotting dates
lineCol	The colour of the sea level curve lines
...	Other arguments to plot (currently ignored)

See Also

BchronCalibrate, Bchronology, BchronRSL, BchronDensity, BchronDensityFast

---

predict.BchronologyRun

Predict ages of other positions for a BchronologyRun object

Description

This function will predict the ages of new positions (usually depths) based on a previous run of the function Bchronology. It will also allow for thickness uncertainties to be included in the resulting ages, for example when the age of a particular event is desired

Usage

```
## S3 method for class 'BchronologyRun'
predict(
  object,
  newPositions,
  newPositionThicknesses = NULL,
  maxExtrap = 500,
  ...
)
```

Arguments

object	Output from a run of Bchronology
newPositions	A vector of new positions at which to find ages
newPositionThicknesses	A vector of thicknesses for the above positions. Must be the same length as newPositions
maxExtrap	The maximum new of extrapolation attempts. It might be worth increasing this if you are extrapolating a long way from the other dated positions
...	Other arguments to predict (not currently supported)

Value

A matrix of dimension num_samples by num_positions so that each row represents a set of monotonic sample predicted ages

See Also

BchronCalibrate, Bchronology BchronRSL, BchronDensity, BchronDensityFast

---

`sampleAges` *Get sample ages from a set of Bchron calibrated dates*

Description

A function for extracting sample ages from Bchron calibrated dates

Usage

```
sampleAges(CalDates, n_samp = 10000)
```

Arguments

`CalDates`	A list created from either `BchronCalibrate`.
`n_samp`	The desired number of samples

Details

Sometimes it is useful to have a set of sample calendar ages for your calibrated dates. For example the samples might be required to create a credible/confidence interval, or to create another non-trivial function of calibrated dates, such as differences. By default the `BchronCalibrate` function provides a grid of ages and an associated density, similar to OxCal. This function extracts that information and uses the `sample` function to output the desired number of samples

Value

A vector of length `n_samp` containing sample ages for the specified date

See Also

BchronCalibrate

Examples

```
# Calibrate multiple ages and summarise them
ages = BchronCalibrate(ages=c(3445,11553,7456),ageSds=c(50,230,110),
                       calCurves=c('intcal20','intcal20','shcal20'))
# Get samples
age_samples = sampleAges(ages)
# Create a credible interval and the median for each date
apply(age_samples, 2, quantile, probs = c(0.05, 0.5, 0.95))
```

shcal13 *Southern hemisphere 2013 calibration curve*

Description

Southern hemisphere 2013 calibration curve

Usage

```
data(shcal13)
```

Format

A data frame with 5141 observations on 5 variables.

Details

For full details and reference see http://intcal.org/blurb.html. For usage details see BchronCalibrate

shcal20 *Southern hemisphere 2020 calibration curve*

Description

Southern hemisphere 2020 calibration curve

Usage

```
data(shcal20)
```

Format

A data frame with 9501 observations on 5 variables.

Details

For full details and reference see http://intcal.org/blurb.html. For usage details see BchronCalibrate

Sluggan *Sluggan Moss data*

Description

Chronology data for Sluggan Moss data set

Usage

```
data(Sluggan)
```

Format

A data frame with 31 observations on the following 6 variables:

`id` ID of each age

`ages` Age in (14C) years BP

`ageSds` Age standard deviations

`position` Depths in cm

`thickness` Thicknesses in cm

`calCurves` Calibration curve for each age

Details

This Sluggan Moss data can be downloaded from the European Pollen Database: http://www.europeanpollendatabase.net. For usage see Bchronology or BchronDensity

Source

Smith, A. G., \& Goddard, I. C. (1991). A 12,500 year record of vegetational history at Sluggan Bog, Co. Antrim, N. Ireland (incorporating a pollen zone scheme for the non-specialist). New Phytologist, 118, 167-187.

summary.BchronCalibratedDates *Summarise a BchronCalibrate object*

Description

Produces summary output from a BchronCalibrate run, including the highest density regions for the calibrated ages for given probability levels

Usage

```
## S3 method for class 'BchronCalibratedDates'
summary(object, prob = 95, ..., digits = max(3, getOption("digits") - 3))
```

Arguments

object	The output of a run of BchronCalibrate
prob	A percentage value (between 0 and 100) at which the highest density regions for each age are calculated
...	Further arguments (not currently supported)
digits	Significant digits to display (not currently supported)

See Also

BchronCalibrate, Bchronology, BchronRSL, BchronDensity, BchronDensityFast

---

summary.BchronDensityRun

Summarise a Bchron density object

Description

Summarise a BchronDensity object

Usage

```
## S3 method for class 'BchronDensityRun'
summary(object, prob = 0.95, ..., digits = max(3, getOption("digits") - 3))
```

Arguments

object	Output from a run of BchronDensity
prob	Probability for identifying phases
...	Other arguments (not currently supported)
digits	Number of digits to report values

See Also

BchronDensity

summary.BchronologyRun *Summarise a Bchronology object*

Description

Summarise a Bchronology object

Usage

```
## S3 method for class 'BchronologyRun'
summary(
  object,
  type = c("quantiles", "outliers", "convergence", "sed_rate", "acc_rate", "max_var"),
  probs = c(0.025, 0.25, 0.5, 0.75, 0.975),
  useExisting = TRUE,
  numPos = 3,
  ...,
  digits = max(3, getOption("digits") - 3)
)
```

Arguments

object	Output from a run of Bchronology
type	Type of output required. The default (quantiles) gives the quantiles of the ages for each position in predictPositions from Bchronology. The other options provide outlier probabilities, convergence diagnostics, accumulation rates, sedimentation rate, and positions of maximum age variance
probs	Probabilities (between 0 and 1) at which to summarise the predicted chronologies
useExisting	Whether to use the predicted chronologies/positions to calculate the sedimentation rate (if TRUE - default) or to re-create them based on a unit-scaled position grid (if FALSE). The latter will be a little bit slower but will provide better sedimentation rate estimates if the original positions are not on a unit scale (e.g. each cm)
numPos	The number of positions at which to provide the maximum variance
...	Other arguments (not currently supported)
digits	Number of digits to report values

See Also

BchronCalibrate, Bchronology BchronRSL, BchronDensity, BchronDensityFast

summary.BchronRSLRun *Summarise a BchronRSL run*

Description

Summarise a BchronRSL run

Usage

```
## S3 method for class 'BchronRSLRun'
summary(
  object,
  type = c("parameters", "RSL", "rate", "accel"),
  age_grid = NULL,
  ...
)
```

Arguments

object	The output from a run of BchronRSL
type	One of parameters, RSL, rate, or accel. If parameters, provides posterior credibility intervals of the regression coefficients. If RSL provides predicted RSL values. If rate, provides rate estimates. If accel provides acceleration estimates.
age_grid	An optional age grid for computing RSL, rate, or acceleration estimates. If not provided uses the age range of the Bchronology run
...	Other arguments to functions (not currently implemented)

See Also

BchronCalibrate, Bchronology, BchronRSL, BchronDensity, BchronDensityFast

TestChronData *Example chronology file for use with the BchronRSL function.*

Description

Some example chronology data for use with the BchronRSL function

Usage

```
data(TestChronData)
```

Format

A data frame with 27 observations on the following 6 variables:

`id` ID names

`ages` Ages in years BP

`ageSds` Ages standard deviations in years BP

`position` Depths in cm

`thickness` Thicknesses in cm

`calCurves` Calibration curve for each age

Source

Andrew C. Parnell and W. Roland Gehrels (2013) 'Using chronological models in late holocene sea level reconstructions from salt marsh sediments' In: I. Shennan, B.P. Horton, and A.J. Long (eds). Handbook of Sea Level Research. Chichester: Wiley

---

TestRSLData *Relative sea level data*

Description

A set of relative sea level data for use with BchronRSL

Usage

```
data(TestRSLData)
```

Format

A data frame with 24 observations on the following 3 variables:

`Depth` Depth in cm

`RSL` Relative sea level in m

`Sigma` Standard deviation of RSL measurement

Source

Andrew C. Parnell and W. Roland Gehrels (2013) 'Using chronological models in late holocene sea level reconstructions from salt marsh sediments' In: I. Shennan, B.P. Horton, and A.J. Long (eds). Handbook of Sea Level Research. Chichester: Wiley

`unCalibrate` *Uncalibrate a Radiocarbon date*

Description

Uncalibrate a Radiocarbon date

Usage

```
unCalibrate(
  calAges,
  calCurve = "intcal20",
  type = c("samples", "ages"),
  pathToCalCurves = system.file("data", package = "Bchron"),
  ...
)
```

Arguments

`calAges`	Either a vector of calibrated ages (when `type = 'ages'`), or a vector of calibrated samples (`type = 'samples'`)
`calCurve`	he calibration curve to use. Only a single calibration curve is currently supported
`type`	Either 'ages' which uncalibrates a calibrated age values without error (i.e. just a lookup on the calibration curve), or a 'samples' which estimates both an uncalibrated mean age and a standard deviation
`pathToCalCurves`	The path to the calibration curve directory. Defaults to the location of the standard calibration curves given in the package
`...`	Other arguments to the `optim` function used to match the probability distributions under `type = 'samples'`

Value

Either a vector of uncalibrated ages (`type = 'ages'`) or a list containing the estimated mean age and standard deviation (`type = 'samples'`)

Examples

```
# Single version outputting just an uncalibrated age
unCalibrate(2350, type = 'ages')

# Vector version giving a vector of uncalibrated ages
unCalibrate(calAge = c(2350, 4750, 11440),
  calCurve = 'shcal20',
  type = 'ages')
```

```
# A version where calibrated standard deviations are required too
calAge = BchronCalibrate(ages = 11255,
  ageSds = 25,
  calCurves = 'intcal20')
calSampleAges = sampleAges(calAge)

# Uncalibrate the above
unCalibrate(calSampleAges,
  type = 'samples')
```

Index