Package: BayLum (via r-universe)

October 18, 2026

Type Package

Title Chronological Bayesian Models Integrating Optically Stimulated Luminescence and Radiocarbon Age Dating

Version 0.3.3

Date 2025-09-19

Maintainer Anne Philippe <anne.philippe@univ-nantes.fr>

Description Bayesian analysis of luminescence data and C-14 age estimates. Bayesian models are based on the following publications: Combes, B. & Philippe, A. (2017) <doi:10.1016/j.quageo.2017.02.003> and Combes et al. (2015) <doi:10.1016/j.quageo.2015.04.001>. This includes, amongst others, data import, export, application of age models and palaeodose model.

License GPL-3

URL https://CRAN.r-project.org/package=BayLum, https://crp2a.github.io/BayLum/

BugReports https://github.com/crp2a/BayLum/issues

Depends coda (>= 0.19), R (>= 4.4), utils

Imports cli (>= 3.6.5), hexbin (>= 1.28.5), KernSmooth (>= 2.23), Luminescence (>= 1.1.1), methods, rjags (>= 4-17), runjags (>= 2.2.2), yaml (>= 2.3.10)

Suggests knitr, rmarkdown, testthat (>= 3.2.3)

VignetteBuilder knitr

Encoding UTF-8

Language en-GB

LazyData true

NeedsCompilation no

RoxygenNote 7.3.3

SystemRequirements JAGS >= 4.3.2 (https://mcmc-jags.sourceforge.io/)

Author Claire Christophe [aut], Anne Philippe [aut, cre] (ORCID: <https://orcid.org/0000-0002-5331-5087>), Sebastian Kreutzer [aut] (ORCID: <https://orcid.org/0000-0002-0734-2199>), Guillaume Guérin [aut] (ORCID: <https://orcid.org/0000-0001-6298-5579>), Frederik Harly Baumgarten [aut] (ORCID: <https://orcid.org/0000-0002-4374-5948>), Nicolas Frerebeau [aut] (ORCID: <https://orcid.org/0000-0001-5759-4944>), Université Bordeaux Montaigne [cph, fnd], CNRS [fnd], LabEx Sciences archéologiques de Bordeaux [fnd]

Config/pak/sysreqs jags

Repository https://cran.r-universe.dev

Date/Publication 2025-09-25 14:39:40 UTC

RemoteUrl https://github.com/cran/BayLum

RemoteRef HEAD

RemoteSha 23a9467d61d47807af10d486f215e0c84b28208d

Contents

Age_Computation . . . 3
Age_OSLC14 . . . 7
AgeC14_Computation . . . 15
AgeS . . . 20
AgeS_Computation . . . 21
combine_DataFiles . . . 28
create_DataFile . . . 29
create_FolderTemplates . . . 31
create_ThetaMatrix . . . 32
DATA_C14 . . . 35
DATA1 . . . 36
DATA2 . . . 37
DATA3 . . . 38
Generate_DataFile-deprecated . . . 39
Generate_DataFile_MG-deprecated . . . 43
IntCal13 . . . 47
IntCal20 . . . 48
LT_RegenDose-deprecated . . . 49
Marine13 . . . 51
Marine20 . . . 52
MCMCsample . . . 52
Model_Age . . . 53
Model_AgeC14 . . . 54
Model_AgeS . . . 55
Model_Palaeodose . . . 57
ModelC14 . . . 58
ModelOSL . . . 59

ModelPrior 60
Palaeodose_Computation 61
plot_Ages 64
plot_MCMC 66
plot_RegDosePoints 68
plot_Scatterplots 69
SC_Ordered 71
SCMatrix 72
SHCal13 73
SHCal20 74
write_BayLumFiles 74
write_YAMLConfigFile 77

Index **80**

`Age_Computation` *Bayesian analysis for the OSL age estimation of one sample*

Description

This function computes the age (in ka) of a sample according to the model developed in Combes and Philippe (2017), based on an output of Generate_DataFile or Generate_DataFile_MG.
A sample, for which data is available in several BIN files, can be analysed.

Usage

```
Age_Computation(
  DATA,
  SampleName = DATA$SampleNames[1],
  PriorAge = c(0.01, 100),
  BinPerSample = c(1),
  SavePdf = FALSE,
  OutputFileName = c("MCMCplot"),
  OutputFilePath = c(""),
  SaveEstimates = FALSE,
  OutputTableName = c("DATA"),
  OutputTablePath = c(""),
  LIN_fit = TRUE,
  Origin_fit = FALSE,
  distribution = c("cauchy"),
  I = 1,
  Iter = 50000,
  t = 5,
  n.chains = 3,
  quiet = FALSE,
  roundingOfValue = 3
)
```

Arguments

DATA	list of objects: LT, sLT, ITimes, dLab, ddot_env, regDose, J, K, Nb_measurement, provided by the function Generate_DataFile or Generate_DataFile_MG. DATA can contain information for more than one sample.
SampleName	character: name of the sample.
PriorAge	numeric (with default): lower and upper bounds for the sample age parameter (in ka). Note that, length(PriorAge)=2.
BinPerSample	integer (with default): vector with the number of BIN files per sample. If in DATA there is more than one sample, the BinPerSample vector must be the same as that used to run the function Generate_DataFile or in Generate_DataFile_MG for generating the DATA object.
SavePdf	logical (with default): if TRUE save graph in pdf file named OutputFileName in folder OutputFilePath.
OutputFileName	character (with default): name of the pdf file that will be generated by the function if SavePdf = TRUE; length(OutputFileName = 2, see **PLOT OUTPUT** in **Value** section for more informations.
OutputFilePath	character (with default): path to the pdf file that will be generated by the function if SavePdf = TRUE. If it is not equal to "", it must be terminated by "/".
SaveEstimates	logical (with default): if TRUE save Bayes estimates and credible interval at level 68% and 95% and the result of the gelman en Rubin test of convergency, in a csv table named OutputFileName in folder OutputFilePath.
OutputTableName	character (with default): name of the table that will be generated by the function if SaveEstimates = TRUE.
OutputTablePath	character (with default): path to the table that will be generated by the function if SaveEstimates = TRUE. If it is not equal to "", it must be terminated by "/".
LIN_fit	logical (with default): if TRUE (default) allows a linear component, on top of the (default) saturating exponential curve, for the fitting of dose response curves. See details section for more informations on the proposed dose response curves.
Origin_fit	logical (with default): if TRUE, forces the dose response curves to pass through the origin. See details section for more informations on the proposed growth curves.
distribution	character (with default): type of distribution that defines how individual equivalent dose values are distributed around the palaeodose. Allowed inputs are "cauchy", "gaussian", "lognormal_A" and "lognormal_M", see details section for more informations.
I	integer (with default): if DATA contains data from more than one sample, I indicates the ID number of the sample to be analysed.
Iter	integer (with default): number of iterations for the MCMC computation (for more information see rjags::jags.model).
t	integer (with default): 1 every t iterations of the MCMC is considered for sampling the posterior distribution (for more information see rjags::jags.model).

`n.chains` integer (with default): number of independent chains for the model (for more information see rjags::jags.model).

`quiet` logical (with default): enables/disables rjags::rjags-package messages

`roundingOfValue` integer (with default): Integer indicating the number of decimal places to be used, default = 3.

Details

Option on growth curves

As for AgeS_Computation and Palaeodose_Computation, the user can choose from 4 dose response curves:

- **Saturating exponential plus linear growth** (`AgeMultiBF_EXPLIN`):
  for all x in IR+, $f(x) = a(1 - exp(-x/b)) + cx + d$; select
  - `LIN_fit=TRUE`
  - `Origin_fit=FALSE`
- **Saturating exponential growth** (`AgeMultiBF_EXP`):
  for all x in IR+, $f(x) = a(1 - exp(-x/b)) + d$; select
  - `LIN_fit = FALSE`
  - `Origin_fit = FALSE`
- **Saturating exponential plus linear growth and fitting through the origin** (`AgeMultiBF_EXPLINZO`):
  for all x in IR+, $f(x) = a(1 - exp(-x/b)) + cx$; select
  - `LIN_fit=TRUE`
  - `Origin_fit=TRUE`
- **Saturating exponential growth and fitting through the origin** (`AgeMultiBF_EXPZO`):
  for all x in IR+, $f(x) = a(1 - exp(-x/b))$; select
  - `LIN_fit=FALSE`
  - `Origin_fit=TRUE`

Option on equivalent dose distribution around the palaeodose

The use can choose between :

- `cauchy`: a Cauchy distribution with location parameter equal to the palaeodose of the sample
- `gaussian`: a Gaussian distribution with mean equal to the palaeodose of the sample
- `lognormal_A`: a log-normal distribution with mean or **A**verage equal to the palaeodose of the sample
- `lognormal_M`: a log-normal distribution with **M**edian equal to the palaeodose of the sample

Value

NUMERICAL OUTPUT

1. **A list containing the following objects**:
   - **Sampling** that corresponds to a sample of the posterior distributions of the age (in ka), palaeodose (in Gy) and equivalent dose dispersion (in Gy) parameters.
   - **Model_GrowthCurve**, stating which dose response fitting option was chosen;
   - **Distribution**, stating which distribution was chosen to model the dispersion of individual equivalent dose values around the palaeodose of the sample;
   - **PriorAge**, stating the priors used for the age parameter (in ka).
2. **The Gelman and Rubin test of convergence**: prints the result of the Gelman and Rubin test of convergence for the age, palaeodose and equivalent dose dispersion parameters. A result close to one is expected.
   In addition, the user must visually assess the convergence of the trajectories by looking at the graph generated by the function (see **PLOT OUTPUT** for more informations).
   If both convergences (Gelman and Rubin test and plot checking) are satisfactory, the user can consider the printed estimates as valid. Otherwise, the user may try increasing the number of MCMC iterations (`Iter`), or being more precise on the `PriorAge` parameter (for example specify if it is a young sample `c(0.01,10)` an old sample `c(10,100)`), or changing the parameter `distribution` or the growth curve, to reach convergence.to reach convergence.
3. **Credible intervals and Bayes estimates**: prints the Bayes estimates, the credible intervals at 95% and 68% for the age, palaeodose and equivalent dose dispersion parameters of the sample.

PLOT OUTPUT

A graph with the MCMC trajectories and posterior distributions of the age, palaeodose and equivalent dose dispersion parameters is displayed.
The first line of the figure correponds to the age parameter, the second to the palaeodose parameter and the third to the equivalent dose dispersion parameter. On each line, the plot on the left represents the MCMC trajectories, and the one on the right the posterior distribution of the parameter.

To give the results in a publication, we recommend to give the Bayes estimate of the parameter as well as the credible interval at 95% or 68%.

Note

Please note that the initial values for all MCMC are currently all the same for all chains since we rely on the automatic initial value generation of JAGS. This is not optimal and will be changed in future. However, it does not affect the quality of the age estimates if the chains have converged.

Author(s)

Claire Christophe, Sebastian Kreutzer, Anne Philippe, Guillaume Guérin

References

Combes, Benoit and Philippe, Anne, 2017. Bayesian analysis of multiplicative Gaussian error for multiple ages estimation in optically stimulated luminescence dating. Quaternary Geochronology (39, 24-34)

Combes, B., Philippe, A., Lanos, P., Mercier, N., Tribolo, C., Guerin, G., Guibert, P., Lahaye, C., 2015. A Bayesian central equivalent dose model for optically stimulated luminescence dating. Quaternary Geochronology 28, 62-70. doi:10.1016/j.quageo.2015.04.001

See Also

Generate_DataFile, Generate_DataFile_MG, rjags::rjags-package, plot_MCMC, AgeS_Computation, Palaeodose_Computation

Examples

```
## load data file generated by the function Generate_DataFile
data(DATA1,envir = environment())
priorage <- c(10,60) # GDB3 is an old sample
Age <- Age_Computation(
 DATA = DATA1,
 SampleName = "GDB3",
 PriorAge = priorage,
 Iter = 100,
 quiet = TRUE)
```

`Age_OSLC14` *Bayesian analysis for age estimation of OSL measurements and C-14 ages of various samples*

Description

This function computes an age of OSL data consisting of at least two samples and calibrate C-14 ages of samples to get an age (in ka).
Ages of OSL data are computed according to the model given in Combes and Philippe (2017). Single-grain or Multi-grain OSL measurements can be analysed simultaneously (with output of Generate_DataFile or Generate_DataFile_MG or both of them using combine_DataFiles). Samples, for which data is available in several BIN files, can be analysed.
For C-14 data, the user can choose one of the following radiocarbon calibration curve: Northern or Southern Hemisphere or marine atmospheric.

Usage

```
Age_OSLC14(
  DATA,
  Data_C14Cal,
  Data_SigmaC14Cal,
  Nb_sample = DATA$Nb_sample,
  SampleNames = DATA$SampleNames,
  SampleNature,
  PriorAge = rep(c(10, 60), Nb_sample),
  SavePdf = FALSE,
```

```
  OutputFileName = c("MCMCplot", "HPD_Cal14CCurve", "summary"),
  OutputFilePath = c(""),
  SaveEstimates = FALSE,
  OutputTableName = c("DATA"),
  OutputTablePath = c(""),
  StratiConstraints = c(),
  sepSC = c(","),
  BinPerSample = rep(1, sum(SampleNature[1, ])),
  THETA = c(),
  sepTHETA = c(","),
  LIN_fit = TRUE,
  Origin_fit = FALSE,
  distribution = c("cauchy"),
  Model_C14 = c("full"),
  CalibrationCurve = c("IntCal20"),
  Iter = 10000,
  burnin = 4000,
  adapt = 1000,
  t = 5,
  n.chains = 3,
  jags_method = "rjags",
  autorun = FALSE,
  quiet = FALSE,
  roundingOfValue = 3,
  ...
)
```

Arguments

DATA	Two types of inputs are supported. (1): a list of objects: LT, sLT, ITimes, dLab, ddot_env, regDose, J, K, Nb_measurement, provided by the function Generate_DataFile, Generate_DataFile_MG or combine_DataFiles. DATA contains information for more than one sample. If there is stratigraphic relations between samples, informations in DATA must be ordered by order of increasing ages. See the details section to for more informations. (2): an object of class BayLum.list which is provided by the output of Age_OSLC14. When input of class BayLum.list is identified, no new JAGS model is created. Instead, the JAGS model specified within the BayLum.list is extended. Useful for when convergence was not originally achieved and a complete restart is not desirable.
Data_C14Cal	numeric vector: corresponding to C-14 age estimate (in years, conversion in ka is automatically done in the function). If there is stratigraphic relations between samples, Data_C14Cal must be ordered by order of increasing ages.
Data_SigmaC14Cal	numeric: corresponding to the error of C-14 age estimates.
Nb_sample	numeric: number of samples (OSL data and C-14 age), (Nb_sample>3, at least to sample of OSL data and one sample of C-14 age).
SampleNames	character: sample names for both OSL data and C14 data. The length of this vector is equal to Nb_sample. If there is stratigraphic relation, this vector must be

ordered by increasing order (to mix OSL samples and C-14 ages if it is needed).

`SampleNature` matrix: states the nature of the sample. Row number of `SampleNature` matrix is equal to 2 and column number is equal to `Nb_sample`. First line of the matrix: `SampleNature[1,i]` states if sample whose number ID is equal to `i`, is an OSL sample 1 or not 0. Second line of the matrix: `SampleNature[2,i]` states if sample whose number ID is equal to `i`, is an C-14 sample 1 or not '0.

`PriorAge` numeric (with default): lower and upper bounds for age parameter of each sample **(in ka)**. Note that, `length(PriorAge) = 2*Nb_sample` sand `PriorAge[2i-1,2i]` corresponds to the lower and upper bounds of sample whose number ID is equal to `i`.

`SavePdf` logical (with default): if `TRUE` save graphs in pdf file named `OutputFileName` in folder `OutputFilePath`.

`OutputFileName` character (with default): name of the pdf file that will be generated by the function if `SavePdf=TRUE`, `length(OutputFileName)=3`, see **PLOT OUTPUT** in **Value** section for more informations.

`OutputFilePath` character (with default): path to the pdf file that will be generated by the function if `SavePdf=TRUE`. If it is not equal to "", it must be terminated by "/".

`SaveEstimates` logical (with default): if TRUE save Bayes' estimates, credible interval at level 68% and 95% and the result of the Gelman en Rubin test of convergence, in a CSV-table named `OutputFileName` in folder `OutputFilePath`.

`OutputTableName`

character (with default): name of the table that will be generated by the function if `SaveEstimates=TRUE`.

`OutputTablePath`

character (with default): path to the table that will be generated by the function if `SaveEstimates=TRUE`. If it is not equal to "", it must be terminated by "/".

`StratiConstraints`

matrix or character (with default): input object for the stratigraphic relation between samples. If there is stratigraphic relation between samples see the details section for instructions regarding how to correctly fill `StratiConstraints`, the user can refer to a matrix or to a CSV-file character. Otherwise, default value is suitable.

`sepSC` character (with default): if `StratiConstraints` is character, indicate column separator in `StratiConstraints` CSV-file.

`BinPerSample` numeric (with default): vector with the number of BIN-files per OSL sample. The length of this vector is equal to the number of OSL samples. `BinPerSample[i]` corresponds to the number of BIN files for the sample whose number ID is equal to `i`. For more information to fill this vector, we refer to details in Generate_DataFile or in Generate_DataFile_MG.

`THETA` numeric matrix or character (with default): input object for systematic and individual error for OSL samples. If systematic errors are considered, see the details section for instructions regarding how to correctly fill `THETA`; the user can refer to a matrix (numeric matrix) or to a csv file (character). Otherwise, default value is suitable, and only individual error is considered.

`sepTHETA` character (with default): if `THETA` is character, indicate column separator in `THETA` CSV-file.

`LIN_fit` logical (with default): if `TRUE` (default) allows a linear component, on top of the (default) saturating exponential curve, for the fitting of dose response curves, for OSL samples. See details for more informations on the proposed dose response curves.

`Origin_fit` plogical (with default): if `TRUE`, forces the dose response curves to pass through the origin. See details for more informations on the proposed growth curves, for OSL samples.

`distribution` character (with default): type of distribution that defines how individual equivalent dose values are distributed around the palaeodose, for OSL samples. Allowed inputs are `"cauchy"`, `"gaussian"`, `"lognormal_A"` and `"lognormal_M"`, see details for more informations.

`Model_C14` character (with default): if `"full"`, error on estimate calibration curve is taken account, for C-14 samples. If `"naive"` this error is not taken account in the age estimate.

`CalibrationCurve`
character (with default): calibration curve chosen, for C-14 samples. Allowed inputs are

- **"Intcal13"** or **"Intcal13"** for Northern Hemisphere atmospheric radiocarbon calibration curve,
- **"Marine13"** or **"Marine13"** for Marine radiocarbon calibration curve,
- **"SHCal13"** or **"SHCal20"** for Southern Hemisphere atmospheric radiocarbon calibration curve
- **a csv file, with tree columns, the first column is dedicated to "Cal.BP", the second to "XC-14.age", the third to "Error". The decimal of this file must be a dot, and the separator must be a comma.**

`Iter` integer (with default): the number of iterations to run and who will be used to assess convergence and ages (see runjags::run.jags]).

`burnin` integer (with default): the number of iterations used to "home in" on the stationary posterior distribution. These are not used for assessing convergence (see runjags::run.jags]).

`adapt` integer (with default): the number of iterations used in the adaptive phase of the simulation (see runjags::run.jags]).

`t` numeric (with default): 1 every `t` iterations of the MCMC is considered for sampling the posterior distribution (for more information see rjags::jags.model.

`n.chains` numeric (with default): number of independent chains for the model (for more information see [rjags::jags.model).

`jags_method` character (with default): select which method to use in order to call JAGS, supported are `"rjags"` (the default), `rjparallel`, `simple`, `interruptible`, `parallel`, and `snow` (for more information about each of these possibilities, see runjags::run.jags])

`autorun` logical (with default): choose to automate JAGS processing. JAGS model will be automatically extended until convergence is reached (for more information see runjags::autorun.jags).

`quiet` logical (with default): enables/disables rjags::rjags-package messages

`roundingOfValue`
: integer (with default): Integer indicating the number of decimal places to be used, default = 3.

`...`
: further arguments that can be passed to control the Bayesian process, see details for supported arguments

Details

Note that there are three types of arguments in the previous list. There are arguments for information concerning only OSL samples: `DATA`, `BinPerSample`, `THETA`, `sepTHETA`, `LIN_fit`, `Origin_fit`, `distribution`.

There are arguments for information concerning only C14 samples: `Data_C14Cal`, `Data_SigmaC14Cal`, `Model_C14`, `CalibrationCurve`.

There are arguments for information concerning all the samples: `Nb_sample`, `SampleNames`, `SampleNature`, `PriorAge`, `SavePdf`, `OutputFileName`, `OutputFilePath`, `SaveEstimates`, `OutputTableName`, `OutputTablePath`, `StratiConstraints`, `sepSC`.

Supported ... arguments

ARGUMENT	INPUT	METHOD	DEFAULT	DESCRIPTION
`max.time`	character	`rjparallel`	`Inf`	maximum allowed processing time, e.g., `10m` for 10 minutes (cf. ru
`interactive`	logical	`rjparallel`	`FALSE`	enable/disable interactive mode (cf. runjags::autorun.jags)
`startburnin`	integer	`rjparallel`	`4000`	number of burn-in iterations (cf. runjags::autorun.jags)
`startsample`	integer	`rjparallel`	`10000`	total number of samples to assess convergence (cf. runjags::autoru
`inits`	named list	`rjparallel`	`NA`	fine control over random seeds and random number generator runja

How to fill 'StratiConstraints?

If there are stratigraphic relations between samples, **14C estimate age in** `Data_C14Cal` **must be ordered by order of increasing ages, as informations in** `DATA`. Names in `SampleNames` must be ordered and corresponds to the order in `Data_C14Cal` and in `DATA`, also if it is needed to mix names of OSL samples and 14C samples.

The user can fill the `StratiConstraints` matrix as follow.

1. **Size of the matrix**: row number of `StratiConstraints` matrix is equal to `Nb_sample+1`, and column number is equal to `Nb_sample`.
2. **First line of the matrix**: for all `i in {1,...,Nb_Sample}`, `StratiConstraints[1,i]=1` that means the lower bound of the sample age (given in `PriorAge[2i-1]`) for the sample whose number ID is equal to `i`, is taken into account.
3. **Sample relations**: for all `j in {2,...,Nb_Sample+1}` and all `i in {j,...,Nb_Sample}`, `StratiConstraints[j,i]=1` if sample age whose number ID is equal to `j-1` is lower than sample age whose number ID is equal to `i`. Otherwise, `StratiConstraints[j,i]=0`.

Note that `StratiConstraints_{2:Nb_sample+1,1:Nb_sample}` is a upper triangular matrix.

The user can also use `SCMatrix` or `SC_Ordered` (if all samples are ordered) function to construct the `StratiConstraints` matrix.

The user can also refer to a csv file that contains the relation between samples as defined above. The user must be careful about which separator is used in the csv file using the argument sepSC.

How to fill THETA covariance matrix concerning common and individual error?

If systematic errors are considered, the user can fill the THETA matrix as follow.

- row number of THETA is equal the column number, equal to Nb_sample.
- For all i in {1,...,Nb_sample}, THETA[i,i] contains individual error plus systematic error of the sample whose number ID is equal to i.
- For all i,j in {1,...,Nb_sample} and i different from j , THETA[i,j] contains common error between samples whose number ID are equal to i and j.

Note that THETA[i,j] is a symmetric matrix.

The user can also refer to a .csv file that contains the errors as defined above.

Option on growth curves

As for Age_Computation and Palaeodose_Computation, the user can choose from 4 dose response curves:

- **Saturating exponential plus linear growth** (AgesMultiCS2_EXPLIN):
  for all x in IR+, f(x)=a(1-exp(-x/b))+cx+d; select
  - LIN_fit=TRUE
  - Origin_fit=FALSE
- **Saturating exponential growth** (AgesMultiCS2_EXP):
  for all x in IR+, f(x)=a(1-exp(-x/b))+d; select
  - LIN_fit=FALSE
  - Origin_fit=FALSE
- **Saturating exponential plus linear growth and fitting through the origin** (AgesMultiCS2_EXPLINZO):
  for all x in IR+, f(x)=a(1-exp(-x/b))+cx; select
  - LIN_fit=TRUE
  - Origin_fit=TRUE
- **Saturating exponential growth and fitting through the origin** (AgesMultiCS2_EXPZO):
  for all x in IR+, f(x)=a(1-exp(-x/b)); select
  - LIN_fit=FALSE
  - Origin_fit=TRUE

Option on equivalent dose distribution around the palaeodose

The use can choose between :

- cauchy: a Cauchy distribution with location parameter equal to the palaeodose of the sample
- gaussian: a Gaussian distribution with mean equal to the palaeodose of the sample

- `lognormal_A`: a log-normal distribution with mean or **A**verage equal to the palaeodose of the sample
- `lognormal_M`: a log-normal distribution with **M**edian equal to the palaeodose of the sample

More precision on `Model`

We propose two models "full" or "naive". If `Model='full'` that means measurement error and error on calibration curve are taken account in the Bayesian model; if `Model="naive"` that means only error on measurement are taken account in the mode.

More precisely, the model considered here, as the one developed by Christen, JA (1994), assume multiplicative effect of errors to address the problem of outliers. In addition, to not penalise variables that are not outliers and damage theirs estimation, we introduce a structure of mixture, that means only variable that are considered as outlier have in addition a multiplicative error.

Value

NUMERICAL OUTPUT

1. **A list containing the following objects:**
   - **Sampling**: that corresponds to a sample of the posterior distributions of the age parameters (in ka for both C14 samples and OSL samples);
   - **PriorAge**: stating the priors used for the age parameter;
   - **StratiConstraints**: stating the stratigraphic relations between samples considered in the model;
   - **Model_OSL_GrowthCurve**: stating which dose response fitting option was chosen;
   - **Model_OSL_Distribution**: stating which distribution was chosen to model the dispersion of individual equivalent dose values around the palaeodose of the sample;
   - **Model_C14**: stating which model was chosen (`"full"` or `"naive"`);
   - **CalibrationCurve**: stating which radiocarbon calibration curve was chosen;
   - **Outlier**: stating the names of samples that must be outliers.
2. **The Gelman and Rubin test of convergency**: prints the result of the Gelman and Rubin test of convergence for the age estimate for each sample. A result close to one is expected.
   In addition, the user must visually assess the convergence of the trajectories by looking at the graph generated by the function (see **PLOT OUTPUT** for more informations).
   If both convergences (Gelman and Rubin test and plot checking) are satisfactory, the user can consider the estimates as valid. Otherwise, the user may try increasing the number of MCMC iterations (`Iter`) or be more precise on the `PriorAge` parameter to reach convergence.
3. **Credible intervals and Bayes estimates**: prints the Bayes' estimates, the credible intervals at 95% and 68% for the age parameters for each sample.
4. **JAGS model output**: returns the JAGS model with class "runjags".

PLOT OUTPUT

1. **MCMC trajectories**: A graph with the MCMC trajectories and posterior distributions of the age parameter is displayed.
   On each line, the plot on the left represents the MCMC trajectories, and the one on the right the posterior distribution of the parameter.

2. **Age estimate and HPD at 95% of 14C samples on calibration curve**: plot age estimate and HPD on calibration plot.
3. **Summary of sample age estimates**: plot credible intervals and Bayes estimate of each sample age on a same graph.

Note

Please note that the initial values for all MCMC are currently all the same for all chains since we rely on the automatic initial value generation of JAGS. This is not optimal and will be changed in future. However, it does not affect the quality of the age estimates if the chains have converged.

Author(s)

Claire Christophe, Anne Philippe, Guillaume Guerin, Sebastian Kreutzer, Frederik Harly Baumgarten

References

Reimer PJ, Bard E, Bayliss A, Beck JW, Blackwell PC, Bronl Ramsey C, Buck CE, Cheng H, Edwards RL, Friedrich M, Grootes PM, Guilderson TP, Haflidason H, Hajdas I, Hatte C, Heaton TJ, Hoffmann DL, Hogg AG, Hughen KA, Kaiser KF, Kromer B, Manning SW, Niu M, Reimer RW, Richards DA, Scott EM, Southon JR, Staff RA, Turney CSM, van der Plicht J. 2013. IntCal13 ans Marine13 radiocarbon age calibration curves 0-50000 years cal BP. Radiocarbon 55(4)=1869-1887.

Hogg AG, Hua Q, Blackwell PG, Niu M, Buck CE, Guilderson TP, Heaton TJ, Palmer JG, Reimer PJ, Reimer RW, Turney CSM, Zimmerman SRH. 2013. SHCal13 Southern Hemisphere calibration, 0-50000 years cal BP. Radiocarbon 55(4):1889-1903

See Also

runjags::run.jags, plot_MCMC, SCMatrix, plot_Ages

Examples

```
## Load data
# OSL data
data(DATA1,envir = environment())
data(DATA2,envir = environment())
Data <- combine_DataFiles(DATA2,DATA1)

# 14C data
C14Cal <- DATA_C14$C14[1,1]
SigmaC14Cal <- DATA_C14$C14[1,2]
Names <- DATA_C14$Names[1]

# Prior Age
prior <- rep(c(1,60),3)
samplenature <- matrix(
 data = c(1,0,1,0,1,0),
 ncol = 3,
 nrow = 2,
```

```
 byrow = TRUE)

SC <- matrix(
 data = c(1,1,1,0,1,1,0,0,1,0,0,0),
 ncol = 3,
 nrow =4 ,
 byrow = TRUE)

## Age computation of samples
## Not run:
Age <- Age_OSLC14(
 DATA = Data,
 Data_C14Cal = C14Cal,
 Data_SigmaC14Cal = SigmaC14Cal,
 SampleNames = c("GDB5",Names,"GDB3"),
 Nb_sample = 3,
 SampleNature = samplenature,
 PriorAge = prior,
 StratiConstraints = SC,
 Iter = 20,
 burnin = 20,
 adapt = 20,
 n.chains = 2)

## End(Not run)
```

`AgeC14_Computation` *Bayesian analysis for C-14 age estimations of various samples*

Description

This function calibrates the C-14 age of samples to get an age (in ka). The user can choose one of the following radiocarbon calibration curve: Northern or Southern Hemisphere or marine atmospheric. It must be the same curve for all samples.

Usage

```
AgeC14_Computation(
  Data_C14Cal,
  Data_SigmaC14Cal,
  SampleNames,
  Nb_sample,
  PriorAge = rep(c(10, 50), Nb_sample),
  SavePdf = FALSE,
  OutputFileName = c("MCMCplot", "HPD_CalC-14Curve", "summary"),
  OutputFilePath = c(""),
  SaveEstimates = FALSE,
  OutputTableName = c("DATA"),
```

```
  OutputTablePath = c(""),
  StratiConstraints = c(),
  sepSC = c(","),
  Model = c("full"),
  CalibrationCurve = c("IntCal20"),
  Iter = 50000,
  t = 5,
  n.chains = 3,
  quiet = FALSE,
  roundingOfValue = 3
)
```

Arguments

`Data_C14Cal`	numeric (**required**): corresponding to C-14 age estimate.
`Data_SigmaC14Cal`	numeric (**required**): corresponding to the error of C-14 age estimates.
`SampleNames`	character (**required**): names of sample. The length of this vector is equal to `Nb_sample`.
`Nb_sample`	integer: number of samples.
`PriorAge`	numeric (with default): lower and upper bounds for age parameter of each sample in years (not in ka). Note that, `length(PriorAge) == 2 * Nb_sample` and `PriorAge[2i-1,2i]` corresponds to the lower and upper bounds of sample whose number ID is equal to `i`.
`SavePdf`	logical (with default): if TRUE save graphs in pdf file named `OutputFileName` in folder `OutputFilePath`.
`OutputFileName`	character (with default): name of the pdf file that will be generated by the function if `SavePd=TRUE`, `length(OutputFileName) = =3`, see **PLOT OUTPUT** in **Value** section for more informations.
`OutputFilePath`	character (with default): path to the pdf file that will be generated by the function if `SavePdf=TRUE`. If it is not equal to "", it must be terminated by "/".
`SaveEstimates`	logical (with default): if TRUE save Bayes' estimates, credible interval at level 68% and 95% and the result of the Gelman and Rubin test of convergence, in a csv table named `OutputFileName` in folder `OutputFilePath`.
`OutputTableName`	logical (with default): name of the table that will be generated by the function if `SaveEstimates=TRUE`.
`OutputTablePath`	character (with default): path to the table that will be generated by the function if `SaveEstimates=TRUE`. If it is not equal to "", it must be terminated by "/".
`StratiConstraints`	numeric matrix or character(with default): input object for the stratigraphic relation between samples. If there is stratigraphic relation between samples see the details section for instructions regarding how to correctly fill `StratiConstraints`; the user can refer to a matrix (numeric matrix) or to a csv file (character). If there is no stratigraphic relation default value is suitable.

`sepSC` character (with default): if `StratiConstraints` is character, indicate column separator in `StratiConstraints` csv file.

`Model` character (with default): if **"full"**, error on estimate calibration curve is taken account. If **"naive"** this error is not taken account in the age estimate.

`CalibrationCurve`
character (with default): calibration curve chosen. Allowed inputs are

- **"Intcal13"** or **"Intcal13"** for Northern Hemisphere atmospheric radiocarbon calibration curve,
- **"Marine13"** or **"Marine13"** for Marine radiocarbon calibration curve,
- **"SHCal13"** or **"SHCal20"** for Southern Hemisphere atmospheric radiocarbon calibration curve
- **a csv file, with tree columns, the first column is dedicated to** `"Cal.BP"`**, the second to** `"XC-14.age"`**, the third to** `"Error"`**. The decimal of this file must be a dot, and the separator must be a comma.**

`Iter` integer (with default): number of iterations for the MCMC computation (for more information see rjags::jags.model).

`t` integer (with default): 1 every `t` iterations of the MCMC is considered for sampling the posterior distribution (for more information see rjags::jags.model.

`n.chains` integer (with default): number of independent chains for the model (for more information see rjags::jags.model.

`quiet` logical (with default): enables/disables rjags::rjags-package messages

`roundingOfValue`
integer (with default): Integer indicating the number of decimal places to be used, default set to 3.

Details

How to fill `StratiConstraints`?

If there is stratigraphic relations between samples, *C-14 age in* `Data_C14Cal` *must be ordered by order of increasing ages*.

The user can fill the `StratiConstraints` matrix as follow.

1. **Size of the matrix**: row number of `StratiConstraints` matrix is equal to `Nb_sample+1`, and column number is equal to `Nb_sample`.
2. **First line of the matrix**: for all `i in {1,...,Nb_Sample}`, `StratiConstraints[1,i]=1` that means the lower bound of the sample age (given in `PriorAge[2i-1]`) for the sample whose number ID is equal to `i`, is taken into account.
3. **Sample relations**: for all `j in {2,...,Nb_Sample+1}` and all `i in {j,...,Nb_Sample}`, `StratiConstraints[j,i]=1` if sample age whose number ID is equal to `j-1` is lower than sample age whose number ID is equal to `i`. Otherwise, `StratiConstraints[j,i]=0`.

Note that `StratiConstraints_{2:Nb_sample+1,1:Nb_sample}` is a upper triangular matrix.

The user can also use `SCMatrix` or SC_Ordered (if all samples are ordered) functions to construct the `StratiConstraints` matrix.

The user can also refer to a .csv file that contains the relation between samples as defined above. The user must take care about the separator used in the csv file using the argument `sepSC`.

** More precision on `Model` **

We propose two models "full" or "naive". If `Model = 'full'` that means measurement error and error on calibration curve are taken account in the Bayesian model; if `Model = "naive"` that means only error on measurement are taken account in the mode.

More precisely, the model considered here, as the one developed by Christen, JA (1994), assume multiplicative effect of errors to address the problem of outliers. In addition, to not penalise variables that are not outliers and damage theirs estimation, we introduce a structure of mixture, that means only variable that are considered as outlier have in addition a multiplicative error.

Value

NUMERICAL OUTPUT

1. **A list containing the following objects:**
   - **Sampling**: that corresponds to a sample of the posterior distributions of the age parameters;
   - **Outlier**: stating the names of samples that are considered as outliers;
   - **Model**: stating which model was chosen (`"full"` or `"naive"`);
   - **CalibrationCurve**: stating which radiocarbon calibration curve was chosen;
   - **PriorAge**: stating the priors used for the age parameter;
   - **StratiConstraints**: stating the stratigraphic relations between samples considered in the model.
2. **The Gelman and Rubin test of convergency**: print the result of the Gelman and Rubin test of convergence for the age estimate for each sample. A result close to one is expected.
   In addition, the user must visually assess the convergence of the trajectories by looking at the graph generated by the function (see **PLOT OUTPUT** for more informations).
   If both convergences (Gelman and Rubin test and plot checking) are satisfactory, the user can consider the estimates as valid. Otherwise, the user may try increasing the number of MCMC iterations (`Iter`) or being more precise if it is possible on the `PriorAge` parameter to reach convergence.
3. **Credible intervals and Bayes estimates**: prints the Bayes' estimates, the credible intervals at 95% and 68% for the age parameters for each sample.

PLOT OUTPUT

1. **MCMC trajectories**: A graph with the MCMC trajectories and posterior distributions of the age parameter is displayed.
   On each line, the plot on the left represents the MCMC trajectories, and the one on the right the posterior distribution of the parameter.
2. **Summary of sample age estimates**: plot credible intervals and Bayes' estimate of each sample age on one graph.

To give the results in a publication, we recommend to give the Bayes' estimate of the parameters as well as the credible interval at 95% or 68%.

Note

Please note that the initial values for all MCMC are currently all the same for all chains since we rely on the automatic initial value generation of JAGS. This is not optimal and will be changed in future. However, it does not affect the quality of the age estimates if the chains have converged.

Author(s)

Claire Christophe, Anne Philippe, Guillaume Guérin, Sebastian Kreutzer

References

Christen, JA (1994). Summarizing a set of radiocarbon determinations: a robust approach. Applied Statistics, 489-503.

Reimer PJ, Bard E, Bayliss A, Beck JW, Blackwell PC, Bronl Ramsey C, Buck CE, Cheng H, Edwards RL, Friedrich M, Grootes PM, Guilderson TP, Haflidason H, Hajdas I, Hatte C, Heaton TJ, Hoffmann DL, Hogg AG, Hughen KA, Kaiser KF, Kromer B, Manning SW, Niu M, Reimer RW, Richards DA, Scott EM, Southon JR, Staff RA, Turney CSM, van der Plicht J. 2013. IntCal13 ans Marine13 radiocarbon age calibration curves 0-50000 years cal BP. Radiocarbon 55(4)=1869-1887.

Hogg AG, Hua Q, Blackwell PG, Niu M, Buck CE, Guilderson TP, Heaton TJ, Palmer JG, Reimer PJ, Reimer RW, Turney CSM, Zimmerman SRH. 2013. SHCal13 Southern Hemisphere calibration, 0-50000 years cal. BP Radiocarbon 55(4):1889-1903

See Also

rjags::rjags-package, plot_MCMC, SCMatrix, plot_Ages

Examples

```
## Load data
data(DATA_C14,envir = environment())
C14Cal <- DATA_C14$C14[,1]
SigmaC14Cal <- DATA_C14$C14[,2]
Names <- DATA_C14$Names
nb_sample <- length(Names)

## Age computation of samples without stratigraphic relations
Age <- AgeC14_Computation(
 Data_C14Cal = C14Cal,
 Data_SigmaC14Cal = SigmaC14Cal,
 SampleNames = Names,
 Nb_sample = nb_sample,
 PriorAge = rep(c(20,60),nb_sample),
 Iter = 500,
 quiet = TRUE,
 roundingOfValue = 3)
```

AgeS *Output of* AgeS_Computation *function for the samples: "GDB5" and "GDB3"*

Description

Output of AgeS_Computation function for the samples: "GDB5" and "GDB3", there is no stratigraphic relation neither systematic errors.

Usage

```
data("AgeS")
```

Format

A list containing

`Sampling` MCMC.list that corresponds to a sample of the posterior distributions of the ages (in ka), palaeodoses (in Gy) and equivalent dose dispersions (in Gy) parameters of samples "GDB5" and "GDB3";

`Model_GrowthCurve` stating which dose response fitting option was chosen to run the function

`Distribution` stating which distribution was chosen to model the dispersion of individual equivalent dose values around the palaeodose of the sample;

`PriorAge` stating the priors used for the age parameter (in ka);

`StratiConstraints` stating the matrix of stratigraphic relations between samples considered in the model;

`CovarianceMatrix` stating the covariance matrix of error used in the model, highlighting not common errors between samples in our cases (diagonal matrix).

References

Tribolo, C., Asrat, A., Bahain, J. J., Chapon, C., Douville, E., Fragnol, C., Hernandez, M., Hovers, E., Leplongeon, A., Martin, L, Pleurdeau, D, Pearson, O , Puaud, S, Assefa, Z. (2017). Across the Gap: Geochronological and Sedimentological Analyses from the Late Pleistocene-Holocene Sequence of Goda Buticha, Southeastern Ethiopia. PloS one, 12(1), e0169418.

Examples

```
data(AgeS)
str(AgeS)
```

AgeS_Computation *Bayesian analysis for OSL age estimation of various samples*

Description

This function computes the age (in ka) of at least two samples according to the model developed in Combès and Philippe (2017), based on outputs of Generate_DataFile or Generate_DataFile_MG or both of them using combine_DataFiles.
Samples, for which data is available in several BIN files, can be analysed.
Single-grain or Multi-grain OSL measurements can be analysed simultaneously.

Usage

```
AgeS_Computation(
  DATA,
  SampleNames = DATA$SampleNames,
  Nb_sample = DATA$Nb_sample,
  PriorAge = rep(c(0.01, 100), Nb_sample),
  BinPerSample = rep(1, Nb_sample),
  SavePdf = FALSE,
  OutputFileName = c("MCMCplot", "summary"),
  OutputFilePath = c(""),
  SaveEstimates = FALSE,
  OutputTableName = c("DATA"),
  OutputTablePath = c(""),
  THETA = c(),
  sepTHETA = c(","),
  StratiConstraints = c(),
  sepSC = c(","),
  LIN_fit = TRUE,
  Origin_fit = FALSE,
  distribution = c("cauchy"),
  model = NULL,
  Iter = 10000,
  burnin = 4000,
  adapt = 1000,
  t = 5,
  n.chains = 3,
  jags_method = "rjags",
  autorun = FALSE,
  quiet = FALSE,
  roundingOfValue = 3,
  ...
)
```

Arguments

DATA	(**required**) Two inputs are possible: (1): DATA list of objects: LT, sLT, ITimes, dLab, ddot_env, regDose, J, K, Nb_measurement, provided by the function Generate_DataFile, Generate_DataFile_MG or combine_DataFiles. DATA contains informations for more than one sample. If there is stratigraphic relations between samples, informations in DATA must be ordered by order of increasing ages. See the details section to for more informations. (2): An object of class BayLum.list which is provided by the output of AgeS_Computation. When input of class BayLum.list is identified, no new JAGS model is created. Instead, the JAGS model specified by the AgeS_Computation output is extended. Useful for when convergence was not originally achieved and a complete restart is not desirable.
SampleNames	character vector: names of samples. The length of this vector is equal to Nb_sample.
Nb_sample	integer: number of samples, Nb_sample>1.
PriorAge	numeric vector (with default): lower and upper bounds for age parameter of each sample (in ka). Note that, length(PriorAge)=2*Nb_sample and PriorAge[2i-1,2i] corresponds to the lower and upper bounds of sample whose number ID is equal to i.
BinPerSample	integer vector (with default): vector with the number of BIN files per sample. The length of this vector is equal to Nb_sample. BinPerSample[i] corresponds to the number of BIN files for the sample whose number ID is equal to i. For more information to fill this vector, we refer to details in Generate_DataFile or in Generate_DataFile_MG.
SavePdf	logical (with default): if TRUE save graphs in pdf file named OutputFileName in folder OutputFilePath.
OutputFileName	character (with default): name of the pdf file that will be generated by the function if SavePdf = TRUE; length(OutputFileName)=2, see **PLOT OUTPUT** in **Value** section for more informations.
OutputFilePath	character (with default): path to the pdf file that will be generated by the function if SavePdf=TRUE. If it is not equal to "", it must be terminated by "/".
SaveEstimates	logical (with default): if TRUE save Bayes' estimates, credible interval at level 68% and 95% and the result of the Gelman en Rubin test of convergence, in a csv table named OutputFileName in folder OutputFilePath.
OutputTableName	character (with default): name of the table that will be generated by the function if SaveEstimates = TRUE.
OutputTablePath	character (with default): path to the table that will be generated by the function if SaveEstimates = TRUE. If it is not equal to "", it must be terminated by "/".
THETA	numeric matrix or character (with default): input object for systematic and individual error. If systematic errors are considered, see the details section for instructions regarding how to correctly fill THETA; the user can refer to a matrix (numeric matrix) or to a csv file (character). Otherwise, default value is suitable, and only individual errors are considered.

`sepTHETA` character (with default): if `THETA` is character, indicate column separator in `THETA` CSV-file.

`StratiConstraints`
numeric matrix or character(with default): input object for the stratigraphic relation between samples. If there is stratigraphic relation between samples see the details section for instructions regarding how to correctly fill `StratiConstraints`; the user can refer to a matrix (numeric matrix) or to a csv file (character). If there is no stratigraphic relation default value is suitable.

`sepSC` character (with default): if `StratiConstraints` is character, indicate column separator in `StratiConstraints` .csv file.

`LIN_fit` logical (with default): if TRUE (default) allows a linear component, on top of the (default) saturating exponential curve, for the fitting of dose response curves. See details section for more informations on the proposed dose response curves.

`Origin_fit` logical (with default): if TRUE, forces the dose response curves to pass through the origin. See details section for more informations on the proposed growth curves.

`distribution` character (with default): type of distribution that defines how individual equivalent dose values are distributed around the palaeodose. Allowed inputs are `"cauchy"`, `"gaussian"`, `"lognormal_A"` and `"lognormal_M"`, see details section for more informations.

`model` character (*optional*): allows to provide a custom model to the function as text string. Please note that if this option is chosen the parameter `distribution` is ignored and no safety net is applied. If the function crashes it is up to the user.

`Iter` integer (with default): the number of iterations to run which will be used to assess convergence and ages (see runjags::run.jags).

`burnin` integer (with default): the number of iterations used to "home in" on the stationary posterior distribution. These are not used for assessing convergence (see runjags::run.jags).

`adapt` integer (with default): the number of iterations used in the adaptive phase of the simulation (see runjags::run.jags).

`t` integer (with default): 1 every `t` iterations of the MCMC is considered for sampling the posterior distribution. (for more information see runjags::run.jags).

`n.chains` integer (with default): number of independent chains for the model (for more information see runjags::run.jags).

`jags_method` character (with default): select which method to use in order to call JAGS. jags_methods `"rjags"` (the default) and `"rjparallel"` have been tested. (for more information about these possibilities and others, see runjags::run.jags)

`autorun` logical (with default): choose to automate JAGS processing. JAGS model will be automatically extended until convergence is reached (for more information see runjags::autorun.jags).

`quiet` logical (with default): enables/disables `rjags` messages

`roundingOfValue`
integer (with default): Integer indicating the number of decimal places to be used, default = 3.

... further arguments that can be passed to control the Bayesian process. 1) When creating a new JAGS model, see details for supported arguments. 2) If extending a JAGS model see arguments of runjags::extend.JAGS.

Details

Supported ... arguments

ARGUMENT	INPUT	METHOD	DEFAULT	DESCRIPTION
max.time	character	rjparallel	Inf	maximum allowed processing time, e.g., 10m for 10 minutes (cf. ru
interactive	logical	rjparallel	FALSE	enable/disable interactive mode (cf. runjags::autorun.jags)
startburnin	integer	rjparallel	4000	number of burn-in iterations (cf. runjags::autorun.jags)
startsample	integer	rjparallel	10000	total number of samples to assess convergence (cf. runjags::autoru
inits	named list	rjparallel	NA	fine control over random seeds and random number generator runja

How to fill StratiConstraints

If there is stratigraphic relations between samples, *informations in DATA must be ordered by order of increasing ages*. To do this the user can either fill right Names in Generate_DataFile or in Generate_DataFile_MG (as it is indicated in Details section of these function), or ordered by order of increasing ages outputs of Generate_DataFile or Generate_DataFile_MG in combine_DataFiles

The user can fill the StratiConstraints matrix as follow.

1. **Size of the matrix**: row number of StratiConstraints matrix is equal to Nb_sample+1, and column number is equal to Nb_sample.
2. **First line of the matrix**: for all i in {1,...,Nb_Sample}, StratiConstraints[1,i]=1 that means the lower bound of the sample age (given in PriorAge[2i-1]) for the sample whose number ID is equal to i, is taken into account.
3. **Sample relations**: for all j in {2,...,Nb_Sample+1} and all i in {j,...,Nb_Sample}, StratiConstraints[j,i]=1 if sample age whose number ID is equal to j-1 is lower than sample age whose number ID is equal to i. Otherwise, StratiConstraints[j,i]=0.

Note that StratiConstraints_{2:Nb_sample+A,1:Nb_sample} is a upper triangular matrix.

The user can also use SCMatrix or SC_Ordered (if all samples are ordered) functions to construct the StratiConstraints matrix.

The user can also refer to a csv file that contains the relation between samples as defined above. The user must take care about the separator used in the csv file using the argument sepSC.

How to fill THETA covariance matrix concerning common and individual error?

If systematic errors are considered, the user can fill the THETA matrix as follows.

- row number of THETA is equal the column number, equal to Nb_sample.
- For all i in {1,...,Nb_sample}, THETA[i,i] contains individual error plus systematic error of the sample whose number ID is equal to i.
- For all i,j in {1,...,Nb_sample} and i different from j , THETA[i,j] contains common error between samples whose number ID are equal to i and j.

Note that `THETA[i,j]` is a symetric matrix.

The user can also refer to a .csv file that contains the errors as defined above.

Alternatively you can use the function create_ThetaMatrix.

Option on growth curves

As for `Age_Computation` and `Palaeodose_Computation`, the user can choose from 4 dose response curves:

- **Saturating exponential plus linear growth** (`AgesMultiCS2_EXPLIN`):
  for all x in IR+, `f(x)=a(1-exp(-x/b))+cx+d`; select
  - `LIN_fit=TRUE`
  - `Origin_fit=FALSE`
- **Saturating exponential growth** (`AgesMultiCS2_EXP`):
  for all x in IR+, `f(x)=a(1-exp(-x/b))+d`; select
  - `LIN_fit=FALSE`
  - `Origin_fit=FALSE`
- **Saturating exponential plus linear growth and fitting through the origin** (`AgesMultiCS2_EXPLINZO`):
  for all x in IR+, `f(x)=a(1-exp(-x/b))+cx`; select
  - `LIN_fit=TRUE`
  - `Origin_fit=TRUE`
- **Saturating exponential growth and fitting through the origin** (`AgesMultiCS2_EXPZO`):
  for all x in IR+, `f(x)=a(1-exp(-x/b))`; select
  - `LIN_fit=FALSE`
  - `Origin_fit=TRUE`

Option on equivalent dose distribution around the palaeodose

The use can choose between :

- `cauchy`: a Cauchy distribution with location parameter equal to the palaeodose of the sample;
- `gaussian`: a Gaussian distribution with mean equal to the palaeodose of the sample;
- `lognormal_A`: a log-normal distribution with mean or **A**verage equal to the palaeodose of the sample:
- `lognormal_M`: a log-normal distribution with **M**edian equal to the palaeodose of the sample.

Value

NUMERICAL OUTPUT

1. **A list of type** `BayLum.list` **containing the following objects:**
   - **Sampling**: that corresponds to a sample of the posterior distributions of the age (in ka), palaeodose (in Gy) and equivalent dose dispersion (in Gy) parameters for each sample;
   - **Model_GrowthCurve**: stating which dose response fitting option was chosen;

- **Distribution**: stating which distribution was chosen to model the dispersion of individual equivalent dose values around the palaeodose of the sample;
- **PriorAge**: stating the priors used for the age parameter (in ka);
- **StratiConstraints**: stating the stratigraphic relations between samples considered in the model;
- **CovarianceMatrix**: stating the covariance matrix of error used in the model, highlighting common errors between samples or not.
- **model**: returns the model that was used for the Bayesian modelling as a character
- **JAGS model output**: returns the JAGS model with class "runjags".

2. **The Gelman and Rubin test of convergency**: prints the result of the Gelman and Rubin test of convergence for the age, palaeodose and equivalent dose dispersion parameters for each sample. A result close to one is expected.
   In addition, the user must visually assess the convergence of the trajectories by looking at the graph generated by the function (see **PLOT OUTPUT** for more informations).
   If both convergences (Gelman and Rubin test and plot checking) are satisfactory, the user can consider the estimates as valid. Otherwise, the user may try increasing the number of MCMC iterations (`Iter`) or being more precise on the `PriorAge` parameter (for example specify if it is a young sample `c(0.01,10)` an old sample `c(10,100)`), or changing the parameter `distribution` or the growth curve, to reach convergence.
3. **Credible intervals and Bayes estimates**: prints the Bayes estimates, the credible intervals at 95% and 68% for the age, palaeodose and equivalent dose dispersion parameters for each sample.

PLOT OUTPUT

1. **MCMC trajectories**: A graph with the MCMC trajectories and posterior distributions of the age, palaeodose and equivalent dose dispersion parameters is displayed, there is one page per sample.
   The first line of the figure corresponds to the age parameter, the second to the palaeodose parameter and the third to the equivalent dose dispersion parameter. On each line, the plot on the left represents the MCMC trajectories, and the one on the right the posterior distribution of the parameter.
2. **Summary of sample age estimates**: plot credible intervals and Bayes estimate of each sample age on a same graph.

To give the results in a publication, we recommend to give the Bayes' estimate of the parameters as well as the credible interval at 95% or 68%.

Note

Please note that the initial values for all MCMC are currently all the same for all chains since we rely on the automatic initial value generation of JAGS. This is not optimal and will be changed in future. However, it does not affect the quality of the age estimates if the chains have converged.

Author(s)

Claire Christophe, Anne Philippe, Guillaume Guérin, Sebastian Kreutzer

References

Combes, Benoit and Philippe, Anne, 2017. Bayesian analysis of multiplicative Gaussian error for multiple ages estimation in optically stimulated luminescence dating. Quaternary Geochronology (39, 24-34)

Combes, B., Philippe, A., Lanos, P., Mercier, N., Tribolo, C., Guerin, G., Guibert, P., Lahaye, C., 2015. A Bayesian central equivalent dose model for optically stimulated luminescence dating. Quaternary Geochronology 28, 62-70. doi:10.1016/j.quageo.2015.04.001

See Also

Generate_DataFile, Generate_DataFile_MG, rjags::rjags-package, plot_MCMC, SCMatrix, Age_Computation, Palaeodose_Computation, plot_Ages, create_ThetaMatrix, runjags::autorun.jags

Examples

```
## Age computation of samples GDB5 and GDB3,
## load data
data(DATA2) # GD85
data(DATA1) # GD83

## produce DataFile
Data <- combine_DataFiles(DATA2, DATA1)

## without common error, assuming GDB5 age younger than GDB3 age
Nb_sample <- 2
SC <- matrix(
  data = c(1,1,0,1,0,0),
  ncol = 2,
  nrow = (Nb_sample+1),
  byrow = TRUE)

## Not run:
## run standard
Age <- AgeS_Computation(
  DATA = Data,
  Nb_sample = Nb_sample,
  SampleNames = c("GDB5","GDB3"),
  PriorAge = rep(c(1,100), 2),
  StratiConstraints = SC,
  Iter = 100,
  quiet = FALSE,
  jags_method = "rjags"
)

## extend model
Age_extended <- AgeS_Computation(
  DATA = Age,
  Nb_sample = Nb_sample,
  SampleNames = c("GDB5","GDB3"),
  PriorAge = rep(c(1,100), 2),
  StratiConstraints = SC,
```

```
  adapt = 0,
  burnin = 500,
  Iter = 1000,
  quiet = FALSE,
  jags_method = "rjags"
)

## autorun mode
Age <- AgeS_Computation(
  DATA = Data,
  Nb_sample = Nb_sample,
  SampleNames = c("GDB5","GDB3"),
  PriorAge = rep(c(1,100), 2),
  StratiConstraints = SC,
  Iter = 10000,
  quiet = FALSE,
  jags_method = "rjags",
  autorun = TRUE)

## parallel mode
Age <- AgeS_Computation(
  DATA = Data,
  Nb_sample = Nb_sample,
  SampleNames = c("GDB5","GDB3"),
  PriorAge = rep(c(1,100), 2),
  StratiConstraints = SC,
  Iter = 10000,
  quiet = FALSE,
  jags_method = "rjparallel")

## End(Not run)
```

`combine_DataFiles` *Combine objects*

Description

Combine objects generated by Generate_DataFile and Generate_DataFile_MG

Usage

```
combine_DataFiles(...)

Concat_DataFile(...)
```

Arguments

`...`	list objects generated by Generate_DataFile or Generate_DataFile_MG

Details

The function allows to combine data already generated by Generate_DataFile or Generate_DataFile_MG. The number of input objects is not limited and the function works similar to the standard base R function `c()`, but preserves the particular structure of the objects imported and generated by 'BayLum'. The elements are combined by list element names.

Combining such data is rather useful in two scenarios:

- The data have been already imported and treated and then stored in RData-files. Using the function `combine_DataFiles()` will significantly speed up the processing time,
- simultaneous analysis of single and multi-grain OSL measurements.

Value

A nested list combining the input objects.

Function version

0.1.1

Author(s)

Sebastian Kreutzer, IRAMAT-CRP2A, UMR 5060, CNRS - Université Bordeaux Montaigne (France), adapting the idea from the function 'Concat_DataFile()' by Claire Christophe.

See Also

Generate_DataFile, Generate_DataFile_MG

Examples

```
# load data files
data(DATA1,envir = environment())
data(DATA2,envir = environment())

#combine objects
DATA3 <- combine_DataFiles(DATA1, DATA2)
str(DATA3)
```

`create_DataFile` *Prepare input data for subsequent BayLum Analysis*

Description

The function pre-processes input data from BIN/BINX file, XSYG files or Luminescence::RLum.Analysis objects for `'BayLum'`. The parameters for the modelling are controlled by a to be supplied YAML configuration file (please read package vignette).

Usage

```
create_DataFile(config_file, verbose = TRUE)
```

Arguments

config_file	character (**required**): path to YAML configuration file; alternatively the config_file can be a list similar to the R representation of the imported YAML file. This enables on-the fly modifications
verbose	logical (*with default*): enable/disable terminal feedback

Details

The function uses a single configuration file based on the YAML format and operates in two modes:

(1) The YAML file contains the path to the files and the function attempts to import them. In such a case, all files must be thoroughly prepared (e.g., strictly follow the SAR protocol etc.).

(2) Alternatively, the YAML file contains no file paths but the data were imported and processed before `create_DataFile()` was called (recommended). Then the function is searching for objects with the sample name in the global environment. Example: `samp1 <- Luminescence::read_BIN2R(...)` with `samp1` the sample name as specified in the YAML file.

For more details, please see the package vignette.

Value

Returns a list that can be processed by the modelling functions of 'BayLum'

- **LT** (one list per sample); each list contains all L/T values for the corresponding sample;
- **sLT** (one list per sample); each list contains all uncertainties on L/T values for the corresponding sample;
- **ITimes** (one list per sample); each list contains irradiation time values for the corresponding sample;
- **dLab**, a matrix containing in line `i`, the laboratory dose rate and its variance for sample `i`;
- **ddot_env**, a matrix containing in line `i`, the environmental dose rate and its variance (excluding the common error terms) for sample `i`;
- **regDose** (one list per sample); each list contains all regenerated doses;
- **J**, a vector giving, for each BIN file, the number of aliquots selected for the analysis;
- **K**, a vector giving, for each BIN file, the number of regenerative doses in the SAR protocol;
- **Nb_measurement**, a vector giving, for each BIN file, the number of measurements;
- **SampleNames**, a character vector with the sample names;
- **Nb_sample**, the number of samples in the dataset

Function version

0.1.0

Author(s)

Sebastian Kreutzer, Institute of Geography, Ruprecht-Karl University of Heidelberg (Germany), in parts based on code by Claire Christophe

See Also

write_YAMLConfigFile, yaml::read_yaml, Luminescence::read_BIN2R, Luminescence::read_XSYG2R, Luminescence::subset_SingleGrainData

Examples

```
##set path to YAML file
yaml_file <- system.file("extdata/example.yml", package = "BayLum")

samp1_file <- system.file("extdata/samp1/bin.bin", package = "BayLum")
samp2_file <- system.file("extdata/samp2/bin.bin", package = "BayLum")

## import BIN files
samp1 <- Luminescence::read_BIN2R(samp1_file, verbose = FALSE) |>
 subset(POSITION == 2 & GRAIN == 32)
samp2 <- Luminescence::read_BIN2R(samp2_file, verbose = FALSE) |>
 subset(POSITION == 2 & GRAIN == 32)

## create file
create_DataFile(yaml_file)
```

---

create_FolderTemplates

Create Folder Templates

Description

Create file and folder structure templates on the user hard drive as expected by Generate_DataFile and Generate_DataFile_MG. Files and data in the folders must then be overwritten manually with user data. The function intends to minimise the errors going along with the creation of these folder structures. The function uses the example data of `BayLum` to create the templates.

Usage

```
create_FolderTemplates(
  path,
  mode = "SG",
  n_folders = 1,
  names = paste("Sample_", 1:n_folders),
  verbose = TRUE
)
```

Arguments

path	character (**required**): path to the folder where the templates should be created
mode	character (*with default*): depending on the dataset you can create templates or single grain (SG) or multi-grain (MG) data
n_folders	numeric (*with default*): number of template folders to be created
names	character (*optional*): allows give own names to the subfolders.
verbose	logical (*with default*): enables/disables verbose mode

Value

If the templates were created successfully on the hard drive, the function returns nothing.

Function version

0.1.0

Author(s)

Sebastian Kreutzer, Geography & Earth Sciences, Aberystwyth University (United Kingdom)

See Also

Generate_DataFile, Generate_DataFile_MG

Examples

```
create_FolderTemplates(tempdir())
```

---

create_ThetaMatrix *Create Theta Matrix*

Description

Create the Θ matrix with the shared uncertainties that can used as input in, e.g., AgeS_Computation and Age_OSLC14 which is used for the covariance matrix Σ (Combès & Philippe, 2017)

Usage

```
create_ThetaMatrix(input, output_file = NULL, sigma_s, ...)
```

Arguments

`input`	character or data.frame (*optional*): input data frame or file connection to import a CSV-file with the needed information. If nothing is provided the function returns an input template. The argument `output_file` can be used to write this input template to the file system
`output_file`	character (optional): file path for the output CSV-file, the field separator is hard set to `","`. Please use utils::write.table for more flexibility.
`sigma_s`	numeric (**required**): named character with values for systematic uncertainties. Those values are lab-specific. Can be set to `NULL` to remove systematic uncertainties. The order of the *named* vector is not important, but the naming! **Note**: some of the uncertainties have a unit, please check details.
`...`	further arguments that can be passed to utils::read.table (for the CSV-file import)

Details

The function intends to ease the creation of the $Theta$ matrix, which cannot be created straight forward, e.g., base R functions such as `stats::cov`. The relationship between the covariance matrix $Sigma$ and $Theta$ is given with

$$\Sigma_i j = A_i * A_j * \Theta_i j$$

For details see Combès & Philippe, 2017 and Guérin et al. (2021).

Input modes

The function supports two different operation modes:

1. `input` is left empty: the function returns a data.frame template that can be used as input (the option `output_file` works as well)
2. `input` is fed with a data.frame or a `character` (file path), the Θ matrix is returned

Input format

The function expects either a CSV-file or a data.frame as input. To create template you can run the function leaving the argument `input` empty (see example). Please note the format of the input table (data.frame) needs to kept as specified in the template.

The following table lists the meaning of the columns:

COLUMN	DESCRIPTION	UNIT
SAMPLE_ID	sample name	-
DR_BETA_K	standard error beta-dose rate K	Gy/ka
DR_BETA_U	standard error beta-dose rate U	Gy/ka
DR_BETA_Th	standard error beta-dose rate Th	Gy/ka
DR_GAMMA_K	standard error gamma-dose rate K	Gy/ka
DR_GAMMA_U	standard error gamma-dose rate U	Gy/ka
DR_GAMMA_Th	standard error gamma-dose rate Th	Gy/ka
DR_GAMMA_TOTAL	standard error total gamma-dose rate	Gy/ka
DR_TOTAL	total dose rate	Gy/ka
DR_TOTAL_X	standard error total dose rate	Gy/ka

Note: All columns can be set to 0 or `NA` *but no column must be left empty! If a value > 0 is provided for* `DR_GAMMA_TOTAL` *this value is taken and values in, e.g.,* `DR_GAMMA_K` *are discarded (set to 0)!*

Systematic uncertainties

The following table provides information on the named argument that can be provided via the argument `sigma_s`. Missing values are not allowed, all values must be set.

ARGUMENT	DESCRIPTION	UNIT
`s_betaK`	relative uncertainty K concentration	-
`s_betaU`	relative uncertainty U concentration	-
`s_betaTh`	relative uncertainty Th concentration	-
`s_gammaK`	relative uncertainty K concentration	-
`s_gammaU`	relative uncertainty U concentration	-
`s_gammaTh`	relative uncertainty Th concentration	-
`s_gammaDR`	relative uncertainty gamma-dose rate	-
`s_CAL`	relative uncertainty beta-source calibration	-
`s_intDR`	absolute uncertainty internal dose rate	Gy/ka

Value

A symmetric $Theta$ matrix or if `input` is missing, a data.frame with an input template

Function version

0.1.0

Author(s)

Sebastian Kreutzer, IRAMAT-CRP2A, UMR 5060, CNRS-Université Bordeaux Montaigne (France), based on an 'MS Excel' sheet by Guillaume Guérin, IRAMAT-CRP2A, UMR 5060, CNRS-Université Bordeaux Montaigne (France)

References

Combès, B., Philippe, A., 2017. Bayesian analysis of individual and systematic multiplicative errors for estimating ages with stratigraphic constraints in optically stimulated luminescence dating. Quaternary Geochronology 39, 24–34. doi:10.1016/j.quageo.2017.02.003

Guérin, G., Lahaye, C., Heydari, M., Autzen, M., Buylaert, J.-P., Guibert, P., Jain, M., Kreutzer, S., Lebrun, B., Murray, A.S., Thomsen, K.J., Urbanova, P., Philippe, A., 2021. Towards an improvement of optically stimulated luminescence (OSL) age uncertainties: modelling OSL ages with systematic errors, stratigraphic constraints and radiocarbon ages using the R package BayLum. Geochronology 3, 229—245. doi:10.5194/gchron32292021

See Also

AgeS_Computation, Age_OSLC14, utils::read.table, utils::write.table

Examples

```
##(1) return template data.frame (no file output)
create_ThetaMatrix()

## Not run:
##(2) return template as data.frame + file
file_path <- tempfile(fileext = ".csv")
create_ThetaMatrix(output_file = file_path )

##NOT RUNNING EXAMPLE for sigma_s
calc_ThetaMatrix(...,
sigma_s =  c(
 s_betaK = 0.010,
 s_betaU = 0.007,
 s_betaTh = 0.006,
 s_gammaK = 0.010,
 s_gammaU = 0.007,
 s_gammaTh = 0.006,
 s_gammaDR = 0.05,
 s_CAL = 0.020,
 s_intDR = 0.030))


## End(Not run)
```

DATA_C14 *C14 cal age estiamte and its error*

Description

C14 cal age estiamtes and theirs error of samples S-EVA-26510, S-EVA-26506, S-EVA-26507, S-EVA-26508.

Usage

```
data("DATA_C14")
```

Format

A list containing:

`Names:` character vector of the sample names;

`C14:` numeric matrix, in the first column the 14C Cal age of the samples, and in the second column theirs errors.

References

For more informations on this sample we refer to the following publication:

Guerin, G., Frouin, M., Talamo, S., Aldeias, V., Bruxelles, L., Chiotti, L., Goldberg, P., Hublin, J.J., Jain, M., Lahaye, C., Madelaine, S., Maureille, B., McPherron, S., Mercier, N., Murray, A., Sandgathe, D., Steele, T., Thomsen, K., Turq, A. (2015). A multi-method luminescence dating of the Palaeolithic sequence of La Ferrassie based on new excavations adjacent to the La Ferrassie 1 and 2 skeletons. Journal of Archaeological Science, 58, 147-166.

Examples

```
data(DATA_C14)
(DATA_C14)
```

DATA1 *DATA of sample named GDB3*

Description

list of objects: LT, sLT, ITimes, dLab, ddot_env, regDose, J,K,Nb_measurement obtained using Generate_DataFile function with single-grain OSL measurementsl of the sample GDB3.

Usage

```
data("DATA1")
```

Format

A list containing:

`LT:` (one list per sample): each list contains all L/T values for the corresponding sample;

`sLT:` (one list per sample): each list contains all uncertainties on L/T values for the corresponding sample;

`ITimes:` (one list per sample): each list contains irradiation time values for the corresponding sample;

`dLab=` a matrix containing in line `i`, the laboratory dose rate and its variance for sample `i`;

`ddot_env:` a matrix containing in line `i`, the environmental dose rate and its variance (excluding the common error terms) for sample `i`;

`regDose:` (one list per sample): each list contains all regenerated doses;

`J:` a vector giving, for each BIN file, the number of aliquots selected for the analysis;

`K:` a vector giving, for each BIN file, the number of regenerative doses in the SAR protocol;

`Nb_measurement:` a vector giving, for each BIN file, the number of measurements;

References

For more informations on this sample we refer to the following publication:

Tribolo, C., Asrat, A., Bahain, J. J., Chapon, C., Douville, E., Fragnol, C., Hernandez, M., Hovers, E., Leplongeon, A., Martin, L, Pleurdeau, D, Pearson, O , Puaud, S, Assefa, Z. (2017). Across the Gap: Geochronological and Sedimentological Analyses from the Late Pleistocene-Holocene Sequence of Goda Buticha, Southeastern Ethiopia. PloS one, 12(1), e0169418.

Examples

```
data(DATA1)
str(DATA1)
```

DATA2 *DATA on sample named GDB5*

Description

list of objects: LT, sLT, ITimes, dLab, ddot_env, regDose, J,K,Nb_measurement obtained using `Generate_DataFile` function with single-grain OSL measurementsl of the sample GDB5.

Usage

```
data("DATA2")
```

Format

A data frame containing:

`LT:` (one list per sample): each list contains all L/T values for the corresponding sample;

`sLT:` (one list per sample): each list contains all uncertainties on L/T values for the corresponding sample;

`ITimes:` (one list per sample): each list contains irradiation time values for the corresponding sample;

`dLab:` a matrix containing in line `i`, the laboratory dose rate and its variance for sample `i`;

`ddot_env:` a matrix containing in line `i`, the environmental dose rate and its variance (excluding the common error terms) for sample `i`;

`regDose:` (one list per sample): each list contains all regenerated doses;

`J:` a vector giving, for each BIN file, the number of aliquots selected for the analysis;

`K:` a vector giving, for each BIN file, the number of regenerative doses in the SAR protocol;

`Nb_measurement:` , a vector giving, for each BIN file, the number of measurements;

References

For more informations on this sample we refer to the following publication:

Tribolo, C., Asrat, A., Bahain, J. J., Chapon, C., Douville, E., Fragnol, C., Hernandez, M., Hovers, E., Leplongeon, A., Martin, L, Pleurdeau, D, Pearson, O , Puaud, S, Assefa, Z. (2017). Across the Gap: Geochronological and Sedimentological Analyses from the Late Pleistocene-Holocene Sequence of Goda Buticha, Southeastern Ethiopia. PloS one, 12(1), e0169418.

Examples

```
data(DATA2)
str(DATA2)
```

DATA3 *DATA of sample named FER1*

Description

list of objects: LT, sLT, ITimes, dLab, ddot_env, regDose, J,K,Nb_measurement obtained using `Generate_DataFile` function with multi-grain OSL measurementsl of the sample FER1.

Usage

```
data("DATA3")
```

Format

A list containing:

`LT:` (one list per sample): each list contains all L/T values for the corresponding sample;

`sLT:` (one list per sample): each list contains all uncertainties on L/T values for the corresponding sample;

`ITimes:` (one list per sample): each list contains irradiation time values for the corresponding sample;

`dLab=` a matrix containing in line `i`, the laboratory dose rate and its variance for sample `i`;

`ddot_env:` a matrix containing in line `i`, the environmental dose rate and its variance (excluding the common error terms) for sample `i`;

`regDose:` (one list per sample): each list contains all regenerated doses;

`J:` a vector giving, for each BIN file, the number of aliquots selected for the analysis;

`K:` a vector giving, for each BIN file, the number of regenerative doses in the SAR protocol;

`Nb_measurement:` a vector giving, for each BIN file, the number of measurements;

References

For more informations on this sample we refer to the following publication:

Guerin, G., Frouin, M., Talamo, S., Aldeias, V., Bruxelles, L., Chiotti, L., Goldberg, P., Hublin, J.J., Jain, M., Lahaye, C., Madelaine, S., Maureille, B., McPherron, S., Mercier, N., Murray, A., Sandgathe, D., Steele, T., Thomsen, K., Turq, A. (2015). A multi-method luminescence dating of the Palaeolithic sequence of La Ferrassie based on new excavations adjacent to the La Ferrassie 1 and 2 skeletons. Journal of Archaeological Science, 58, 147-166.

Examples

```
data(DATA3)
str(DATA3)
```

Generate_DataFile-deprecated

Generates, from one (or several) BIN-file(s) of Single-grain OSL measurements, a list of luminescence data and information before statistical analysis (DEPRECATED)

Description

This function is used to generate, from the BIN file(s), a list of values of: **Single-grain** OSL intensities and associated uncertainties, regenerative doses, etc., which will be the input of the Bayesian models. To be easy-to-use, this function requires a rigorous organisation - all needed files should be arranged in one folder - of informations concerning each BIN file.
It is possible to process data for various samples simultaneously and to consider more than one BIN file per sample.

Usage

```
Generate_DataFile(
  Path,
  FolderNames,
  Nb_sample,
  Nb_binfile = length(FolderNames),
  BinPerSample = rep(1, Nb_sample),
  sepDP = c(","),
  sepDE = c(","),
  sepDS = c(","),
  sepR = c("="),
  verbose = TRUE,
  ...
)
```

Arguments

`Path`	character (**required**): the path to the project folder, containing one or more sub folders in which the BIN files are located. If it is not equal to "", it must be terminated by "/".
`FolderNames`	character (**required**): list of names of the sub-folders containing the BIN files • each sub folder must contain a BIN file and associated csv files. See details for more informations on associated csv files required in the sub folders. If there is more than one BIN file per sample, see the details section for instructions regarding how to correctly fill the `FolderNames` vector.
`Nb_sample`	integer (**required**): number of samples.
`Nb_binfile`	integer (with default): number of BIN files. It must be equal to, or greater than `Nb_sample`.
`BinPerSample`	integer vector (with default): vector with the number of BIN files per sample. The length of this vector must be equal to `Nb_sample` and the sum of entries of this vector must be equal to `Nb_binfile`. If there is more than one BIN file per sample, see the details section for instructions regarding how to correctly fill `BinPerSample` vector. Otherwise, this vector must contain a list of 1 values.
`sepDP`	character (with default): column separator in the `DiscPose.csv` files.
`sepDE`	character (with default): column separator in the `DoseEnv.csv` files.
`sepDS`	character (with default): column separator in the 'DoseLab.csv" files.
`sepR`	character (with default): column separator in the Rule.csv files.
`verbose`	logical (with default): enable/disable verbose mode
`...`	further arguments that can be passed to Luminescence::read_BIN2R.

Details

With `Path` and `FolderNames`, this function goes to the sub folders containing the BIN files and associated information to compute the luminescence data.

**** What are the required files in each subfolder? ****

Each sub folder can be named, for example, as the sample name followed by a number; it must contain:

- **bin.bin**: the bin file renamed as bin.BIN (note: the name of all files matters);
- **DiscPos.csv**: a two columns csv file containing the list of disc and grain position number of the previously selected grains (typically this list will include the position of grains based on their sensitivity, recycling or other properties);
- **DoseEnv.csv**: a two columns file containing the observation of the natural (or environmental), dose rate, and its non-shared variance (i.e. after removing all shared errors), both in Gy. Note: the user shall provide the squared value of the error associated with the dose rate experienced by the sample grains in nature;
- **DoseSourve.csv**: a two columns file containing the observation of the laboratory dose rate, and its variance (squared error) both in Gy;

- **rule.csv**: a csv file containing information on
  - `beginSignal=` the first channel for summing the natural or regenerative OSL signal (typically 1 or 6);
  - `endSignal=` the last channel for summing the natural or regenerative OSL signal (typically 5 or 10);
  - `beginBackground=` the first channel for background estimation of the natural or regenerative OSL signal (typically 76 or 81);
  - `endBackground=` the last channel for background estimation of the natural or regenerative OSL signal (typically 95 or 100);
  - `beginTest=`,
  - `endTest=`,
  - `beginTestBackground=`,
  - `endTestBackground=` same values as above, for the test dose response (typically the same values should be used);
  - `inflatePercent=` uncertainty arising from the instrument reproducibility (typically 0.02, i.e. 2\
  - `nbOfLastCycleToRemove=` number of cycles at the end of the SAR protocol which should not be included in the dose response curve fitting (typically 1 if only a recycling test is performed, or 2 if both recycling and IR depletion are tested).

**** How to fill the `FolderNames` vector? ****

`FolderNames` is a vector of length `Nb_binfile`. `FolderNames[i]` is the name (e.g., Sample1-File1, or successive names separated by "/" signs, if BIN files are in subfolders, e.g. Sample1/File1) of the subfolder containing all informations on the BIN file of ID number `i`. The names in `FolderNames` are ordered following two rules:

- The names in the `FolderNames` vector must be ordered following the sample order (the names of subfolders containing BIN files for the same sample should follow each other in the `FolderNames` vector, e.g. Sample1, Sample2-File1, Sample2-File2, etc.).
- If stratigraphic constraints apply to samples, and so a **Bayesian model with stratigraphic constraints** is implemented, then the names in the `FolderNames` vector must be ordered by order of increasing ages.
  For example, `FolderNames=c(noun1,noun2)`, in which case `noun1` (respectively, `noun2`) corresponds to the subfolder name containing the BIN file of sample 1 (respectively of sample 2). In addition, if we know that sample 1 is younger than sample 2, then `FolderNames` vector is correctly filled.
  If conversely, `FolderNames=c(noun2,noun1)`, the analysis performed by `AgeS_Computation` would not be consistent.

**** How to fill the `BinPerSample` vector? ****

`BinPerSample[i]` correponds to the number of BIN files for the sample whose number ID is equal to `i`.
For example, let us consider a case with two samples (Sample1 and Sample2), with 2 BIN files

for Sample1 and 1 for Sample2. In this case, `Nb_binfile=3` and `Nb_sample=2`. The user may then set `FolderNames=c("Sample1-File1", "Sample1-File2", "Sample2-File1")`, in which case `"Sample1-File1"` is the name of the subfolder containing the first BIN file for Sample1, `"Sample1-File2"` the name of the subfolder for the second BIN file of Sample1; eventually, `"Sample2-File1"` is the name of the subfolder containing the BIN file for the second sample. In this case, `BinPerSample=c(2,1)`.

For the general BIN-file structure, the reader is referred to the following website: `http://www.nutech.dtu.dk/`

The function Luminescence::read_BIN2R developed in Luminescence::Luminescence-package package is used to read the BIN files.

Value

A list containing the following objects:

- **LT** (one list per sample); each list contains all L/T values for the corresponding sample;
- **sLT** (one list per sample); each list contains all uncertainties on L/T values for the corresponding sample;
- **ITimes** (one list per sample); each list contains irradiation time values for the corresponding sample;
- **dLab**, a matrix containing in line `i`, the laboratory dose rate and its variance for sample `i`;
- **ddot_env**, a matrix containing in line `i`, the environmental dose rate and its variance (excluding the common error terms) for sample `i`;
- **regDose** (one list per sample); each list contains all regenerated doses;
- **J**, a vector giving, for each BIN file, the number of aliquots selected for the analysis;
- **K**, a vector giving, for each BIN file, the number of regenerative doses in the SAR protocol;
- **Nb_measurement**, a vector giving, for each BIN file, the number of measurements.

** How to save this list **

You can save this list in a `.RData` object. To do this, you can use the function `save`. Then, to load this list you can use the function `load` (see example section fore more details).

Author(s)

Claire Christophe, Sebastian Kreutzer, Anne Philippe, Guillaume Guerin

See Also

Luminescence::read_BIN2R, `combine_DataFiles`, `Generate_DataFile_MG`, `LT_RegenDose` `Age_Computation`, `AgeS_Computation`, `Palaeodose_Computation`

Examples

```
## Not run:
## Example for one sample with one Bin File
path<- system.file("extdata/samp1", "", package="BayLum")
folder=""
nbsample=1  # give the number of sample
Data <- Generate_DataFile(
 Path = path,
 FolderNames = folder,
 Nb_sample = nbsample,
 verbose = FALSE)
str(Data)

## to save information in RData object in folder containing bin file
# save(Data,file=c(paste(path,folder,'Data.RData',sep="")))
## to load information containing Data.RData object
# load(file=c(paste(path,folder,"Data.RData",sep="")))

## End(Not run)
```

Generate_DataFile_MG-deprecated

Generates, from one (or several) BIN file(s) of Multi-grain OSL measurements a list of luminescence data and information before statistical analysis (DEPRECATED)

Description

This function is used to generate, from the BIN file(s), a list of values of:

Multi-grain OSL intensities and associated uncertainties, regenerative doses, etc., which will be the input of the Bayesian models. To be easy-to-use, this function requires a rigorous organisation - all needed files should be arranged in one folder - of informations concerning each BIN file.
It is possible to process data for various samples simultaneously and to consider more than one BIN-file per sample.

Usage

```
Generate_DataFile_MG(
  Path,
  FolderNames,
  Nb_sample,
  Nb_binfile = length(FolderNames),
  BinPerSample = rep(1, Nb_sample),
  sepD = c(","),
  sepDE = c(","),
  sepDS = c(","),
  sepR = c("="),
```

```
  verbose = TRUE,
  force_run1_at_a_time = FALSE,
  ...
)
```

Arguments

`Path`	character (**required**): the path to the project folder, containing one or more sub folders in which the BIN files are located. If it is not equal to `""`, it must end with `"/"`.
`FolderNames`	character (**required**) vector: list of names of the sub-folders containing the BIN files • each sub folder must contain a BIN file and associated csv files. See details for more informations on associated csv files required in the sub folders. If there is more than one BIN file per sample, see the details section for instructions regarding how to correctly fill the `FolderNames` vector.
`Nb_sample`	integer (**required**): number of samples
`Nb_binfile`	integer (with default): number of BIN files. It must be equal to, or greater than `Nb_sample`.
`BinPerSample`	integer vector (with default): vector with the number of BIN files per sample. The length of this vector must be equal to `Nb_sample` and the sum of entries of this vector must be equal to `Nb_binfile`. If there is more than one BIN file per sample, see the details section for instructions regarding how to correctly fill `BinPerSample` vector. Otherwise, this vector must contain a list of 1 values.
`sepD`	character (with default): column separator in the `DiscPose.csv` files.
`sepDE`	character (with default): column separator in the `DoseEnv.csv` files.
`sepDS`	character (with default): column separator in the `DoseLab.csv` files.
`sepR`	character (with default): column separator in the `Rule.csv` files.
`verbose`	logical (with default): enable/disable verbose mode
`force_run1_at_a_time`	logical (*with default*): if set to `TRUE`, the order of the records is pushed to follow the one "Run 1 at a time" order (this is, all sequence steps were performed on one aliquot before moving to the next aliquot), regardless of their original sequence. The default is `FALSE` because `'BayLum'` assumes that the sample was measured with the "Run 1 at a time" option (only Risø readers, lexsyg readers do not have another option). In other words, the argument allows you to automatically correct your input data to follow the order `'BayLum'` expects. Why isn't the default value `TRUE`?. Because this re-ordering must fail if a measurement position was used more than once for different samples! This typically happens when different BIN/BINX files are merged.
`...`	further arguments that can be passed to Luminescence::read_BIN2R.

Details

With `Path` and `FolderNames`, this function goes to the sub folders containing the BIN files and associated information to compute the luminescence data.

** What are the required files in each subfolder? **

Each subfolder can be named, for example, as the sample name followed by a number; it must contain:

- **bin.bin**, the bin file renamed as bin.BIN (note: the name of all files matters);
- **Disc.csv**, a one columns csv file containing the list of disc number of the previously selected grains (typically this list will include the position of grains based on their sensitivity, recycling or other properties);
- **DoseEnv.csv**, a two columns file containing the observation of the natural (or environmental), dose rate, and its non-shared variance (i.e. after removing all shared errors), both in Gy. Note: the user shall provide the squared value of the error associated with the dose rate experienced by the sample grains in nature;
- **DoseSourve.csv**, a two columns file containing the observation of the laboratory dose rate, and its variance (squared error), both in Gy;
- **rule.csv**, a csv file containing information on
  - `beginSignal=` the first channel for summing the natural or regenerative OSL signal (typically 1 or 6);
  - `endSignal=` the last channel for summing the natural or regenerative OSL signal (typically 5 or 10);
  - `beginBackground=` the first channel for background estimation of the natural or regenerative OSL signal (typically 76 or 81);
  - `endBackground=` the last channel for background estimation of the natural or regenerative OSL signal (typically 95 or 100);
  - `beginTest`,
  - `endTest`,
  - `beginTestBackground`,
  - `endTestBackground=` same values as above, for the test dose response (typically the same values should be used);
  - `inflatePercent=` uncertainty arising from the instrument reproducibility (typically 0.02, i.e. 2\
  - `nbOfLastCycleToRemove=` number of cycles at the end of the SAR protocol which should not be included in the dose response curve fitting (typically 1 if only a recycling test is performed, or 2 if both recycling and IR depletion are tested).

** How to fill the `FolderNames` vector? **

`FolderNames` is a vector of length `Nb_binfile`. `FolderNames[i]` is the name (e.g., Sample1-File1, or successive names separated by "/" signs, if BIN files are in subfolders, e.g. Sample1/File1) of the subfolder containing all informations on the BIN file of ID number `i`. The names in `FolderNames` are ordered following two rules:

- The names in the `FolderNames` vector must be ordered following the sample order (the names of subfolders containing BIN files for the same sample should follow each other in the `FolderNames` vector, e.g. Sample1, Sample2-File1, Sample2-File2, etc.).

- If stratigraphic constraints apply to samples, and so a **Bayesian model with stratigraphic constraints** is implemented, then the names in the `FolderNames` vector must be ordered by order of increasing ages.
  For example, `FolderNames=c(noun1,noun2)`, in which case `noun1` (respectively, `noun2`) corresponds to the subfolder name containing the BIN file of sample 1 (respectively of sample 2). In addition, if we know that sample 1 is younger than sample 2, then `FolderNames` vector is correctly filled.
  If conversely, `FolderNames=c(noun2,noun1)`, the analysis performed by `AgeS_Computation` would not be consistent.

** How to fill the `BinPerSample` vector? **

`BinPerSample[i]` corresponds to the number of BIN files for the sample whose number ID is equal to `i`.
For example, let us consider a case with two samples (Sample1 and Sample2), with 2 BIN files for Sample1 and 1 for Sample2. In this case, `Nb_binfile=3` and `Nb_sample=2`. The user may then set `FolderNames=c("Sample1-File1", "Sample1-File2", "Sample2-File1")`, in which case `"Sample1-1"` is the name of the subfolder containing the first BIN file for Sample1, `"Sample1-File2"` the name of the subfolder for the second BIN file of Sample1; eventually, `"Sample2-1"` is the name of the subfolder containing the BIN file for the second sample. In this case, `BinPerSample=c(2,1)`.

For the general BIN-file structure, the reader is referred to the following website: `http://www.nutech.dtu.dk/`

The function Luminescence::read_BIN2R is used to read the BIN files.

Value

A list containing the following objects:

- **LT** (one list per sample); each list contains all L/T values for the corresponding sample;
- **sLT** (one list per sample); each list contains all uncertainties on L/T values for the corresponding sample;
- **ITimes** (one list per sample); each list contains irradiation time values for the corresponding sample;
- **dLab**, a matrix containing in line `i`, the laboratory dose rate and its variance for sample `i`;
- **ddot_env**, a matrix containing in line `i`, the environmental dose rate and its variance (excluding the common error terms) for sample `i`;
- **regDose** (one list per sample); each list contains all regenerated doses;
- **J**, a vector giving, for each BIN file, the number of aliquots selected for the analysis;
- **K**, a vector giving, for each BIN file, the number of regenerative doses in the SAR protocol;
- **Nb_measurement**, a vector giving, for each BIN file, the number of measurements;

** How to save this list **

You can save this list in a `.RData` object. To do this, you can use the function `save`. Then, to load this list you can use the function `load` (see example section fore more details).

Note

The function imports only BIN/BINX-file records which have been previously selected.

Author(s)

Claire Christophe, Sebastian Kreutzer, Anne Philippe, Guillaume Guérin

See Also

Luminescence::read_BIN2R, combine_DataFiles, LT_RegenDose Age_Computation, AgeS_Computation, Palaeodose_Computation

Examples

```
## Not run:
path <- system.file("extdata/FER1", "", package="BayLum")
folder <- ""
# give the number of sample
nbsample <- 1
DATA <- Generate_DataFile_MG(
 Path = path,
 FolderNames = folder,
 Nb_sample = nbsample)
str(DATA)

# to save information in RData object in folder containing bin file
#save(DATA,file=c(paste(path,folder,'DATA.RData',sep="")))
# to load information containing DATA.RData object
#load(file=c(paste(path,folder,"DATA.RData",sep="")))

## End(Not run)
```

IntCal13 *Atmospheric North data for calibration of 14C age*

Description

As 14C years is not equal to calendar years because atmospheric 14C concentration varies through time. Hence, data in AtmosphericNorth_CalC14 allows a calibration for mid-latitude Northern Hemisphere atmospher reservoir.

Usage

```
data("IntCal13")
```

Format

A data frame with 3 variables.

`CAL.BP` a numeric vector correpondig to calendar years befor present

`X14C.age` a numeric vector correponding to 14C age

`Error` a numeric vector correponding to error arround 14C age measurement

References

Reimer PJ, Bard E, Bayliss A, Beck JW, Blackwell PC, Bronl Ramsey C, Buck CE, Cheng H, Edwards RL, Friedrich M, Grootes PM, Guilderson TP, Haflidason H, Hajdas I, Hatte C, Heaton TJ, Hoffmann DL, Hogg AG, Hughen KA, Kaiser KF, Kromer B, Manning SW, Niu M, Reimer RW, Richards DA, Scott EM, Southon JR, Staff RA, Turney CSM, van der Plicht J. 2013. IntCal13 ans Marine13 radiocarbon age calibration curves 0-50000 years cal BP. Radiocarbon 55(4)=1869-1887.

Examples

```
data(IntCal13)
## maybe str(IntCal13) ; head(IntCal13) ...
```

IntCal20 *Atmospheric North data for calibration of 14C age*

Description

As 14C years is not equal to calendar years because atmospheric 14C concentration varies through time. Hence, data in AtmosphericNorth_CalC14 allows a calibration for mid-latitude Northern Hemisphere atmospher reservoir.

Usage

```
data("IntCal20")
```

Format

A data frame with 3 variables.

`CAL.BP` a numeric vector correpondig to calendar years befor present

`X14C.age` a numeric vector correponding to 14C age

`Error` a numeric vector correponding to error arround 14C age measurement

References

Reimer, P., Austin, W., Bard, E., Bayliss, A., Blackwell, P., Bronk Ramsey, C., . . . Talamo, S. (2020). The IntCal20 Northern Hemisphere Radiocarbon Age Calibration Curve (0–55 cal kBP). Radiocarbon, 62(4), 725-757. doi:10.1017/RDC.2020.41

Examples

```
data(IntCal20)
## maybe str(IntCal20) ; head(IntCal20) ...
```

`LT_RegenDose-deprecated`

Plots Lx/Tx as a function of the regenerative dose (DEPRECATED)

Description

This function plots `Lx/Tx` values as a function of regenerative dose, for every selected aliquot and for each sample.

Usage

```
LT_RegenDose(
  DATA,
  Path,
  FolderNames,
  SampleNames = FolderNames,
  Nb_sample,
  BinPerSample = rep(1, Nb_sample),
  SG = rep(TRUE, Nb_sample),
  sepDP = c(","),
  nrow = 3L,
  ncol = nrow
)
```

Arguments

`DATA`	list (**required**): list of objects `LT`, `sLT`, `ITimes`, `dLab`, `ddot_env`, `regDose`, `J`, `K`, `Nb_measurement`, #' provided by Generate_DataFile or Generate_DataFile_MG or combine_DataFiles.`DATA` can contain information from more than one sample.
`Path`	character (**required**): path to the project folder (the same as the one used in Generate_DataFile or Generate_DataFile_MG to provide `DATA`)
`FolderNames`	character (**required**): vector of names of the sub-folders containing the BIN-files, which were used by Generate_DataFile or Generate_DataFile_MG to generate the `DATA` object.
`SampleNames`	character (with default): Names of samples. To use if there is more than one bin file per sample.
`Nb_sample`	integer (**required**): ID number (in `[1,Nb_sample]`) of the sample selected for plotting L/T as a function of regenerative doses. Required if the `DATA` object contains information for more than one sample.

BinPerSample	integer (with default): integer vector (with default): vector with the number of BIN files per sample, which was used in Generate_DataFile or Generate_DataFile_MG to generate the DATA object.
SG	logical (with default): vector to set the type of measurement for each sample length(SG)=Nb_sample.If the sample of number ID equal to i, SG[i]=TRUE if it is a Single-grain OSL measurements, SG[i]=FALSE if it is a Multi-grain OSL measurements.
sepDP	character (with default): column separator in the DiscPos.csv file or in Disc.csv file. It must be the same separator for all samples, for single-grain OSL measurements or multi-grain OSL measurements.
nrow	integer (with default): controls the arrangement of the plots, here the number of rows. Can be set to NULL.
ncol	integer (with default): controls the arrangement of the plots, here the number of columns. Can be set to NULL.

Details

To fill FolderNames and BinPerSample, we refer to the **Detail** section from the Generate_DataFile or Generate_DataFile_MG function. As well for a precise description of input DATA.

Value

Lx/Tx plots; there are as many plots as selected aliquots in the DiscPos.csv file. There are 9 plots per page. There is not interpolation.

Author(s)

Claire Christophe, Sebastian Kreutzer, Anne Philippe, Guillaume Guérin

See Also

Generate_DataFile, Generate_DataFile_MG

Examples

```
## Not run:
## load data file generated by the function Generate_DataFile
data(DATA3,envir = environment())
path<- system.file("extdata/FER1", "", package="BayLum")
folder=""
samplename <- "FER1"
LT_RegenDose(
 DATA = DATA3,
 Path = path,
 FolderNames = folder,
 SampleNames = samplename,
 Nb_sample = 1,
 SG = FALSE)

## End(Not run)
```

Marine13 *Marine data for calibration of 14C age*

Description

As 14C years is not equal to calendar years because atmospheric 14C concentration varies through time. Hence, data in marine_CalC14 allows a calibration for hypothetical "global" marine reservoir.

Usage

```
data("Marine13")
```

Format

A data frame with 3 variables.

`CAL.BP` a numeric vector correpondig to calendar years befor present

`X14C.age` a numeric vector correponding to 14C age

`Error` a numeric vector correponding to error arround 14C age measurement

References

Reimer PJ, Bard E, Bayliss A, Beck JW, Blackwell PC, Bronl Ramsey C, Buck CE, Cheng H, Edwards RL, Friedrich M, Grootes PM, Guilderson TP, Haflidason H, Hajdas I, Hatte C, Heaton TJ, Hoffmann DL, Hogg AG, Hughen KA, Kaiser KF, Kromer B, Manning SW, Niu M, Reimer RW, Richards DA, Scott EM, Southon JR, Staff RA, Turney CSM, van der Plicht J. 2013. IntCal13 ans Marine13 radiocarbon age calibration curves 0-50000 years cal BP. Radiocarbon 55(4)=1869-1887.

Examples

```
data(Marine13)
## maybe str(Marine13) ; head(Marine13) ...
```

Marine20 *Marine data for calibration of 14C age*

Description

As 14C years is not equal to calendar years because atmospheric 14C concentration varies through time. Hence, data in marine_CalC14 allows a calibration for hypothetical "global" marine reservoir.

Usage

```
data("Marine20")
```

Format

A data frame with 3 variables.

`CAL.BP` a numeric vector correpondig to calendar years befor present

`X14C.age` a numeric vector correponding to 14C age

`Error` a numeric vector correponding to error arround 14C age measurement

References

Heaton, T., Köhler, P., Butzin, M., Bard, E., Reimer, R., Austin, W., . . . Skinner, L. (2020). Marine20—The Marine Radiocarbon Age Calibration Curve (0–55,000 cal BP). Radiocarbon, 62(4), 779-820. doi:10.1017/RDC.2020.68

Examples

```
data(Marine20)
## maybe str(Marine20) ; head(Marine20) ...
```

MCMCsample *MCMC sample from the posterior distribution of the dataset GDB5*

Description

MCMC samples from the posterior distribution of "A" for age, "D" for palaeodose and "sD" for dispersion of equivalent doses around "D", of the data set GDB5.

Usage

```
data("MCMCsample")
```

Format

It is a matric with 6000 row and tree column.

`A` The first column of the matrice are sampled from the posterior distribution of the paramete `A`

`D` The first column of the matrice are sampled from the posterior distribution of the paramete `D`

`sD` The first column of the matrice are sampled from the posterior distribution of the paramete `sD`

References

Tribolo, C., Asrat, A., Bahain, J. J., Chapon, C., Douville, E., Fragnol, C., Hernandez, M., Hovers, E., Leplongeon, A., Martin, L, Pleurdeau, D, Pearson, O , Puaud, S, Assefa, Z. (2017). Across the Gap: Geochronological and Sedimentological Analyses from the Late Pleistocene-Holocene Sequence of Goda Buticha, Southeastern Ethiopia. PloS one, 12(1), e0169418.

Examples

```
data(MCMCsample)
## maybe str(MCMCsample) ; plot(MCMCsample[,1],type="l") ...
```

`Model_Age` *JAGS models use in* `Age_Computation`

Description

A list of JAGS models use to a Bayesian analysis of OSL age of one sample. There are models for various growth curves and various distrubution to describe equivalent dose distribution around the palaeodose.

Usage

```
data("Model_Age")
```

Format

This list contains:

`AgeMultiBF_EXPLIN` a list of 4 models that all consider a saturating exponential plus linear growth. These 4 models have different distribution to describe equivalent dose values around the palaeodose.

`AgeMultiBF_EXP` a list of 4 models that all consider a saturating exponential growth. These 4 models have different distribution to describe equivalent dose values around the palaeodose.

`AgeMultiBF_EXPZO` a list of 4 models that all consider a saturating exponential plus linear growth and fitting through the origin. These 4 models have different distribution to describe equivalent dose values around the palaeodose.

`AgeMultiBF_EXPLINZO` a list of 4 models that all consider a saturating exponential growth and fitting through the origin. These 4 models have different distribution to describe equivalent dose values around the palaeodose.

Details

The different distibutions to describe equivalent dose values around the palaeodose are:

`cauchy` a Cauchy distribution with postition parameter equal to the palaeodose of the sample

`gaussian` a Gaussian distribution with mean equal to the palaeodose of the sample

`lognormal_A` a log-normal distribution with mean or **A**verage equal to the palaeodose of the sample

`lognormal_M` a log-normal distribution with **M**edian equal to the palaeodose of the sample

For more information we refer to the function `Age_Computation`, section Details.

References

Plummer, M. (2003). JAGS: A program for analysis of Bayesian graphical models using Gibbs sampling. In Proceedings of the 3rd international workshop on distributed statistical computing, volume 124, page 125. Technische Universit at Wien, Austria.

Plummer, M. (2015). JAGS Version 4.0. 0 user manual.

See Also

rjags::rjags-package

Examples

```
data(Model_Age)
## Terminal print
## The JAGS model for a saturating exponential plus linear growth
## (a function of the type \code{f(x)=a(1-exp(-x/b))+cx+d})
## and a gaussian distribution of equivalent doses around the palaeodose:
writeLines(Model_Age$AgeMultiBF_EXPLIN$cauchy)
```

`Model_AgeC14` *JAGS models use in* `AgeC14_Computation`

Description

A list of JAGS models use to a Bayesian analysis of C14 calibration age of various sample. Stratigraphic relations can be taken in count to calibrate C14 ages. This ages take into account that some data can be an outlier.

Usage

```
data("Model_AgeC14")
```

Format

This list contains:

`full` a model considering error on calibration curve.

`naive` a model not considering error on calibration curve.

References

Reimer PJ, Bard E, Bayliss A, Beck JW, Blackwell PC, Bronl Ramsey C, Buck CE, Cheng H, Edwards RL, Friedrich M, Grootes PM, Guilderson TP, Haflidason H, Hajdas I, Hatte C, Heaton TJ, Hoffmann DL, Hogg AG, Hughen KA, Kaiser KF, Kromer B,Manning SW, Niu M, Reimer RW, Richards DA, Scott EM, Southon JR, Staff RA, Turney CSM, van der Plicht J. 2013. IntCal13 ans Marine13 radiocarbon age calibration curves 0-50000 years cal BP. Radiocarbon 55(4)=1869-1887.

Hogg AG, Hua Q, Blackwell PG, Niu M, Buck CE, Guilderson TP, Heaton TJ, Palmer JG, Reimer PJ, Reimer RW, Turney CSM, Zimmerman SRH. 2013. SHCal13 Southern Hemisphere calibration, 0-50000 years cal BP. Radiocarbon 55(4):1889-1903

See Also

rjags::rjags-package

Examples

```
data(Model_AgeC14)
writeLines(Model_AgeC14$full)
```

Model_AgeS *JAGS models use in* AgeS_Computation

Description

A list of JAGS models use to a Bayesian analysis of OSL age of various samples. There are models for various growth curves and various distrubution to describe equivalent dose distribution around the palaeodose.

Usage

```
data("Model_AgeS")
```

Format

This list contains:

`AgesMultiCS2_EXPLIN` a list of 4 models that all consider a saturating exponential plus linear growth. These 4 models have different distribution to describe equivalent dose values around the palaeodose.

`AgesMultiCS2_EXP` a list of 4 models that all consider a saturating exponential growth. These 4 models have different distribution to describe equivalent dose values around the palaeodose.

`AgesMultiCS2_EXPZO` a list of 4 models that all consider a saturating exponential plus linear growth and fitting through the origin. These 4 models have different distribution to describe equivalent dose values around the palaeodose.

`AgesMultiCS2_EXPLINZO` a list of 4 models that all consider a saturating exponential growth and fitting through the origin. These 4 models have different distribution to describe equivalent dose values around the palaeodose.

Details

The different distibutions to describe equivalent dose values around the palaeodose are:

`cauchy` a Cauchy distribution with postition parameter equal to the palaeodose of the sample

`gaussian` a Gaussian distribution with mean equal to the palaeodose of the sample

`lognormal_A` a log-normal distribution with mean or **A**verage equal to the palaeodose of the sample

`lognormal_M` a log-normal distribution with **M**edian equal to the palaeodose of the sample

For more information we refer to the function `AgeS_Computation`, section Details.

References

Plummer, M. (2003). JAGS: A program for analysis of Bayesian graphical models using Gibbs sampling. In Proceedings of the 3rd international workshop on distributed statistical computing, volume 124, page 125. Technische Universit at Wien, Austria.

Plummer, M. (2015). JAGS Version 4.0. 0 user manual.

See Also

rjags::rjags-package

Examples

```
data(Model_AgeS)
## The JAGS model for a saturating exponential plus linear growth
## (a function of the type \code{f(x)=a(1-exp(-x/b))+cx+d})
## and a gaussian distribution of equivalent doses around the palaeodose:
writeLines(Model_AgeS$AgesMultiCS2_EXP$gaussian)
```

Model_Palaeodose *JAGS models use in* Palaeodose_Computation

Description

A list of JAGS models use to a Bayesian analysis of OSL palaeodose of one or various samples. There are models for various growth curves and various distrubution to describe equivalent dose distribution around the palaeodose.

Usage

```
data("Model_Palaeodose")
```

Format

This list contains:

`PalaeodosesMultiBF_EXPLIN` a list of 4 models that all consider a saturating exponential plus linear growth. These 4 models have different distribution to describe equivalent dose values around the palaeodose.

`PalaeodosesMultiBF_EXP` a list of 4 models that all consider a saturating exponential growth. These 4 models have different distribution to describe equivalent dose values around the palaeodose.

`PalaeodosesMultiBF_EXPZO` a list of 4 models that all consider a saturating exponential plus linear growth and fitting through the origin. These 4 models have different distribution to describe equivalent dose values around the palaeodose.

`PalaeodosesMultiBF_EXPLINZO` a list of 4 models that all consider a saturating exponential growth and fitting through the origin. These 4 models have different distribution to describe equivalent dose values around the palaeodose.

Details

The different distibutions to describe equivalent dose values around the palaeodose are:

`cauchy` a Cauchy distribution with postition parameter equal to the palaeodose of the sample

`gaussian` a Gaussian distribution with mean equal to the palaeodose of the sample

`lognormal_A` a log-normal distribution with mean or **A**verage equal to the palaeodose of the sample

`lognormal_M` a log-normal distribution with **M**edian equal to the palaeodose of the sample

For more information we refer to the function `Palaeodose_Computation`, section Details.

References

Plummer, M. (2003). JAGS: A program for analysis of Bayesian graphical models using Gibbs sampling. In Proceedings of the 3rd international workshop on distributed statistical computing, volume 124, page 125. Technische Universit at Wien, Austria.

Plummer, M. (2015). JAGS Version 4.0. 0 user manual.

See Also

rjags::rjags-package

Examples

```
data(Model_Palaeodose)
writeLines(Model_Palaeodose$PalaeodosesMultiBF_EXPLIN$gaussian)
```

ModelC14 *Likelihood of C14 samples for JAGS models use in* Age_OSLC14

Description

A list of models for C14 data to define likelyhood in JAGS models.

Usage

```
data("ModelC14")
```

Format

This list contains:

`full` a model considering error on calibration curve.

`naive` a model not considering error on calibration curve.

References

Reimer PJ, Bard E, Bayliss A, Beck JW, Blackwell PC, Bronl Ramsey C, Buck CE, Cheng H, Edwards RL, Friedrich M, Grootes PM, Guilderson TP, Haflidason H, Hajdas I, Hatte C, Heaton TJ, Hoffmann DL, Hogg AG, Hughen KA, Kaiser KF, Kromer B,Manning SW, Niu M, Reimer RW, Richards DA, Scott EM, Southon JR, Staff RA, Turney CSM, van der Plicht J. 2013. IntCal13 ans Marine13 radiocarbon age calibration curves 0-50000 years cal BP. Radiocarbon 55(4)=1869-1887.

Hogg AG, Hua Q, Blackwell PG, Niu M, Buck CE, Guilderson TP, Heaton TJ, Palmer JG, Reimer PJ, Reimer RW, Turney CSM, Zimmerman SRH. 2013. SHCal13 Southern Hemisphere calibration, 0-50000 years cal BP. Radiocarbon 55(4):1889-1903

Examples

```
data(Model_AgeC14)
writeLines(Model_AgeC14$full)
```

ModelOSL *Likelihood of OSL samples for JAGS models use in* Age_OSLC14

Description

A list of models for OSL data to define likelyhood in JAGS models.

Usage

```
data("ModelOSL")
```

Format

This list contains:

`AgesMultiCS2_EXPLIN` a list of 4 models that all consider a saturating exponential plus linear growth. These 4 models have different distribution to describe equivalent dose values around the palaeodose.

`AgesMultiCS2_EXP` a list of 4 models that all consider a saturating exponential growth. These 4 models have different distribution to describe equivalent dose values around the palaeodose.

`AgesMultiCS2_EXPZO` a list of 4 models that all consider a saturating exponential plus linear growth and fitting through the origin. These 4 models have different distribution to describe equivalent dose values around the palaeodose.

`AgesMultiCS2_EXPLINZO` a list of 4 models that all consider a saturating exponential growth and fitting through the origin. These 4 models have different distribution to describe equivalent dose values around the palaeodose.

Details

The different distibutions to describe equivalent dose values around the palaeodose are:

`cauchy` a Cauchy distribution with postition parameter equal to the palaeodose of the sample

`gaussian` a Gaussian distribution with mean equal to the palaeodose of the sample

`lognormal_A` a log-normal distribution with mean or **A**verage equal to the palaeodose of the sample

`lognormal_M` a log-normal distribution with **M**edian equal to the palaeodose of the sample

For more information we refer to the function `AgeS_Computation`, section Details.

References

Plummer, M. (2003). JAGS: A program for analysis of Bayesian graphical models using Gibbs sampling. In Proceedings of the 3rd international workshop on distributed statistical computing, volume 124, page 125. Technische Universit at Wien, Austria.

Plummer, M. (2015). JAGS Version 4.0. 0 user manual.

Examples

```
data(ModelOSL)
## The JAGS model of the likelyhood for a saturating exponential plus linear growth
## (a function of the type \code{f(x)=a(1-exp(-x/b))+cx+d})
## and a gaussian distribution of equivalent doses around the palaeodose:
writeLines(ModelOSL$AgesMultiOSL_EXPLIN$gaussian)
```

`ModelPrior` *Prior for JAGS models use in* `Age_OSLC14`

Description

A list to define prior in JAGS models, taking acount OSL data and C14 data in stratigraphic constraint. The difficulty is in the fact that each cases is different. The youngest sample can be a C14 as well as a OSL sample. To resolve this problem we consider diferent cases thanks to this list.

Usage

```
data("ModelPrior")
```

Format

This list contains:

`Sample1_C14` model considering that the youngest sample is a C14 sample

`Sample1_OSL` model considering that the youngest sample is a OSL sample

`C14_OSL` model considering that the second sample is a C14 sample

`OSL_C14` model considering that the second sample is a OSL sample

`C14` model considering that the last sample is a C14 sample

`OSL` model considering that the last sample is a OSL sample

References

Plummer, M. (2003). JAGS: A program for analysis of Bayesian graphical models using Gibbs sampling. In Proceedings of the 3rd international workshop on distributed statistical computing, volume 124, page 125. Technische Universit at Wien, Austria.

Plummer, M. (2015). JAGS Version 4.0. 0 user manual.

Examples

```
data(ModelPrior)
## ModelPrior[[OSL]]
writeLines(ModelPrior$OSL)
```

`Palaeodose_Computation` *Bayesian analysis for the palaeodose estimation of various samples*

Description

This function computes the palaeodose (in Gy) of one or various samples according to the model developed in Combes et al (2015), based on an output of `Generate_DataFile` or `Generate_DataFile_MG` or both of them using `combine_DataFiles`.
Samples, for which data is avalilable in several BIN files, can be analysed.
Single-grain or Multi-grain OSL measurements can be analysed simultaneouly.

Usage

```
Palaeodose_Computation(
  DATA,
  SampleNames,
  Nb_sample,
  BinPerSample = rep(1, Nb_sample),
  SavePdf = FALSE,
  OutputFileName = c("MCMCplot"),
  OutputFilePath = c(""),
  SaveEstimates = FALSE,
  OutputTableName = c("DATA"),
  OutputTablePath = c(""),
  LIN_fit = TRUE,
  Origin_fit = FALSE,
  distribution = c("cauchy"),
  Iter = 50000,
  t = 5,
  n.chains = 3
)
```

Arguments

`DATA`	list of objects: LT, sLT, ITimes, dLab, ddot_env, regDose, J, K, Nb_measurement, provided by `Generate_DataFile` or `Generate_DataFile_MG`. `DATA` contains information for more than one sample.
`SampleNames`	character vector: names of sample. The length of this vector is equal to `Nb_sample`.
`Nb_sample`	integer: number of samples.
`BinPerSample`	integer vector (with default): vector with the number of BIN files per sample. The length of this vector is equal to `Nb_sample`. `BinPerSample[i]` corresponds to the number of BIN files for the sample whose number ID is equal to `i`. For more information to fill this vector, we refer to details in `Generate_DataFile` or `Generate_DataFile_MG`.

SavePdf	boolean (with default): if TRUE save graph in pdf file named OutputFileName in folder OutputFilePath.
OutputFileName	character (with default): name of the pdf files that will be generated by the function.
OutputFilePath	character (with default): path to the pdf files that will be generated by the function.
SaveEstimates	boolean (with default): if TRUE save Bayes estimates and credible interval at level 68 in a csv table named OutputFileName in folder OutputFilePath.
OutputTableName	character (with default): name of the table that will be generated by the function if SaveEstimates=TRUE.
OutputTablePath	character (with default): path to the table that will be generated by the function if SaveEstimates=TRUE.
LIN_fit	logical (with default): if TRUE (default) allows a linear component, on top of the (default) saturating exponential curve, for the fitting of dose response curves. Please see details for more informations on the proposed dose response curves.
Origin_fit	logical (with default): if TRUE, forces the dose response curves to pass through the origin. Please see details for more informations on the proposed growth curves.
distribution	character (with default): type of distribution that defines how individual equivalent dose values are distributed around the palaeodose. Allowed inputs are **"cauchy"**, **"gaussian"**, **"lognormal_A"** and **"lognormal_M"**.
Iter	integer (with default): number of iterations for the MCMC computation (for more information see [rjags::jags.model]).
t	integer (with default): 1 every t iterations of the MCMC is considered for sampling the posterior distribution (for more information see [rjags::jags.model]).
n.chains	integer (with default): number of independent chains for the model (for more information see [rjags::jags.model]).

Details

**** Option on growth curves ****

As for Age_Computation and AgeS_Computation, the user can choose from 4 dose response curves:

- **Saturating exponential plus linear growth** (PalaeodosesMultiBF_EXPLIN):
  for all x in IR+, f(x)=a(1-exp(-x/b))+cx+d; select
  - LIN_fit=TRUE
  - Origin_fit=FALSE
- **Saturating exponential growth** (PalaeodosesMultiBF_EXP):
  for all x in IR+, f(x)=a(1-exp(-x/b))+d; select
  - LIN_fit=FALSE
  - Origin_fit=FALSE

- **Saturating exponential plus linear growth and fitting through the origin** (`PalaeodosesMultiBF_EXPLINZO`): for all x in IR+, `f(x)=a(1-exp(-x/b))+cx`; select
  - `LIN_fit=TRUE`
  - `Origin_fit=TRUE`
- **Saturating exponential growth and fitting through the origin** (`PalaeodosesMultiBF_EXPZO`): for all x in IR+, `f(x)=a(1-exp(-x/b))`; select
  - `LIN_fit=FALSE`
  - `Origin_fit=TRUE`

**** Option on equivalent dose distribution around the palaeodose ****

The use can choose between :

- `cauchy`: a Cauchy distribution with location parameter equal to the palaeodose of the sample
- `gaussian`: a Gaussian distribution with mean equal to the palaeodose of the sample
- `lognormal_A`: a log-normal distribution with mean or **A**verage equal to the palaeodose of the sample
- `lognormal_M`: a log-normal distribution with **M**edian equal to the palaeodose of the sample

Value

NUMERICAL OUTPUT

1. **A list containing the following objects:**
   - **Sampling** that corresponds to a sample of the posterior distributions of palaeodose and equivalent dose dispersion parameters (both in Gy).
   - **Model_GrowthCurve**, stating which dose response fitting option was chosen;
   - **Distribution**, stating which distribution was chosen to model the dispersion of individual equivalent dose values around the palaeodose of the sample.
2. **The Gelman and Rubin test of convergency**: prints the result of the Gelman and Rubin test of convergency for palaeodose and equivalent dose dispersion parameters for each sample. A result close to one is expected.
   In addition, the user must visually assess the convergency of the trajectories by looking at the pdf file generated by the function (see **PLOT OUTPUT** for more informations).
   If both convergencies (Gelman and Rubin test and plot checking) are satisfactory, the user can consider the printed estimates as valid. Otherwise, the user may try increasing the number of MCMC interations (`Iter`) to reach convergency.
3. **Credible intervals and Bayes estimates**: prints the Bayes esitmates, the credible intervals at 95 the palaeodose and equivalent dose dispersion parameters for each sample.

PLOT OUTPUT

1. **MCMC trajectories** A graph with the MCMC trajectories and posterior distributions of the palaeodose and equivalent dose dispersion parameters is displayed, there is one page per sample.
   The first line of the figure corresponds to the palaeodose parameter and the second to the equivalent dose dispersion parameter. On each line, the plot on the left represents the MCMC trajectories, and the one on the right the posterior distribution of the parameter.

2. **Summary of palaeodose estimates**: plot credible intervals and Bayes estimate of each sample palaeodose on a same graph.

To give result in a publication, we recommend to give the Bayes estimate of the parameters as well as the credible interval at 95

Note

Please note that the initial values for all MCMC are currently all the same for all chains since we rely on the automatic initial value generation of JAGS. This is not optimal and will be changed in future. However, it does not affect the quality of the age estimates if the chains have converged.

Author(s)

Claire Christophe, Sebastian Kreutzer, Anne Philippe, Guillaume Guérin

References

Combes, B., Philippe, A., Lanos, P., Mercier, N., Tribolo, C., Guerin, G., Guibert, P., Lahaye, C., 2015. A Bayesian central equivalent dose model for optically stimulated luminescence dating. Quaternary Geochronology 28, 62-70. doi:10.1016/j.quageo.2015.04.001

See Also

Generate_DataFile, Generate_DataFile_MG, combine_DataFiles, [rjags::rjags-package], [plot_MCMC], Age_Computation, AgeS_Computation

Examples

```
## Load data
data(DATA1,envir = environment())
## Palaeodose computation of samples GDB3
P=Palaeodose_Computation(DATA=DATA1,Nb_sample=1,SampleNames=c("GDB5"),Iter=100)
```

---

plot_Ages *Create Age Plot*

Description

Create Age Plot

Usage

```
plot_Ages(
  object,
  sample_names = NULL,
  sample_order = NULL,
  plot_mode = "ages",
  ...
)
```

Arguments

`object`	list or data.frame (**required**): Output as created by functions like AgeC14_Computation, which is a list of class `BayLum.list`. Alternatively the function supports a data.frame as input, however, in such a case the data.frame must resemble the ages data.frame created by the computation functions otherwise the input will be silently ignored.
`sample_names`	character (optional): alternative sample names used for the plotting. If the length of the provided character vector is shorter than the real number of samples, the names are recycled.
`sample_order`	numeric (optional): argument to rearrange the sample order, e.g., `sample_order = c(4:1)` plots the last sample first.
`plot_mode`	character (*with default*): allows to switch from displaying ages as points with lines (`"ages"`) for the credible intervals to densities (`"density"`)
`...`	further arguments to control the plot output, standard arguments are: `cex`, `xlim`, `main`, `xlab`, `col` further (non-standard) arguments are: `grid` (`TRUE/FALSE`), `legend` (`TRUE/FALSE`), `legend.text` (character input needed), `legend.pos` graphics::legend, `legend.cex`. Additional arguments: `d_scale` (scales density plots), `show_ages` (add ages to density plots)

Details

This function creates an age plot showing the mean ages along with the credible intervals. The function provides various arguments to modify the plot output, however, for an ultimate control the function returns the data.frame extracted from the input object for own plots.

Value

The function returns a plot and the data.frame used to display the data

Function version

0.1.5

Author(s)

Sebastian Kreutzer, Institute of Geography, Ruprecht-Karl-University of Heidelberg (Germany), based on code written by Claire Christophe

See Also

AgeC14_Computation, AgeS_Computation

Examples

```
## load data
data(DATA_C14,envir = environment())
C14Cal <- DATA_C14$C14[,1]
SigmaC14Cal <- DATA_C14$C14[,2]
Names <- DATA_C14$Names
nb_sample <- length(Names)

## Age computation
Age <- AgeC14_Computation(
   Data_C14Cal = C14Cal,
   Data_SigmaC14Cal = SigmaC14Cal,
   SampleNames = Names,
   Nb_sample = nb_sample,
   PriorAge = rep(c(20,60),nb_sample),
   Iter = 500,
   quiet = TRUE)

## plot output
plot_Ages(Age)

## plot output
plot_Ages(Age, plot_mode = "density")
```

---

plot_MCMC *Plot MCMC trajectories and posterior distributions*

Description

This function uses the output of rjags::jags.model to visualise the traces of the MCMC and the corresponding densities. In particular it displays the posterior distributions of the age, if it is calculated, palaeodose and the equivalent dose dispersion parameters of the sample. The function output is very similar to plot output produced with the 'coda' package, but tailored to meet the needs in the context of the 'BayLum' package.

Usage

```
plot_MCMC(
  object,
  sample_names = NULL,
  variables = c("A", "D", "sD"),
  axes_labels = c(A = "Age (ka)", D = "D (Gy)", sD = "sD (Gy)"),
  n.chains = NULL,
  n.iter = 1000,
```

```
  smooth = FALSE,
  rug = TRUE,
  plot_single = FALSE,
  ...
)
```

Arguments

`object`	coda::mcmc.list or coda::mcmc (**required**): Output generated by rjags::jags.model, e.g., in Age_Computation. Alternatively, limited support is provided for `BayLum.list` object input
`sample_names`	character (optional): Names of the used samples. This argument overrides the optional argument `mtext`.
`variables`	character (with default): Variables in your coda::mcmc object to be plotted.
`axes_labels`	character (with default): Axes labels used for the trace and density plots. The labels should be provided as named character vector with the parameter names as the names used to assign the axes labelling. The labelling for the x-axis (trace plots) and y-axis (density plot) cannot be modified.
`n.chains`	numeric (optional): Set the number of chains to visualise, if nothing is provided the number of chains is determined from the input object
`n.iter`	integer (with default): Set the number of iterations to be visualised in the trace plots, regardless of the size of the input dataset as long as the real number of iterations is > `n.iter`. Please note that large numbers impact the plot performance.
`smooth`	logical (with default): Enable/disables smooth of trace plots using stats::smooth
`rug`	logical (with default): Enable/disables rug under density plots
`plot_single`	logical (with default): Enables/disables the single plot mode of the function, i.e. if set to `TRUE` every plot is returned in a single plot and own par settings can be applied.
`...`	further arguments that can be passed to modify the plot output. Supported arguments are `lwd`, `lty`, `col`, `type`, `cex`,`mtext`, cf. mtext for `mtext` and plot.default for the other arguments.

Details

The function is used in the function Age_Computation, AgeS_Computation and Palaeodose_Computation, but can be used also as standalone plot function.

Value

Two plots: Traces of the MCMC chains and the corresponding density plots. This plots are similar to coda::traceplot and coda::densplot.

Function version

0.1.5

Author(s)

Sebastian Kreutzer, Institute of Geography, Ruprecht-Karl University of Heidelberg (Germany). This function is a re-written version of the function 'MCMC_plot()' by Claire Christophe

See Also

Age_Computation, AgeS_Computation, Palaeodose_Computation, rjags::coda.samples and rjags::rjags-package

Examples

```
data(MCMCsample,envir = environment())
object <- coda::as.mcmc(MCMCsample)
plot_MCMC(object)
```

---

plot_RegDosePoints *Plot Regeneration Dose Points*

Description

Simple plot functionality to visualise L_x/T_x values against the dose extracted from data created by create_DataFile

Usage

```
plot_RegDosePoints(object, nrow = 3L, ncol = nrow, ...)
```

Arguments

object	list (**required**): input object created by create_DataFile
nrow	integer (*with default*): number of rows used for the plot panel
ncol	integer (*with default*): number of columns in the plot panel
...	further plot arguments passed down to modify the plot output. Supported arguments are xlab, ylab, type, pch, col, cex

Value

The function returns a plot

Author(s)

Sebastian Kreutzer, Institute of Geography, Heidelberg University (Germany)

See Also

create_DataFile

Examples

```
data(DATA3,envir = environment())
plot_RegDosePoints(DATA3)
```

`plot_Scatterplots` *Display Scatter Plot Matrix of the Bayesian Age Results*

Description

Create a hexbin plot matrix (hexbin::hexplom) of age results returned by the Bayesian age calculation.

Usage

```
plot_Scatterplots(
  object,
  variables = c("A"),
  sample_names = NULL,
  sample_selection = NULL,
  n.chains = NULL,
  plot_type = "hexbin",
  plot_mode = "matrix",
  ...
)

ScatterSamples(...)
```

Arguments

`object`	coda::mcmc.list or a data.frame (**required**): mcmc list objects generated by rjags::jags.model in AgeS_Computation, AgeC14_Computation or Age_OSLC14. If a data.frame is provided, only the first two columns are taken and `NA` values are automatically removed. The function can also handle `BayLum.list` objects directly for certain functions
`variables`	character (*with default*): variable to be selected for the scatter plot, e.g., `"A"`. Please note that you can only select one variable at the time
`sample_names`	character (*optional*): sample names shown in the plot matrix
`sample_selection`	numeric (*with default*): vector of samples to be plotted in the scatter matrix, e.g., `c(1,2)` will plot the first two samples, `c(1,3)` will plot samples 1 and 3 and `c(1:3)` will plot the first three samples
`n.chains`	integer (*with default*): allows to limit the number of chains shown, by default the results of all chains are plotted.

`plot_type` character (*with default*): switch between different plot types, `"hexbin"` (the default), based on the function hexbin::hexplom and `smoothScatter` (the alternative) based on a highly customised plot function using the function graphics::smoothScatter

`plot_mode` character (*with default*): switch between a `matrix` plot mode and a `single` plot mode. The plot mode `single` only works for `plot_type = smoothScatter` and creates a single plot panel for each sample. Please note that this cannot be further combined with other par settings.

`...` further arguments to control the plot output, standard plot arguments supported are `main`, `xlab`, `ylab`, `xlim`, `ylim`, `cex`. For additional arguments supporting a fine tuning of the plot, see details.

Details

Additional supported plot arguments

The following table lists additional arguments supported by the function in order to fine tune the graphical output. Such arguments, can just be added in the function call. Example, for disabling the graphics::rug in the plot mode `smoothScatter` you can type `plot_Scatterplots(..., rug = FALSE)` Please note that not all arguments are supported by all plot types.

ARGUMENT	SUPPORTED BY PLOT TYPE	DESCRIPTION
`colramp`	`hexbin` and `smoothScatter`	Option to define an own colour ramp, by defining an own function.
`pscales`	`hexbin` and `smoothScatter`	Controls the number of ticks shown on the plot axes, please note th
`bw_smoothScatter`	`smoothScatter`	Controls the bandwidth of the smooth scatter, cf. graphics::smooth
`rug`	`smoothScatter`	enables/disables rugs
`nlevels`	`smoothScatter`	controls the number of isolines shown (cf. graphics::contour)
`nrpoints`	`smoothScatter`	defines the number of `nrpoints` to be plotted graphics::smoothSca
`col_contour`	`smoothScatter`	defines the colour of the contour lines
`col_nrpoints`	`smoothScatter`	sets colour of the `nrpoints` in the scatter plot

Value

A scatter plot based on hexbin::hexplom

Function version

0.3.2

Author(s)

Sebastian Kreutzer, Institute of Geography, Ruprecht-Karl University of Heidelberg (Germany) , based on the function 'ScatterSamples()' by Claire Christophe, Anne Philippe, Guillaume Guérin

See Also

Age_Computation, AgeS_Computation, AgeC14_Computation, and rjags::rjags-package packages.

Examples

```
data(AgeS,envir = environment())

##hexbin
plot_Scatterplots(
   object = AgeS$Sampling,
   sample_names = c("GDB5", "GDB3"),
   sample_selection = c(1,2)
 )

##scatter smooth (matrix)
plot_Scatterplots(
   object = AgeS$Sampling,
   sample_names = c("GDB5", "GDB3"),
   sample_selection = c(1,2),
   plot_type = "smoothScatter")


##scatter smooth (single)
plot_Scatterplots(
   object = AgeS$Sampling,
   sample_names = c("GDB5", "GDB3"),
   sample_selection = c(1,2),
   plot_type = "smoothScatter",
   plot_mode = "single")
```

SC_Ordered *Create Stratigraphically Ordered Sample Matrix*

Description

Construct the stratigraphic matrix used in the functions AgeS_Computation and AgeC14_Computation for samples that are all ordered by increasing age.

Usage

```
SC_Ordered(Nb_sample)
```

Arguments

Nb_sample	integer (**required**): the number of samples; alternatively an object of class `BayLum.list` can be provided as input (such as produced by create_DataFile)

Value

Stratigraphic matrix where each sample are ordered by increasing order. This matrix can be integrated in the function AgeS_Computation. Please see AgeS_Computation for more information on this matrix.

Author(s)

Claire Christophe, Anne Philippe, Sebastian Kreutzer, Guillaume Guérin

See Also

AgeS_Computation, SCMatrix

Examples

```
SC <- SC_Ordered(Nb_sample = 3)
```

SCMatrix *Construct the Stratigraphic Constrain Matrix Interactively*

Description

This function helps to define the stratigraphic relation between samples using questions. The output of this function can be used in the function AgeS_Computation.

Usage

```
SCMatrix(DATA = NULL, Nb_sample, SampleNames)
```

Arguments

DATA	BayLum.list (*with default*): Object of class BayLum.list, if provided the other parameters are not any longer mandatory.
Nb_sample	integer (**required**): the sample number, if DATA is provided, the input is not required
SampleNames	character (**required**): sample names, if DATA is provided, the input is not required

Details

The function will ask if sample i is younger than sample j to construct the stratigraphic constrain matrix.

Value

Returns a matrix that summarise the ordered relation between samples. This matrix can be integrate in AgeS_Computation function. We refer to detail on AgeS_Computation for more information concerning this matrix.

Author(s)

Claire Christophe, Anne Philippe, Guillaume Guérin, Sebastian Kreutzer

See Also

AgeS_Computation

Examples

```
## Not run:
SCMatrix(
 Nb_sample = 2,
 SampleNames = c("sample1","sample2"))

## End(Not run)
```

SHCal13 *Atmospheric South data for calibration of 14C age*

Description

As 14C years is not equal to calendar years because atmospheric 14C concentration varies through time. Hence, data in AtmosphericSouth_CalC14 allows a calibration for mid-latitude Southern Hemisphere atmospher reservoir.

Usage

```
data("SHCal13")
```

Format

A data frame with 3 variables.

`CAL.BP` a numeric vector correpondig to calendar years (in ky) befor present

`X14C.age` a numeric vector correponding to 14C age

`Error` a numeric vector correponding to error arround 14C age measurement

References

Reimer PJ, Bard E, Bayliss A, Beck JW, Blackwell PC, Bronl Ramsey C, Buck CE, Cheng H, Edwards RL, Friedrich M, Grootes PM, Guilderson TP, Haflidason H, Hajdas I, Hatte C, Heaton TJ, Hoffmann DL, Hogg AG, Hughen KA, Kaiser KF, Kromer B, Manning SW, Niu M, Reimer RW, Richards DA, Scott EM, Southon JR, Staff RA, Turney CSM, van der Plicht J. 2013. IntCal13 ans Marine13 radiocarbon age calibration curves 0-50000 years cal BP. Radiocarbon 55(4)=1869-1887.

Examples

```
data(SHCal13)
## maybe str(SHCal13) ; head(SHCal13) ...
```

SHCal20 *Atmospheric South data for calibration of 14C age*

Description

As 14C years is not equal to calendar years because atmospheric 14C concentration varies through time. Hence, data in AtmosphericSouth_CalC14 allows a calibration for mid-latitude Southern Hemisphere atmospher reservoir.

Usage

```
data("SHCal20")
```

Format

A data frame with 3 variables.

`CAL.BP` a numeric vector correpondig to calendar years (in ky) befor present

`X14C.age` a numeric vector correponding to 14C age

`Error` a numeric vector correponding to error arround 14C age measurement

References

Hogg, A., Heaton, T., Hua, Q., Palmer, J., Turney, C., Southon, J., . . . Wacker, L. (2020). SHCal20 Southern Hemisphere Calibration, 0–55,000 Years cal BP. Radiocarbon, 62(4), 759-778. doi:10.1017/RDC.2020.59

Examples

```
data(SHCal20)
## maybe str(SHCal20) ; head(SHCal20) ...
```

write_BayLumFiles *Write BayLum .csv-files*

Description

This function allows the user to write all .csv files expected by Generate_DataFile and Generate_DataFile_MG. Unlike create_FolderTemplates, this function makes it possible to write .csv files with all information directly from R. No further modification of .csv files are required. The purpose of this function is (i) to reduce tedious manual editing of .csv-files and the errors that result (ii) to introduce an easy way to review information inside .csv-files (by revisiting code rather than opening individual .csv-files) and (iii) to streamline folder and file creation when preparing data to run BayLum's modelling functions. Note: the user will still need to move the appropriate .bin-files into all the sample folders.

Usage

```
write_BayLumFiles(
  folder,
  SampleNames = "Sample_1",
  BinPerSample = rep(1, length(SampleNames)),
  SubSampleNames = NULL,
  DiscPos = NULL,
  DRenv = 1,
  DRenv.error = 0.04,
  DRsource = 0.1,
  DRsource.error = 0.002,
  signal.integral.min = 6,
  signal.integral.max = 10,
  background.integral.min = 346,
  background.integral.max = 395,
  inflatePercent = 0.025,
  nbOfLastCycleToRemove = 2
)
```

Arguments

`folder`	character (*required**): The name of the main folder in which all subsequent BayLum files and folders will be located. This could be a path to an already existing folder, or the path/name of a folder to be created.
`SampleNames`	character (*required*): Vector of sample names.
`BinPerSample`	numeric (*with default*): Vector of numbers indicating the number of .bin-files per sample.
`SubSampleNames`	character (*optional*): Vector of names to give each subfolder within a sample when the number of .bin-files in a sample counts more than one. If omitted or NULL, the subfolders are named by the subfolder count number.
`DiscPos`	numeric (*with default*): List of data frames with each data frame having one or two columns to identify aliquots/grains to be included in the analysis. The first column corresponds to the position number, and the second column corresponds to the grain number. If the data frame has only one column, a Disc.csv will be written. If the data.frame has two columns, a DiscPos.csv will be written. The length of the list should be the number of .bin-files included.
`DRenv`	numeric (*with default*): Vector where `DRenv[i]` corresponds to environmental dose rate for `.bin-file[i]`. Length should be one or the number of .bin-files included in the analysis.
`DRenv.error`	numeric (*with default*): Vector where `DRenv.error[i]` corresponds to environmental dose rate error for `.bin-file[i]`. Length should be one or the number of .bin-files included in the analysis.
`DRsource`	numeric (*with default*): Vector where `DRsource[i]` corresponds to source dose rate for `.bin-file[i]`. Length should be one or the number of .bin-files included in the analysis.

`DRsource.error` numeric (*with default*): Vector where `DRsource.error[i]` corresponds to source dose rate error for `.bin-file[i]`. Length should be one or the number of .bin-files included in the analysis.

`signal.integral.min`
numeric (*with default*): Vector where `signal.integral.min[i]` corresponds to the channel number where the OSL signal should be summed from for`bin-file[i]` Length should be one or the number of .bin-files included in the analysis.

`signal.integral.max`
numeric (*with default*): Vector where `signal.integral.max[i]` corresponds to the channel number where the OSL signal should be summed from to for `bin-file[i]` Length should be one or the number of .bin-files included in the analysis.

`background.integral.min`
numeric (*with default*): Vector where `background.integral.min[i]` corresponds to the channel number where the OSL background signal should be summed from for`bin-file[i]` Length should be one or the number of .bin-files included in the analysis.

`background.integral.max`
numeric (*with default*): Vector where `background.integral.max[i]` corresponds to the channel number where the OSL background signal should be summed to for `.bin-file[i]`. Length should be one or the number of .bin-files included in the analysis.

`inflatePercent` numeric (*with default*): Vector where `inflatePercent[i]` corresponds to uncertainty due to instrumental reproducibility to`bin-file[i]` Length should be one or the number of .bin-files included in the analysis.

`nbOfLastCycleToRemove`
numeric (*with default*): Vector where `nbOfLastCycleToRemove[i]` corresponds to the number of regeneration points to remove in analysis for `bin-file[i]` Length should be one or the number of .bin-files included in the analysis.

Value

The function returns nothing, but writes the folder structure.

Function version

0.1.0

Author(s)

Frederik Baumgarten, RadPhys, DTU Physics, Technical University of Denmark (Denmark)

See Also

Generate_DataFile, Generate_DataFile_MG

Examples

```
# example samples
SampleNames <- c("OSL-1-MG","OSL-2-SG")

# number of .bin-files for each sample
BinPerSample <- c(1,3)

# List of data.frames of accepted aliquot/grain to be included
# in the analysis for each .bin-file.
DiscPos <- list(
data.frame("position" = 1:48),
data.frame("position" = c(1,1,1,1), "grain" = c(4,67,92,99)),
data.frame("position" = c(2,2,2,2), "grain" = c(7,13,41,72)),
data.frame("position" = c(3,3,3,3), "grain" = c(7,52,67,88)))

# example 1: write to disk (all together)
write_BayLumFiles(
folder = paste(tempdir(),"new_BayLum_folder",sep = "/"),
SampleNames = SampleNames,
BinPerSample = BinPerSample,
DiscPos = DiscPos,
DRenv = c(1.75, 1.52, 1.52, 1.52),
DRenv.error = c(0.04, 0.03, 0.03, 0.03),
DRsource = c(0.2075, 0.1501, 0.1501, 0.1501),
DRsource.error = c(0.0010, 0.0008, 0.0008, 0.0008))

# example 2: write to disk (by sample)
write_BayLumFiles(
folder = paste(tempdir(),"new_BayLum_folder",sep = "/"),
SampleNames = "OSL-1-MG",
BinPerSample = 1,
DiscPos = DiscPos[[1]],
DRenv = 1.75,
DRenv.error = 0.04,
DRsource = 0.2075,
DRsource.error = 0.0010)

write_BayLumFiles(
folder = paste(tempdir(),"new_BayLum_folder",sep = "/"),
SampleNames = "OSL-2-SG",
BinPerSample = 3,
DiscPos = DiscPos[2:4],
DRenv = 1.75,
DRenv.error = 0.04,
DRsource = 0.2075,
DRsource.error = 0.0010)
```

`write_YAMLConfigFile` *Write Auto Generated YAML BayLum Configuration File to the Disc*

Description

This little function helps to auto-generate a the BayLum YAML configuration file or a list that can be passed to create_DataFile. The YAML file itself can be modified in any text editor. The allowed parameters are extracted from the reference YAML file

Usage

```
write_YAMLConfigFile(output_file = NULL, ...)
```

Arguments

`output_file`	character (*with default*): valid file path of the output file
`...`	parameters to be preset in the YAML file (run `write_YAMLConfigFile()` to see allowed parameters) The parameter `sample` is special, because it can be provided as a character vector of any length. The number of elements in the vector (sample names) are then used to multiply the records in the configuration file.

Value

The function has two output modes:

- (1) `output_file = <file_path>`: Writes a YAML into the specified path and returns this path.
- (2) `output_file = NULL`: Returns a list of allowed function parameters that can be passed to the function **and** it returns a list that can be used a passed on to create_DataFile.

Function version

0.1.0

Author(s)

Sebastian Kreutzer, Institute of Geography, Ruprecht-Karl University of Heidelberg (Germany)

See Also

create_DataFile, yaml::read_yaml, Luminescence::read_BIN2R, Luminescence::read_XSYG2R

Examples

```
## generate list
write_YAMLConfigFile(
sample = c("samp1", "samp2"),
settings.rules.endTest = 10)

## generate file (here written in tempdir)
write_YAMLConfigFile(
 output_file = tempfile("configuration.yml"),
 sample = c("samp1", "samp2"),
```

```
    settings.rules.endTest = 10)
```

Index

∗ **IO**
create_FolderTemplates, 31
create_ThetaMatrix, 32
write_BayLumFiles, 74
∗ **datagen**
create_FolderTemplates, 31
create_ThetaMatrix, 32
SC_Ordered, 71
write_BayLumFiles, 74
∗ **datasets**
AgeS, 20
DATA1, 36
DATA2, 37
DATA3, 38
DATA_C14, 35
IntCal13, 47
IntCal20, 48
Marine13, 51
Marine20, 52
MCMCsample, 52
Model_Age, 53
Model_AgeC14, 54
Model_AgeS, 55
Model_Palaeodose, 57
ModelC14, 58
ModelOSL, 59
ModelPrior, 60
SHCal13, 73
SHCal20, 74
∗ **dplot**
plot_MCMC, 66

Age_Computation, 3, *12*, *25*, *27*, *42*, *47*, *54*, *62*, *64*, *67, 68*, *70*
Age_OSLC14, 7, *8*, *32*, *34*, *69*
AgeC14_Computation, 15, *65, 66*, *69–71*
AgeS, 20
AgeS_Computation, *5*, *7*, *20*, 21, *22*, *32*, *34*, *41, 42*, *46, 47*, *56*, *59*, *62*, *64*, *66–73*
character, *4*, *8–11*, *16, 17*, *22–24*, *26*, *30*, *32, 33*, *40*, *44*, *49, 50*, *65*, *67*, *69, 70*, *72*, *75*, *78*
coda::densplot, *67*
coda::mcmc, *67*
coda::mcmc.list, *67*, *69*
coda::traceplot, *67*
combine_DataFiles, *7, 8*, *21, 22*, *24*, 28, *42*, *47*, *49*, *61*, *64*
Concat_DataFile *(*combine_DataFiles*)*, 28
create_DataFile, 29, *68*, *71*, *78*
create_FolderTemplates, 31, *74*
create_ThetaMatrix, *25*, *27*, 32

data.frame, *33, 34*, *65*, *69*
DATA1, 36
DATA2, 37
DATA3, 38
DATA_C14, 35

Generate_DataFile, *3, 4*, *7–9*, *21, 22*, *24*, *27–29*, *31, 32*, *36–38*, *49, 50*, *61*, *64*, *74*, *76*
Generate_DataFile *(*Generate_DataFile-deprecated*)*, 39
Generate_DataFile-deprecated, 39
Generate_DataFile_MG, *3, 4*, *7–9*, *21, 22*, *24*, *27–29*, *31, 32*, *42*, *49, 50*, *61*, *64*, *74*, *76*
Generate_DataFile_MG *(*Generate_DataFile_MG-deprecated*)*, 43
Generate_DataFile_MG-deprecated, 43
graphics::contour, *70*
graphics::legend, *65*
graphics::rug, *70*
graphics::smoothScatter, *70*

hexbin::hexplom, *69, 70*

IntCal13, 47
IntCal20, 48
integer, *4, 5*, *10, 11*, *16, 17*, *22–24*, *40*, *44*, *49, 50*, *67–69*, *71, 72*

list, *4*, *11*, *22*, *24*, *30*, *49*, *65*, *68*, *78*
load, *42*, *46*
logical, *4, 5*, *9–11*, *16, 17*, *22–24*, *30*, *32*, *40*, *44*, *50*, *67*
LT_RegenDose, *42*, *47*
LT_RegenDose *(*LT_RegenDose-deprecated*)*, 49
LT_RegenDose-deprecated, 49
Luminescence::Luminescence-package, *42*
Luminescence::read_BIN2R, *31*, *40*, *42*, *44*, *46, 47*, *78*
Luminescence::read_XSYG2R, *31*, *78*
Luminescence::RLum.Analysis, *29*
Luminescence::subset_SingleGrainData, *31*

Marine13, 51
Marine20, 52
matrix, *9*, *22*, *72*
MCMCsample, 52
Model_Age, 53
Model_AgeC14, 54
Model_AgeS, 55
Model_Palaeodose, 57
ModelC14, 58
ModelOSL, 59
ModelPrior, 60
mtext, *67*

numeric, *4*, *8–10*, *16*, *22*, *32, 33*, *65*, *67*, *69*, *75, 76*

Palaeodose_Computation, *5*, *7*, *12*, *25*, *27*, *42*, *47*, *57*, 61, *67, 68*
par, *67*
plot.default, *67*
plot_Ages, *14*, *19*, *27*, 64
plot_MCMC, *7*, *14*, *19*, *27*, 66
plot_RegDosePoints, 68
plot_Scatterplots, 69

rjags::coda.samples, *68*
rjags::jags.model, *4, 5*, *10*, *17*, *66, 67*, *69*
rjags::rjags-package, *5*, *7*, *10*, *17*, *19*, *27*, *54–56*, *58*, *68*, *70*
rug, *67*
runjags::autorun.jags, *10, 11*, *23, 24*, *27*
runjags::extend.JAGS, *24*
runjags::run.jags, *10*, *14*, *23*

save, *42*, *46*
SC_Ordered, *11*, *17*, *24*, 71
ScatterSamples *(*plot_Scatterplots*)*, 69
SCMatrix, *11*, *14*, *19*, *24*, *27*, 72, *72*
SHCal13, 73
SHCal20, 74
stats::smooth, *67*

utils::read.table, *33, 34*
utils::write.table, *33, 34*

vector, *67*

write_BayLumFiles, 74
write_YAMLConfigFile, *31*, 77

yaml::read_yaml, *31*, *78*