Package ‘TLdating’

June 29, 2017

Type Package

Title Tools for Thermoluminescences Dating

Version 0.1.3

Date 2016-08-01

Maintainer David Strebler <david.strebler@uni-koeln.de>

Description
A series of function to make thermoluminescence dating using the MAAD or the SAR protocol. This package completes the R package ``Luminescence.''

License GPL-3

Encoding UTF-8

Depends R (>= 2.0.0)

Imports methods, Luminescence (>= 0.5.0), gplots

Collate 'analyse_TL.MAAD.R' 'analyse_TL.SAR.R' 'analyse_TL.plateau.R'
'calc_TL.LxTx.R' 'calc_TL.MAAD.fit.I.R' 'calc_TL.MAAD.fit.Q.R'
'calc_TL.MAAD.separate.R' 'calc_TL.MAAD.average.R'
'calc_TL.SAR.fit.R' 'calc_TL.plateau.R' 'calc_TL.temperature.R'
'mod_align.peaks.R' 'mod_extract.TL.R' 'mod_extract.aliquot.R'
'mod_extract.preheat.R' 'mod_update.dType.R'
'mod_update.error.R' 'mod_remove.preheat.R'
'mod_remove.aliquot.R' 'mod_substract.background.R'
'plot_align.peaks.R' 'plot_extract.TL.R'
'plot_remove.preheat.R' 'plot_substract.background.R'
'plot_TL.plateau.R' 'plot_TL.MAAD.R' 'plot_TL.SAR.R'
'script_TL.export.R' 'script_TL.import.R' 'script_TL.MAAD.R'
'script_TL.plateau.R' 'script_TL.pretreatment.R'
'script_TL.SAR.R' 'TLum-class.R' 'TLum.Data-class.R'
'TLum.BIN.File-class.R' 'TLum.Data.Curve-class.R'
'TLum.Analysis-class.R' 'TLum.Results-class.R'
'Risoe.BINfileData2TLum.BIN.File.R'
'TLum.BIN.File2Risoe.BINfileData.R'
'TLum.BIN.File2TLum.Analysis.R' 'RLum.Analysis2TLum.Analysis.R'
'TLum.BIN.File2TLum.Data.Curve.R'
'TLum.Analysis2TLum.BIN.File.R' 'TLum.Analysis2RLum.Analysis.R'
'template_DRAC4flint.R' 'use_DRAC4flint.R'
'template_DRAC4ceramic.R' 'use_DRAC4ceramic.R'
'template_DRAC4cave.R' 'use_DRAC4cave.R'
'template_DRAC4brick.R' 'use_DRAC4brick.R' 'TLdating-package.R'

RoxygenNote 5.0.1

NeedsCompilation no

Author David Strebler [aut, cre],
Dominik Brill [ths],
Helmut Brückner [ths]

Repository CRAN

Date/Publication 2016-08-31 14:28:21

R topics documented:

TLdating-package *Tools for Thermoluminescences Dating*

Description

A series of functions for thermoluminescence dating using the MAAD or the SAR protocol. This package adds to the R package Luminescence.

Details

Package:	TLdating
Type:	Package
Version:	0.1.2
Date:	2016-03-01
License:	GPL-3

Author(s)

Authors

David Strebler, University of Cologne, Germany

Beta-tester

Anja Zander, University of Cologne, Germany

Supervisor

Helmut Brückner, University of Cologne, Germany
Dominik Brill, University of Cologne, Germany

Project source code repository
https://github.com/dstreble/TLdating

Related package projects
http://www.r-luminescence.de
http://cran.r-project.org/package=Luminescence

Package maintainer
David Strebler, Geographisches Institut, Universitat zu Koeln, Cologne, Germany.
<david.strebler@uni-koeln.de>

Acknowledgement
This project is realized in the context of the CRC 806 “Our Way to Europe” (http://www.sfb806.uni-koeln.de/) which is funded by the German Research foundation (DFG).
The code and the structure of this package is partially based on those from the R package Luminescence version 0.5.1 developed by S. Kreutzer, M. Dietze, C. Burow, M.C. Fuchs, C. Schmidt, M. Fisher and R.K. Smedley.

---

analyse_TL.MAAD *MAAD protocol for TL dating*

Description

Function to estimate the ED in TL dating using the MAAD protocol.
It provides an estimation of the palaeodose (Q) and/or the supralinearity correction (I). The equivalent dose (ED) is estimated by the addition of Q and I.
See details for more information.

Usage

```
analyse_TL.MAAD(object, eval.Tmin, eval.Tmax,
  rejection.criteria = list(testdose.error = 10, paleodose.error = 10),
  fitting.parameters = list(fit.method = "LIN", fit.weighted = FALSE,
  fit.use.slope = FALSE, fit.aDoses.min = 0, fit.aDoses.max = NA, fit.rDoses.min
  = 0, fit.rDoses.max = NA), plotting.parameters = list(plot.Tmin = 0,
  plot.Tmax = NA, no.plot = FALSE))
```

Arguments

object	TLum.Analysis (**required**): object containing the TL curves used for the ED estimation.
eval.Tmin	integer (**required**): Temperature (°C) of the lower boundary for the signal integration.
eval.Tmax	integer (**required**): Temperature (°C) of the upper boundary for the signal integration.

`rejection.criteria`
: list (with default): list containing the rejection criteria (in %). See details.

`fitting.parameters`
: list (with default): list containing the fitting parameters. See details.

`plotting.parameters`
: list (with default): list containing the plotting parameters. See details.

Details

This function estimates the equivalent dose for the thermoluminescence dating with the MAAD protocol. It can provide an estimation of the palaeodose (Q) and the supralinearity correction (I) simultaniously or separately. These are estimated using the growth curve approach (QC) (Aitken, 1985) and the dose plateau approach (DP). Both approaches should provide a similar result. The equivalent dose is estimated by the addition of Q and I
The Lx/Tx matrix is estimated using calc_TL.LxTx.
The average TL curves for each dose step are estimate using calc_TL.MAAD.average.
The plateau test values are estimated using calc_TL.plateau.

Rejection criteria

The rejection criteria are:

`testdose.error` numeric: Maximum error accepted on Tx (in %).

`paleodose.error` numeric: Maximum error accepted on Lx (in %).

Fitting parameters

The fitting parameters are:

`method` character: Fitting method (`LIN`, `EXP`, `EXP+LIN` or `EXP+EXP`).

`fit.weighted` logical: If the fitting is weighted or not.

`fit.use.slope` logical: If the slope of the Q growth curve is reused for the supralinearity correction.

`fit.aDoses.min` numeric: Lowest additive dose used for the fitting.

`fit.aDoses.max` numeric: Highest additive dose used for the fitting.

`fit.rDoses.min` numeric: Lowest regenerative dose used for the fitting.

`fit.rDoses.max` numeric: Highest regenerative dose used for the fitting.

See also calc_TL.MAAD.fit.Q and calc_TL.MAAD.fit.I.

Plotting parameters

The plotting parameters are:

`plot.Tmin` numeric: Lowest temperature plotted.

`plot.Tmax` numeric: Highest temperature plotted.

`no.plot` logical: If `TRUE`, the results will not be plotted.

See also plot_TL.MAAD.

Value

The results are plotted using plot_TL.MAAD.

The function also provides a TLum.Results containing:

De.GC list: Results obtained with the dose plateau approach and their uncertainties (`De`, `De.error`, `Q`, `Q.error`, `I`, `I.error`)

De.DP list: Results obtained with the growth curve approach and their uncertainties (`De`, `De.error`, `Q`, `Q.error`, `I`, `I.error`)

LnLxTnTx.table matrix: Lx/Tx values

RC.Status character: The acceptance result.

Author(s)

David Strebler, University of Cologne (Germany)

References

Aitken, M.J. (1985) Thermoluminescence Dating, Academic Press, London

See Also

calc_TL.LxTx, calc_TL.plateau, calc_TL.MAAD.average, calc_TL.MAAD.separate, calc_TL.MAAD.fit.I, calc_TL.MAAD.fit.Q, analyse_TL.SAR.

Examples

```
##load data

##perform analysis
```

`analyse_TL.plateau` *Plateau test function for TL dating*

Description

This function performs the plateau test for TL curves (Ln/Lx).

Usage

```
analyse_TL.plateau(object, plotting.parameters = list(plot.Tmin = 0, plot.Tmax
  = NA, plateau.Tmin = 0, plateau.Tmax = NA, no.plot = FALSE))
```

Arguments

`object` TLum.Analysis (**required**): object containing the TL curves used for the Plateau test.

`plotting.parameters` list (with default): list containing the plotting parameters. See details.

Details

Plotting parameters
The plotting parameters are:

`plot.Tmin` numeric: Lowest temperature plotted.

`plot.Tmax` numeric: Highest temperature plotted.

`no.plot` logical: If `TRUE`, the results will not be plotted.

See also plot_TL.plateau.

Value

The results are plotted using plot_TL.plateau.

Author(s)

David Strebler, University of Cologne (Germany)

References

Aitken, M.J. (1985) Thermoluminescence Dating, Academic Press, London

See Also

calc_TL.LxTx, calc_TL.plateau, analyse_TL.MAAD

`analyse_TL.SAR` *SAR protocol for thermoluminescence dating*

Description

This function calculates the equivalent dose (ED) using the SAR protocol.
See details for more information.

Usage

```
analyse_TL.SAR(object, eval.Tmin, eval.Tmax,
  rejection.criteria = list(recycling.ratio = 10, recuperation.rate = 10,
  testdose.error = 10, paleodose.error = 10),
  fitting.parameters = list(fit.method = "LIN", fit.weighted = FALSE,
  fit.rDoses.min = NA, fit.rDoses.max = NA),
  plotting.parameters = list(plot.Tmin = 0, plot.Tmax = NA, no.plot = FALSE))
```

Arguments

`object` `TLum.Analysis` (**required**): object containing the TL curves used for the ED calculation.

`eval.Tmin` integer (**required**): Temperature (°C) of the lowest boundary for the signal integration.

`eval.Tmax` integer (**required**): Temperature (°C) of the upper boundary for the signal integration.

`rejection.criteria`
list (with default): list containing the rejection criteria (in %). See details.

`fitting.parameters`
list (with default): list containing the fitting parameters. See details.

`plotting.parameters`
list (with default): list containing the plotting parameters. See details.

Details

This function estimates the equivent dose in thermoluminescence dating using the SAR protocol. The equivalent dose is estimated for each disc using the growth curve approaches (QC) (Aitken, 1985) and the dose plateau approach (DP). Both approach should provide a similar result.

The Lx/Tx matrix is estimated using calc_TL.LxTx.
The plateau test values are estimated using calc_TL.plateau.

Rejection criteria

The rejection criteria are:

`recycling.ratio` numeric: Maximum recycling ratio accepted (in %).

`recuperation.rate` numeric: Maximum recuperation rate accepted (in %).

`paleodose.error` numeric: Maximum error accepted on Lx (in %).

`testdose.error` numeric: Maximum error accepted on Tx (in %).

Fitting parameters

The fitting parameters are:

`method` character: Fitting method (`LIN`, `EXP`, `EXP+LIN` or `EXP+EXP`).

`fit.weighted` logical: If the fitting is weighted or not.

`fit.rDoses.min` numeric: lowest regenerative dose used for the fitting.

`fit.rDoses.max` numeric: highest regenerative dose used for the fitting.

See also calc_TL.SAR.fit.

Plotting parameters

The plotting parameters are:

`plot.Tmin` numeric: lowest temperature plotted.

`plot.Tmax` numeric: highest temperature plotted.

`no.plot` logical: If `TRUE`, the results will not be plotted.

See also plot_TL.SAR.

Value

The results are plotted using plot_TL.SAR.

The function also provides an TLum.Results containing:

`De.GC` list: Results obtained with the dose plateau approach and their uncertainties. (`De`, `De.error`)

`De.DP` list: Results obtained with the growth curve approach and their uncertainties. (`De`, `De.error`)

`LnLxTnTx.table` matrix: Lx/Tx values

`RC.Status` character: Results of the rejection tests.

Author(s)

David Strebler, University of Cologne (Germany),
David Strebler

References

Aitken, M.J. (1985) Thermoluminescence Dating, Academic Press, London

Murray & Wintle (2000). Luminescence dating of quartz using an improved single-aliquot regenerative-dose protocol. Radiation Measurements, Vol.32, No.1, p.57-73.

See Also

calc_TL.LxTx, calc_TL.plateau, calc_TL.SAR.fit, analyse_TL.MAAD.

`calc_TL.LxTx` *calculation of the Lx/Tx matrix*

Description

Internal function called by analyse_TL.MAAD and analyse_TL.SAR.
This function separates the Lx matrix from the Tx matrix. Then, it estimates the Lx/Tx matrix. It also provides a name for each of the curves.

Usage

```
calc_TL.LxTx(object)
```

Arguments

`object`	`TLum.Analysis` (**required**): `TLum.Analysis` object

Details

Warning: This function is an internal function and should not be used except for development purposes. Internal functions can be heavily modified and even renamed or removed in new version of the package.

Value

The function provides an TLum.Results containing:

`Temperatures` numeric: Vector with the temperature values.

`Names` character: Vector with the curve names.

`Datatype` character: Vector with the curve type.

`Doses` numeric: Vector with the curve doses.

`Testdoses` numeric: Vector with the curve test-doses.

`Lx` Lx matrix.

`Lx.error` Absolute error for the Lx matrix.

`Tx` Tx matrix.

`Tx.error` Absolute error for the Tx matrix

`LxTx` Lx/Tx matrix.

`LxTx.error` Absolute error for the Lx/Tx matrix.

Author(s)

David Strebler, University of Cologne (Germany).

References

Aitken, M.J. (1985) Thermoluminescence Dating, Academic Press, London

Murray & Wintle (2000). Luminescence dating of quartz using an improved single-aliquot regenerative-dose protocol. Radiation Measurements, Vol.32, No.1, p.57-73.

See Also

analyse_TL.MAAD, analyse_TL.SAR.

`calc_TL.MAAD.average` *Estimate average additive curves for the MAAD protocol.*

Description

Internal function called by analyse_TL.MAAD.
This function estimates the average curves for each additive dose.

Usage

```
calc_TL.MAAD.average(names, doses, Lx, Lx.error)
```

Arguments

`names` character (**required**): Names of the TL curves

`doses` numeric (**required**): additive doses used for the TL curve

`Lx` numeric (**required**): Lx matrix

`Lx.error` numeric (**required**): Error for the Lx matrix.

Details

Warning: This function is an internal function and should not be used except for development purposes. Internal functions can be heavily modified and even renamed or removed in new version of the package.

Value

The function provide an TLum.Results object containing:

`names` character: Vector with the names of the average additive curves.

`doses` character: Vector with the additive doses corresponding to each average additive curve.

`Lx` numeric: new average additive curve Lx matrix.

`Lx.error` numeric: Error on the new Lx matrix.

Author(s)

David Strebler, University of Cologne (Germany).

`calc_TL.MAAD.fit.I` *Estimation of the supralinearity value for the MAAD protocol*

Description

Internal function called by analyse_TL.MAAD.
This function estimates the supralinearity correction based on the dose vector and the Lx/Tx vector provided.
See details for more information.

Usage

```
calc_TL.MAAD.fit.I(LxTx, LxTx.error, doses, slope = NULL,
  fitting.parameters = list(fit.method = "LIN", fit.weighted = FALSE,
  fit.use.slope = FALSE))
```

Arguments

`LxTx` numeric (**required**): Lx/Tx vector

`LxTx.error` numeric (**required**): Error for the Lx/Tx vector

`doses` numeric (**required**): doses vector

`slope` list (with default): Property of the additive growth curve.

`fitting.parameters`
list (with default): fitting parameters. See details.

Details

This function estimates the supralinearity correction based on the doses vector and the Lx/Tx matrix provided.
Different fitting methods are available (`LIN`, `EXP`, `EXP+LIN` or `EXP+EXP`). Morover, the fitting can be weigthed or not.
If the fitting parameter `fit.use.slope` is `TRUE`, the function will use the data from `slope` to define the fitting curve for the supralinearity correction. In that case, the supralinearity correction growth curve will be parallel to the additive growth curve.

#' **Fitting parameters**
The fitting parameters are:

`method` character: Fitting method (`LIN`, `EXP`, `EXP+LIN` or `EXP+EXP`).

`fit.weighted` logical: If the fitting is weighted or not.

`fit.use.slope` logical: If the slope of the Q growth curve is reused for the supralinearity correction.

`fit.rDoses.min` numeric: lowest regenerative dose used for the fitting.

`fit.rDoses.max` numeric: Highest regenerative dose used for the fitting.

Warning: This function is an internal function and should not be used except for development purposes. Internal functions can be heavily modified and even renamed or removed in new version of the package.

Value

The function provides an TLum.Results object containing:

`GC` lm: The fitting result.

`i` numeric: The supralinearity correction estimation for the given equivalent dose

`I.error` numeric: The error for the supralinearity correction estimation

`summary` numeric: The parameters of the fitting result.

Author(s)

David Strebler, University of Cologne (Germany).

See Also

calc_TL.MAAD.fit.Q, analyse_TL.MAAD.

calc_TL.MAAD.fit.Q *Estimation of the palaeodose (Q) with the MAAD protocol*

Description

Internal function called by analyse_TL.MAAD.
This function makes a first estimation of the palaeodose based on a doses vector and a Lx/Tx vector provided.
See details for more information.

Usage

```
calc_TL.MAAD.fit.Q(LxTx, LxTx.error, doses,
  fitting.parameters = list(fit.method = "LIN", fit.weighted = FALSE))
```

Arguments

`LxTx`	numeric (**required**): Lx/Tx vector
`LxTx.error`	numeric (**required**): Error for the Lx/Tx vector
`doses`	numeric (**required**): doses vector
`fitting.parameters`	list (with default): fitting parameters. See details.

Details

This function estimates the equivalent dose before any supralineary correction based on the doses vector and the Lx/Tx matrix provided.
Different fitting methods are available (`LIN`, `EXP`, `EXP+LIN` or `EXP+EXP`). Moreover, the fitting can be weigthed or not.

#' **Fitting parameters**
The fitting parameters are:

`method` character: Fitting method (`LIN`, `EXP`, `EXP+LIN` or `EXP+EXP`).

`fit.weighted` logical: If the fitting is weighted or not.

`fit.use.slope` logical: If the slope of the Q growth curve is reused for the supralinearity correction.

`fit.rDoses.min` numeric: lowest regenerative dose used for the fitting.

`fit.rDoses.max` numeric: Highest regenerative dose used for the fitting.

Warning: This function is an internal function and should not be used except for development purposes. Internal functions can be heavily modified and even renamed or removed in new version of the package.

Value

The function provides an TLum.Results object containing:

`GC` lm: fitting result.

`Q` numeric: equivalent dose estimation

`Q.error` numeric: Error for the equivalent dose estimation

`summary` numeric: parameters of the fitting result.

Author(s)

David Strebler, University of Cologne (Germany).

See Also

calc_TL.MAAD.fit.I, analyse_TL.MAAD.

`calc_TL.MAAD.separate` *Separate the additive and the regenerative curves*

Description

Internal function called by analyse_TL.MAAD.
This function separates the additive curves from the regenerative curves using a vector containing the data type associate with each curve. Additive curves have "Natural" or "N+dose" as datatype. Regenerative curves have "Bleach" or "Bleach+dose" as datatype. Other datatypes are not supported.

Usage

```
calc_TL.MAAD.separate(Lx, Lx.error, doses, dTypes)
```

Arguments

`Lx`	numeric (**required**): Lx matrix
`Lx.error`	numeric (**required**): Error for the Lx matrix.
`doses`	numeric (**required**): doses vector
`dTypes`	character (**required**): data type vector.

Details

Warning: This function is an internal function and should not be used except for development purposes. Internal functions can be heavily modified and even renamed or removed in new version of the package.

Author(s)

David Strebler, University of Cologne (Germany).

calc_TL.plateau *Estimate value for plateau test*

Description

The function estimates the quotient between the natural and the additive/regenerate signal.

Usage

```
calc_TL.plateau(Ln, Ln.error, Lx, Lx.error)
```

Arguments

`Ln`	numeric (**required**): Ln vector
`Ln.error`	numeric (**required**): Error for the Ln vector
`Lx`	numeric (**required**): Ln matrix
`Lx.error`	numeric (**required**): Error for the Lx matrix

Details

Warning: This function is an internal function and should not be used except for development purposes. Internal functions can be heavily modified and even renamed or removed in new version of the package.

Value

The function provides an TLum.Results object containing:

`LnLx` numeric: Ln/Lx matrix

`LnLx.error` numeric: Error for the Ln/Lx matrix.

Author(s)

David Strebler, University of Cologne (Germany).

calc_TL.SAR.fit *Estimation of the equivalent dose (ED) value for the SAR protocol*

Description

Internal function called by analyse_TL.SAR.
This function estimates the equivalent dose (ED) based on a doses vector and a Lx/Tx vector provided.
See details for more information.

Usage

```
calc_TL.SAR.fit(doses, LnTn, LnTn.error, LxTx, LxTx.error,
  fitting.parameters = list(fit.method = "LIN", fit.weighted = FALSE))
```

Arguments

doses	numeric (**required**): doses vector
LnTn	numeric (**required**): Ln/Tn.
LnTn.error	numeric (**required**): Error for the Ln/Tn.
LxTx	numeric (**required**): Lx/Tx vector
LxTx.error	numeric (**required**): Error for the Lx/Tx vector
fitting.parameters	list (with default): fitting parameters. See details.

Details

This function estimates the equivalent dose based on the doses vector, Ln/Tn and the Lx/Tx matrix provided.
Different fitting methods are available (LIN, EXP, EXP+LIN or EXP+EXP). Moreover, the fitting can be weigthed or not.

Fitting parameters

The fitting parameters are:

method character: Fitting method (LIN, EXP, EXP+LIN or EXP+EXP).

fit.weighted logical: If the fitting is weighted or not.

fit.use.slope logical: If the slope of the Q growth curve is reused for the sublinearity correction.

fit.rDoses.min numeric: Lowest regenerative dose used for the fitting.

fit.rDoses.max numeric: Highest regenerative dose used for the fitting.

Warning: This function is an internal function and should not be used except for development purposes. Internal functions can be heavily modified and even renamed or removed in new version of the package.

Value

The function provides an TLum.Results object containing:

GC list: fitting curve.

Q numeric: equivalent dose estimation

Q.error numeric: Error for the equivalent dose estimation

summary list: parameters of the fitting result.

Author(s)

David Strebler, University of Cologne (Germany).

See Also

analyse_TL.SAR.

calc_TL.temperature *calculate temperature vector*

Description

This function calculates the temperature vector.

Usage

```
calc_TL.temperature(nPoints, Tmax, Hrate, an_temp = 0, an_time = 0,
  rec_ramp2PH = FALSE, rec_duringPH = FALSE)
```

Arguments

nPoints	numeric (**required**): number of points.
Tmax	numeric (**required**): Maximum temperature.
Hrate	numeric (**required**): Heating rate.
an_temp	numeric (with default): Annealing temperature.
an_time	numeric (with default): Annealing time.
rec_ramp2PH	logical (with default): Indicate if the signal was record during the ramp up to the preheat temperature.
rec_duringPH	logical (with default): Indicate if the signal was record during the preheat plateau.

Details

Warning: This function is an internal function and should not be used except for development purposes. Internal functions can be heavily modified and even renamed or removed in new version of the package.

Value

This function provides a new `TLum.Results` object containing the times and temperature vectors.

Author(s)

David Strebler, University of Cologne (Germany).

mod_align.peaks *Aligning the TL peaks*

Description

This function detects the peak position for each TL curve of the object and aligns them. It uses the average of the testdose maximum positions as reference for the new peak position.

Usage

```
mod_align.peaks(object, aligning.parameters = list(peak.Tmin = 0, peak.Tmax =
  NA, no.testdose = FALSE), plotting.parameters = list(plot.Tmin = 0,
  plot.Tmax = NA, no.plot = FALSE))
```

Arguments

`object` `TLum.Analysis` (**required**): object containing the TL curves that have to be aligned.

`aligning.parameters` list (with default): list containing the aligning parameters. See details.

`plotting.parameters` list (with default): list containing the plotting parameters. See details.

Details

Aligning parameters

The aligning parameters are:

`peak.Tmin` numeric: Lower boundary for looking at the peak maximum position.

`peak.Tmax` numeric: Upper boundary for looking at the peak maximum position.

`no.testdose` logical: If `TRUE`, the function will use the Lx curves rather than the Tx curves as reference for the peak maximum position.

Plotting parameters

The plotting parameters are:

`no.plot` logical: If `TRUE`, the results will not be plotted.

Value

This function provides a new `TLum.Analysis` object with the same TL curves but aligned.
It also plots the original TL curves, the TL curves used to determine the peak maximum position, and the shiffted TL curves using plot_align.peaks.

Author(s)

David Strebler, University of Cologne (Germany).

See Also

plot_align.peaks.

`mod_extract.aliquot` *extract aliquots*

Description

This function extracts a list of aliquots from a `TLum.Analysis` object.

Usage

```
mod_extract.aliquot(object, list)
```

Arguments

`object`	`TLum.Analysis` (**required**): object containing the initial TL curves.
`list`	numeric (**required**): list containing the position of the aliquots that shall be used.

Value

This function provides a `TLum.Data.Curve` object containing only the aliquots specified in the list.

Author(s)

David Strebler, University of Cologne (Germany).

`mod_extract.preheat` *Extract preheat*

Description

This function provides a new `TLum.Analysis` object containing only the preheat curves.

Usage

```
mod_extract.preheat(object, plotting.parameters = list(no.plot = FALSE))
```

Arguments

`object`	`TLum.Analysis` (**required**): object containing the initial TL curves.
`plotting.parameters`	list (with default): list containing the plotting parameters. See details.

Details

Plotting parameters
The plotting parameters are:

`no.plot` logical: If `TRUE`, the results will not be plotted.

Value

This function provides a new `TLum.Analysis` object with only the preheat curve.
It also plots the preheat curves and the TL curves using plot_remove.preheat.

Author(s)

David Strebler, University of Cologne (Germany).

See Also

plot_remove.preheat

`mod_extract.TL` *Extract TL*

Description

This function provides a new `TLum.Analysis` object containing only the TL curves.

Usage

```
mod_extract.TL(object, plotting.parameters = list(no.plot = FALSE),
  record.parameters = list(separatePreheat = TRUE, recDuringPreheatRamp =
  FALSE, recDuringPreheatPlateau = FALSE))
```

Arguments

`object` `TLum.Analysis` (**required**): object containing the initial TL curves.

`plotting.parameters`
list (with default): list containing the plotting parameters. See details.

`record.parameters`
list (with default): list containing the record parameters. See details.

Details

Plotting parameters
The plotting parameters are:

`no.plot` logical: If `TRUE`, the results will not be plotted.

Record parameters
The record parameters are:

`includePreheat` logical: If `TRUE`, the preheat was included in the TL recording. If `FALSE`, the preheat was recorded separately.

`recDuringPreheatRamp` logical: Only used when `includePreheat` is `TRUE`. If `TRUE`, the signal was recorded during the preheat ramp.

`recDuringPreheatPlateau` logical: Only used when `includePreheat` is `TRUE`. If `TRUE`, the signal was recorded during the preheat plateau.

Value

This function provides a new `TLum.Analysis` with only the TL curve.
It also plots the TL curves using plot_extract.TL.

Author(s)

David Strebler, University of Cologne (Germany).

See Also

plot_extract.TL

`mod_remove.aliquot` *Remove aliquots*

Description

This function removes a list of aliquots from a `TLum.Analysis` object.

Usage

```
mod_remove.aliquot(object, list)
```

Arguments

`object`	`TLum.Analysis` (**required**): object containing the initial TL curves.
`list`	numeric (**required**): list containing the position of the aliquots to remove.

Value

This function provides a `TLum.Analysis` without the aliquots specified in the list.

Author(s)

David Strebler, University of Cologne (Germany).

`mod_remove.preheat` *Remove preheat*

Description

This function provides a new `TLum.Analysis` object from which the preheat curves were removed.

Usage

```
mod_remove.preheat(object, plotting.parameters = list(no.plot = FALSE))
```

Arguments

`object`	`TLum.Analysis` (**required**): object containing the initial TL curves.
`plotting.parameters`	list (with default): list containing the plotting parameters. See details.

Details

Plotting parameters
The plotting parameters are:

`no.plot` logical: If `TRUE`, the results will not be plotted.

Value

This function provides a new `TLum.Analysis` object without the preheat curves.
It also plots the preheat curves and the TL curves using plot_remove.preheat.

Author(s)

David Strebler, University of Cologne (Germany).

See Also

plot_remove.preheat

`mod_substract.background`

Creates a new TLum.Analysis object where the background was removed from the signal.

Description

Creates a new TLum.Analysis object where the background was removed from the signal.

Usage

```
mod_substract.background(object, keep.background = FALSE,
  plotting.parameters = list(no.plot = FALSE))
```

Arguments

`object`	`TLum.Analysis` (**required**): object containing the initial TL curves.
`keep.background`	logical (with default): Parameter indicating if the background curve have to be kept or suppressed.
`plotting.parameters`	list (with default): list containing the plotting parameters. See details.

Details

Plotting parameters
The plotting parameters are:

`no.plot` logical: If `TRUE`, the results will not be plotted.

Value

This function provides a new `TLum.Analysis` object with the TL curves after background subtraction.
It also plots the TL curves, the background curves and the background substracted curves using plot_remove.preheat.

Author(s)

David Strebler, University of Cologne (Germany).

See Also

plot_substract.background

`mod_update.dType` *mod identify dType*

Description

This function identify the data type of each curve from a `TLum.Analysis` object. It also add the new data type "testdose" and "preheat" based on the `comment` present in the `TLum.Analysis` object or a sequence vector.

Usage

```
mod_update.dType(object, method = "comment", ref = list(sequence =
  c("Preheat", "Natural", "Background", "Testdose", "Background"), oneByOne =
  FALSE, protocol = "SAR", preheat = NA, testdose = NA))
```

Arguments

`object`	`TLum.Analysis` (**required**): object containing the initial TL curves.
`method`	character (with default): Defines the methode use to identify the new data type ("comment", "sequence", "temperature+dose").
`ref`	list (with default): Contains the reference values to identify the new data type.

Value

This function provides a new `TLum.Analysis` with the new dtype.

Author(s)

David Strebler, University of Cologne (Germany).

`mod_update.error` *Update the error matrix*

Description

This function update the error vector of each curve from a `TLum.Analysis` object.

Usage

```
mod_update.error(object, method, absolute.error = NULL,
  relative.error = NULL, k = 1)
```

Arguments

`object`	`TLum.Analysis` (**required**): object containing the initial TL curves.
`method`	character (with default): Defines the methode use to update the error matrix ("poisson", "absolute", "relative", "combine").
`absolute.error`	numeric (with default): absolute error of the TL signals (used by the "absolute" and "combine" methods).
`relative.error`	numeric (with default): Relative error of the TL signals (used by the "relative" and "combine" methods).
`k`	numeric (with default): corrective factor to use when using a poisson distribution for the uncertainties (used by the "poisson" method).

Value

This function provides a new `TLum.Analysis` with the new error matrix.

Author(s)

David Strebler, University of Cologne (Germany).

`plot_align.peaks` *Plots mod_alignPeaks results*

Description

This function plots the results obtained by mod_alignPeaks.

Usage

```
plot_align.peaks(temperatures, old.TL, new.TL, ref.TL, pos.peak,
  plotting.parameters = list(plot.Tmin = 0, plot.Tmax = NA))
```

Arguments

`temperatures` numeric: Vector containing the temperature step

`old.TL` numeric: Matrix containing the luminescence signal before the peak alignment.

`new.TL` numeric: Matrix containing the luminescence signal after the peak alignment.

`ref.TL` numeric: Matrix containing the luminescence signal used as reference to define the peak position.

`pos.peak` numeric: Average peak position.

`plotting.parameters` list (with default): list containing the plotting parameters. See details.

Details

Plotting parameters
The plotting parameters are:

`plot.Tmin` logical: Minimum temperature which is plotted.

`plot.Tmax` logical: Maximum temperature which is plotted.

Author(s)

David Strebler

See Also

mod_align.peaks

`plot_extract.TL` *plot the TL curves*

Description

This function plots the results obtained by mod_extract.TL.

Usage

```
plot_extract.TL(temperatures, TL)
```

Arguments

`temperatures` numeric: matrix containing the temperature steps for each TL curve.

`TL` numeric: Matrix containing the luminescence signal for the TL curves.

Author(s)

David Strebler

See Also

mod_extract.TL

`plot_remove.preheat` *Plotting of the preheat and TL curves*

Description

This functions plots the results obtained by mod_remove.preheat

Usage

```
plot_remove.preheat(PH.signal, PH.temperatures, PH.times, TL.signal,
  TL.temperatures)
```

Arguments

`PH.signal`	numeric: matrix containing the preheat curves.
`PH.temperatures`	numeric: matrix containing the temperature steps for each preheat curve.
`PH.times`	numeric: matrix containing the time steps for each preheat curve.
`TL.signal`	numeric: matrix containing the TL curves.
`TL.temperatures`	numeric: matrix containing the temperature steps for each TL curve.

Author(s)

David Strebler

See Also

mod_remove.preheat

`plot_substract.background` *Plotting function for mod_substract.background.*

Description

This function plots the results of the mod_substract.background function. It plots the TL curves, the background (BG) curves and the background substracted curves.

Usage

```
plot_substract.background(old.TL, BG, new.TL, temperatures)
```

Arguments

`old.TL`	numeric: Matrix containing the luminescence signal before background subtraction.
`BG`	numeric: Matrix containing the luminescence signal from the background curves.
`new.TL`	numeric: Matrix containing the luminescence signal after background subtraction.
`temperatures`	numeric: Vector containing the temperature step

Author(s)

David Strebler

See Also

mod_substract.background

`plot_TL.MAAD` *plot MAAD result*

Description

This function plots the results for analyse_TL.MAAD. The first page regroups all the information about the additive curves (names, doses, intensity vs. temperature and plateau test for Lx, Tx and Lx/Tx). The second page regroups all the information about the regenerative curves (names, doses, intensity vs. temperature and plateau test for Lx, Tx and Lx/Tx). The third page regroups all the information about the equivalent dose (dose plateau for the palaeodose and the supralinearity correction, growth curves, rejection criteria,...).

Usage

```
plot_TL.MAAD(sample.name, temperatures, eval.Tmin, eval.Tmax, aNames, aDoses,
  aLx, aTx, aLxTx, aLx.plateau, aTx.plateau, aLxTx.plateau, rNames, rDoses, rLx,
  rTx, rLxTx, rLx.plateau, rTx.plateau, rLxTx.plateau, DP.Q.line,
  DP.Q.line.error, GC.Q.line, GC.Q.slope, GC.Q.LxTx, GC.Q.LxTx.error,
  GC.Q.doses, GC.Q.names, DP.I.line, DP.I.line.error, GC.I.line, GC.I.slope,
  GC.I.LxTx, GC.I.LxTx.error, GC.I.doses, GC.I.names, Q.DP, Q.DP.error, Q.GC,
  Q.GC.error, I.DP, I.DP.error, I.GC, I.GC.error, De.GC, De.GC.error, De.DP,
  De.DP.error, rejection.values, fitting.parameters,
  plotting.parameters = list(plot.Tmin = 0, plot.Tmax = NA))
```

Arguments

`sample.name`	character (**required**): Sample name.
`temperatures`	numeric (**required**): temperature vector
`eval.Tmin`	integer (**required**): Temperature (°C) of the lower boundary for the signal integration.
`eval.Tmax`	integer (**required**): Temperature (°C) of the upper boundary for the signal integration.
`aNames`	character (**required**): Name vector for the additive curves.
`aDoses`	numeric (**required**): Dose vector for the additive curves.
`aLx`	numeric (**required**): Lx matrix for the additive curves.
`aTx`	numeric (**required**): Tx matrix for the additive curves.
`aLxTx`	numeric (**required**): Lx/Tx matrix for the additive curves.
`aLx.plateau`	numeric (**required**): Ln/Lx matrix for the additive curves.
`aTx.plateau`	numeric (**required**): Ln/Tx matrix for the additive curves.
`aLxTx.plateau`	numeric (**required**): (Ln/Tn)/(Lx/Tx) matrix for the additive curves.

`rNames` character (**required**): Name vector for the regenerative curves.

`rDoses` numeric (**required**): Dose vector for the regenerative curves.

`rLx` numeric (**required**): Lx matrix for the regenerative curves.

`rTx` numeric (**required**): Tx matrix for the regenerative curves.

`rLxTx` numeric (**required**): Lx/Tx matrix for the regenerative curves.

`rLx.plateau` numeric (**required**): Ln/Lx matrix for the regenerative curves.

`rTx.plateau` numeric (**required**): Tn/Tx matrix for the regenerative curves.

`rLxTx.plateau` numeric (**required**): (Ln/Tn)/(Lx/Tx) matrix for the regenerative curves.

`DP.Q.line` numeric (**required**): Vector containing the estimation of Q for each T° step.

`DP.Q.line.error` numeric (**required**): Vector containing the uncertainty on the estimation of Q for each T° step.

`GC.Q.line` numeric (**required**): growth curve for Q

`GC.Q.slope` numeric (**required**): growth curve parameters for Q

`GC.Q.LxTx` numeric (**required**): Lx/Tx vector used for Q estimation using the growth curve approach.

`GC.Q.LxTx.error` numeric (**required**): Error on the Lx/tx vector used for Q estimation using the growth curve approach.

`GC.Q.doses` numeric (**required**): Doses used for Q estimation using the growth curve approach.

`GC.Q.names` numeric (**required**): Names of the Lx/tx vector used for Q estimation using the growth curve approach.

`DP.I.line` numeric (**required**): Vector containing I for each temperature step.

`DP.I.line.error` numeric (**required**): Vector containing the uncertainty on I for each temperature step.

`GC.I.line` numeric (**required**): growth curve for I

`GC.I.slope` numeric (**required**): growth curve parameters for I.

`GC.I.LxTx` numeric (**required**): Lx/tx vector used for I estimation using the growth curve approach.

`GC.I.LxTx.error` numeric (**required**): Error on the Lx/tx vector used for I estimation using the growth curve approach.

`GC.I.doses` numeric (**required**): Doses used for I estimation using the growth curve approach.

`GC.I.names` numeric (**required**): Names of the Lx/Tx vector used for I estimation using the growth curve approach.

`Q.DP` numeric (**required**): Q estimation using the dose plateau approach

`Q.DP.error` numeric (**required**): Uncertainty on the Q estimation using the dose plateau approach

`Q.GC` numeric (**required**): Q estimation using the growth curve approach

`Q.GC.error` numeric (**required**): Uncertainty on the Q estimation using the growth curve approach

`I.DP` numeric (**required**): I estimation using the dose plateau approach

`I.DP.error` numeric (**required**): Uncertainty on the I estimation using the dose plateau approach

`I.GC` numeric (**required**): I estimation using the growth curve approach

`I.GC.error` numeric (**required**): Uncertainty on the I estimation using the growth curve approach

`De.GC,` numeric (**required**): ED (Q+I) estimation using the growth curve approach

`De.GC.error,` numeric (**required**): Uncertainty on the ED (Q+I) estimation using the growth curve approach

`De.DP,` numeric (**required**): ED (Q+I) estimation using the dose plateau approach

`De.DP.error` numeric (**required**): Uncertainty on the ED (Q+I) estimation using the dose plateau approach

`rejection.values`
list (**required**): result of the rejection tests.

`fitting.parameters`
list (with default): list containing the fitting parameters. See details.

`plotting.parameters`
list (with default): list containing the plotting parameters. See details.

Details

Fitting parameters
The fitting parameters are:

`method` character: Fitting method (`LIN`, `EXP`, `EXP+LIN` or `EXP+EXP`).

`fit.weighted` logical: If the fitting is weighted or not.

`fit.use.slope` logical: If the slope of the Q growth curve is reused for the supralinearity correction.

`fit.aDoses.min` numeric: Lowest additive dose used for the fitting.

`fit.aDoses.max` numeric: Highest additive dose used for the fitting.

`fit.rDoses.min` numeric: Lowest regenerative dose used for the fitting.

`fit.rDoses.max` numeric: Highest regenerative dose used for the fitting.

See also analyse_TL.MAAD, calc_TL.MAAD.fit.Q and calc_TL.MAAD.fit.I.

Plotting parameters
The plotting parameters are:

`plot.Tmin` numeric: Lower temperature plotted.

`plot.Tmax` numeric: Higher temperature plotted.

`no.plot` logical: If `TRUE`, the results will not be plotted.

See also analyse_TL.MAAD.

Author(s)

David Strebler

See Also

analyse_TL.MAAD, calc_TL.MAAD.fit.Q, calc_TL.MAAD.fit.I.

---

plot_TL.plateau *plot plateau test result*

Description

This function plots the results for analyse_TL.plateau.

Usage

```
plot_TL.plateau(sample.name, temperatures, names, doses, Lx, Lx.a, Lx.plateau,
  LxTx, LxTx.a, LxTx.plateau, plotting.parameters = list(plateau.Tmin = 0,
  plateau.Tmax = NA, plot.Tmin = 0, plot.Tmax = NA))
```

Arguments

`sample.name`	character (**required**): Sample name.
`temperatures`	numeric (**required**): temperature vector
`names`	character (**required**): Name vector for the additive curves.
`doses`	numeric (**required**): Dose vector for the additive curves.
`Lx`	numeric (**required**): Lx matrix for the additive curves.
`Lx.a`	numeric (**required**): Lx matrix for the average additive curves.
`Lx.plateau`	numeric (**required**): Ln/Lx matrix for the additive curves.
`LxTx`	numeric (**required**): Lx/Tx matrix for the additive curves.
`LxTx.a`	numeric (**required**): Lx/Tx matrix for the average additive curves.
`LxTx.plateau`	numeric (**required**): (Ln/Tn)/(Lx/Tx) matrix for the additive curves.
`plotting.parameters`	list (with default): list containing the plotting parameters. See details.

Details

Plotting parameters
The plotting parameters are:

`plot.Tmin`	numeric: Lowest temperature plotted.
`plot.Tmax`	numeric: Highest temperature plotted.
`no.plot`	logical: If `TRUE`, the results will not be plotted.

See also analyse_TL.MAAD.

Author(s)

David Strebler

See Also

analyse_TL.plateau, calc_TL.MAAD.fit.Q, calc_TL.MAAD.fit.I.

plot_TL.SAR *plots MAAD results*

Description

This function plots the results obtained by the analyse_TL.MAAD function.

Usage

```
plot_TL.SAR(sample.name, sample.position, fitting.parameters = list(fit.method
  = "LIN", fit.weighted = FALSE, fit.rDoses.min = NA, fit.rDoses.max = NA),
  eval.Tmin, eval.Tmax, temperatures, names, names.duplicated, doses, Lx, Tx,
  LxTx, Lx.plateau, Tx.plateau, LxTx.plateau, DP.Q.line, DP.Q.line.error,
  GC.Q.line, GC.Q.LxTx, GC.Q.LxTx.error, GC.Q.slope, Q.DP, Q.DP.error, Q.GC,
  Q.GC.error, TxTn, rejection.values, plotting.parameters = list(plot.Tmin =
  0, plot.Tmax = NA))
```

Arguments

`sample.name`	character (**required**): Sample name.
`sample.position`	integer (**required**): aliquot position.
`fitting.parameters`	list (with default): list containing the fitting parameters. See details.
`eval.Tmin`	integer (**required**): Temperature (°C) of the lower boundary for the signal integration.
`eval.Tmax`	integer (**required**): Temperature (°C) of the upper boundary for the signal integration.
`temperatures`	numeric (**required**): temperature vector
`names`	character (**required**): Name vector for the regenerative curves.
`names.duplicated`	character (**required**): Name vector for the duplicated doses.
`doses`	numeric (**required**): Dose vector for the regenerative curves.
`Lx`	numeric (**required**): Lx matrix for the regenerative curves.
`Tx`	numeric (**required**): Tx matrix for the regenerative curves.
`LxTx`	numeric (**required**): Lx/Tx matrix for the regenerative curves.
`Lx.plateau`	numeric (**required**): Ln/Lx matrix for the regenerative curves.
`Tx.plateau`	numeric (**required**): Tn/Tx matrix for the regenerative curves.
`LxTx.plateau`	numeric (**required**): (Ln/Tn)/(Lx/Tx) matrix for the regenerative curves.
`DP.Q.line`	numeric (**required**): Vector containing the estimation of Q for each T° step.
`DP.Q.line.error`	numeric (**required**): Vector containing the uncertainty on the estimation of Q for each T° step.
`GC.Q.line`	numeric (**required**): growth curve for Q
`GC.Q.LxTx`	numeric (**required**): Lx/Tx vector used for Q estimation using the growth curve approach.

`GC.Q.LxTx.error`
: numeric (**required**): Error on the Lx/tx vector used for Q estimation using the growth curve approach.

`GC.Q.slope` numeric (**required**): growth curve parameters for Q

`Q.DP` numeric (**required**): Q estimation using the dose plateau approach

`Q.DP.error` numeric (**required**): Uncertainty on the Q estimation using the dose plateau approach

`Q.GC` numeric (**required**): Q estimation using the growth curve approach

`Q.GC.error` numeric (**required**): Uncertainty on the Q estimation using the growth curve approach

`TxTn` numeric (**required**): average Tx/Tn value for the regenerative curves.

`rejection.values`
: list (**required**): result of the rejection tests.

`plotting.parameters`
: list (with default): list containing the plotting parameters. See details.

Details

Fitting parameters
The fitting parameters are:

`method` character: Fitting method (`LIN`, `EXP`, `EXP+LIN` or `EXP+EXP`).

`fit.weighted` logical: If the fitting is weighted or not.

`fit.rDoses.min` numeric: Lower regenerative dose used for the fitting.

`fit.rDoses.max` numeric: Higher regenerative dose used for the fitting.

See also calc_TL.SAR.fit.

Plotting parameters
The plotting parameters are:

`plot.Tmin` numeric: Lower temperature plotted.

`plot.Tmax` numeric: Higher temperature plotted.

`no.plot` logical: If `TRUE`, the results will not be plotted.

See also plot_TL.SAR.

Author(s)

David Strebler

`Risoe.BINfileData2TLum.BIN.File` *Convert Risoe.BINfileData object to an TLum.BIN.File object.*

Description

This function convert a Risoe.BINfileData object, created using the package'Luminescence', into a TLum.BIJ.file object, which will be used by this package. The TL package is implemented to only works with its own class of object (TLum.BIN.File, TLum.Analysis and TLum.Data.Curve).

Usage

```
Risoe.BINfileData2TLum.BIN.File(object, k)
```

Arguments

`object`	`Risoe.BINfileData` (**required**): object containing the TL curves used for the ED estimation.
`k`	numeric (**required**): Relative error of the TL signals. Generally, it is between 0.02 and 0.1.

Details

This function use the data from the Risoe.BINFileData and the k specified to create a absolute error matrix. Then it create a new TLum.BIN.File including all the information from the Risoe.BINFileData and the new absolute error matrix. For practical reason, the TLdating package considers the error as random. It means that the systematic component of the error will be ignored.

Author(s)

David Strebler, University of Cologne (Germany),
David Strebler

See Also

TLum.BIN.File2Risoe.BINfileData, TLum.BIN.File2TLum.Analysis and TLum.BIN.File2TLum.Data.Curve.

`RLum.Analysis2TLum.Analysis` *Convert RLum.Analysis object to an TLum.Analysis.*

Description

This function convert a RLum.Analysis object, created using the package'Luminescence', into a TLum.Analysis object, which will be used by this package. The TL package is implemented to only works with its own class of object (TLum.Analysis, TLum.Analysis and TLum.Data.Curve).

Usage

```
RLum.Analysis2TLum.Analysis(object, relative.error)
```

Arguments

`object` RLum.Analysis (**required**): object containing the TL curves used for the ED estimation.

`relative.error` numeric (**required**): Relative error of the TL signals. Generally, it is between 0.02 and 0.1.

Details

This function use the data from the RLum.Analysis and the relative.error specified to create a absolute error matrix. Then it create a new TLum.Analysis including all the information from the RLum.Analysis and the new absolute error matrix. For practical reason, the TLdating package considers the error as random. It means that the systematic component of the error will be ignored.

Author(s)

David Strebler, University of Cologne (Germany),
David Strebler

`script_TL.export` *Script for data export*

Description

This script creates a .binx file from a TLum.Analysis object. It just requires the name of the file and the TLum.Analysis object.

Usage

```
script_TL.export(object, file.name, file.parameters = list(file.extension =
  ".binx", folder.out = "./"))
```

Arguments

`object` TLum.Analysis (**required**): object containing the TL curves to export.

`file.name` character (**required**): Name of the file containing the luminescence data.

`file.parameters` list (with default): list containing the file parameters. See details.

File parameters

The file parameters are:

`file.extension` character (with default): extension of the file containing the luminescence data (.bin or .binx)

`folder.out` character (with default): Folder containing the file with the luminescene data.

Value

This function returns a TLum.Analysis object.

Author(s)

David Strebler, University of Cologne (Germany).

See Also

write_R2BIN, TLum.BIN.File2Risoe.BINfileData, TLum.Analysis2TLum.BIN.File.

`script_TL.import` *Script for data pretreatment*

Description

This script opens a .binx file and creates a TLum.Analysis object from it. It just requires the name of the file with the TL curves and the relative error on the measurements. It extracts the TL curves and updates the data types.

Usage

```
script_TL.import(file.name, k = 1, protocol = "Unknown",
  file.parameters = list(file.extension = ".binx", folder.in = "./"),
  plotting.parameters = list(plot.Tmin = 0, plot.Tmax = NA, no.plot = FALSE))
```

Arguments

`file.name`	character (**required**): Name of the file containing the luminescence data.
`k`	numeric (with default): Corrective factor for estimating the uncertainties using a poisson distribution.
`protocol`	character (**required**): Measurment protocol used.
`file.parameters`	list (with default): list containing the file parameters. See details.
`plotting.parameters`	list (with default): list containing the plotting parameters. See details.

Details

Plotting parameters
The plotting parameters are:

`plot.Tmin` numeric: Lowest temperature plotted.

`plot.Tmax` numeric: Highest temperature plotted.

`no.plot` logical: If `TRUE`, the results will not be plotted.

See also plot_extract.TL.

File parameters
The file parameters are:

`file.extension` character (with default): extension of the file containing the luminescence data (.bin or .binx)

`folder.in` character (with default): Folder containing the file with the luminescene data.

Value

This function returns a `TLum.Analysis` object.

Author(s)

David Strebler, University of Cologne (Germany).

See Also

read_BIN2R, Risoe.BINfileData2TLum.BIN.File, TLum.BIN.File2TLum.Analysis, mod_extract.TL, mod_update.dType.

---

script_TL.MAAD *Script for the MAAD protocol*

Description

This script calls a series of data functions before estimating the ED using the MAAD protocol. It just requires the name of the files with the TL curves, the relative error on the measurement and the temperature boundaries for the signal integration.

Usage

```
script_TL.MAAD(file.name, eval.Tmin, eval.Tmax, k = 1, remove.discs = NULL,
  file.parameters = list(file.extension = ".binx", folder.in = "./",
  folder.out = "./"), aligning.parameters = list(peak.Tmin = NULL, peak.Tmax =
  NULL, no.testdose = FALSE), fitting.parameters = list(fit.method = "LIN",
  fit.weighted = FALSE, fit.use.slope = FALSE, fit.aDoses.min = 0,
  fit.aDoses.max = NA, fit.rDoses.min = 0, fit.rDoses.max = NA),
  plotting.parameters = list(plot.Tmin = 0, plot.Tmax = NA, no.plot = FALSE),
  rejection.criteria = list(testdose.error = 10, paleodose.error = 10))
```

Arguments

`file.name`	character (**required**): Name of the file containing the luminescence data.
`eval.Tmin`	integer (**required**): Temperature (°C) of the lower boundary for the signal integration.
`eval.Tmax`	integer (**required**): Temperature (°C) of the upper boundary for the signal integration.
`k`	numeric (with default): Corrective factor for estimating the uncertainties using a poisson distribution.
`remove.discs`	numeric (with default): list containing the position of the aliquots to shall be removed.
`file.parameters`	list (with default): list containing the input/output parameters. See details.
`aligning.parameters`	list (with default): list containing the aligning parameters. See details.
`fitting.parameters`	list (with default): list containing the fitting parameters. See details.

`plotting.parameters`
: list (with default): list containing the plotting parameters. See details.

`rejection.criteria`
: list (with default): list containing the rejection criteria (in %). See details.

Details

File parameters
The file parameters are:

`file.extension` character (with default): extension of the file containing the luminescence data (.bin or .binx)

`folder.in` character (with default): Folder containing the file with the luminescene data.

`folder.out` character (with default): Folder containing the file with the new luminescene data.

see also script_TL.pretreatment.

Aligning parameters
The aligning parameters are:

`peak.Tmin` numeric: Lower boundary for looking for the peak maximum position.

`peak.Tmax` numeric: Upper boundary for looking for the peak maximum position.

`no.testdose` logical: If `TRUE`, the function will use the Lx curves rather than the Tx curves as reference for the peak maximum position.

Fitting parameters
The fitting parameters are:

`method` character: Fitting method (`LIN`, `EXP`, `EXP+LIN` or `EXP+EXP`).

`fit.weighted` logical: If the fitting is weighted or not.

`fit.use.slope` logical: If the slope of the Q growth curve is reused for the supralinearity correction.

`fit.aDoses.min` numeric: Lowest additive dose used for the fitting.

`fit.aDoses.max` numeric: Highest additive dose used for the fitting.

`fit.rDoses.min` numeric: Lowest regenerative dose used for the fitting.

`fit.rDoses.max` numeric: Highest regenerative dose used for the fitting.

See also calc_TL.MAAD.fit.Q and calc_TL.MAAD.fit.I.

Plotting parameters
The plotting parameters are:

`plot.Tmin` numeric: Lowest temperature plotted.

`plot.Tmax` numeric: Highest temperature plotted.

`no.plot` logical: If `TRUE`, the results will not be plotted.

See also plot_TL.MAAD.

Rejection criteria
The rejection criteria are:

`testdose.error` numeric: Maximum error accepted on the testdose signal within the selected temperature interval.

`paleodose.error` numeric: Maximum error accepted on the palaeodose signal within the selected temperature interval.

Value

This function plots the results from the differents functions called using: plot_extract.TL, plot_remove.preheat plot_substract.background plot_align.peaks and plot_TL.MAAD.

This function saves a file containing the luminescence data after the pretreatment in the specified folder.

Finally, it also provides an list containing:

`De.GC` data.frame: Results obtained with the dose plateau approach and their uncertainties (`De`, `De.error`

`De.DP` data.frame: Results obtained with the growth curve approach and their uncertainties (`De`, `De.error`

Author(s)

David Strebler, University of Cologne (Germany),
David Strebler

See Also

read_BIN2R, Risoe.BINfileData2TLum.BIN.File, mod_extract.TL, mod_update.dType, mod_remove.aliquot, mod_remove.preheat, mod_substract.background, mod_align.peaks, write_R2BIN, TLum.BIN.File2TLum.Analysis and analyse_TL.MAAD.

`script_TL.plateau` *Script for the plateau test*

Description

This script calls a series of data pretreatment functions before performing the plateau test. It just requires the name of the file with the TL curves and the relative error on the measurements.

Usage

```
script_TL.plateau(file.name, k = 1, remove.discs = NULL,
  file.parameters = list(file.extension = ".binx", folder.in = "./",
  folder.out = "./"), aligning.parameters = list(peak.Tmin = NULL, peak.Tmax =
  NULL, no.testdose = FALSE), plotting.parameters = list(plot.Tmin = 0,
  plot.Tmax = NA, plateau.Tmin = 0, plateau.Tmax = 0, no.plot = FALSE))
```

Arguments

`file.name` character (**required**): Name of the file containing the luminescence data.

`k` numeric (with default): Corrective factor for estimating the uncertainties using a poisson distribution.

`remove.discs` numeric (with default): list containing the position of the aliquots that shall be removed.

`file.parameters` list (with default): list containing the file parameters. See details.

`aligning.parameters` list (with default): list containing the aligning parameters. See details.

`plotting.parameters` list (with default): list containing the plotting parameters. See details.

Details

Aligning parameters
The aligning parameters are:

`peak.Tmin` numeric: Lower boundary for looking for the peak maximum position.

`peak.Tmax` numeric: Upper boundary for looking for the peak maximum position.

`no.testdose` logical: If `TRUE`, the function will use the Lx curves rather than the Tx curves as reference for the peak maximum position.

File parameters
The file parameters are:

`file.extension` character (with default): extension of the file containing the luminescence data (.bin or .binx)

`folder.in` character (with default): Folder containing the file with the luminescene data.

`folder.out` character (with default): Folder containing the file with the new luminescene data.

see also script_TL.pretreatment.

Plotting parameters
The plotting parameters are:

`plot.Tmin` numeric: Lowest temperature plotted.

`plot.Tmax` numeric: Highest temperature plotted.

`no.plot` logical: If `TRUE`, the results will not be plotted.

See also plot_TL.MAAD.

Value

This function plots the results from the differents functions called using: plot_extract.TL, plot_remove.preheat plot_substract.background plot_align.peaks and plot_TL.plateau.

Author(s)

David Strebler, University of Cologne (Germany).

See Also

read_BIN2R, Risoe.BINfileData2TLum.BIN.File, mod_extract.TL, mod_update.dType, mod_remove.aliquot, mod_remove.preheat, mod_substract.background, mod_align.peaks, write_R2BIN, TLum.BIN.File2TLum.Analysis and analyse_TL.plateau.

---

script_TL.pretreatment

Script for data pretreatment

Description

This script call a series of data pretreatment functions for TL dating. It only requires the name of the files with the TL curves and the relative error on the measurement.

Usage

```
script_TL.pretreatment(file.name, k = 1, remove.discs = NULL,
  file.parameters = list(file.extension = ".binx", folder.in = "./",
  folder.out = "./"), aligning.parameters = list(peak.Tmin = NULL, peak.Tmax =
  NULL, no.testdose = FALSE), plotting.parameters = list(plot.Tmin = 0,
  plot.Tmax = NA, no.plot = FALSE))
```

Arguments

`file.name` character (**required**): Name of the file containing the luminescence data.

`k` numeric (with default): Corrective factor for estimating the uncertainties using a poisson distribution.

`remove.discs` numeric (with default): list containing the position of the aliquots to remove.

`file.parameters` list (with default): list containing the file parameters. See details.

`aligning.parameters` list (with default): list containing the aligning parameters. See details.

`plotting.parameters` list (with default): list containing the plotting parameters. See details.

Details

Aligning parameters
The aligning parameters are:

`peak.Tmin` numeric: Lower boundary for looking at the peak maximum position.

`peak.Tmax` numeric: Upper boundary for looking at the peak maximum position.

`no.testdose` logical: If `TRUE`, the function will use the Lx curves rather the Tx curves as reference for the peak maximum position.

Plotting parameters
The plotting parameters are:

`plot.Tmin` numeric: Lower temperature plotted.

`plot.Tmax` numeric: Higher temperature plotted.

`no.plot` logical: If `TRUE`, the results will not be plotted.

See also plot_TL.MAAD.

File parameters
The plotting parameters are:

`file.extension` character (with default): extension of the file containing the luminescence data (.bin or .binx)

`folder.in` character (with default): Folder containing the file with the luminescene data.

`folder.out` character (with default): Folder containing the file with the new luminescene data.

see also mod_update.dType.

Value

This function return a `TLum.Analysis` where the preheat were removed, the background substract and the peaks aligned. Its save the result as a .binx file il the specified folder. And, its plots the results from the differents functions called using: plot_extract.TL, plot_remove.preheat, plot_substract.background and plot_align.peaks.

Author(s)

David Strebler, University of Cologne (Germany),
David Strebler

See Also

read_BIN2R, Risoe.BINfileData2TLum.BIN.File, mod_extract.TL, mod_update.dType, mod_remove.aliquot, mod_remove.preheat, mod_substract.background, mod_align.peaks, write_R2BIN.

`script_TL.SAR` *Easy script for the SAR protocol*

Description

This function provides and estimation of the ED using the SAR protocol. It only requires the name of the files with the TL curves, the relative error on the measurements and the temperature boundaries for the signal integration. Extra parameters can be provided to improve the ED estimation.

Usage

```
script_TL.SAR(file.name, eval.Tmin, eval.Tmax, k = 1, remove.discs = NULL,
  file.parameters = list(file.extension = ".binx", folder.in = "./",
  folder.out = "./"), aligning.parameters = list(peak.Tmin = NULL, peak.Tmax =
  NULL, no.testdose = FALSE), fitting.parameters = list(fit.method = "LIN",
  fit.weighted = FALSE, fit.rDoses.min = 0, fit.rDoses.max = NA),
  plotting.parameters = list(plot.Tmin = 0, plot.Tmax = NA, no.plot = FALSE),
  rejection.criteria = list(recycling.ratio = 10, recuperation.rate = 10,
  testdose.error = 10, paleodose.error = 10))
```

Arguments

`file.name` character (**required**): Name of the file containing the luminescence data.

`eval.Tmin` integer (**required**): Temperature (°C) of the lower boundary for the signal integration.

`eval.Tmax` integer (**required**): Temperature (°C) of the upper boundary for the signal integration.

`k` numeric (with default): Corrective factor for estimating the uncertainties using a poisson distribution.

`remove.discs` numeric (with default): list containing the position of the aliquots that shall be removed

`file.parameters` list (with default): list containing the input/output parameters. See details.

`aligning.parameters` list (with default): list containing the aligning parameters. See details.

`fitting.parameters` list (with default): list containing the fitting parameters. See details.

`plotting.parameters` list (with default): list containing the plotting parameters. See details.

`rejection.criteria` list (with default): list containing the rejection criteria (in %). See details.

Details

File parameters

The file parameters are:

`file.extension` character (with default): extension of the file containing the luminescence data (.bin or .binx)

`folder.in` character (with default): Folder containing the file with the luminescene data.

`folder.out` character (with default): Folder containing the file with the new luminescene data.

see also script_TL.pretreatment.

Aligning parameters

The aligning parameters are:

`peak.Tmin` numeric: Lower boundary for looking for the peak maximum position.

`peak.Tmax` numeric: Upper boundary for looking for the peak maximum position.

`no.testdose` logical: If `TRUE`, the function will use the Lx curves rather the Tx curves as reference for the peak maximum position.

Rejection criteria
The rejection criteria are:

`recycling.ratio` numeric: Maximum recycling ratio accepted (in %).

`recuperation.rate` numeric: Maximum recuparation rate accepted (in %).

`paleodose.error` numeric: Maximum error accepted on the regenative signals (in %).

`testdose.error` numeric: Maximum error accepted on the testdose signals (in %).

Fitting parameters
The fitting parameters are:

`method` character: Fitting method (`LIN`, `EXP`, `EXP+LIN` or `EXP+EXP`).

`fit.weighted` logical: If the fitting is weighted or not.

`fit.rDoses.min` numeric: Lowest regenerative dose used for the fitting.

`fit.rDoses.max` numeric: Highest regenerative dose used for the fitting.

See also calc_TL.SAR.fit.

Plotting parameters
The plotting parameters are:

`plot.Tmin` numeric: Lower temperature plotted.

`plot.Tmax` numeric: Higher temperature plotted.

`no.plot` logical: If `TRUE`, the results will not be plotted.

See also plot_TL.SAR.

Value

This function plots the results from the differents functions called using: plot_extract.TL, plot_remove.preheat plot_substract.background plot_align.peaks and plot_TL.SAR.

This function saves a file containing the luminescence data after the pretreatment in the specified folder.

Finally, it also provides an list containing:

`De.GC` list: Results obtained with the dose plateau approach and their uncertainties (`De`, `De.error`)

`De.DP` list: Results obtained with the growth curve approach and their uncertainties (`De`, `De.error`)

Author(s)

David Strebler, University of Cologne (Germany).

See Also

read_BIN2R, Risoe.BINfileData2TLum.BIN.File, mod_extract.TL, mod_update.dType, mod_remove.aliquot, mod_remove.preheat, mod_substract.background, mod_align.peaks, write_R2BIN, TLum.BIN.File2TLum.Analysis and analyse_TL.SAR.

`template_DRAC4brick` *Creation of a template for use_DRAC4brick input.*

Description

This function generates a template that can be fill in and use with use_DRAC4brick.

Usage

```
template_DRAC4brick()
```

Author(s)

David Strebler

`template_DRAC4cave` *Creation of a template for use_DRAC4cave input.*

Description

This function generates a template that can be fill in and use with use_DRAC4cave.

Usage

```
template_DRAC4cave()
```

Author(s)

David Strebler

`template_DRAC4ceramic` *Creation of a template for use_DRAC4flint input.*

Description

This function generates a template that can be fill in and use with use_DRAC4ceramic.

Usage

```
template_DRAC4ceramic()
```

Author(s)

David Strebler

template_DRAC4flint *Creation of a template for use_DRAC4flint input.*

Description

This function generates a template that can be fill in and use with use_DRAC4flint.

Usage

```
template_DRAC4flint()
```

Author(s)

David Strebler

TLum-class *Class* "TLum"

Description

Abstract class for data in the package Luminescence

Note

`TLum` is a virtual class.

Author(s)

David Strebler

TLum.Analysis-class *Class* "TLum.Analysis"

Description

Object class containing analysis data for protocol analysis.

Usage

```
## S4 method for signature 'TLum.Analysis'
show(object)

set_TLum.Analysis(records, protocol, history, plotHistory)

## S4 method for signature 'list,ANY,character,list'
set_TLum.Analysis(records, protocol,
  history, plotHistory)

get_TLum.Analysis(object, record.id, recordType, curveType, TLum.type,
  get.index, keep.object = FALSE)

## S4 method for signature 'TLum.Analysis'
get_TLum.Analysis(object, record.id, recordType,
  curveType, TLum.type, get.index, keep.object = FALSE)
```

Arguments

`object`	TLum.Analysis: an object of class TLum.Analysis.
`records`	list: list of TLum.Data.Curve objects
`protocol`	character: protocol type for analysis object.
`history`	character: Vector containing the previous modification made on the data set.
`plotHistory`	list: Data for plotting the evolution of the data set.
`record.id`	numeric: IDs of specific records.
`recordType`	character: record type.
`curveType`	character: curve type.
`TLum.type`	character: TLum object type.
`get.index`	logical: return a numeric vector with the index of each element in the TLum.Analysis object.
`keep.object`	logical: return a TLum.Analysis object.

Slots

`protocol` character: Protocol used for the analysis.

`records` list: TLum.Data.Curve included in the analysis.

`history` character: Vector containing the previous modification made on the data set.

`plotHistory` list: Data for plotting the evolution of the data set.

Note

The code and the structure of this class is based on the RLum.Analysis class from the Luminescence package.

Author(s)

David Strebler

`TLum.Analysis2RLum.Analysis` *Convert RLum.Analysis object to an TLum.Analysis.*

Description

This function convert a TLum.Analysis object into a RLum.Analysis object, from the 'Luminescence' package. The 'TLdating' package is implemented to only works with its own class of object (TLum.Analysis, TLum.Analysis and TLum.Data.Curve).

Usage

```
TLum.Analysis2RLum.Analysis(object)
```

Arguments

`object`	`TLum.Analysis` (**required**): object containing the TL curves used for the ED estimation.

Details

This function use the data from the TLum.Analysis to create a new RLum.Analysis. During the process, all information relative to the uncertainties and stored in the TLum.Analysis object are lost. The original data-type of each luminescence curve is also restored.

Author(s)

David Strebler, University of Cologne (Germany),
David Strebler

`TLum.Analysis2TLum.BIN.File` *Convert TLum.Analysis object to an TLum.BIN.File object.*

Description

This function convert a TLum.BIN.File in a TLum.Analysis object. A TLum.Analysis object is a list of TLum.Data.Curve object. It is possible to specify which luminescence curves will be keeped.

Usage

```
TLum.Analysis2TLum.BIN.File(object)
```

Arguments

`object`	`TLum.Analysis` (**required**): object containing the luminescence curves.

Value

This function will return a TLum.BIN.File object.

Author(s)

David Strebler, University of Cologne (Germany),
David Strebler

See Also

TLum.Analysis, TLum.BIN.File, TLum.Data.Curve and TLum.BIN.File2TLum.Data.Curve.

TLum.BIN.File-class *Class* "TLum.BIN.File"

Description

Class for luminescence curves data.

Usage

```
## S4 method for signature 'TLum.BIN.File'
show(object)

set_TLum.BIN.File(METADATA, DATA, ERROR, .RESERVED)

## S4 method for signature 'data.frame,list,list'
set_TLum.BIN.File(METADATA, DATA, ERROR,
  .RESERVED)

get_TLum.BIN.File(object)

## S4 method for signature 'TLum.BIN.File'
get_TLum.BIN.File(object)
```

Arguments

`object`	TLum.BIN.File: an object of class 'TLum.BIN.File'.
`METADATA`	data.frame: Object containing the meta information for each curve.
`DATA`	list: Object containing numeric vector with count data.
`ERROR`	list: Object containing numeric vector with count data absolute uncertainty.
`.RESERVED`	list: Object containing list of undocumented raw values for internal use only.

Slots

`METADATA` data.frame: Object containing the meta information for each curve.

`DATA` list: Object containing numeric vector with count data.

`ERROR` list: Object containing numeric vector with count data absolute uncertainty.

`.RESERVED` list: Object containing list of undocumented raw values for internal use only.

Note

The code and the structure of this class is based on the Risoe.BINfileData class from the Luminescence package.

Author(s)

David Strebler

TLum.BIN.File2Risoe.BINfileData

Convert TLum.BIN.File object to an Risoe.BINfileData object.

Description

This function convert TLum.BIN.File object into a Risoe.BINfileData object that is usable by the Luminescence package.

Usage

```
TLum.BIN.File2Risoe.BINfileData(object)
```

Arguments

`object`	`TLum.BIN.File` (**required**): object containing the TL curves used for the ED estimation.

Value

This function return an Risoe.BINfileData containing all information previously stored in the TLum.BIN.File except the uncertainties matrix. To avoid conflicts with other software, the original data type of each curves is restored.

Author(s)

David Strebler

TLum.BIN.File2TLum.Analysis

Convert TLum.BIN.File object to an TLum.Analysis object.

Description

This function convert a TLum.BIN.File in a TLum.Analysis object. A TLum.Analysis object is a list of TLum.Data.Curve object. It is possible to specify which luminescence curves will be keeped.

Usage

```
TLum.BIN.File2TLum.Analysis(object, protocol = "unknown",
  rec_duringPH = TRUE, rec_ramp2PH = TRUE)
```

Arguments

`object`	`TLum.BIN.File` (**required**): object containing the luminescence curves.
`protocol`	character (with default): protocol used.
`rec_duringPH`	logical (with default): Indicate if the signal was record during the preheat plateau.
`rec_ramp2PH`	logical (with default): Indicate if the signal was record during the ramp up to the preheat temperature.

Value

This function will return a TLum.Analysis object.

Author(s)

David Strebler, University of Cologne (Germany),
David Strebler

See Also

TLum.Analysis, TLum.BIN.File, TLum.Data.Curve and TLum.BIN.File2TLum.Data.Curve.

`TLum.BIN.File2TLum.Data.Curve`

Convert an element from a TLum.BIN.File object into a TLum.Data.Curve objet

---

Description

This function extract a curve from a TLum.BIN.File object and convert it into a TLum.Data.Curve objet. The extract element can be identify either by its id or by its position, run and set.

Usage

```
TLum.BIN.File2TLum.Data.Curve(object, id, pos, run, set, rec_duringPH = TRUE,
  rec_ramp2PH = TRUE)
```

Arguments

`object`	`TLum.BIN.File` (**required**): object containing the luminescence curves.
`id`	integer (with default): id of the curve.
`pos`	integer (with default): position of the curve.
`run`	integer (with default): run of the curve.
`set`	integer (with default): set of the curve.
`rec_duringPH`	logical (with default): Indicate if the signal was record during the preheat plateau.
`rec_ramp2PH`	logical (with default): Indicate if the signal was record during the ramp up to the preheat temperature.

Details

The element that is extracted to be converted into a TLum.Data.Curve objet can be identify either by its id or by its position, run and set.

Value

This function return a TLum.Data.Curve objet.

TLum.Data-class *Class* "TLum.Data"

Description

Generalized virtual data class for luminescence data.

Note

The code and the structure of this class is based on the RLum.Data class from the Luminescence package.

Author(s)

David Strebler

TLum.Data.Curve-class *Class* "TLum.Data.Curve"

Description

Class for luminescence curve data.

Usage

```
## S4 method for signature 'TLum.Data.Curve'
show(object)

set_TLum.Data.Curve(recordType, curveType, metadata, temperatures, data, error,
  analysis, .RESERVED)

## S4 method for signature 'ANY'
set_TLum.Data.Curve(recordType, curveType, metadata,
  temperatures, data, error, analysis, .RESERVED)

get_TLum.Data.Curve(object, ref)

## S4 method for signature 'ANY'
get_TLum.Data.Curve(object, ref)
```

Arguments

`object`	TLum.Data.Curve: an object of class 'TLum.Data.Curve'.
`recordType`	character: record type.
`curveType`	character: curve type
`metadata`	list: metadata elements.
`temperatures`	numeric: Object containing numeric vector with temperature data.
`data`	numeric: Object containing numeric vector with count data
`error`	numeric: Object containing numeric vector with count data absolute uncertainty.
`analysis`	list: data produced by analysis function.
`.RESERVED`	list: Object containing list of undocumented raw values for internal use only.
`ref`	character: name of the wanted element.

Slots

`recordType` character: record type.

`curveType` character: curve type

`metadata` list: metadata elements.

`temperatures` numeric: Object containing numeric vector with temperature data.

`data` numeric: Object containing numeric vector with count data

`error` numeric: Object containing numeric vector with count data absolute uncertainty.

`analysis` list: data produced by analysis function.

`.RESERVED` list: Object containing list of undocumented raw values for internal use only.

Note

The code and the structure of this class is based on the RLum.Data.Curve class from the Luminescence package.

Author(s)

David Strebler

`TLum.Results-class` *Class* `"TLum.Results"`

Description

Object class contains results data from functions.

Usage

```
## S4 method for signature 'TLum.Results'
show(object)

set_TLum.Results(originator, data, plotData)

## S4 method for signature 'ANY,list'
set_TLum.Results(originator, data, plotData)

get_TLum.Results(object, ref)

## S4 method for signature 'TLum.Results'
get_TLum.Results(object, ref)
```

Arguments

`object`	TLum.Results: object to be evaluated.
`originator`	character: : contains the name of the calling function.
`data`	list: the data to be stored in the object.
`plotData`	list: a list containing the data required for plotting.
`ref`	character: name of the 'data' slot to be returned.

Slots

`originator` character: contains the name of the calling function (the function that produces this object).

`data` list: a list containing the data to be stored in the object.

`plotData` list: a list containing the data required for plotting.

Note

The code and the structure of this class is based on the RLum.Results class from the Luminescence package.

Author(s)

David Strebler

`use_DRAC4brick` *Calculation of the dose rate for a brick sample coming from a cave*

Description

This function allows to estimating the dose rate for a grain which was surrounded by brick and mortar. It call the use_DRAC and calc_CosmicDoseRate functions from the R package 'Luminescence'. The function 'use_DRAC' is only compatible wth DRAC version 1.1.

Usage

```
use_DRAC4brick(data, notification = TRUE)
```

Arguments

`data`	list: data object create throught the function template_DRAC4brick.
`notification`	logical (default): set to `FALSE` if you don't want to see the legal notification.

Value

This function return a TLum.Results object containing the Age estimation, the dose rates (total, internal, external, environmental, alpha, beta, gamma and cosmic), the equivalent dose used and their uncertainties.

Author(s)

David Strebler

`use_DRAC4cave` *Calculation of the dose rate for a sediment sample coming from a cave*

Description

This function allows to estimating the dose rate for a grain which was surrounded by sediment and rocks. It call the use_DRAC and calc_CosmicDoseRate functions from the R package 'Luminescence'. The function 'use_DRAC' is only compatible wth DRAC version 1.1.

Usage

```
use_DRAC4cave(data, notification = TRUE)
```

Arguments

`data`	list: data object create throught the function template_DRAC4cave.
`notification`	logical (default): set to `FALSE` if you don't want to see the legal notification.

Value

This function return a TLum.Results object containing the Age estimation, the dose rates (total, internal, external, environmental, alpha, beta, gamma and cosmic), the equivalent dose used and their uncertainties.

Author(s)

David Strebler

`use_DRAC4ceramic` *Calculation of the dose rate for a ceramic sample*

Description

This function allows to estimating the dose rate for a ceramic which was surrounded by sediment. It call the use_DRAC and calc_CosmicDoseRate functions from the R package 'Luminescence'. The function 'use_DRAC' is only compatible wth DRAC version 1.1.

Usage

```
use_DRAC4ceramic(data, notification = TRUE)
```

Arguments

`data`	list: data object create throught the function template_DRAC4flint.
`notification`	logical (default): set to `FALSE` if you don't want to see the legal notification.

Value

This function return a TLum.Results object containing the Age estimation, the dose rates (total, internal, external, environmental, alpha, beta, gamma and cosmic), the equivalent dose used and their uncertainties.

Author(s)

David Strebler

`use_DRAC4flint` *Calculation of the dose rate for a flint*

Description

This function allows to estimating the dose rate for a burnt flint object which was surrounded by sediment. It call the use_DRAC and calc_CosmicDoseRate functions from the R package 'Luminescence'. The function 'use_DRAC' is only compatible wth DRAC version 1.1.

Usage

```
use_DRAC4flint(data, notification = TRUE)
```

Arguments

`data`	list: data object create throught the function template_DRAC4flint.
`notification`	logical (default): set to `FALSE` if you don't want to see the legal notification.

Value

This function return a TLum.Results object containing the Age estimation, the dose rates (total, internal, external, environmental, alpha, beta, gamma and cosmic), the equivalent dose used and their uncertainties.

Author(s)

David Strebler

Index