Package 'Luminescence'

December 15, 2012

Type Package

Title Package for Luminescence Dating data analysis

Version 0.2

Date 2012-12-15

Maintainer Sebastian Kreutzer <sebastian.kreutzer@geogr.uni-giessen.de>

Description
Package provides a collection of various R functions for Luminescence Dating data analysis.

License GPL-3

Depends R (>= 2.15.1)

Imports methods

URL http://CRAN.R-project.org/package=Luminescence

Collate Analyse_SAR.OSLdata.R CW2pLM.R CW2pLMi.R CW2pHMi.R CW2pPMi.R
Calc_FadingCorr.R Calc_FuchsLang2001.R Calc_OSLLxTxRatio.R
Second2Gray.R fit_LMCurve.R fit_CWCurve.R plot_BINfileData.R
plot_KDE.R plot_GrowthCurve.R plot_Histogram.R
plot_RadialPlot.R readBIN2R.R RisoeBINfileData-class.R
Calc_CentralDose.R Calc_FiniteMixture.R Calc_MinDose3.R
Calc_MinDose4.R Calc_CommonDose.R

Author Sebastian Kreutzer [aut, trl, cre],Christoph Schmidt [aut],Margret C. Fuchs [aut],Michael Dietze [aut],Manfred Fischer [aut, trl],Christoph Burow [aut],Markus Fuchs [ths]

R topics documented:

`Luminescence-package` *Collection of functions for luminescence dating data analysis*

Description

This package provides various functions developed for the purpose of Luminescence Dating data analysis.

Details

Package:	Luminescence
Type:	Package
Version:	0.2
Date:	2012-12-15
License:	GPL-3

Author(s)

Authors

Christoph Burow	University of Cologne
Michael Dietze	TU Dresden
Manfred Fischer	University of Bayreuth
Margret C. Fuchs	TU Bergakademie Freiberg

Sebastian Kreutzer JLU Giessen
Christoph Schmidt University of Bayreuth

Supervisor

Markus Fuchs, JLU Giessen, Germany

Support contact

<team@r-luminescence.de>

Package maintainer

Sebastian Kreutzer, JLU Giessen, Germany,
<sebastian.kreutzer@geogr.uni-giessen.de>

References

Kreutzer, S., Schmidt, C., Fuchs, M.C., Dietze, M., Fischer, M. & Fuchs, M., 2012. Introducing an R package for luminescence dating analysis. Ancient TL, 30, pp. 1-8.

Analyse_SAR.OSLdata *Analyse SAR CW-OSL measurements.*

Description

The function analyses SAR CW-OSL curve data and provides a summary of the measured data for every position. The output of the function is optimized for SAR OSL measurements on quartz.

Usage

```
Analyse_SAR.OSLdata(input.data,
                    signal.integral,
                    background.integral,
                    position, run, set,
                    info.measurement = "unkown measurement",
                    log = "",
                    output.plot = FALSE,
                    cex.global = 1)
```

Arguments

`input.data`	Risoe.BINfileData-class (**required**): input data from a Risoe BIN file, produced by the function readBIN2R.
`signal.integral`	vector (**required**): channels used for the signal integral, e.g. `signal.integral=c(1:2)`
`background.integral`	vector (**required**): channels used for the background integral, e.g. `background.integral=c(85:100)`
`position`	vector (optional): reader positions that have to be analysed (e.g. `position=c(1:48)`. Empty positions are automatically omitted. If no value is given all positions are analysed by default.

`run` vector (optional): range of runs used for the analysis. If no value is given the range of the runs in the sequence is deduced from the Risoe.BINfileData object.

`set` vector (optional): range of sets used for the analysis. If no value is given the range of the sets in the sequence is deduced from the Risoe.BINfileData object.

`info.measurement` character (with default): option to provide information on the measurement on the plot output (e.g. name of the BIN file).

`log` character (with default): a character string which contains "x" if the x axis is to be logarithmic, "y" if the y axis is to be logarithmic and "xy" or "yx" if both axes are to be logarithmic. See plot.default).

`output.plot` logical (with default): plot output (`TRUE/FALSE`)

`cex.global` numeric (with default): global scaling factor.

Details

The functions works only for standard SAR protocol measurements introduced by Murray and Wintle (2000) with CW-OSL curves. For the calculation of the Lx/Tx value the function Calc_OSLLxTxRatio is used.

Provided rejection criteria

'recyling ratio': calculated for every repeated regeneration dose point.

'recuperation': recuperation rate calculated by comparing the Lx/Tx values of the zero regeneration point with the Ln/Tn value (the Lx/Tx ratio of the natural signal). For methodological background see Aitken and Smith (1988)

'IRSL/BOSL': the integrated counts (`signal.integral`) of a IRSL curve are compared with the integrated counts of the first regenerated dose point. It is assumed that IRSL curves got the same dose as the first regenerated dose point. **Note:** This is not the IR depletation ratio described by Duller (2003).

Value

A plot (optional) and list is returned containing the following elements:

`LnLxTnTx` data.frame of all calculated Lx/Tx values including signal, background counts and the dose points.

`RejectionCriteria` data.frame with values that might by used as rejection criteria. NA is produced if no R0 dose point exists.

`SARParameters` data.frame of additional measurement parameters obtained from the BIN file, e.g. preheat or reard temperature (not valid for all types of measurements).

Note

Rejection criteria are calculated but not considered during the analysis to discard values.

Author(s)

Sebastian Kreutzer (JLU Giessen, Germany),
Margret C. Fuchs (TU Bergakadmie Freiberg, Germany)

References

Aitken, M.J. & Smith, B.W., 1988. Optical dating: recuperation after bleaching. Quaternary Science Reviews, 7, pp. 387-393.

Duller, G., 2003. Distinguishing quartz and feldspar in single grain luminescence measurements. Radiation Measurements, 37 (2), pp. 161-165.

Murray, A.S. & Wintle, A.G., 2000. Luminescence dating of quartz using an improved single-aliquot regenerative-dose protocol. Radiation Measurements, 32, pp. 57-73.

See Also

Calc_OSLLxTxRatio, Risoe.BINfileData-class, readBIN2R and for further analysis plot_GrowthCurve

Examples

```
##load data
data(ExampleData.BINfileData)

##analyse data
output<-Analyse_SAR.OSLdata(input.data=BINfileData,
                    signal.integral=c(1:5),
                    background.integral=c(900:1000),
                    position=c(1:1),output.plot=TRUE
                    )

##combine results relvant for further analysis
output.SAR <- cbind(output$LnLxTnTx[[1]]$Dose, output$LnLxTnTx[[1]]$LxTx, output$LnLxTnTx[[1]]$LxTx.Error)
output.SAR
```

Calc_CentralDose *Apply the central age model (CAM) after Galbraith et al. (1999) to a given De distribution*

Description

This function calculates the central dose and dispersion of the De distribution, their standard errors and the profile log likelihood function for sigma.

Usage

```
Calc_CentralDose(input.data,
                 sigmab = 0,
                 sample.id = "unknown sample",
                 print.iterations = FALSE,
                 output.plot = TRUE)
```

Arguments

`input.data`	data.frame (**required**): two column data frame with De values and corresponding De errors
`sigmab`	numeric (with default): spread in De values given as a fraction (e.g. 0.2). This value represents the expected overdispersion in the data should the sample be well-bleached (Cunningham & Walling 2012, p. 100).
`sample.id`	character (with default): sample id
`print.iterations`	logical (with default): terminal output of calculation iterations
`output.plot`	logical (with default): plot output

Details

This function uses the equations of Galbraith et al. (1999, pp. 358-359). The parameter `sigma` is estimated using the maximum likelihood approach. A detailed explanation on maximum likelihood estimation can be found in the appendix of Galbraith & Laslett (1993, pp. 468-470)

Value

Returns a plot and terminal output. In addition a list is returned containing the following element:

`results`	data frame with statistical parameters.

Author(s)

Original S script: Rex Galbraith, University College London (UK), 2010

Revised R script: Christoph Burow, University of Cologne (Germany), 2012

References

Galbraith, R.F. & Laslett, G.M., 1993. Statistical models for mixed fission track ages. Nuclear Tracks Radiation Measurements, 4, pp. 459-470.

Galbraith, R.F., Roberts, R.G., Laslett, G.M., Yoshida, H. & Olley, J.M., 1999. Optical dating of single grains of quartz from Jinmium rock shelter, northern Australia. Part I: experimental design and statistical models. Archaeometry, 41, pp. 339-364.

Galbraith, R.F. & Roberts, R.G., 2012. Statistical aspects of equivalent dose and error calculation and display in OSL dating: An overview and some recommendations. Quaternary Geochronology, 11, pp. 1-27.

Further reading

Arnold, L.J. & Roberts, R.G., 2009. Stochastic modelling of multi-grain equivalent dose (De) distributions: Implications for OSL dating of sediment mixtures. Quaternary Geochronology, 4, pp. 204-230.

Bailey, R.M. & Arnold, L.J., 2006. Statistical modelling of single grain quartz De distributions and an assessment of procedures for estimating burial dose. Quaternary Science Reviews, 25, pp. 2475-2502.

Cunningham, A.C. & Wallinga, J., 2012. Realizing the potential of fluvial archives using robust OSL chronologies. Quaternary Geochronology, 12, pp. 98-106.

Rodnight, H., Duller, G.A.T., Wintle, A.G. & Tooth, S., 2006. Assessing the reproducibility and accuracy of optical dating of fluvial deposits. Quaternary Geochronology, 1, pp. 109-120.

Rodnight, H., 2008. How many equivalent dose values are needed to obtain a reproducible distribution?. Ancient TL, 26, pp. 3-10.

See Also

plot, Calc_CommonDose, Calc_FiniteMixture, Calc_FuchsLang2001, Calc_MinDose3, Calc_MinDose4

Examples

```
## load example data
data(ExampleData.DeValues)

## apply the central dose model
Calc_CentralDose(ExampleData.DeValues)
```

`Calc_CommonDose` *Apply the (un-)logged common age model after Galbraith et al. (1999) to a given De distribution*

Description

Function to calculate the common dose of a De distribution.

Usage

```
Calc_CommonDose(input.data,
                sigmab = 0,
                log = TRUE,
                sample.id = "unknown sample")
```

Arguments

`input.data`	data.frame (**required**): two column data frame with De values and corresponding De errors
`sigmab`	numeric (with default): spread in De values given as a fraction (e.g. 0.2). This value represents the expected overdispersion in the data should the sample be well-bleached (Cunningham & Walling 2012, p. 100).
`log`	logical (with default): fit the (un-)logged common age model to De data
`sample.id`	character (with default): sample id

Details

(Un-)logged model

When `log = TRUE` this function calculates the weighted mean of logarithmic De values. Each of the estimates is weighted by the inverse square of its relative standard error. The weighted mean is then transformed back to the dose scale (Galbraith & Roberts 2012, p. 14).

The log transformation is not applicable if the De estimates are close to zero or negative. In this case the un-logged model can be applied instead (`log = FALSE`). The weighted mean is then calculated using the un-logged estimates of De and their absolute standard error (Galbraith & Roberts 2012, p. 14).

Value

Returns a terminal output. In addition a list is returned containing the following element:

`results`	data frame with statistical parameters.

Author(s)

Original S script: Rex Galbraith, University College London (UK), 2010

Revised R script: Christoph Burow, University of Cologne (Germany), 2012

References

Galbraith, R.F. & Laslett, G.M., 1993. Statistical models for mixed fission track ages. Nuclear Tracks Radiation Measurements, 4, pp. 459-470.

Galbraith, R.F., Roberts, R.G., Laslett, G.M., Yoshida, H. & Olley, J.M., 1999. Optical dating of single grains of quartz from Jinmium rock shelter, northern Australia. Part I: experimental design and statistical models. Archaeometry, 41, pp. 339-364.

Galbraith, R.F. & Roberts, R.G., 2012. Statistical aspects of equivalent dose and error calculation and display in OSL dating: An overview and some recommendations. Quaternary Geochronology, 11, pp. 1-27.

Further reading

Arnold, L.J. & Roberts, R.G., 2009. Stochastic modelling of multi-grain equivalent dose (De) distributions: Implications for OSL dating of sediment mixtures. Quaternary Geochronology, 4, pp. 204-230.

Bailey, R.M. & Arnold, L.J., 2006. Statistical modelling of single grain quartz De distributions and an assessment of procedures for estimating burial dose. Quaternary Science Reviews, 25, pp. 2475-2502.

Cunningham, A.C. & Wallinga, J., 2012. Realizing the potential of fluvial archives using robust OSL chronologies. Quaternary Geochronology, 12, pp. 98-106.

Rodnight, H., Duller, G.A.T., Wintle, A.G. & Tooth, S., 2006. Assessing the reproducibility and accuracy of optical dating of fluvial deposits. Quaternary Geochronology, 1, pp. 109-120.

Rodnight, H., 2008. How many equivalent dose values are needed to obtain a reproducible distribution?. Ancient TL, 26, pp. 3-10.

See Also

Calc_CentralDose, Calc_FiniteMixture, Calc_FuchsLang2001, Calc_MinDose3, Calc_MinDose4

Examples

```
## load example data
data(ExampleData.DeValues)

## apply the common dose model
Calc_CommonDose(ExampleData.DeValues)
```

`Calc_FadingCorr` *Apply a fading correction according to Huntley & Lamothe (2001) for a given g-value.*

Description

This function runs the itrerations that are needed to calculate the corrected age including the error for a given g-value according to Huntley & Lamothe (2001).

Usage

```
Calc_FadingCorr(g_value, tc, age.faded, n.MCruns = 500)
```

Arguments

`g_value`	vector (**required**): g-value and error obtained from separate fading measurements (see example)
`tc`	numeric (**required**): time in seconds (time between irradiation and the prompt measurement, cf. Huntely & Lamothe 2001).
`age.faded`	numeric vector (**required**): uncorrected age with error in ka (see example)
`n.MCruns`	integer (with default): number of Monte Carlo simulation runs for error estimation

Details

The error of the fading-corrected age is determined using a Monte Carlo simulation approach. Hugh values for n.MCruns will significantly increase the computation time.

Value

A data.frame containing the fading-corrected age is returned.

Note

The upper age limit is set to 500 ka!

Author(s)

Sebastian Kreutzer, JLU Giessen, 2012

References

Huntley, D.J. & Lamothe, M., 2001. Ubiquity of anomalous fading in K-feldspars and the measurement and correction for it in optical dating. Canadian Journal of Earth Sciences, 38, pp. 1093-1106.

See Also

##

Examples

```
Calc_FadingCorr(g_value = c(3.3,0.03), tc = 752, age.faded = c(100,10))
```

`Calc_FiniteMixture` *Apply the finite mixture model (FMM) after Galbraith (2005) to a given De distribution*

Description

This function fits a k-component mixture to a De distribution with differing known standard errors. Parameters (doses and mixing proportions) are estimated by maximum likelihood assuming that the log dose estimates are from a mixture of normal distributions.

Usage

```
Calc_FiniteMixture(input.data,
                   sigmab,
                   n.components,
                   sample.id = "unknown sample",
                   n.iterations = 200,
                   grain.probability = FALSE,
                   output.file = FALSE,
                   output.filename = "default")
```

Arguments

`input.data`	data.frame (**required**): two column data frame with De values and corresponding De errors
`sigmab`	numeric (**required**): spread in De values given as a fraction (e.g. 0.2). This value represents the expected overdispersion in the data should the sample be well-bleached (Cunningham & Walling 2012, p. 100).
`n.components`	numeric (**required**): number of components to be fitted
`sample.id`	character (with default): sample id
`n.iterations`	numeric (with default): number of iterations for maximum likelihood estimates
`grain.probability`	logical (with default): prints the estimated probabilities of which component each grain is in
`output.file`	logical (with default): save results to file. See `output.filename`.
`output.filename`	character (with default): desired filename, else results are saved to default.res

Details

This model uses the maximum likelihood and Bayesian Information Criterion (BIC) approaches.

Indications of overfitting are:

- increasing BIC
- repeated dose estimates
- covariance matrix not positive definite
- convergence problems

Value

Returns a terminal output and a file containing statistical results if wanted. In addition a list is returned containing the following elements:

`mle.matrix`	covariance matrix of maximum likelihood estimates.
`grain.probability`	matrix with estimated probabilities of which component each grain is in.
`meta`	data frame containing model parameters (sample.id, sigmab, n.components, llik, bic).
`components`	data frame containing fitted components.
`single.comp`	data frame containing log likelihood and BIC for a single component.

Author(s)

Original S script: Rex Galbraith, University College London (UK), 2006

Revised R script: Christoph Burow, University of Cologne (Germany), 2012

References

Galbraith, R.F. & Green, P.F., 1990. Estimating the component ages in a finite mixture. Nuclear Tracks and Radiation Measurements, 17, pp. 197-206.

Galbraith, R.F. & Laslett, G.M., 1993. Statistical models for mixed fission track ages. Nuclear Tracks Radiation Measurements, 4, pp. 459-470.

Galbraith, R.F. & Roberts, R.G., 2012. Statistical aspects of equivalent dose and error calculation and display in OSL dating: An overview and some recommendations. Quaternary Geochronology, 11, pp. 1-27.

Roberts, R.G., Galbraith, R.F., Yoshida, H., Laslett, G.M. & Olley, J.M., 2000. Distinguishing dose populations in sediment mixtures: a test of single-grain optical dating procedures using mixtures of laboratory-dosed quartz. Radiation Measurements, 32, pp. 459-465.

Galbraith, R.F., 2005. Statistics for Fission Track Analysis, Chapman & Hall/CRC, Boca Raton.

Further reading

Arnold, L.J. & Roberts, R.G., 2009. Stochastic modelling of multi-grain equivalent dose (De) distributions: Implications for OSL dating of sediment mixtures. Quaternary Geochronology, 4, pp. 204-230.

Cunningham, A.C. & Wallinga, J., 2012. Realizing the potential of fluvial archives using robust OSL chronologies. Quaternary Geochronology, 12, pp. 98-106.

Rodnight, H., Duller, G.A.T., Wintle, A.G. & Tooth, S., 2006. Assessing the reproducibility and accuracy of optical dating of fluvial deposits. Quaternary Geochronology, 1, pp. 109-120.

Rodnight, H. 2008. How many equivalent dose values are needed to obtain a reproducible distribution?. Ancient TL, 26, pp. 3-10.

See Also

Calc_CentralDose, Calc_CommonDose, Calc_FuchsLang2001, Calc_MinDose3, Calc_MinDose4

Examples

```
## load example data
data(ExampleData.DeValues)

## apply the finite mixture model
Calc_FiniteMixture(ExampleData.DeValues,
                   sigmab = 0.08, n.components = 2,
                   grain.probability = TRUE, output.file = FALSE)
```

`Calc_FuchsLang2001` *Apply the model after Fuchs & Lang (2001) to a given De distribution.*

Description

This function applies the method according to Fuchs & Lang (2001) for heterogeneously bleached samples with a given coefficient of variation threshold.

Usage

```
Calc_FuchsLang2001(sample,
                   sample.mtext = "unkown sample",
                   sample.id = sample.mtext,
                   cvThreshold = 5,
                   startDeValue = 1,
                   output.plot = TRUE, output.terminal = TRUE,
                   main = "Fuchs & Lang (2001)",
                   xlab = expression(paste(D[e], " [Gy]")),
                   cex.global = 1)
```

Arguments

`sample`	data.frame (**required**): two column data frame, e.g. De and De error
`sample.mtext`	character (optional): mtext for optional plot (top)
`sample.id`	character (with default): sample id, with default the sample.mtext is used.
`cvThreshold`	numeric (with default): coefficient of variation in percent, as threshold for the method, e.g. `cvThreshold = 3`. See details.
`startDeValue`	numeric (with default): number of the first aliquot that is used for the calculations
`output.plot`	logical (with default): plot output `TRUE/FALSE`
`output.terminal`	logical (with default): terminal output `TRUE/FALSE`
`main`	character (with default): title of the plot (works as in plot)
`xlab`	character (with default): xlab works as in plot
`cex.global`	numeric (with default): global scaling factor

Details

Used values

If the coefficient of variation (c[v]) of the first two values is larger than the threshold c[v_treshold], the frist value is skipped. Use the `startDeValue` parameter to define a start value for calculation (e.g. 2nd or 3rd value).

Basic steps of the approach

(1) Estimate natural relative variation of the sample using a dose recovery test

(2) Order the input values ascendingly

(3) Calculate a running mean, starting with the lowermost two values and add values iteratively.

(4) Stop if the calculated c[v] exceeds the specified `cvThreshold`

Value

A plot and terminal output is provided if desired. In addition, a list is returned containing two elements:

`results`	data.frame with stastical parameters, e.g. mean, sd,...
`usedDeValues`	data.frame containing the used values for the calculation

Note

Please consider the requirments and the constraints of this method (see Fuchs & Lang, 2001)

Author(s)

Sebastian Kreutzer, JLU Giessen, 2012

References

Fuchs, M. & Lang, A., 2001. OSL dating of coarse-grain fluvial quartz using single-aliqout protocols on sediments from NE Peloponnese, Greece. In: Quaternary Science Reviews (20), pp. 783-787.

Fuchs, M. & Wagner, G.A., 2003. Recognition of insufficient bleaching by small aliquots of quartz for reconstructing soil erosion in Greece. Quaternary Science Reviews, 22, pp. 1161-1167.

See Also

`plot`

Examples

```
##load example data
data(ExampleData.DeValues)

##calculate De according to Fuchs & Lang (2001)
Calc_FuchsLang2001(ExampleData.DeValues, cvThreshold = 5)
```

Calc_MinDose3 *Apply the (un-)logged three parameter minimum age model (MAM 3) after Galbraith et al. (1999) to a given De distribution*

Description

Function to fit the (un-)logged three parameter minimum dose model (MAM 3) to De data.

Usage

```
Calc_MinDose3(input.data,
              sigmab,
              log = TRUE,
              sample.id = "unknown sample",
              gamma.xlb = 0.1,
              gamma.xub = 100,
              sigma.xlb = 0.001,
              sigma.xub = 5.00,
              init.gamma = 10,
              init.sigma = 1.2,
              init.p0 = 0.01,
              calc.ProfileLikelihoods = TRUE,
              console.ProfileLikelihoods=FALSE,
              console.extendedOutput=FALSE,
              output.file = FALSE,
              output.filename = "default",
              output.plot = FALSE,
              output.indices = 3)
```

Arguments

input.data	data.frame (**required**): two column data frame with De values and corresponding De errors
sigmab	numeric (**required**): spread in De values given as a fraction (e.g. 0.2). This value represents the expected overdispersion in the data should the sample be well-bleached (Cunningham & Walling 2012, p. 100).
log	logical (with default): fit the (un-)logged three parameter minimum dose model to De data
sample.id	character (with default): sample id
gamma.xlb	numeric (with default): lower boundary of gamma
gamma.xub	numeric (with default): upper boundary of gamma
sigma.xlb	numeric (with default): lower boundary of sigma
sigma.xub	numeric (with default): upper boundary of sigma
init.gamma	numeric (with default): starting value of gamma
init.sigma	numeric (with default): starting value of sigma
init.p0	numeric (with default): starting value of p0

`calc.ProfileLikelihoods`
: logical (with default): calculate profile log likelihood functions for gamma, sigma, p0. See `output.indices`.

`console.ProfileLikelihoods`
: logical (with default): print profile log likelihood functions for gamma, sigma, p0 to console.

`console.extendedOutput`
: logical (with default): extended terminal output

`output.file`
: logical (with default): save results to file. See `output.filename`.

`output.filename`
: character (with default): desired filename, else results are saved to default-3R(-UL).res

`output.plot`
: logical (with default): plot output (`TRUE`/`FALSE`)

`output.indices`
: numeric (with default): requires `calc.ProfileLikelihoods = TRUE`. Indices: 1 = gamma, 2 = gamma/sigma, 3 = gamma/sigma/p0.

Details

Parameters

This model has three parameters:

`gamma`:	minimum dose on the log scale
`sigma`:	spread in ages above the minimum
`p0`:	proportion of grains at gamma

(Un-)logged model

In the original version of the three-parameter minimum dose model, the basic data are the natural logarithms of the De estimates and relative standard errors of the De estimates. This model will be applied if `log = TRUE`.

If `log = FALSE`, the modified un-logged model will be applied instead. This has essentially the same form as the original version. `gamma` and `sigma` are in Gy and `gamma` becomes the minimum true dose in the population.

While the original (logged) version of the mimimum dose model may be appropriate for most samples (i.e. De distributions), the modified (un-logged) version is specially designed for modern-age and young samples containing negative, zero or near-zero De estimates (Arnold et al. 2009, p. 323).

Boundaries

Depending on the data, the upper and lower bounds for gamma (`gamma.xlb` and `gamma.xub`) need to be specified. If the final estimate of gamma is on the boundary, `gamma.xlb` and `gamma.xub` need to be adjusted appropriately, so that gamma lies within the bounds. The same applies for sigma boundaries (`sigma.xlb` and `sigma.xub`).

Initial values

The log likelihood calculations use the nlminb function. Accordingly, initial values for the three parameters `init.gamma`, `init.sigma` and `init.p0` need to be specified.

Value

A terminal output is provided. A plot (postscript) and a file containing statistical results are provided if desired. In addition a list is returned containing the following element:

`results` data frame containing statistical results.

Note

The default boundary and starting values for *gamma*, *sigma* and *p0* may only be appropriate for some De data sets and may need to be changed for other data. This is especially true when the un-logged version is applied.

Author(s)

Original S script: Rex Galbraith, University College London (UK), 2010

Revised R script: Christoph Burow, University of Cologne (Germany), 2012

References

Arnold, L.J., Roberts, R.G., Galbraith, R.F. & DeLong, S.B., 2009. A revised burial dose estimation procedure for optical dating of young and modern-age sediments. Quaternary Geochronology, 4, pp. 306-325.

Galbraith, R.F. & Laslett, G.M., 1993. Statistical models for mixed fission track ages. Nuclear Tracks Radiation Measurements, 4, pp. 459-470.

Galbraith, R.F., Roberts, R.G., Laslett, G.M., Yoshida, H. & Olley, J.M., 1999. Optical dating of single grains of quartz from Jinmium rock shelter, northern Australia. Part I: experimental design and statistical models. Archaeometry, 41, pp. 339-364.

Galbraith, R.F., 2005. Statistics for Fission Track Analysis, Chapman & Hall/CRC, Boca Raton.

Galbraith, R.F. & Roberts, R.G., 2012. Statistical aspects of equivalent dose and error calculation and display in OSL dating: An overview and some recommendations. Quaternary Geochronology, 11, pp. 1-27.

Further reading

Arnold, L.J. & Roberts, R.G., 2009. Stochastic modelling of multi-grain equivalent dose (De) distributions: Implications for OSL dating of sediment mixtures. Quaternary Geochronology, 4, pp. 204-230.

Bailey, R.M. & Arnold, L.J., 2006. Statistical modelling of single grain quartz De distributions and an assessment of procedures for estimating burial dose. Quaternary Science Reviews, 25, pp. 2475-2502.

Cunningham, A.C. & Wallinga, J., 2012. Realizing the potential of fluvial archives using robust OSL chronologies. Quaternary Geochronology, 12, pp. 98-106.

Rodnight, H., Duller, G.A.T., Wintle, A.G. & Tooth, S., 2006. Assessing the reproducibility and accuracy of optical dating of fluvial deposits. Quaternary Geochronology, 1, pp. 109-120.

Rodnight, H., 2008. How many equivalent dose values are needed to obtain a reproducible distribution?. Ancient TL, 26, pp. 3-10.

See Also

nlminb, postscript, Calc_CentralDose, Calc_CommonDose, Calc_FiniteMixture, Calc_FuchsLang2001, Calc_MinDose4

Examples

```
## load example data
data(ExampleData.DeValues)

## apply the logged minimum dose model
Calc_MinDose3(ExampleData.DeValues,
              sigmab = 0.3,gamma.xub = 7000,
              output.file = FALSE, output.plot = FALSE)

## apply the un-logged minimum dose model
## note that the example data set does not meet the un-logged model requirements
Calc_MinDose3(ExampleData.DeValues, log = FALSE,
              sigmab = 0.3,gamma.xub = 5000,
              output.file = FALSE, output.plot = FALSE)
```

`Calc_MinDose4` *Apply the (un-)logged four parameter minimum age model (MAM 4) after Galbraith et al. (1999) to a given De distribution*

Description

Function to fit the (un-)logged four parameter minimum dose model (MAM 4) to De data.

Usage

```
Calc_MinDose4(input.data,
              sigmab,
              log = TRUE,
              sample.id = "unknown sample",
              gamma.xlb = 0.1,
              gamma.xub = 100,
              mu.xlb = 1,
              mu.xub = 100,
              sigma.xlb = 0.001,
              sigma.xub = 5.00,
              init.gamma = 5,
              init.mu = 10,
              init.sigma = 0.6,
              init.p0 = 0.01,
              calc.ProfileLikelihoods = TRUE,
              console.ProfileLikelihoods = FALSE,
              console.extendedOutput = FALSE,
              output.file = FALSE,
              output.filename = "default",
              output.plot = FALSE,
              output.indices = 4)
```

Arguments

`input.data`	data.frame (**required**): two column data frame with De values and corresponding De errors

`sigmab`	numeric (**required**): spread in De values given as a fraction (e.g. 0.2). This value represents the expected overdispersion in the data should the sample be well-bleached (Cunningham & Walling 2012, p. 100).
`log`	logical (with default): fit the (un-)logged three parameter minimum dose model to De data
`sample.id`	character (with default): sample id
`gamma.xlb`	numeric (with default): lower boundary of gamma
`gamma.xub`	numeric (with default): upper boundary of gamma
`mu.xlb`	numeric (with default): lower boundary of mu
`mu.xub`	numeric (with default): upper boundary of mu
`sigma.xlb`	numeric (with default): lower boundary of sigma
`sigma.xub`	numeric (with default): upper boundary of sigma
`init.gamma`	numeric (with default): starting value of gamma
`init.mu`	numeric (with default): starting value of mu
`init.sigma`	numeric (with default): starting value of sigma
`init.p0`	numeric (with default): starting value of p0
`calc.ProfileLikelihoods`	logical (with default): calculate profile log likelihood functions for gamma, mu, sigma, p0. See `output.indices`.
`console.ProfileLikelihoods`	logical (with default): print profile log likelihood functions for gamma, sigma, p0 to console.
`console.extendedOutput`	logical (with default): extended terminal output
`output.file`	logical (with default): save results to `output.filename`-4R(-UL).res file. See `output.filename`.
`output.filename`	character (with default): desired filename, else results are saved to default-4R(-UL).res
`output.plot`	logical (with default): plot output
`output.indices`	numeric (with default): requires `calc.ProfileLikelihoods=TRUE`. Indices: 1 = gamma, 2 = gamma/mu, 3 = gamma/mu/sigma, 4 = gamma/mu/sigma/p0

Details

Parameters

This model has four parameters:

`gamma`:	minimum dose on the log scale
`mu`:	mean of the non-truncated normal distribution
`sigma`:	spread in ages above the minimum
`p0`:	proportion of grains at gamma

(Un-)logged model

In the original version of the three-parameter minimum dose model, the basic data are the natural

logarithms of the De estimates and relative standard errors of the De estimates. This model will be applied if `log = TRUE`.

If `log = FALSE`, the modified un-logged model will be applied instead. This has essentially the same form as the original version. `gamma` and `sigma` are in Gy and `gamma` becomes the minimum true dose in the population.

While the original (logged) version of the mimimum dose model may be appropriate for most samples (i.e. De distributions), the modified (un-logged) version is specially designed for modern-age and young samples containing negative, zero or near-zero De estimates (Arnold et al. 2009, p. 323).

Boundaries

Depending on the data, the upper and lower bounds for gamma (`gamma.xlb` and `gamma.xub`) and mu (`mu.xlb` and `mu.xub`) need to be specified. If the final estimate of gamma or mu is on the boundary, `gamma.xlb` and `gamma.xub` (`mu.xlb` and `mu.xub` respectively) need to be adjusted appropriately, so that gamma and mu lie within the bounds. The same applies for sigma boundaries (`sigma.xlb` and `sigma.xub`)

Initial values

The log likelihood calculations use the nlminb function. Accordingly, initial values for the four parameters `init.gamma`, `init.sigma`, `init.mu` and `init.p0` need to be specified.

Value

A terminal output is provided. A plot (postscript) and a file containing statistical results are provided if wanted. In addition a list is returned containing the following element:

`results`	data frame containing statistical results.

Note

The default boundary and starting values for *gamma*, *mu*, *sigma* and *p0* may only be appropriate for some De data sets and may need to be changed for other data. This is especially true when the un-logged version is applied.

Author(s)

Original S script: Rex Galbraith, University College London (UK), 2010

Revised R script: Christoph Burow, University of Cologne (Germany), 2012

References

Arnold, L.J., Roberts, R.G., Galbraith, R.F. & DeLong, S.B., 2009. A revised burial dose estimation procedure for optical dating of young and modern-age sediments. Quaternary Geochronology, 4, pp. 306-325.

Galbraith, R.F. & Laslett, G.M., 1993. Statistical models for mixed fission track ages. Nuclear Tracks Radiation Measurements, 4, pp. 459-470.

Galbraith, R.F., Roberts, R.G., Laslett, G.M., Yoshida, H. & Olley, J.M., 1999. Optical dating of single grains of quartz from Jinmium rock shelter, northern Australia. Part I: experimental design and statistical models. Archaeometry, 41, pp. 339-364.

Galbraith, R.F., 2005. Statistics for Fission Track Analysis, Chapman & Hall/CRC, Boca Raton.

Galbraith, R.F. & Roberts, R.G., 2012. Statistical aspects of equivalent dose and error calculation and display in OSL dating: An overview and some recommendations. Quaternary Geochronology, 11, pp. 1-27.

Further reading

Arnold, L.J. & Roberts, R.G., 2009. Stochastic modelling of multi-grain equivalent dose (De) distributions: Implications for OSL dating of sediment mixtures. Quaternary Geochronology, 4, pp. 204-230.

Bailey, R.M. & Arnold, L.J., 2006. Statistical modelling of single grain quartz De distributions and an assessment of procedures for estimating burial dose. Quaternary Science Reviews, 25, pp. 2475-2502.

Cunningham, A.C. & Wallinga, J., 2012. Realizing the potential of fluvial archives using robust OSL chronologies. Quaternary Geochronology, 12, pp. 98-106.

Rodnight, H., Duller, G.A.T., Wintle, A.G. & Tooth, S., 2006. Assessing the reproducibility and accuracy of optical dating of fluvial deposits. Quaternary Geochronology, 1, pp. 109-120.

Rodnight, H., 2008. How many equivalent dose values are needed to obtain a reproducible distribution?. Ancient TL, 26, pp. 3-10.

See Also

nlminb, postscript, Calc_CentralDose, Calc_CommonDose, Calc_FiniteMixture, Calc_FuchsLang2001, Calc_MinDose3

Examples

```
## load example data
data(ExampleData.DeValues)

## apply the logged minimum dose model
Calc_MinDose4(ExampleData.DeValues,
              sigmab = 0.05, gamma.xub = 10000, mu.xub = 10000, init.p0 = 0.4,
              output.file = FALSE,output.plot = FALSE)
```

Calc_OSLLxTxRatio *Calculate Lx/Tx ratio for CW-OSL curves.*

Description

Calculate Lx/Tx ratios from two given OSL curves.

Usage

```
Calc_OSLLxTxRatio(Lx.data,Tx.data,
                  signal.integral,
                  background.integral)
```

Arguments

Lx.data	data.frame (**required**): requires a CW-OSL shine down curve (x = time, y= counts)
Tx.data	data.frame (optional): requires a CW-OSL shine down curve (x = time, y = counts). If no input is given the Tx.data will be treated as NA and no Lx/Tx ratio is calculated.

`signal.integral`
: vector (**required**): requires a vector with the limits for the signal integral.

`background.integral`
: vector (**required**): requires a vector with the bounds for the background integral.

Details

The integrity of the chosen values for the signal and background integral is checked by the function; the signal integral limits have to be lower than the background integral limits. If a vector is given as input instead of a data.frame, an artificial `data.frame` is produced. The error calculation is done according to Galbraith (2002).

Value

Returns a data.frame with the following structure:

$ LnLx
$ LnLx.BG
$ TnTx
$ TnTx.BG
$ Net_LnLx
$ Net_LnLx.Error
$ Net_TnTx.Error

Note

The results of this function have been cross-checked with the Analyst (vers. 3.24b).

Author(s)

Sebastian Kreutzer, JLU Giessen, Germany

References

Duller, G., 2007. Analyst.

Galbraith, R.F., 2002. A note on the variance of a background-corrected OSL count. Ancient TL, 20 (2), 49-51.

See Also

`Analyse_SAR.OSLdata`, `plot_GrowthCurve`

Examples

```
##load data
data(ExampleData.LxTxOSLData)

##calculate Lx/Tx ratio
results<-Calc_OSLLxTxRatio(Lx.data,Tx.data, signal.integral = c(1:2), background.integral = c(85:100))
print(results)
```

CW2pHMi *Transform a CW-OSL curve into a pHM-OSL curve via interpolation under hyperbolic modulation conditions*

Description

This function transforms a conventionally measured continuous-wave (CW) OSL-curve to a pseudo hyperbolic modulated (pHM) curve under hyperbolic modulation conditions using the interpolation procedure described by Bos & Wallinga (2012).

Usage

```
CW2pHMi(values, delta)
```

Arguments

values	data.frame (**required**): measured curve data of type stimulation time (t) (values[,1]) and measured counts (cts) (values[,2])
delta	vector (optional): stimulation rate parameter, if no value is given, the optimal value is estimated automatically (see details). Smaller values of delta produce more points in the rising tail of the curve.

Details

The complete procedure of the transformation is described in Bos & Wallinga (2012). The input data.frame consists of two columns: time (t) and count values (CW(t))

Internal transformation steps

(1) log(CW-OSL) values

(2) Calculate t' which is the transformed time:

$$t' = t - (1/delta) * log(1 + delta * t)$$

(3) Interpolate CW(t'), i.e. use the log(CW(t)) to obtain the count values for the transformed time (t'). Values beyond min(t) and max(t) produce NA values.

(4) Select all values for t' < min(t), i.e. values beyond the time resolution of t. Select the first two values of the transformed data set which contain no NA values and use these values for a linear fit using lm.

(5) Extrapolate values for t' < min(t) based on the previously obtained fit parameters.

(6) Transform values using

$$pHM(t) = (delta * t/(1 + delta * t)) * c * CW(t')$$
$$c = (1 + delta * P)/delta * P$$
$$P = length(stimulation\ period)$$

(7) Combine all values and truncate all values for t' > max(t)

The number of values for t' < min(t) depends on the stimulation rate parameter delta. *To avoid the production of too many artificial data at the raising tail of the determined pHM curve, it is recommended to use the automatic estimation routine for* delta, *i.e. provide no value for* delta.

Value

Returns a data.frame with three columns:

$x	: time
$y.t	: transformed count values
$x.t	: transformed time values
$method	: used method for the production of the new data points

Note

According to Bos & Wallinga (2012), the number of extrapolated points should be limited to avoid artificial intensity data. If `delta` is provided manually and more than two points are extrapolated, a warning message is returned.

The function approx may produce some `Inf` and `NaN` data. The function tries to manually interpolate these values by calculating the `mean` using the adjacent channels. If two invalid values are succeeding, the values are removed and no further interpolation is attempted.

Author(s)

Sebastian Kreutzer, JLU Giessen, Germany, 2012

Based on comments and suggestions from:
Adrie J.J. Bos, Delft University of Technology, The Netherlands

References

Bos, A.J.J. & Wallinga, J., 2012. How to visualize quartz OSL signal components. Radiation Measurements 47, 752-758.

Further

Bulur, E., 1996. An Alternative Technique For Optically Stimulated Luminescence (OSL) Experiment. Radiation Measurements, 26, 701-709.

Bulur, E., 2000. A simple transformation for converting CW-OSL curves to LM-OSL curves. Radiation Measurements, 32, 141-145.

See Also

CW2pLM, CW2pLMi, CW2pPMi, fit_LMCurve, lm

Examples

```
##(1) - simple transformation

##load CW-OSL curve data
data(ExampleData.CW_OSL_Curve)

##transform values
values.transformed<-CW2pHMi(ExampleData.CW_OSL_Curve)

##plot
plot(values.transformed$x, values.transformed$y.t, log = "x")

##(2) - load CW-OSL curve from BIN-file and plot transformed values
```

```
##load BINfile
#BINfileData<-readBIN2R("[path to BIN-file]")
data(ExampleData.BINfileData)

##grep first CW-OSL curve from ALQ 1

  curve.ID<-BINfileData@METADATA[BINfileData@METADATA[,"LTYPE"]=="OSL" &
                                 BINfileData@METADATA[,"POSITION"]==1
                                 ,"ID"]

  curve.HIGH<-BINfileData@METADATA[BINfileData@METADATA[,"ID"]==curve.ID[1]
                                   ,"HIGH"]

  curve.NPOINTS<-BINfileData@METADATA[BINfileData@METADATA[,"ID"]==curve.ID[1]
                                     ,"NPOINTS"]

##combine curve to data set

  curve<-data.frame(x = seq(curve.HIGH/curve.NPOINTS,curve.HIGH,
                            by = curve.HIGH/curve.NPOINTS),
                    y=unlist(BINfileData@DATA[curve.ID[1]])
                    )

##transform values

  curve.transformed<-CW2pHMi(curve)

##plot curve
  plot(curve.transformed$x, curve.transformed$y.t, log="x")


##(3) - produce Fig. 4 from Bos & Wallinga (2012)

##load data
data(ExampleData.CW_OSL_Curve)
values<-CW_Curve.BosWallinga2012

##open plot area
plot(NA, NA,
     xlim=c(0.001,10),
     ylim=c(0,8000),
     ylab="pseudo OSL (cts/0.01 s)",
     xlab="t [s]",
     log="x",
     main="Fig. 4 - Bos & Wallinga (2012)",
     )

values.t<-CW2pLMi(values, P=1/20)
lines(values[1:length(values.t[,1]),1],CW2pLMi(values, P=1/20)[,2], col="red" ,lwd=1.3)
text(0.03,4500,"LM", col="red" ,cex=.8)

values.t<-CW2pHMi(values, delta=40)
lines(values[1:length(values.t[,1]),1],CW2pHMi(values, delta=40)[,2], col="black", lwd=1.3)
text(0.005,3000,"HM", cex=.8)

values.t<-CW2pPMi(values, P=1/10)
lines(values[1:length(values.t[,1]),1],CW2pPMi(values, P=1/10)[,2], col="blue", lwd=1.3)
```

```
text(0.5,6500,"PM", col="blue" ,cex=.8)
```

CW2pLM *Transform a CW-OSL curve into a pLM-OSL curve*

Description

Transforms a conventionally measured continuous-wave (CW) curve into a pseudo linearly modulated (pLM) curve using the equations given in Bulur (2000).

Usage

```
CW2pLM(values)
```

Arguments

values	data.frame (**required**): measured curve data of type stimulation time (t) (values[,1]) and measured counts (cts) (values[,2])

Details

According to Bulur (2000), the curve data are transformed by introducing two new parameters P (stimulation period) and u (transformed time):

$$P = 2 * max(t)$$

$$u = \sqrt{(2 * t * P)}$$

The new count values are then calculated by

$$ctsNEW = cts(u/P)$$

And the returned data.frame is produced by: data.frame(u,ctsNEW)

Value

Returns a data.frame of the same form as the input of the transformed curve data.

Note

The transformation is recommended for curves recorded with a channel resolution of at least 0.05 s/channel.

Author(s)

Sebastian Kreutzer, JLU Giessen, Germany

References

Bulur, E., 2000. A simple transformation for converting CW-OSL curves to LM-OSL curves. Radiation Measurements, 32, 141-145.

See Also

CW2pLM

The output of the function can be further used for LM-OSL fitting:

CW2pLMi, CW2pHMi, CW2pPMi, fit_LMCurve

Examples

```
##read curve from BINfileData transform curve an plot values
data(ExampleData.BINfileData)

##read id for the 1st OSL curve
id.OSL<-BINfileData@METADATA[BINfileData@METADATA[,"LTYPE"]=="OSL","ID"]

##produce x and y (time and count data for the data set)
x<-seq(BINfileData@METADATA[id.OSL[1],"HIGH"]/BINfileData@METADATA[id.OSL[1],"NPOINTS"],
       BINfileData@METADATA[id.OSL[1],"HIGH"],
       by=BINfileData@METADATA[id.OSL[1],"HIGH"]/BINfileData@METADATA[id.OSL[1],"NPOINTS"])
y<-unlist(BINfileData@DATA[id.OSL[1]])
values<-data.frame(x,y)

##transform values
values.transformed<-CW2pLM(values)

##plot
plot(values.transformed)
```

CW2pLMi *Transform a CW-OSL curve into a pLM-OSL curve via interpolation under linear modulation conditions*

Description

Transforms a conventionally measured continuous-wave (CW) OSL-curve into a pseudo linearly modulated (pLM) curve under linear modulation conditions using the interpolation procedure described by Bos & Wallinga (2012).

Usage

```
CW2pLMi(values, P)
```

Arguments

values	data.frame (**required**): measured curve data of type stimulation time (t) (values[,1]) and measured counts (cts) (values[,2])
P	vector (optional): stimulation time in seconds, if no value is given the optimal value is estimated automatically (see details). Greater values of P produce more points in the rising tail of the curve.

Details

The complete procedure of the transformation is given in Bos & Wallinga (2012). The input `data.frame` consists of two columns: time (t) and count values (CW(t))

Nomenclature

P = stimulation time (s)
1/P = stimulation rate (1/s)

Internal transformation steps

(1) log(CW-OSL) values

(2) Calculate t' which is the transformed time:

$$t' = 1/2 * 1/P * t^2$$

(3) Interpolate CW(t'), i.e. use the log(CW(t)) to obtain the count values for the transformed time (t'). Values beyond `min(t)` and `max(t)` produce `NA` values.

(4) Select all values for t' < `min(t)`, i.e. values beyond the time resolution of t. Select the first two values of the transformed data set which contain no `NA` values and use these values for a linear fit using `lm`.

(5) Extrapolate values for t' < `min(t)` based on the previously obtained fit parameters.

(6) Transform values using

$$pLM(t) = t/P * CW(t')$$

(7) Combine values and truncate all values for t' > `max(t)`

The number of values for t' < `min(t)` depends on the stimulation period (P) and therewith on the stimulation rate 1/P. To avoid the production of too many artificial data at the raising tail of the determined pLM curves it is recommended to use the automatic estimation routine for `P`, i.e. provide no own value for `P`.

Value

Returns a `data.frame` with three columns:

$x	: time
$y.t	: transformed count values
$x.t	: transformed time values
$method	: used method for the production of the new data points

Note

According to Bos & Wallinga (2012), the number of extrapolated points should be limited to avoid artificial intensity data. If `P` is provided manually and more than two points are extrapolated, a warning message is returned.

Author(s)

Sebastian Kreutzer, JLU Giessen, Germany, 2012

Based on comments and suggestions from:
Adrie J.J. Bos, Delft University of Technology, The Netherlands

References

Bos, A.J.J. & Wallinga, J., 2012. How to visualize quartz OSL signal components. Radiation Measurements, 47, 752-758.

Further read

Bulur, E., 1996. An Alternative Technique For Optically Stimulated Luminescence (OSL) Experiment. Radiation Measurements 26, 701-709.

Bulur, E., 2000. A simple transformation for converting CW-OSL curves to LM-OSL curves. Radiation Measurements, 32, 141-145.

See Also

CW2pLM, CW2pHMi, CW2pPMi, fit_LMCurve

Examples

```
##(1)
##load CW-OSL curve data
data(ExampleData.CW_OSL_Curve)

##transform values
values.transformed<-CW2pLMi(ExampleData.CW_OSL_Curve)

##plot
plot(values.transformed$x, values.transformed$y.t, log = "x")

##(2) - produce Fig. 4 from Bos & Wallinga (2012)
##load data
data(ExampleData.CW_OSL_Curve)
values<-CW_Curve.BosWallinga2012

##open plot area
plot(NA, NA,
     xlim=c(0.001,10),
     ylim=c(0,8000),
     ylab="pseudo OSL (cts/0.01 s)",
     xlab="t [s]",
     log="x",
     main="Fig. 4 - Bos & Wallinga (2012)",
     )

values.t<-CW2pLMi(values, P=1/20)
lines(values[1:length(values.t[,1]),1],CW2pLMi(values, P=1/20)[,2], col="red" ,lwd=1.3)
text(0.03,4500,"LM", col="red" ,cex=.8)

values.t<-CW2pHMi(values, delta=40)
lines(values[1:length(values.t[,1]),1],CW2pHMi(values, delta=40)[,2], col="black", lwd=1.3)
text(0.005,3000,"HM", cex=.8)

values.t<-CW2pPMi(values, P=1/10)
lines(values[1:length(values.t[,1]),1],CW2pPMi(values, P=1/10)[,2], col="blue", lwd=1.3)
text(0.5,6500,"PM", col="blue" ,cex=.8)
```

CW2pPMi *Transform a CW-OSL curve into a pPM-OSL curve via interpolation under parabolic modulation conditions*

Description

Transforms a conventionally measured continuous-wave (CW) OSL-curve into a pseudo parabolic modulated (pPM) curve under parabolic modulation conditions using the interpolation procedure described by Bos & Wallinga (2012).

Usage

```
CW2pPMi(values, P)
```

Arguments

values	data.frame (**required**): measured curve data of type stimulation time (t) (values[,1]) and measured counts (cts) (values[,2])
P	vector (optional): stimulation period in seconds, if no value is given, the optimal value is estimated automatically (see details). Greater values of P produce more points in the rising tail of the curve.

Details

The complete procedure of the transformation is given in Bos & Wallinga (2012). The input data.frame consists of two columns: time (t) and count values (CW(t))

Nomenclature

P = stimulation time (s)
1/P = stimulation rate (1/s)

Internal transformation steps

(1) log(CW-OSL) values

(2) Calculate t' which is the transformed time:

$$t' = (1/3) * (1/P^2)t^3$$

(3) Interpolate CW(t'), i.e. use the log(CW(t)) to obtain the count values for the transformed time (t'). Values beyond min(t) and max(t) produce NA values.

(4) Select all values for t' < min(t), i.e. values beyond the time resolution of t. Select the first two values of the transformed data set which contain no NA values and use these values for a linear fit using lm.

(5) Extrapolate values for t' < min(t) based on the previously obtained fit parameters. The extrapolation is limited to two values. Other values at the beginning of the transformed curve are set to 0.

(6) Transform values using

$$pLM(t) = t^2/P^2 * CW(t')$$

(7) Combine all values and truncate all values for t' > max(t)

The number of values for t' < `min(t)` *depends on the stimulation period* `P`*. To avoid the production of too many artificial data at the raising tail of the determined pPM curve, it is recommended to use the automatic estimation routine for* `P`*, i.e. provide no value for* `P`*.*

Value

Returns a `data.frame` with three columns:

$x	: time
$y.t	: transformed count values
$x.t	: transformed time values
$method	: used method for the production of the new data points

Note

According to Bos & Wallinga (2012), the number of extrapolated points should be limited to avoid artificial intensity data. If `P` is provided manually not more than two points are extrapolated.

Author(s)

Sebastian Kreutzer, JLU Giessen, Germany, 2012

Based on comments and suggestions from:
Adrie J.J. Bos, Delft University of Technology, The Netherlands

References

Bos, A.J.J. & Wallinga, J., 2012. How to visualize quartz OSL signal components. Radiation Measurements, 47, 752-758.

Further

Bulur, E., 1996. An Alternative Technique For Optically Stimulated Luminescence (OSL) Experiment. Radiation Measurements, 26, 701-709.

Bulur, E., 2000. A simple transformation for converting CW-OSL curves to LM-OSL curves. Radiation Measurements, 32, 141-145.

See Also

`CW2pLM`, `CW2pLMi`, `CW2pHMi`, `fit_LMCurve`

Examples

```
##(1)
##load CW-OSL curve data
data(ExampleData.CW_OSL_Curve)

##transform values
values.transformed<-CW2pPMi(ExampleData.CW_OSL_Curve)

##plot
plot(values.transformed$x,values.transformed$y.t, log = "x")

##(2) - produce Fig. 4 from Bos & Wallinga (2012)

##load data
data(ExampleData.CW_OSL_Curve)
```

```
values<-CW_Curve.BosWallinga2012

##open plot area
plot(NA, NA,
     xlim=c(0.001,10),
     ylim=c(0,8000),
     ylab="pseudo OSL (cts/0.01 s)",
     xlab="t [s]",
     log="x",
     main="Fig. 4 - Bos & Wallinga (2012)",
     )

values.t<-CW2pLMi(values, P=1/20)
lines(values[1:length(values.t[,1]),1],CW2pLMi(values, P=1/20)[,2], col="red" ,lwd=1.3)
text(0.03,4500,"LM", col="red" ,cex=.8)

values.t<-CW2pHMi(values, delta=40)
lines(values[1:length(values.t[,1]),1],CW2pHMi(values, delta=40)[,2], col="black", lwd=1.3)
text(0.005,3000,"HM", cex=.8)

values.t<-CW2pPMi(values, P=1/10)
lines(values[1:length(values.t[,1]),1],CW2pPMi(values, P=1/10)[,2], col="blue", lwd=1.3)
text(0.5,6500,"PM", col="blue" ,cex=.8)
```

`ExampleData.BINfileData` *Example data from a SAR OSL measurement for package Luminescence*

Description

Example data from a SAR OSL measurement for package Luminescence diretly extracted from a Risoe BIN file and provided in an object of type Risoe.BINfileData-class

Usage

```
ExampleData.BINfileData
```

Format

Class object containing two slots: (a) `METADATA` is a data.frame with all metadata stored in the BIN file of the measurements and (b) `DATA` contains a list of vectors of the measured data (usually count values).

Source

Lab:	Luminescence Laboratory Bayreuth
Lab-Code:	BT607
Location:	Saxony/Germany
Material:	Middle grain quartz measured on aluminum cups on a Risoe DA-15 reader

References

unpublished data

Examples

```
##show first 5 elements of the METADATA and DATA elements in the terminal
data(ExampleData.BINfileData)
BINfileData@METADATA[1:5,]
BINfileData@DATA[1:5]
```

ExampleData.CW_OSL_Curve

Example CW-OSL curve data for package Luminescence

Description

`data.frame` containing CW-OSL curve data (time, counts)

Usage

```
data(ExampleData.CW_OSL_Curve)
```

Format

Data frame with 1000 observations on the following 2 variables.

`x` a numeric vector, time

`y` a numeric vector, counts

Details

see source

Source

ExampleData.CW_OSL_Curve

Lab:	Luminescence Laboratory Bayreuth
Lab-Code:	BT607
Location:	Saxony/Germany
Material:	Middle grain quartz measured on aluminum cups on a Risoe DA-15 reader.
Reference:	unpublished data

CW_Curve.BosWallinga2012

Lab:	Netherlands Centre for Luminescence Dating (NCL)
Lab-Code:	NCL-2108077
Location:	Guadalentin Basin, Spain
Material:	Coarse grain quartz.

Reference: Bos & Wallinga (2012) and Baartman et al. (2011)

References

Baartman, J.E.M., Veldkamp, A., Schoorl, J.M., Wallinga, J., Cammeraat, L.H., 2011. Unravelling Late Pleistocene and Holocene landscape dynamics: The Upper Guadalentin Basin, SE Spain. Geomorphology, 125, pp. 172-185.

Bos, A.J.J. & Wallinga, J., 2012. How to visualize quartz OSL signal components. Radiation Measurements, 47, 752-758.

Examples

```
data(ExampleData.CW_OSL_Curve)
plot(ExampleData.CW_OSL_Curve)
```

ExampleData.DeValues *Example De data for package Luminescence*

Description

25 equivalent dose (De) values measured for a fine grain quartz sample from a loess section in Rottewitz (Saxony/Germany). Internal lab code: BT998.

Usage

```
ExampleData.DeValues
```

Format

A data.frame with two columns (De and De_Error).

Source

Lab:	Luminescence Laboratory Bayreuth
Lab-Code:	BT998
Location:	Rottewitz (Saxony/Germany)
Material:	Fine grain quartz measured on aluminum discs on a Risoe DA-15 reader
Units:	Values are given in seconds
Dose Rate:	Dose rate of the beta-source ca. 0.0438 +/- 0.0019 Gy/s

References

unpublished data

Examples

```
##plot histogram
data(ExampleData.DeValues)
hist(ExampleData.DeValues[,1])
```

ExampleData.FittingLM *Example data for fit_LMCurve() in package Luminescence*

Description

Lineraly modulated (LM) measurement data from a quartz sample from Norway including background measurement. Measurements carried out at the luminescence lab at the University of Bayreuth.

Usage

```
ExampleData.FittingLM
```

Format

Two objects (data.frames) with two columns (time and counts).

Source

Lab:	Luminescence Laboratory Bayreuth
Lab-Code:	BT900
Location:	Norway
Material:	Beach deposit, coarse grain quartz measured on aluminum discs on a Risoe DA-15 reader

References

Fuchs, M., Kreutzer, S., Fischer, M., Sauer, D., Soerensen, R., 2012. OSL and IRSL dating of raised beach sand deposits along the southeastern coast of Norway. Quaternary Geochronology, 10, pp. 195-200.

Examples

```
##show LM data
data(ExampleData.FittingLM)
plot(values.curve,log="x")
```

ExampleData.LxTxData *Example Lx/Tx data from CW-OSL SAR measurement*

Description

LxTx data from a SAR measurement

Usage

```
ExampleData.LxTxData
```

Format

A `data.frame` with 4 columns (Dose, LxTx, LxTx.Error, TnTx).

Source

Lab:	Luminescence Laboratory Bayreuth
Lab-Code:	BT607
Location:	Ostrau (Saxony-Anhalt/Germany)
Material:	middle grain quartz measured on a Risoe DA-15 reader.

References

unpublished data

Examples

```
##plot Lx/Tx data vs dose [s]
data(ExampleData.LxTxData)
plot(LxTxData$Dose,LxTxData$LxTx)
```

ExampleData.LxTxOSLData

Example Lx and Tx curve data from an artificial OSL measurement

---

Description

Lx and Tx data of continous wave (CW-) OSL signal curves.

Usage

```
ExampleData.LxTxOSLData.RData
```

Format

Two `data.frames` containing time and count values.

Source

Artificial OSL measurement.

References

unpublished data

Examples

```
##load data
data(ExampleData.LxTxOSLData)

##plot data
plot(Lx.data)
plot(Tx.data)
```

`fit_CWCurve` *Nonlinear Least Squares Fit for CW-OSL curves*

Description

The function determines the weighted least-squares estimates of the component parameters of a CW-OSL for a given maximum number of components and returns various component parameters. The fitting procedure uses the `nls` function with the `port` algorithm.

Usage

```
fit_CWCurve(values,
            n.components.max,
            fit.failure_threshold = 3,
            fit.trace = FALSE,
            fit.calcError = FALSE,
            LED.power = 36,
            LED.wavelength = 470,
            log = "",
            cex.global = 0.6,
            main = "CW-OSL Curve Fit",
            sample_code = "Default",
            ylab, xlab,
            output.path,
            output.terminal = TRUE,
            output.terminalAdvanced = TRUE,
            output.plot = TRUE)
```

Arguments

`values`	data.frame (**required**): x,y data of measured values (time and counts). See examples.
`n.components.max`	vector (optional): maximum number of components that are to be used for fitting. The upper limit is 7.
`fit.failure_threshold`	vector (with default): limits the failed fitting attempts.
`fit.trace`	logical (with default): trace the fitting process on the terminal.
`fit.calcError`	logical (with default): calculate 1-sigma error range of components using confint
`LED.power`	numeric (with default): LED power (max.) used for intensity ramping in mW/cm^2. **Note:** The value is used for the calculation of the absolute photoionisation cross section.

`LED.wavelength` numeric (with default): LED wavelength used for stimulation in nm. **Note:** The value is used for the calculation of the absolute photoionisation cross section.

`log` character (optional): option for log-scaled axis, works as in plot

`cex.global` numeric (with default): global scaling factor.

`main` character (with default): header for plot output.

`sample_code` character (optional): sample code used for the plot and the optional output table (mtext).

`ylab` character (with default): alternative y-axis labelling

`xlab` character (with default): alternative x-axis labelling

`output.path` character (optional): output path for table output containing the results of the fit. The file name is set automatically. If the file already exists in the directory, the values are appended.

`output.terminal` logical (with default): terminal ouput with fitting results.

`output.terminalAdvanced` logical (with default): enhanced terminal output. Requires `output.terminal = TRUE`. If output.terminal = FALSE no advanced output is possible.

`output.plot` logical (with default): returns a plot of the fitted curves.

Details

Fitting function

The function for the fitting has the general form :

$$y = I0_1 * exp(-lambda_1 * x) + , \ldots , + I0_i * exp(-lambda_i * x)$$

where $1 >= i <= 7$

Start values

Start values are estimated automatically by fitting a linear function on the log input data set. Currently, there is no option to manually provide start parameters.

goodness of fit

The goodness of the fit is given by a expression(pseudo-R^2) value (pseudo coefficient of determination). According to Lave (1970), the value is calculated as:

$$pseudoR^2 = 1 - RSS/TSS$$

where $RSS = Residual\ Sum\ of\ Squares$
and $TSS = Total\ Sum\ of\ Squares$

Error of fitted component parameters

The 1-sigma error for the components is calculated using the function confint. Due to considerable calculation time, this option is deactived by default. In addition, the error for the components can be estimated by using internal R functions like summary. See the nls help page for more information.

For details on the nonlinear regression in R, see Ritz & Streibig (2008).

Value

`plot`	(optional) the fitted CW-OSL curves are returned as plot.
`table`	(optional) an output table (*.csv) with the fitted components is provided if the `output.path` is set.
`list object`	beside the plot and table output options a list is returned. The list contains: (a) a `nls` object (`$fit`) for which generic R functions are provided, e.g. summary, confint, profile. For more details, see nls. (b) a data.frame containing the summarized parameters including the error (`$output.table`).

Note

The pseudo-R^2 may not be the best parameter to describe the goodness of the fit. The trade off between the `n.components` and the pseudo-R^2 value has not yet been considered.

The function **does not** ensure that the fitting procedure has reached a global minimum rather than a local minimum! In any case of doubt, the use of manual start values is highly recommended.

Author(s)

Sebastian Kreutzer, JLU Giessen, Germany, 2012

References

Lave, C.A.T., 1970. The Demand for Urban Mass Transportation. The Review of Economics and Statistics, 52 (3), pp. 320-323.

Ritz, C. & Streibig, J.C., 2008. Nonlinear Regression with R R. Gentleman, K. Hornik, & G. Parmigiani, eds., Springer.

See Also

`fit_LMCurve`, `plot`,`nls`

Examples

```
##load data
data(ExampleData.CW_OSL_Curve)

##fit data
fit <- fit_CWCurve(values = ExampleData.CW_OSL_Curve,
                   main = "CW Curve Fit",
                   n.components.max = 4,
                   log = "x")
```

`fit_LMCurve` *Nonlinear Least Squares Fit for LM-OSL Curves*

Description

The function determines weighted nonlinear least-squares estimates of the component parameters of a LM-OSL curve (Bulur 1996) for a given number of components and returns various component parameters. The fitting procedure uses the function `nls` with the `port` algorithm.

Usage

```
fit_LMCurve(values,
            values.bg,
            n.components = 3,
            start_values,
            input.dataType = "LM",
            main = "Default",
            sample_code = "",
            sample_ID = "",
            LED.power = 36, LED.wavelength = 470,
            log_scale = "", cex.global = 0.8,
            fit.trace = FALSE,
            fit.advanced = FALSE,
            fit.calcError = FALSE,
            bg.subtraction = "polynomial",
            output.path,
            output.terminal = TRUE, output.terminaladvanced = TRUE,
            output.plot = TRUE, output.plotBG = FALSE
            )
```

Arguments

`values`	data.frame (**required**): x,y data of measured values (time and counts). See examples.
`values.bg`	data.frame (optional): x,y data of measured values (time and counts) for background subtraction
`n.components`	integer (with default): fixed number of components that are to be recognised during fitting (min = 1, max = 7).
`start_values`	data.frame (optional): start parameters for lm and xm data for the fit. If no start values are given, an automatic start value estimation is attempted (see details).
`input.dataType`	character (with default): alter the plot output depending on the input data: "LM" or "pLM" (pseudo-LM). See: CW2pLM
`main`	character (with default): header for plot output.
`sample_code`	character (optional): sample code used for the plot and the optional output table (mtext).
`sample_ID`	character (optional): additional identifier used as column for the table output.
`LED.power`	numeric (with default): LED power (max.) used for intensity ramping in mW/cm^2. **Note:** The value is used for the calculation of the absolute photoionisation cross section.

`LED.wavelength`	numeric (with default): LED wavelength used for stimulation in nm. **Note:** The value is used for the calculation of the absolute photoionisation cross section.
`log_scale`	character: option for logarithm scales on the plot. Works as in plot.default.
`cex.global`	numeric (with default): global scaling factor.
`fit.trace`	logical (with default): trace the fitting process on the terminal.
`fit.advanced`	logical (with default): enables advanced fitting attempt for automatic start parameter recognition. Works only if no start parameters are provided. **Note:** It may take a while.
`fit.calcError`	logical (with default): calculate 1-sigma error range of components using confint
`bg.subtraction`	character (with default): specify method for background subtraction (`polynomial`, `linear`, `channel`, see Details). **Note:** input for `values.bg` is required.
`output.path`	character (optional): output path for table output containing the results of the fit. The file name is set automatically. If the file already exists in the directory, the values are appended.
`output.terminal`	logical (with default): terminal output with fitting results.
`output.terminaladvanced`	logical (with default): enhanced terminal output. Requires `output.terminal = TRUE`. If output.terminal = FALSE no advanced output is possible.
`output.plot`	logical (with default): returns a plot of the fitted curves.
`output.plotBG`	logical (with default): returns a plot of the background values with the fit used for the background subtraction.

Details

Fitting function

The function for the fitting has the general form :

$$y = exp(0.5)*Im_1*x/xm_1)*exp(-x^2/(2*xm_1^2))+,\ldots,+exp(0.5)*Im_i*x/xm_i)*exp(-x^2/(2*xm_i^2))$$

where $1 >= i <= 7$

This function and the equations for the conversion to b (detrapping probability) and n0 (proportional to initially trapped charge) have been taken from Kitis et al. (2008):

$$xm_i = \sqrt{max(t)/b_i}$$

$$Im_i = exp(-0.5)n0/xm_i$$

Background subtraction

Three methods for background subtraction are provided for a given background signal (`values.bg`).

`polynomial`: default method. A polynomial function is fitted using glm and the resulting function is used for the background subtraction:

$$y = a * x^4 + b * x^3 + c * x^2 + d * x + e$$

`linear`: a linear function is fitted using glm and the resulting function is used for the background subtraction:

$$y = a * x + b$$

`channel`: the measured background signal is subtracted channelwise from the measured signal.

Start values

The choice of the initial parameters for the nls-fitting is a crucial point and the fitting procedure may mainly fail due to ill chosen start parameters. Here, three options are provided:

(a) If no start values (`start_values`) are provided by the user, a cheap guess is made by using the detrapping values found by Jain et al. (2003) for quartz for a maximum of 7 components. Based on these values, the pseudo start parameters xm and Im are recalculated for the given data set. In all cases, the fitting starts with the ultra-fast component and (depending on `n.components`) steps through the following values. If no fit could be achieved, an error plot (for `output.plot = TRUE`) with the pseudo curve (based on the pseudo start parameters) is provided. This may give the opportunity to identify appropriate start parameters visually.

(b) If start values are provided, the function works like a simple `nls` fitting approach.

(c) If no start parameters are provided and the option `fit.advanced == TRUE` is chosen, an advanced start paramter estimation is applied using a stochastical attempt. Therefore the recalculated start parameters **(a)** are used to construct a normal distribution. The start parameters are then sampled randomly from this distribution. A maximum of 100 attempts will be made. **Note:** This process may be really time consuming.

Goodness of fit

The goodness of the fit is given by a pseudo-R^2 value (pseudo coefficient of determination). According to Lave (1970), the value is calculated as:

$$pseudoR^2 = 1 - RSS/TSS$$

where $RSS = Residual\ Sum\ of\ Squares$
and $TSS = Total\ Sum\ of\ Squares$

Error of fitted component parameters

The 1-sigma error for the components is calculated using the function confint. Due to considerable calculation time, this option is deactived by default. In addition, the error for the components can be estimated by using internal R functions like summary. See the nls help page for more information.

For more details on the nonlinear regression in R, see Ritz & Streibig (2008).

Value

`plot`	(optional) various types of plots are returned for details see above.
`table`	(optional) an output table (*.csv) with the fitted components is provided if the `output.path` is set.
`list object`	beside the plot and table output options a list is returned. The list contains: (a) a `nls` object (`$fit`) for which generic R functions are prov, e.g. summary, confint, profile. For more details, see nls. (b) a data.frame containing the summarised parameters including the error (`$output.table`).

Note

The pseudo-R^2 may not be the best parameter to describe the goodness of the fit. The trade off between the `n.components` and the pseudo-R^2 value has not yet been considered.

The function **does not** ensure that the fitting procedure has reached a global minimum rather than a local minimum! In any case of doubt, the use of manual start values is highly recommended.

Author(s)

Sebastian Kreutzer, JLU Giessen, Germany, 2011

References

Bulur, E., 1996. An Alternative Technique For Optically Stimulated Luminescence (OSL) Experiment. Radiation Measurements, 26 (5), pp. 701-709.

Jain, M., Murray, A.S., Boetter-Jensen, L., 2003. Characterisation of blue-light stimulated luminescence components in different quartz samples: implications for dose measurement. Radiation Measurements, 37 (4-5), pp. 441-449.

Kitis, G. & Pagonis, V., 2008. Computerized curve deconvolution analysis for LM-OSL. Radiation Measurements, 43, pp. 737-741.

Lave, C.A.T., 1970. The Demand for Urban Mass Transportation. The Review of Economics and Statistics, 52 (3), pp. 320-323.

Ritz, C. & Streibig, J.C., 2008. Nonlinear Regression with R R. Gentleman, K. Hornik, & G. Parmigiani, eds., Springer.

See Also

`fit_CWCurve`, `plot`,`nls`

Examples

```
##(1) fit LM data without background subtraction
data(ExampleData.FittingLM)
fit_LMCurve(values = values.curve, n.components = 3, log_scale = "x")

##(2) fit LM data with background subtraction and export as JPEG
## -alter file path for your prefered system
##jpeg(file = "~/Desktop/Fit_Output%03d.jpg", quality = 100,
    ## height = 3000, width = 3000, res = 300)
data(ExampleData.FittingLM)
fit_LMCurve(values = values.curve, values.bg = values.curveBG,
     n.components = 2, log_scale = "x", output.plotBG = TRUE)
##dev.off()

##(3) fit LM data with manual start parameters
data(ExampleData.FittingLM)
fit_LMCurve(values = values.curve, values.bg = values.curveBG, n.components = 3, log_scale = "x",
            start_values = data.frame(Im = c(170,25,400), xm = c(56,200,1500)))
```

plot_BINfileData *Plot single luminescence curves from a BIN file object.*

Description

Plot single luminescence curves from an object returned by the readBIN2R function.

Usage

```
plot_BINfileData(BINfileData, position, run, set,
                 sorter = "POSITION",
                 ltype = c("IRSL","OSL","TL","RIR","RBR"),
                 dose_rate,
                 temp.lab = "deg. C",
                 cex.global = 1)
```

Arguments

BINfileData	Risoe.BINfileData-class (**required**): requires an S4 object returned by the readBIN2R function.
position	vector (optional): option to limit the plotted curves by position (e.g. position = 1, position = c(1,3,5)).
run	vector (optional): option to limit the plotted curves by run (e.g. run = 1, set = c(1,3,5)).
set	vector (optional): option to limit the plotted curves by set (e.g. run = 1, set = c(1,3,5)).
sorter	character (with default): the plot output can be ordered by "POSITION","SET" or "RUN". POSITION, SET and RUN are options defined in the Risoe Sequence Editor.
ltype	character (with default): option to limit the plotted curves by the type of luminescence stimulation. Allowed values: "IRSL","OSL","TL","RIR" and "RBR". All type of curves are plotted by default.
dose_rate	numeric (optional): dose rate of the irradition source at the measurement date. If set, the given irradiation dose will be shown in Gy. See details.
temp.lab	character (optional): option to plot a circle (for degree) before the C (Celsius) instead of 'deg.'. This argument is just a technical issue (s. details).
cex.global	numeric (with default): global scaling factor.

Details

Nomenclature

The nomenclature used for the function (e.g. ltype, position) are taken from the Analyst manual (Duller, 2007, p. 42):

[,1]	ID	: Unique record ID (same ID as in slot DATA)	numeric
[,2]	SEL	: Record selection	logical
[,3]	VERSION	: Data format version number	raw

[,4]	LENGTH	: Length of this record	integer
[,5]	PREVIOUS	: Length of previous record	integer
[,6]	NPOINTS	: Number of data points in the record	integer
[,7]	LTYPE	: Luminescence type	factor
[,8]	LOW	: Low (temperature, time, wavelength)	numeric
[,10]	HIGH	: High (temperature, time, wavelength)	numeric
[,11]	RATE	: Rate (heating rate, scan rate)	numeric
[,12]	TEMPERATURE	: Sample temperature	integer
[,13]	XCOORD	: X position of a single grain	integer
[,14]	YCOORD	: Y position of a single grain	integer
[,15]	TOLDELAY	: TOL 'delay' channels	integer
[,16]	TOLON	: TOL 'on' channels	integer
[,17]	TOLOFF	: TOL 'off' channels	integer
[,18]	POSITION	: Carousel position	integer
[,19]	RUN	: Run number	integer
[,20]	TIME	: Data collection time (hh-mm-ss)	factor
[,21]	DATA	: Data collection date (dd-mm-yy)	factor
[,22]	SEQUENCE	: Sequence name	factor
[,23]	USER	: User name	factor
[,24]	DTYPE	: Data type	factor
[,25]	IRR_TIME	: Irradiation time	numeric
[,26]	IRR_TYPE	: Irradiation type (alpha, beta or gamma)	integer
[,27]	IRR_UNIT	: Irradiation unit (Gy, Rads, secs, mins, hrs)	integer
[,28]	BL_TIME	: Bleaching time	numeric
[,29]	BL_UNIT	: Bleaching unit (mJ, J, secs, mins, hrs)	integer
[,30]	AN_TEMP	: Annealing temperature	numeric
[,31]	AN_TIME	: Annealing time	numeric
[,32]	NORM1	: Normalisation factor (1)	numeric
[,33]	NORM2	: Normalisation factor (2)	numeric
[,34]	NORM3	: Normalisation factor (3)	numeric
[,35]	BG	: Background level	numeric
[,36]	SHIFT	: Number of channels to shift data	integer
[,37]	SAMPLE	: Sample name	factor
[,38]	COMMENT	: Comment	factor
[,39]	LIGHTSOURCE	: Light source	factor
[,40]	SET	: Set Number	integer
[,41]	TAG	: Tag	integer
[,42]	GRAIN	: Grain number	integer
[,43]	LPOWER	: Optical Stimulation Power	numeric
[,44]	SYSTEMID	: System ID	integer

Irradiation time

Plotting the irradiation time (s) or the given dose (Gy) requires that the variable `IRR_TIME` has been set within the BIN-file. This is normally done by using the 'Run Info' option within the Sequence Editor or by editing in R.

Circle instead of 'deg.'

Unfortunately, the circle (for degree) is not an ASCII character and therefore a warning is produced during compiling the R-package. To avoid the warning message the default scale unit is 'deg. C'.

Value

Returns a plot.

Note

The function has been successfully tested for the Sequence Editor file output version 3 and 4.

Author(s)

Sebastian Kreutzer, JLU Giessen (Germany), 2012

References

Duller, G., 2007. Analyst. pp. 1-45.

See Also

readBIN2R

Examples

```
##load data
data(ExampleData.BINfileData)

##plot all curves from the first position to the desktop
#pdf(file = "~/Desktop/CurveOutput.pdf", paper = "a4", height = 11, onefile = TRUE)

  ##example - load from *.bin file
  #BINfile<-"[your path]"
  #BINfileData<-readBIN2R(BINfile)

#par(mfrow = c(4,3), oma = c(0.5,1,0.5,1))
#plot_BINfileData(BINfileData,position = 1)
#mtext(side = 4, BINfile, outer = TRUE, col = "blue", cex = .7)
#dev.off()
```

---

plot_GrowthCurve *Fit and plot a growth curve for luminescence data (Lx/Tx against dose)*

Description

A dose response curve is produced for luminescence measurements using a regenerative protocol.

Usage

```
plot_GrowthCurve(sample, main = "Growth Curve", mtext = "",
                 fit.method = "EXP",
                 fit.weights = TRUE,
                 fit.includingRepeatedRegPoints = TRUE,
                 fit.NumberRegPoints,
                 fit.NumberRegPointsReal,
                 fit.bounds = TRUE,
                 NumberIterations.MC = 100, xlab = "s",
```

```
output.plot = TRUE, output.plotExtended = TRUE,
cex.global = 1)
```

Arguments

`sample`	data.frame (**required**): data frame with three columns for x=Dose,y=LxTx,z=LxTx.Error, y1=TnTx. The column for the test dose response is optional, but requires 'TnTx' as column name if used.
`main`	character (with default): header of the plot.
`mtext`	character (optional): additional text on the right side of the plot.
`fit.method`	character (with default): functions used for fitting. Possible options are: `LIN`, `EXP`, `EXP OR LIN`,`EXP+LIN` or `EXP+EXP`. See details.
`fit.weights`	logical (with default): option whether the fitting is done with or without weights. See details.
`fit.includingRepeatedRegPoints`	logical (with default): includes repeated points for fitting (TRUE/FALSE)
`fit.NumberRegPoints`	integer (optional): set number of regeneration points manually. By default the number of all(!) regeneration points is grepped automatically.
`fit.NumberRegPointsReal`	integer (optional): if the number of regeneration points is provided manually the value of the real regenerations points = all points - repeated points - reg 0 has to be inserted.
`fit.bounds`	logical (with default): set lower fit bounds for fitting parameter to 0. Limited for the use with the fit methods `EXP`, `EXP+LIN` and `EXP OR LIN`. Argument to be inserted for experimental application only!
`NumberIterations.MC`	integer (with default): number of Monte Carlo simulation for the error estimation. See details.
`xlab`	character (with default): unit for x-axis labelling. Possible values are "Gy" and "s".
`output.plot`	logical (with default): plot output (TRUE/FALSE).
`output.plotExtended`	logical (with default): If `TRUE` 3-plots on one plot area are provided: (1) growth curve, (2) histogram from error Monte Carlo simulation and (3) a test dose response plot. If `FALSE`, just the growth curve will be plotted. **Requires:** `output.plot = TRUE`
`cex.global`	numeric (with default): global scale factor.

Details

Fitting methods

For all options (except for the `LIN` and the `EXP OR LIN`), the nls function with the `port` algorithm is used.

`LIN`: fit a linear function to the data using lm:

$$y = mx + n$$

EXP: try to fit a function of the form

$$y = a * (1 - exp(-(x + c)/b))$$

Parameters b and c are approximated by a linear fit using lm.

EXP OR LIN: works for some cases where an EXP fit failes. If the EXP fit failes, a LIN fit is done instead.

EXP+LIN: try to fit an exponential plus linear function of the form:

$$y = a * (1 - exp(-(x + c)/b) + (g * x))$$

The De is calculated by iteration.

Note: In the context of luminescence dating, this function has no physical meaning. Therefore, no D0 value is returned.

EXP+EXP: try to fit a double exponential function of the form

$$y = (a1 * (1 - exp(-(x)/b1))) + (a2 * (1 - exp(-(x)/b2)))$$

This fitting procedure is not robust against wrong start parameters and should be further improved.

Fit weighting (suggested by Michael Dietze and Margret Fuchs)

If the option fit.weights = TRUE is chosen weights are calculated using provided signal errors (Lx/Tx error) and the equation:

$$fit.weights = 1/error/(sum(1/error))$$

Error estimation using Monte Carlo simulation

Error estimation is done using a Monte Carlo (MC) simulation approach. A set of values is constructed by randomly drawing curve data from a normal distribution. The normal distribution is defined by the input values (mean = value, sd = value.error). Then, a growth curve fit is attempted for each dataset which results in new distribution of values. The sd of this distribution is the error of the De. With increasing iterations, the error value is becoming more stable. **Note:** It may take some calculation time with increasing MC runs, especially for the composed functions (EXP+LIN and EXP+EXP).

Each error estimation is done with the function of the chosen fitting method.

Value

list	a list containing the De (De, De Error, D01 value, D02 value and Fit type) and Fit object nls object for EXP, EXP+LIN and EXP+EXP. In case of a linear fit EXP OR LIN, a lm object is returned.
Plot	A plot of the growth curve. A histogram for the error calculation and a test dose response plot.

Note

No D01 is returned for the fit functions EXP+LIN and LIN. A D02 value is provided for the function EXP+EXP only.

Author(s)

Sebastian Kreutzer, JLU Giessen (Germany), 2012

References

Duller, G.A.T., 2007. Assessing the error on equivalent dose estimates derived from single aliquot regenerative dose measurements. Ancient TL, 25, pp. 15-24.

Galbraith, R.F. & Roberts, R.G., 2012. Statistical aspects of equivalent dose and error calculation and display in OSL dating: An overview and some recommendations. Quaternary Geochronology, 11, pp.1-27.

See Also

hist, plot, nls, lm

Examples

```
##(1) plot growth curve for a dummy data.set
data(ExampleData.LxTxData)
plot_GrowthCurve(LxTxData)


##(2) plot the growth curve only

##pdf(file = "~/Desktop/Growth_Curve_Dummy.pdf", paper = "special")
data(ExampleData.LxTxData)
plot_GrowthCurve(LxTxData)
##dev.off()

##(3) plot growth curve with pdf output on desktop (path works for Mac)

##pdf(file = "~/Desktop/Growth_Curve_Dummy.pdf", paper = "special")
data(ExampleData.LxTxData)
plot_GrowthCurve(LxTxData)
##dev.off()
```

---

plot_Histogram *Plot a histogram with a separate error plot*

Description

Plot a predefined histogram with an accompanying error plot as suggested by Rex Galbraith at the UK LED in Oxford 2010.

Usage

```
plot_Histogram(values,
               main = "Histogram",
               mtext = "",
               xlab = expression(paste(D[e]," [Gy]")),
               cex.global = 1,
               breaks = "Sturges", normal_curve = FALSE)
```

Arguments

values	data.frame (**required**): De (values[,1]) and De error (values[,2])
main	character (with default): title of the plot
mtext	character (optional): further sample information (mtext)
xlab	character (with default): label x-axis (xlab, see also plot)
cex.global	numeric (with default): global scaling factor
breaks	(with default): set breakpoints for histogram. Works as in hist.
normal_curve	logical (with default): add a normal curve to the histogram. Mean and sd are calculated from the input data. More see details section.

Details

If the normal curve is added, the y-axis in the histogram will show the probability density.

Value

Returns a plot.

Note

The input data is not restricted to a special type of input data.

Author(s)

Sebastian Kreutzer, JLU Giessen, Germany

References

Galbraith, R., 2010. Statistics in OSL: Some Current Questions; Ask Rex. Oral presentation during the UK TL/OSL/ESR Meeting at the School of Geography and the Environment, University of Oxford, 8-10 September 2010.

Galbraith, R.F. & Roberts, R.G., 2012. Statistical aspects of equivalent dose and error calculation and display in OSL dating: An overview and some recommendations. Quaternary Geochronology, 11, pp.1-27.

See Also

hist, plot

Examples

```
##load data
data(ExampleData.DeValues)

##plot histogram
plot_Histogram(ExampleData.DeValues, xlab = expression(paste(D[e],” [s]”)))
```

`plot_KDE` *Plot a kernel density estimate (KDE), sorted values with associated errors and statistical measures*

Description

Plot a kernel density estimate of measurement values in combination with the actual values and associated error bars in ascending order. Optionally, statistical measures such as mean, median, standard deviation, standard error and quartile range can be provided visually and as numeric values.

Usage

```
plot_KDE(values,
         distribution.parameters = ”“,
         summary = ”“,
         bw = ”nrd0”,
         ...)
```

Arguments

`values`	data.frame (**required**): two columns: De (`values[,1]`) and De error (`values[,2]`)
`distribution.parameters`	character (optional): plot additional distribution parameters. Can be one or more out of "mean", "median", "KDEmax" (maximum value of probability density function), "sd" (standard deviation) and "qr" (quartile range). Example: `distribution.parameter=c(”median”,”mean”,”sd”)`. Mean and median are potted as a line, the standard deviation is shown as a gray polygon.
`summary`	character (optinal): add numerical output to the plot. Can be one or more out of: "n" (number of samples), "mean" (mean De value), "median" (median of the De values), "KDEmax" (maximum value of probability density function), "sdrel" (relative standard deviation), "sdabs" (absolute standard deviation), "serel" (relative standard error) and "seabs" (absolute standard deviation).
`bw`	character (with default): bin-width, choose a numeric value for manual setting.
`...`	further arguments and graphical parameters passed to `plot`

Details

The function allows, to pass several plot arguments, such as main, xlab, cex. However, since the figure is an overlay of two separate plots, ylim must be specified in the order: c(ymin_axis1, ymax_axis1, ymin_axis2, ymax_axis2). Similarly, if other than the default colours are desired, the argument col must be provided with colours in the following order: probability density function, De values, De error bars, sd or qr polygon. See examples for some further explanations.For details on the calculation of the bin-width (parameter `bw`) see `density`.

Value

Returns a plot.

Note

The plot output is no 'PD' plot (cf. the discussion of Berger and Galbraith in Ancient TL; see references)!

Author(s)

Sebastian Kreutzer (JLU Giessen, Germany),
Michael Dietze (TU Dresden, Germany)

References

Berger, G.W., 2010. An alternate form of probability-distribution plot for De values. Ancient TL 28, pp. 11-21.

Berger, G.W., 2011. Response to Galbraith. Ancient TL 29, pp. 48-50.

Galbraith, R.F., 2011. Some comments arising from Berger (2010). Ancient TL 29, pp. 41-47.

Galbraith, R.F. & Roberts, R.G., 2012. Statistical aspects of equivalent dose and error calculation and display in OSL dating: An overview and some recommendations. Quaternary Geochronology, 11, pp.1-27.

See Also

density, plot

Examples

```
# read example data set
data(ExampleData.DeValues)

# native function call
plot_KDE(ExampleData.DeValues)

# function call with some user-defined modifications
plot_KDE(ExampleData.DeValues,
                    main = "Plot of Dose distribution data",
                    col = c("red", "black", "grey", "cyan"),
                    xlab = "Equivalent dose [Gy]",
                    ylabs = c("KDE estimate", "cumulative De values"),
                    distribution.parameters = c("qr", "median"),
                    xlim = c(2000, 5000),
                    ylims = c(0, 0.005, -5, 50),
                    summary = c("n", "median", "serel", "seabs"),
                    cex = 0.8)

# function call with complete numerical statistical description output
plot_KDE(ExampleData.DeValues,
                    distribution.parameters = c("qr", "kdemax"),
                    summary = c("n", "mean", "median", "kdemax", "serel", "sdrel", "sdabs", "seabs"))
```

`plot_RadialPlot` *Produce a Galbraith's radial plot.*

Description

A Galbraith radial plot is produced on a logarithmic or a linear De scale. The function is based on a rewritten S script of Rex Galbraith.

Usage

```
plot_RadialPlot(sample,
                sample.groups,
                sample.legend,
                sample.lty = 1,
                sample.pch = 1,
                sample.col = "black",
                sample.mtext = "default",
                zscale.log = TRUE,
                zaxis.scale, zaxis.group_circle = FALSE, yaxis.scale,
                plot.2sigmaRange = TRUE,
                plot.area_ratio = 4.5/6,
                zlab = expression(paste(D[e], " [Gy]")),
                main = expression(paste(D[e], " Distribution", sep = "")),
                cex.global = 1, xscale_factor = 1.01)
```

Arguments

`sample`	data.frame (**required**): two column data frame with the input values, e.g. 'de' and 'se'.
`sample.groups`	list (optional): option to group the input data set like: `sample.groups = list(c(1:14), c(15:26), c(27:40))`
`sample.legend`	character (optional): character vector for a legend. This option is provided for the parameter `sample.groups`, but can also be used for one sample.
`sample.lty`	vector (with default): line type for the central value (see par. If the sample is grouped, a line type can be defined for each data set.)
`sample.pch`	vector (with default): point type for the presented data (see par. If the sample is grouped, a point type can be defined for each data.set
`sample.col`	vector or character (with default): colour of the points and lines (see colors). If the sample is grouped, a colour can be defined for each group.
`sample.mtext`	character (optional): mtext on the top of the plot. This option is only availabe if the grouping option is used. Otherwise information on the distribution is shown.
`zscale.log`	logical (with default): log De scale (`TRUE/FALSE`)
`zaxis.scale`	numeric (optional): option to set the z-scale manually. Example: `zaxis.scale = seq(50,120, by = 10)`
`zaxis.group_circle`	logical (with default): shows additional group circles for the 2-sigma uncertainties on the z-scale.
`yaxis.scale`	vector (optional): option for manual y-axis scaling. Example: `yaxis.scale=c(-15,15)`

`plot.2sigmaRange`
: logical (with default): plot a grey polygon showing the 2-sigma range of the central value.

`plot.area_ratio`
: vector (with default): option for manual plot ratio adjustments. Example `plot.area_ratio=4.5/5.5`

`zlab`
: character (with default): z-axis (semi circle) label

`main`
: character (with default): title of the plot

`cex.global`
: numeric (with default): global scaling factor

`xscale_factor`
: numeric (with default): scaling factor for the z-scale ticks.

Details

Details and the theoretical background on the radial plot are given in the cited literature.

Value

Returns a plot.

Note

This function is based on an S script of Rex Galbraith. To reduce the manual adjustments, the function has been rewritten. Thanks to Rex Galbraith for useful comments on this function.

Author(s)

Original S script: Rex Gablraith, University College London (UK)

Revised R script: Sebastian Kreutzer, JLU Giessen (Germany), 2012

References

Galbraith, R.F., 1988. Graphical Display of Estimates Having Differing Standard Errors. Technometrics, 30 (3), pp. 271-281.

Galbraith, R.F., 1990. The radial plot: Graphical assessment of spread in ages. International Journal of Radiation Applications and Instrumentation. Part D. Nuclear Tracks and Radiation Measurements, 17 (3), pp. 207-214.

Galbraith, R. & Green, P., 1990. Estimating the component ages in a finite mixture. International Journal of Radiation Applications and Instrumentation. Part D. Nuclear Tracks and Radiation Measurements, 17 (3), pp. 197-206.

Galbraith, R.F. & Laslett, G.M., 1993. Statistical models for mixed fission track ages. Nuclear Tracks And Radiation Measurements, 21 (4), pp. 459-470.

Galbraith, R.F., 1994. Some Applications of Radial Plots. Journal of the American Statistical Association, 89 (428), pp. 1232-1242.

Galbraith, R.F., 2010. On plotting OSL equivalent doses. Ancient TL, 28 (1), pp. 1-10.

Galbraith, R.F. & Roberts, R.G., 2012. Statistical aspects of equivalent dose and error calculation and display in OSL dating: An overview and some recommendations. Quaternary Geochronology, 11, pp.1-27.

See Also

plot, legend

Examples

```
##load data
data(ExampleData.DeValues)

##plot
plot_RadialPlot(ExampleData.DeValues,zscale.log = TRUE, zaxis.scale = seq(1995,4250, by = 500),
                zlab = expression(paste(D[e], " [s]")))
```

`readBIN2R` *Import Risoe BIN-file into R*

Description

Import a *.bin file produced by a Risoe DA15 and DA20 TL/OSL reader into R.

Usage

```
readBIN2R(file)
```

Arguments

`file`	character (**required**):	the bin-file name (including path), e.g. [WIN]: `readBIN2R("C:/Desktop/test.bin")`, [MAC/LINUX]: `readBIN2R("/User/test/Desktop/test.bin")`

Details

The binary data file is parsed byte per byte following the data structure published in the Appendices of the Analyst manual p. 42.

[,1]	ID	: Unique record ID (same ID as in slot `DATA`)	`numeric`
[,2]	SEL	: Record selection	`logical`
[,3]	VERSION	: Data format version number	`raw`
[,4]	LENGTH	: Length of this record	`integer`
[,5]	PREVIOUS	: Length of previous record	`integer`
[,6]	NPOINTS	: Number of data points in the record	`integer`
[,7]	LTYPE	: Luminescence type	`factor`
[,8]	LOW	: Low (temperature, time, wavelength)	`numeric`
[,10]	HIGH	: High (temperature, time, wavelength)	`numeric`
[,11]	RATE	: Rate (heating rate, scan rate)	`numeric`
[,12]	TEMPERATURE	: Sample temperature	`integer`
[,13]	XCOORD	: X position of a single grain	`integer`
[,14]	YCOORD	: Y position of a single grain	`integer`
[,15]	TOLDELAY	: TOL 'delay' channels	`integer`
[,16]	TOLON	: TOL 'on' channels	`integer`
[,17]	TOLOFF	: TOL 'off' channels	`integer`
[,18]	POSITION	: Carousel position	`integer`
[,19]	RUN	: Run number	`integer`
[,20]	TIME	: Data collection time (hh-mm-ss)	`factor`
[,21]	DATA	: Data collection date (dd-mm-yy)	`factor`

[,22]	SEQUENCE	: Sequence name	factor
[,23]	USER	: User name	factor
[,24]	DTYPE	: Data type	factor
[,25]	IRR_TIME	: Irradiation time	numeric
[,26]	IRR_TYPE	: Irradiation type (alpha, beta or gamma)	integer
[,27]	IRR_UNIT	: Irradiation unit (Gy, Rads, secs, mins, hrs)	integer
[,28]	BL_TIME	: Bleaching time	numeric
[,29]	BL_UNIT	: Bleaching unit (mJ, J, secs, mins, hrs)	integer
[,30]	AN_TEMP	: Annealing temperature	numeric
[,31]	AN_TIME	: Annealing time	numeric
[,32]	NORM1	: Normalisation factor (1)	numeric
[,33]	NORM2	: Normalisation factor (2)	numeric
[,34]	NORM3	: Normalisation factor (3)	numeric
[,35]	BG	: Background level	numeric
[,36]	SHIFT	: Number of channels to shift data	integer
[,37]	SAMPLE	: Sample name	factor
[,38]	COMMENT	: Comment	factor
[,39]	LIGHTSOURCE	: Light source	factor
[,40]	SET	: Set Number	integer
[,41]	TAG	: Tag	integer
[,42]	GRAIN	: Grain number	integer
[,43]	LPOWER	: Optical Stimulation Power	numeric
[,44]	SYSTEMID	: System ID	integer

Value

Returns an S4 Risoe.BINfileData-class object containing two slots:

METADATA	A data.frame containing all variables stored in the bin-file.
DATA	A list containing a numeric vector of the measured data. The ID corresponds to the record ID in METADATA

Note

Function has been successfully tested for BIN-file version 3 and 4. The version number depends on used the Sequence Editor.

Author(s)

Sebastian Kreutzer, JLU Giessen, Germany

References

Duller, G., 2007. Analyst.

See Also

readBin

Examples

```
##readBIN2R("~/Desktop/Example.bin")
```

`Risoe.BINfileData-class` *Class* `"Risoe.BINfileData"`

Description

S4 class object for luminescence data in R. The object is produced by the function readBIN2R.

Objects from the Class

Objects can be created by calls of the form `new("Risoe.BINfileData", ...)`.

Slots

`METADATA`: Object of class `"data.frame"` ~~

`DATA`: Object of class `"list"` ~~

`.S3Class`: Object of class `"character"` ~~

Methods

show `signature(object = "Risoe.BINfileData")`: ...

Note

##

Author(s)

Sebastian Kreutzer, JLU Giessen/Germany

References

##

See Also

`plot_BINfileData`

Examples

```
showClass("Risoe.BINfileData")
```

Second2Gray *Converting values from seconds (s) to gray (Gy)*

Description

Conversion of absorbed radiation dose in seconds (s) to the SI unit gray (Gy) using the Gaussian error propagation. Normally used for equivalent dose data.

Usage

```
Second2Gray(values,dose_rate)
```

Arguments

values	data.frame (**required**): measured data (values[,1]) and data error (values [,2])
dose_rate	vector (**required**): dose rate in Gy/s and dose rate error in Gy/s

Details

The functions uses the Gaussian error propagation to convert values with errors from seconds to gray.

Value

Returns a data.frame with converted values.

Note

##

Author(s)

Sebastian Kreutzer, JLU Giessen (Germany), 2012

References

##

See Also

##

Examples

```
##(1) for dose rate = 0.14 +/- 0.001
data(ExampleData.DeValues)
Second2Gray(ExampleData.DeValues,c(0.14,0.001))
```

Index