Package 'RLumCarlo'

September 17, 2025

Type Package

Title Monte-Carlo Methods for Simulating Luminescence Phenomena

Version 0.1.10

Date 2025-09-17

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

Description A collection of functions to simulate luminescence production in dosimetric materials using Monte Carlo methods. Implemented are models for delocalised transitions (e.g., Chen and McKeever (1997) <doi:10.1142/2781>), localised transitions (e.g., Pagonis et al. (2019) <doi:10.1016/j.jlumin.2018.11.024>) and tunnelling transitions (Jain et al. (2012) <doi:10.1088/0953-8984/24/38/385402> and Pagonis et al. (2019) <doi:10.1016/j.jlumin.2018.11.024>). Supported stimulation methods are thermal luminescence (TL), continuous-wave optically stimulated luminescence (CW-OSL), linearly-modulated optically stimulated luminescence (LM-OSL), linearly-modulated infrared stimulated luminescence (LM-IRSL), and isothermal luminescence (ITL or ISO-TL).

Contact Package Developer Team <sebastian.kreutzer@uni-heidelberg.de>

License GPL-3

BugReports https://github.com/R-Lum/RLumCarlo/issues

Depends R (>= 4.4),
utils,
magrittr

URL https://CRAN.R-project.org/package=RLumCarlo

LinkingTo Rcpp (>= 1.1.0),
RcppArmadillo (>= 14.6.0)

Imports abind (>= 1.4-8),
doParallel (>= 1.0.17),
foreach (>= 1.5.2),
khroma (>= 1.16.0),
methods,
parallel,
Rcpp (>= 1.1.0),
scatterplot3d (>= 0.3),
stats

Suggests spelling (>= 2.3.2),
R.rsp (>= 0.46.0),
testthat (>= 3.2.3)

Encoding UTF-8

Language en-GB

VignetteBuilder R.rsp

RoxygenNote 7.3.3

Contents

`RLumCarlo-package` *Monte-Carlo Methods for Simulating Luminescence Phenomena.*

Description

A collection of functions to simulate luminescence production in dosimetric materials using Monte-Carlo methods. Implemented are models for delocalised, localised and tunnelling transitions. Supported stimulation modes are TL, CW-OSL, LM-OSL, LM-IRSL, and ITL (ISO-TL).

Details

Funding

The development of RLumCarlo benefited from the support by various funding bodies:

- The initial work by Johannes Friedrich, Sebastian Kreutzer and Christoph Schmidt was supported by the Deutsche Forschungsgemeinschaft (DFG, 2015–2018, SCHM 3051/4-1, "Modelling quartz luminescence signal dynamics relevant for dating and dosimetry", SCHM 3051/4-1).
- Later work (2018-2019) was secured through the project "ULTIMO: Unifying Luminescence Models of quartz and feldspar DAAD: Deutscher Akademischer Austauschdienst (German Academic Exchange Service). Framework: DAAD PPP USA 2018, ID: 57387041.

- The work of Sebastian Kreutzer as maintainer of the package was supported by LabEx LaSCArBx (ANR - n. ANR-10-LABX-52) between 2017 and 2019.
- From 2020, Sebastian Kreutzer received funding from the European Union's Horizon 2020 research and innovation programme under the Marie Skłodowska-Curie grant agreement No 844457.

Author(s)

Johannes Friedrich, University of Bayreuth (Germany),
Sebastian Kreutzer, Institute of Geography, Heidelberg University (Germany)
Vasilis Pagonis, McDaniel College Westminster (MD, USA),
Christoph Schmidt, University of Bayreuth (Germany),
Ena Rajovic, University of Bayreuth (Germany),
Alex Roy Duncan, University of Bayreuth (Germany),
Christian Laag, Institut de Physique du Globe de Paris, Université de Paris (France)

References

Kreutzer, S., Friedrich, J., Pagonis, V., Laag, C., Rajovic, E., Schmidt, C., 2021. RLumCarlo: Simulating Cold Light using Monte Carlo Methods. The R Journal 13, 351–365. doi:10.32614/RJ2021043

Pagonis, V., Friedrich, J., Discher, M., Müller-Kirschbaum, A., Schlosser, V., Kreutzer, S., Chen, R., Schmidt, C., 2019. Excited state luminescence signals from a random distribution of defects - A new Monte Carlo simulation approach for feldspar. Journal of Luminescence 207, 266–272. doi:10.1016/j.jlumin.2018.11.024

See Also

Useful links:

- https://CRAN.R-project.org/package=RLumCarlo
- Report bugs at https://github.com/R-Lum/RLumCarlo/issues

---

create_ClusterSystem *Create dosimetric cluster system*

Description

In order to allow interaction of an spatial a correlation clusters in RLumCarlo, first a dosimetric system needs to be created in a three-dimensional space, which is the purpose of this function.

Usage

```
create_ClusterSystem(n = 100, h = 0.5, plot = FALSE, ...)
```

Arguments

n	numeric (*with default*): number of clusters to be created in an arbitrary 3-dimensional cube. x, y, z distances range between 0 and 1.
h	numeric (*with default*): numeric scalar the cut the cluster tree using stats::cutree. The number must range between 0 and 1.
plot	logical (*with default*): enables/disables plot output
...	further arguments to be passed to the plot output

Value

The function returns a list of class `RLumCarlo_clusters` consisting of numeric vector of cluster groups and a matrix of the cluster positions in the arbitrary space. If `plot = TRUE` the system is displayed using scatterplot3d::scatterplot3d

Function version

0.1.0

How to cite

Kreutzer, S., 2025. create_ClusterSystem(): Create dosimetric cluster system. Function version 0.1.0. In: Friedrich, J., Kreutzer, S., Pagonis, V., Schmidt, C., 2025. RLumCarlo: Monte-Carlo Methods for Simulating Luminescence Phenomena. R package version 0.1.10. https://CRAN.R-project.org/package=RLumCarlo

Author(s)

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

See Also

stats::dist, stats::hclust, stats::cutree

Examples

```
create_ClusterSystem(n = 10, plot = TRUE)
```

---

plot_RLumCarlo *Plot RLumCarlo Monte-Carlo Simulation Results*

Description

Visualise 'RLumCarlo' modelling results without extracting the values manually. Typically visualised are the averaged signal or the number of remaining electrons, with a polygon indicating modelling uncertainties.

Usage

```
plot_RLumCarlo(
  object,
  plot_value = "mean",
  plot_uncertainty = "range",
  FUN = NULL,
  norm = FALSE,
  add = FALSE,
  ...
)
```

Arguments

`object`	list of class `RLumCarlo_Model_Output` (**required**): input object to be plotted, usually the required input object is generated by one of the functions starting with `run_`. Alternatively a list of such objects can be provided.
`plot_value`	character (*with default*): type of curve value to be displayed. Allowed are `mean` (the default) and `sum` (meaningful if different systems are combined). `NULL` disables the value visualisation.
`plot_uncertainty`	character (*with default*): type of the displayed uncertainty. Allowed values are `range`, `sd` (standard deviation) and `var` (variance). `NULL` disables the uncertainty visualisation.
`FUN`	function (*optional*): own function that can be applied to the y-values before normalisation and plotting
`norm`	logical (*with default*): normalise curve to the highest intensity value
`add`	logical (*with default*): allows overplotting of results by adding curves to an existing plot. This argument is handled automatically if `object` is of type list
`...`	further argument, that can be passed to control the plot output largely following the argument names in graphics::plot.default. Currently supported are: `xlab`, `ylab`, `xlim`, `ylim`, `main`, `lwd`, `type`, `pch`, `lty`,`col`, `grid`, `legend`. The arguments `lwd`, `type`, `pch`, `lty`, `col` can be provided as a vector if `object` is a list

Details

For colouring the curves, the package khroma::khroma-package is used to provide colours that can be best distinguished, in particular by colour-blind users.

Value

This function returns a graphical output which is the visualisation of the modelling output.

Function version

0.1.0

How to cite

Kreutzer, S., Friedrich, J., 2025. plot_RLumCarlo(): Plot RLumCarlo Monte-Carlo Simulation Results. Function version 0.1.0. In: Friedrich, J., Kreutzer, S., Pagonis, V., Schmidt, C., 2025. RLumCarlo: Monte-Carlo Methods for Simulating Luminescence Phenomena. R package version 0.1.10. https://CRAN.R-project.org/package=RLumCarlo

Author(s)

Sebastian Kreutzer, Institute of Geography, Heidelberg University (Germany)
Johannes Friedrich, University of Bayreuth (Germany)

Examples

```
## plain plot
DELOC <- run_MC_TL_DELOC(
  s = 3.5e12,
  E = 1.45,
```

```
  R = 0.1,
  method = 'seq',
  clusters = 100,
  times = 150:350) %T>%
plot_RLumCarlo(legend = TRUE)

## TL with FUN to correct for thermal
## quenching
f <- function(x) x * 1/(1 + (2e+6 * exp(-0.55/(8.617e-5 * (DELOC$time + 273)))))
plot_RLumCarlo(
 object = DELOC,
 FUN = f)
```

`run_MC_CW_IRSL_LOC` *Monte-Carlo Simulation for CW-IRSL (localized transitions)*

Description

Runs a Monte-Carlo (MC) simulation of continuous wave infrared stimulated luminescence (CW-IRSL) using the generalized one trap (GOT) model. Localized transitions refer to transitions which do not involve the conduction or valence band. These transitions take place between the ground state and an excited state of the trapped charge, and also involve an energy state of the recombination centre.

Usage

```
run_MC_CW_IRSL_LOC(
  A,
  times,
  clusters = 10,
  n_filled = 100,
  r,
  method = "par",
  output = "signal",
  ...
)
```

Arguments

`A`	numeric (**required**): The optical excitation rate from the ground state of the trap to the excited state (`s^-1`)
`times`	numeric (**required**): The sequence of time steps within the simulation (s)
`clusters`	numeric (*with default*): The number of created clusters for the MC runs. The input can be the output of create_ClusterSystem. In that case `n_filled` indicate absolute numbers of a system.
`n_filled`	integer (*with default*): The number of filled electron traps at the beginning of the simulation (dimensionless). Can be a vector of `length(clusters)`, shorter values are recycled.
`r`	numeric (**required**): The retrapping ratio for localized transitions

`method`	character (*with default*): Sequential `'seq'` or parallel `'par'`processing. In the parallel mode the function tries to run the simulation on multiple CPU cores (if available) with a positive effect on the computation time.
`output`	character (*with default*): output is either the `'signal'` (the default) or `'remaining_e'` (the remaining charges/electrons in the trap)
`...`	further arguments, such as `cores` to control the number of used CPU cores or `verbose` to silence the terminal

Details

The model

$$I_{LOC}(t) = -dn/dt = A * (n^2/(r+n))$$

where in the function:
A := optical excitation rate from the ground state into the excited state of the trap (s^-1)
r := retrapping ratio for localized transitions
t := time (s)
n := number of filled electron traps

Value

This function returns an object of class `RLumCarlo_Model_Output` which is a list consisting of an array with dimension `length(times)` x clusters and a numeric time vector.

Function version

0.1.0

How to cite

Kreutzer, S., 2025. run_MC_CW_IRSL_LOC(): Monte-Carlo Simulation for CW-IRSL (localized transitions). Function version 0.1.0. In: Friedrich, J., Kreutzer, S., Pagonis, V., Schmidt, C., 2025. RLumCarlo: Monte-Carlo Methods for Simulating Luminescence Phenomena. R package version 0.1.10. https://CRAN.R-project.org/package=RLumCarlo

Author(s)

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

References

Pagonis, V., Friedrich, J., Discher, M., Müller-Kirschbaum, A., Schlosser, V., Kreutzer, S., Chen, R. and Schmidt, C., 2019. Excited state luminescence signals from a random distribution of defects: A new Monte Carlo simulation approach for feldspar. Journal of Luminescence 207, 266–272. doi:10.1016/j.jlumin.2018.11.024

Further reading

Chen, R., McKeever, S.W.S., 1997. Theory of Thermoluminescence and Related Phenomena. WORLD SCIENTIFIC. doi:10.1142/2781

Examples

```
run_MC_CW_IRSL_LOC(
 A = 0.12,
 times = 0:100,
 clusters = 50,
 n_filled = 100,
 r = 1e-7,
 method = "seq",
 output = "signal"
) %>%
plot_RLumCarlo(legend = TRUE)
```

`run_MC_CW_IRSL_TUN` *Run Monte-Carlo Simulation for CW-IRSL (tunnelling transitions)*

Description

Runs a Monte-Carlo (MC) simulation of continuous wave infrared stimulated luminescence (CW-IRSL) using the model for tunnelling transitions. Tunnelling refers to quantum mechanical tunnelling processes from the excited state of the trap, into a recombination centre.

Usage

```
run_MC_CW_IRSL_TUN(
  A,
  rho,
  times,
  clusters = 10,
  r_c = 0,
  delta.r = 0.1,
  N_e = 200,
  method = "seq",
  output = "signal",
  ...
)
```

Arguments

`A`	numeric (**required**): The effective optical excitation rate for the tunnelling process (`s^-1`).
`rho`	numeric (**required**): The density of recombination centres (defined as ρ' in Huntley 2006) (dimensionless).
`times`	numeric (**required**): The sequence of time steps within the simulation (s).
`clusters`	numeric (*with default*): The number of created clusters for the MC runs. The input can be the output of create_ClusterSystem. In that case `n_filled` indicate absolute numbers of a system.
`r_c`	numeric (*with default*): Critical distance (>0) that must be provided if the sample has been thermally and/or optically pretreated. This parameter expresses the fact that electron-hole pairs within a critical radius `r_c` have already recombined.

`delta.r`	numeric (*with default*): Increments of the dimensionless distance parameter r'
`N_e`	numeric (*width default*): The total number of electron traps available (dimensionless). Can be a vector of `length(clusters)`, shorter values are recycled.
`method`	character (*with default*): Sequential `'seq'` or parallel `'par'`processing. In the parallel mode the function tries to run the simulation on multiple CPU cores (if available) with a positive effect on the computation time.
`output`	character (*with default*): output is either the `'signal'` (the default) or `'remaining_e'` (the remaining charges/electrons in the trap)
`...`	further arguments, such as `cores` to control the number of used CPU cores or `verbose` to silence the terminal

Details

The model

$$I_{TUN}(r',t) = -dn/dt = A * exp(-(\rho')^{-1/3} * r') * n(r',t)$$

Where in the function:
A := effective optical excitation rate for the tunnelling process (s^-1)
r' := the dimensionless tunnelling radius
ρ' := `rho'` the dimensionless density of recombination centres (see Huntley (2006))
t := time (s)
n := the instantaneous number of electrons corresponding to the radius r' at time t

Value

This function returns an object of class `RLumCarlo_Model_Output` which is a list consisting of an array with dimension length(times) x length(r) x clusters and a numeric time vector.

Function version

0.2.0

How to cite

Friedrich, J., Kreutzer, S., 2025. run_MC_CW_IRSL_TUN(): Run Monte-Carlo Simulation for CW-IRSL (tunnelling transitions). Function version 0.2.0. In: Friedrich, J., Kreutzer, S., Pagonis, V., Schmidt, C., 2025. RLumCarlo: Monte-Carlo Methods for Simulating Luminescence Phenomena. R package version 0.1.10. https://CRAN.R-project.org/package=RLumCarlo

Author(s)

Johannes Friedrich, University of Bayreuth (Germany), Sebastian Kreutzer, Institute of Geography, Heidelberg University (Germany)

References

Huntley, D.J., 2006. An explanation of the power-law decay of luminescence. Journal of Physics: Condensed Matter, 18(4), 1359.

Pagonis, V., Friedrich, J., Discher, M., Müller-Kirschbaum, A., Schlosser, V., Kreutzer, S., Chen, R. and Schmidt, C., 2019. Excited state luminescence signals from a random distribution of defects:

A new Monte Carlo simulation approach for feldspar. Journal of Luminescence 207, 266–272. doi:10.1016/j.jlumin.2018.11.024

Further reading

Aitken, M.J., 1985. Thermoluminescence dating. Academic Press.

Jain, M., Guralnik, B., Andersen, M.T., 2012. Stimulated luminescence emission from localized recombination in randomly distributed defects. Journal of Physics: Condensed Matter 24, 385402.

Chen, R., McKeever, S.W.S., 1997. Theory of Thermoluminescence and Related Phenomena. WORLD SCIENTIFIC. doi:10.1142/2781

Examples

```
run_MC_CW_IRSL_TUN(
 A = 0.8,
 rho = 1e-4,
 times = 0:50,
 r_c = 0.05,
 delta.r = 0.1,
 method = "seq",
 clusters = 10,
  output = "signal") %>%
 plot_RLumCarlo(norm = TRUE, legend = TRUE)
```

---

run_MC_CW_OSL_DELOC *Run Monte-Carlo Simulation for CW-OSL (delocalized transitions)*

Description

Runs a Monte-Carlo (MC) simulation of continuous wave optically stimulated luminescence (CW-OSL) using the one trap one recombination centre (OTOR) model. The term delocalized here refers to the involvement of the conduction band.

Usage

```
run_MC_CW_OSL_DELOC(
  A,
  times,
  clusters = 10,
  N_e = 200,
  n_filled = N_e,
  R,
  method = "par",
  output = "signal",
  ...
)
```

Arguments

A	numeric (**required**): The optical excitation rate from trap to conduction band (s^-1)

`times`	numeric (**required**): The sequence of temperature steps within the simulation (s)
`clusters`	numeric (*with default*): The number of created clusters for the MC runs. The input can be the output of create_ClusterSystem. In that case `n_filled` indicate absolute numbers of a system.
`N_e`	integer (*with default*): The total number of electron traps available (dimensionless). Can be a vector of `length(clusters)`, shorter values are recycled.
`n_filled`	integer (*with default*): The number of filled electron traps at the beginning of the simulation (dimensionless). Can be a vector of `length(clusters)`, shorter values are recycled.
`R`	numeric (**required**): The retrapping ratio for delocalized transitions (dimensionless)
`method`	character (*with default*): Sequential `'seq'` or parallel `'par'`processing. In the parallel mode the function tries to run the simulation on multiple CPU cores (if available) with a positive effect on the computation time.
`output`	character (*with default*): Output is either the `'signal'` (the default) or `'remaining_e'` (the remaining charges, electrons, in the trap)
`...`	further arguments, such as `cores` to control the number of used CPU cores or `verbose` to silence the terminal

Details

The model

$$I_{DELOC}(t) = -dn/dt = A * (n^2/(N * R + n(1 - R)))$$

Where in the function:
t := time (s)
A := the optical excitation rate from trap to conduction band (1/s)
n := `n_filled`, the instantaneous number of electrons
N := `N_e` the available number of electron traps available
R := retrapping ratio for delocalized transitions

Value

This function returns an object of class `RLumCarlo_Model_Output` which is a list consisting of an array with dimension length(times) x clusters and a numeric time vector.

Function version

0.1.0

How to cite

Kreutzer, S., 2025. run_MC_CW_OSL_DELOC(): Run Monte-Carlo Simulation for CW-OSL (delocalized transitions). Function version 0.1.0. In: Friedrich, J., Kreutzer, S., Pagonis, V., Schmidt, C., 2025. RLumCarlo: Monte-Carlo Methods for Simulating Luminescence Phenomena. R package version 0.1.10. https://CRAN.R-project.org/package=RLumCarlo

Author(s)

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

References

Pagonis, V., Friedrich, J., Discher, M., Müller-Kirschbaum, A., Schlosser, V., Kreutzer, S., Chen, R. and Schmidt, C., 2019. Excited state luminescence signals from a random distribution of defects: A new Monte Carlo simulation approach for feldspar. Journal of Luminescence 207, 266–272. doi:10.1016/j.jlumin.2018.11.024

Further reading

Chen, R., McKeever, S.W.S., 1997. Theory of Thermoluminescence and Related Phenomena. WORLD SCIENTIFIC. doi:10.1142/2781

Examples

```
## brief example
run_MC_CW_OSL_DELOC(
 A = 0.12,
 R = 0.1,
 times = 0:10,
 clusters = 10,
 method = "seq") %>%
plot_RLumCarlo(legend = TRUE)

## A long example
## Not run:
A <- c(0.1,0.3,0.5,1)
times <- seq(0, 60, 1)
s <- 1e12
E <- 1
R <- c(1e-7, 1e-6, 0.01, 0.1) # sequence of different R values
clusters <- 1000 # number of Monte Carlo simulations
N_e <- c(200, 500, 700, 400) # number of free electrons
n_filled <- c(200, 500, 100, 70) # number of filled traps
method <-"par"
output <- "signal"
col <- c(1,2,3,4) # ifferent colours for the individual curves
plot_uncertainty <- c(TRUE,FALSE,TRUE,FALSE)  # do you want to see the uncertainty?
add_TF <- c(FALSE,rep(TRUE, (length(R)-1)))

## loop to plot different curves into one plot
for (u in 1:length(R)){
 results <- run_MC_CW_OSL_DELOC(
  A = A[u],
  times,
  clusters = clusters,
  N_e = N_e[u],
  n_filled = n_filled[u],
  R = R[u],
  method = method,
  output = output)

plot_RLumCarlo(
results,
 add = add_TF[u],
 legend = FALSE,
 col = col[u],
 main = "Delocalised Transition")
}
```

```
# add your legend with your parameters
legend("topright",
  ncol = 4,
  cex = 0.55,
  title = "parameters",
  legend=c(
   paste0("A = ", A),
   paste0("n_filled = ", n_filled),
   paste0("N_e = ", N_e),
   paste0("R = ", R)),
   bty = "n",
   text.col = col)

## End(Not run)
```

`run_MC_ISO_DELOC` *Run Monte-Carlo Simulation for ISO-TL (delocalized transitions)*

Description

Runs a Monte-Carlo (MC) simulation of isothermally stimulated luminescence (ISO-TL or ITL) using the one trap one recombination centre (OTOR) model. Delocalised refers to involvement of the conduction band.

Usage

```
run_MC_ISO_DELOC(
  s,
  E,
  T = 20,
  times,
  clusters = 10,
  N_e = 200,
  n_filled = N_e,
  R,
  method = "par",
  output = "signal",
  ...
)
```

Arguments

`s`	numeric (**required**): The frequency factor of the trap (s^-1)
`E`	numeric (**required**): Thermal activation energy of the trap (eV)
`T`	numeric (*with default*): Constant stimulation temperature (°C)
`times`	numeric (*with default*): The sequence of time steps within the simulation (s)
`clusters`	numeric (*with default*): The number of created clusters for the MC runs. The input can be the output of create_ClusterSystem. In that case `n_filled` indicate absolute numbers of a system.

N_e	integer (*with default*): The total number of electron traps available (dimensionless). Can be a vector of length(clusters), shorter values are recycled.
n_filled	integer (*with default*): The number of filled electron traps at the beginning of the simulation (dimensionless). Can be a vector of length(clusters), shorter values are recycled.
R	numeric (**required**): The delocalized retrapping ratio (dimensionless)
method	character (*with default*): Sequential 'seq' or parallel 'par'processing. In the parallel mode the function tries to run the simulation on multiple CPU cores (if available) with a positive effect on the computation time.
output	character (*with default*): Output is either the 'signal' (the default) or 'remaining_e' (the remaining charges, electrons, in the trap)
...	further arguments, such as cores to control the number of used CPU cores or verbose to silence the terminal

Details

The model

$$I_{DELOC}(t) = -dn/dt = (s * exp(-E/(k_B * T_{ISO}))) * (n^2/(N * R + n(1 - R)))$$

Where in the function:
t := time
k_B := Boltzmann constant (8.617 x 10^-5 eV K^-1)
T_{ISO} = temperature of the isothermal experiment (°C)
n := n_filled, the number of filled electron traps at the beginning of the simulation
E := the trap depth (eV)
s := the frequency factor in (s^-1)
N := N_e, the total number of electron traps available (dimensionless)
R := the retrapping ratio for delocalized transitions

Value

This function returns an object of class RLumCarlo_Model_Output which is a list consisting of an array with dimension length(times) x clusters and a numeric time vector.

Function version

0.1.0

How to cite

Kreutzer, S., 2025. run_MC_ISO_DELOC(): Run Monte-Carlo Simulation for ISO-TL (delocalized transitions). Function version 0.1.0. In: Friedrich, J., Kreutzer, S., Pagonis, V., Schmidt, C., 2025. RLumCarlo: Monte-Carlo Methods for Simulating Luminescence Phenomena. R package version 0.1.10. https://CRAN.R-project.org/package=RLumCarlo

Author(s)

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

References

Pagonis, V., Friedrich, J., Discher, M., Müller-Kirschbaum, A., Schlosser, V., Kreutzer, S., Chen, R. and Schmidt, C., 2019. Excited state luminescence signals from a random distribution of defects: A new Monte Carlo simulation approach for feldspar. Journal of Luminescence 207, 266–272. doi:10.1016/j.jlumin.2018.11.024

Further reading

Chen, R., McKeever, S.W.S., 1997. Theory of Thermoluminescence and Related Phenomena. WORLD SCIENTIFIC. doi:10.1142/2781

Examples

```
run_MC_ISO_DELOC(
 s = 3.5e12,
 E = 1.45,
 T = 200,
 R = 1,
 method = 'seq',
 times = 0:100) %>%
plot_RLumCarlo(legend = TRUE)
```

---

run_MC_ISO_LOC *Run Monte-Carlo simulation for ISO-TL (localized transitions)*

Description

Runs a Monte-Carlo (MC) simulation of isothermally stimulated luminescence (ISO-TL or ITL) using the generalized one trap (GOT) model. Localized transitions refer to transitions which do no involve the conduction or valence band. These transitions take place between the ground state and an excited state of the trapped charge, and also involve an energy state of the recombination centre.

Usage

```
run_MC_ISO_LOC(
  s,
  E,
  T = 20,
  times,
  clusters = 10,
  n_filled = 100,
  r,
  method = "par",
  output = "signal",
  ...
)
```

Arguments

s	numeric (**required**): The frequency factor of the trap (s^-1)
E	numeric (**required**): Thermal activation energy of the trap (eV)

`T`	numeric (*with default*): Constant stimulation temperature (°C)
`times`	numeric (*with default*): The sequence of time steps within the simulation (s)
`clusters`	numeric (*with default*): The number of created clusters for the MC runs. The input can be the output of create_ClusterSystem. In that case `n_filled` indicate absolute numbers of a system.
`n_filled`	integer (*with default*): The number of filled electron traps at the beginning of the simulation (dimensionless). Can be a vector of `length(clusters)`, shorter values are recycled.
`r`	numeric (**required**): The retrapping ratio for localized transitions.
`method`	character (*with default*): Sequential `'seq'` or parallel `'par'`processing. In the parallel mode the function tries to run the simulation on multiple CPU cores (if available) with a positive effect on the computation time.
`output`	character (*with default*): output is either the `'signal'` (the default) or `'remaining_e'` (the remaining charges/electrons in the trap)
`...`	further arguments, such as `cores` to control the number of used CPU cores or `verbose` to silence the terminal

Details

The model

$$I_{LOC}(t) = -dn/dt = (s * exp(-E/(k_B * T_{ISO}))) * (n^2/(r+n)))$$

Where in the function:
t := time (s)
k_B := Boltzmann constant (8.617 x 10^-5 eV K^-1)
T_{ISO} := isothermal temperature (°C)
n := `n_filled`
s := frequency factor of the trap (1/s)
E := activation energy of the trap (eV)
r := retrapping ratio for localized transitions

Value

This function returns an object of class `RLumCarlo_Model_Output` which is a list consisting of an array with dimension length(times) x clusters and a numeric time vector.

Function version

0.1.0

How to cite

Kreutzer, S., 2025. run_MC_ISO_LOC(): Run Monte-Carlo simulation for ISO-TL (localized transitions). Function version 0.1.0. In: Friedrich, J., Kreutzer, S., Pagonis, V., Schmidt, C., 2025. RLumCarlo: Monte-Carlo Methods for Simulating Luminescence Phenomena. R package version 0.1.10. https://CRAN.R-project.org/package=RLumCarlo

Author(s)

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

References

Pagonis, V., Friedrich, J., Discher, M., Müller-Kirschbaum, A., Schlosser, V., Kreutzer, S., Chen, R. and Schmidt, C., 2019. Excited state luminescence signals from a random distribution of defects: A new Monte Carlo simulation approach for feldspar. Journal of Luminescence 207, 266–272. doi:10.1016/j.jlumin.2018.11.024

Examples

```
run_MC_ISO_LOC(
 E = 1.45,
 s = 3.5e12,
 T = 200,
 times = 0:100,
 method = 'seq',
 r = 1) %>%
plot_RLumCarlo(legend = TRUE)
```

`run_MC_ISO_TUN` *Monte-Carlo Simulation for ISO-TL (tunnelling transitions)*

Description

Runs a Monte-Carlo (MC) simulation of isothermally stimulated luminescence (ISO-TL or ITL) using the tunnelling (TUN) model. Tunnelling refers to quantum mechanical tunnelling processes from the excited state of the trapped charge, into the recombination centre.

Usage

```
run_MC_ISO_TUN(
  E,
  s,
  T = 200,
  rho,
  times,
  clusters = 10,
  r_c = 0,
  delta.r = 0.1,
  N_e = 200,
  method = "par",
  output = "signal",
  ...
)
```

Arguments

`E`	numeric (**required**): Thermal activation energy of the trap (eV).
`s`	numeric (**required**): The effective frequency factor for the tunnelling process (`s^-1`).
`T`	numeric (*with default*): Constant stimulation temperature (°C).

`rho`	numeric (**required**): The dimensionless density of recombination centres (defined as ρ' in Huntley 2006) (dimensionless).
`times`	numeric (**required**): The sequence of time steps within the simulation (s).
`clusters`	numeric (*with default*): The number of created clusters for the MC runs. The input can be the output of create_ClusterSystem. In that case `n_filled` indicate absolute numbers of a system.
`r_c`	numeric (*with default*): Critical distance (>0) that must be provided if the sample has been thermally and/or optically pretreated. This parameter expresses the fact that electron-hole pairs within a critical radius `r_c` have already recombined.
`delta.r`	numeric (*with default*): Fractional change of the dimensionless distance of nearest recombination centres (r')
`N_e`	numeric (*width default*): The total number of electron traps available (dimensionless). Can be a vector of `length(clusters)`, shorter values are recycled.
`method`	character (*with default*): Sequential `'seq'` or parallel `'par'`processing. In the parallel mode the function tries to run the simulation on multiple CPU cores (if available) with a positive effect on the computation time.
`output`	character (*with default*): output is either the `'signal'` (the default) or `'remaining_e'` (the remaining charges/electrons in the trap)
`...`	further arguments, such as `cores` to control the number of used CPU cores or `verbose` to silence the terminal

Details

The model

$$I_{TUN}(r',t) = -dn/dt = (s * exp(-E/(k_B * T_{ISO}))) * exp(-(\rho')^{-1/3} * r') * n(r',t)$$

Where in the function:
E := thermal activation energy (eV)
s := the effective frequency factor for the tunnelling process (s^-1)
T_{ISO} := the temperature of the isothermal experiment (°C)
k_B := Boltzmann constant (8.617 x 10^-5 eV K^-1)
r' := the dimensionless tunnelling radius
ρ' := `rho` the dimensionless density of recombination centres see Huntley (2006)
t := time (s)
n := the instantaneous number of electrons corresponding to the radius r'

Value

This function returns an object of class `RLumCarlo_Model_Output` which is a list consisting of an array with dimension length(times) x length(r) x clusters and a numeric time vector.

Function version

0.1.0

How to cite

Friedrich, J., Kreutzer, S., 2025. run_MC_ISO_TUN(): Monte-Carlo Simulation for ISO-TL (tunnelling transitions). Function version 0.1.0. In: Friedrich, J., Kreutzer, S., Pagonis, V., Schmidt, C., 2025. RLumCarlo: Monte-Carlo Methods for Simulating Luminescence Phenomena. R package version 0.1.10. https://CRAN.R-project.org/package=RLumCarlo

Author(s)

Johannes Friedrich, University of Bayreuth (Germany), Sebastian Kreutzer, Institute of Geography, Heidelberg University (Germany)

References

Pagonis, V. and Kulp, C., 2017. Monte Carlo simulations of tunneling phenomena and nearest neighbor hopping mechanism in feldspars. Journal of Luminescence 181, 114–120. doi:10.1016/j.jlumin.2016.09.014

Further reading Aitken, M.J., 1985. Thermoluminescence dating. Academic Press.

Huntley, D.J., 2006. An explanation of the power-law decay of luminescence. Journal of Physics: Condensed Matter, 18(4), 1359.

Jain, M., Guralnik, B., Andersen, M.T., 2012. Stimulated luminescence emission from localized recombination in randomly distributed defects. Journal of Physics: Condensed Matter 24, 385402.

Pagonis, V., Friedrich, J., Discher, M., Müller-Kirschbaum, A., Schlosser, V., Kreutzer, S., Chen, R. and Schmidt, C., 2019. Excited state luminescence signals from a random distribution of defects: A new Monte Carlo simulation approach for feldspar. Journal of Luminescence 207, 266–272. doi:10.1016/j.jlumin.2018.11.024

Examples

```
## short example
run_MC_ISO_TUN(
 E = .8,
 s = 1e16,
 T = 50,
 rho = 1e-4,
 times = 0:100,
 clusters = 10,
 N_e = 100,
 r_c = 0.2,
 delta.r = 0.5,
 method = "seq") %>%
 plot_RLumCarlo(legend = TRUE)

## Not run:
## long (meaningful) example
results <- run_MC_ISO_TUN(
 E = .8,
 s = 1e16,
 T = 50,
 rho = 1e-4,
 times = 0:100,
 clusters = 1000,
 N_e = 200,
 r_c = 0.1,
```

```
 delta.r = 0.05,
 method = "par")

plot_RLumCarlo(results, legend = TRUE)

## End(Not run)
```

`run_MC_LM_OSL_DELOC` *Run Monte-Carlo Simulation for LM-OSL (delocalized transitions)*

Description

Runs a Monte-Carlo (MC) simulation of linearly modulated optically stimulated luminescence (LM-OSL) using the one trap one recombination centre (OTOR) model. Delocalised refers to involvement of the conduction band.

Usage

```
run_MC_LM_OSL_DELOC(
  A,
  times,
  clusters = 10,
  N_e = 200,
  n_filled = N_e,
  R,
  method = "par",
  output = "signal",
  ...
)
```

Arguments

`A`	numeric (**required**): The optical excitation rate from trap to conduction band (s^-1)
`times`	numeric (**required**): The sequence of time steps within the simulation (s)
`clusters`	numeric (*with default*): The number of created clusters for the MC runs. The input can be the output of create_ClusterSystem. In that case `n_filled` indicate absolute numbers of a system.
`N_e`	integer (*with default*): The total number of electron traps available (dimensionless). Can be a vector of `length(clusters)`, shorter values are recycled.
`n_filled`	integer (*with default*): The number of filled electron traps at the beginning of the simulation (dimensionless). Can be a vector of `length(clusters)`, shorter values are recycled.
`R`	numeric (**required**): The retrapping ratio for delocalized transitions
`method`	character (*with default*): Sequential `'seq'` or parallel `'par'`processing. In the parallel mode the function tries to run the simulation on multiple CPU cores (if available) with a positive effect on the computation time.
`output`	character (*with default*): output is either the `'signal'` (the default) or `'remaining_e'` (the remaining charges/electrons in the trap)
`...`	further arguments, such as `cores` to control the number of used CPU cores or `verbose` to silence the terminal

Details

The model

$$I_{DELOC}(t) = -dn/dt = A * t/P * (n^2/(N * R + n(1 - R)))$$

Where in the function:
t := time (s)
A := the optical excitation rate from trap to conduction band (1/s)
n := `n_filled`, the instantaneous number of electrons
R := the retrapping ratio for delocalized transitions
N := `N_e`, the total number of electron traps available (dimensionless)
P := total stimulation time (s)

Value

This function returns an object of class `RLumCarlo_Model_Output` which is a list consisting of an array with dimension length(times) x clusters and a numeric time vector.

Function version

0.1.0

How to cite

Kreutzer, S., 2025. run_MC_LM_OSL_DELOC(): Run Monte-Carlo Simulation for LM-OSL (delocalized transitions). Function version 0.1.0. In: Friedrich, J., Kreutzer, S., Pagonis, V., Schmidt, C., 2025. RLumCarlo: Monte-Carlo Methods for Simulating Luminescence Phenomena. R package version 0.1.10. https://CRAN.R-project.org/package=RLumCarlo

Author(s)

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

References

Pagonis, V., Friedrich, J., Discher, M., Müller-Kirschbaum, A., Schlosser, V., Kreutzer, S., Chen, R. and Schmidt, C., 2019. Excited state luminescence signals from a random distribution of defects: A new Monte Carlo simulation approach for feldspar. Journal of Luminescence 207, 266–272. doi:10.1016/j.jlumin.2018.11.024

Further reading

Chen, R., McKeever, S.W.S., 1997. Theory of Thermoluminescence and Related Phenomena. WORLD SCIENTIFIC. doi:10.1142/2781

Examples

```
run_MC_LM_OSL_DELOC(
 A = 0.12,
 R = 0.1,
 times = 0:50,
 method = "seq",
 clusters = 10) %>%
plot_RLumCarlo(legend = TRUE)
```

`run_MC_LM_OSL_LOC` *Run Monte-Carlo Simulation for LM-OSL (localized transitions)*

Description

Runs a Monte-Carlo (MC) simulation of linearly modulated optically stimulated luminescence (LM-OSL) using the generalized one trap (GOT) model. Localized transitions refer to transitions which do not involve the conduction or valence band. These transitions take place between the ground state and an excited state of the trap, and also involve a an energy state of the recombination centre.

Usage

```
run_MC_LM_OSL_LOC(
  A,
  times,
  clusters = 10,
  n_filled = 100,
  r,
  method = "par",
  output = "signal",
  ...
)
```

Arguments

`A`	numeric (**required**): The optical excitation rate from the ground state into the excited state of the trap (s^-1)
`times`	numeric (**required**): The sequence of time steps within the simulation (s)
`clusters`	numeric (*with default*): The number of created clusters for the MC runs. The input can be the output of create_ClusterSystem. In that case `n_filled` indicate absolute numbers of a system.
`n_filled`	integer (*with default*): The number of filled electron traps at the beginning of the simulation (dimensionless). Can be a vector of `length(clusters)`, shorter values are recycled.
`r`	numeric (**required**): The retrapping ratio for localized transitions
`method`	character (*with default*): Sequential `'seq'` or parallel `'par'`processing. In the parallel mode the function tries to run the simulation on multiple CPU cores (if available) with a positive effect on the computation time.
`output`	character (*with default*): output is either the `'signal'` (the default) or `'remaining_e'` (the remaining charges, electrons, in the trap)
`...`	further arguments, such as `cores` to control the number of used CPU cores or `verbose` to silence the terminal

Details

The model

$$I_{LOC}(t) = -dn/dt = (A * t/P) * (n^2/(r+n))$$

Where in the function:
A := optical excitation rate from the ground state into the excited state of the trap (1/s)
P := total excitation time (s)
t := time (s)
n := `n_filled`, the instantaneous number of electrons
r := the retrapping ratio for localized transitions

Value

This function returns an object of class `RLumCarlo_Model_Output` which is a list consisting of an array with dimension length(times) x clusters and a numeric time vector.

Function version

0.1.0

How to cite

Kreutzer, S., 2025. run_MC_LM_OSL_LOC(): Run Monte-Carlo Simulation for LM-OSL (localized transitions). Function version 0.1.0. In: Friedrich, J., Kreutzer, S., Pagonis, V., Schmidt, C., 2025. RLumCarlo: Monte-Carlo Methods for Simulating Luminescence Phenomena. R package version 0.1.10. https://CRAN.R-project.org/package=RLumCarlo

Author(s)

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

References

Pagonis, V., Friedrich, J., Discher, M., Müller-Kirschbaum, A., Schlosser, V., Kreutzer, S., Chen, R. and Schmidt, C., 2019. Excited state luminescence signals from a random distribution of defects: A new Monte Carlo simulation approach for feldspar. Journal of Luminescence 207, 266–272. doi:10.1016/j.jlumin.2018.11.024

Examples

```
## short example
run_MC_LM_OSL_LOC(
 A = 1,
 times = 0:40,
 clusters = 10,
 n_filled = 100,
 r = 1e-7,
 method = "seq",
 output = "signal") %>%
plot_RLumCarlo(legend = TRUE)

## Not run:
## the long (meaningful) example
results <- run_MC_LM_OSL_LOC(
 A = 1,
 times = 0:100,
 clusters = 100,
 n_filled = 100,
 r = 1e-7,
```

```
 method = "par",
 output = "signal")

## plot
plot_RLumCarlo(results, legend = TRUE)

## End(Not run)
```

`run_MC_LM_OSL_TUN` *Run Monte-Carlo Simulation for LM-OSL (tunnelling transitions)*

Description

Runs a Monte-Carlo (MC) simulation of linearly modulated optically stimulated luminescence (LM-OSL) using the tunnelling (TUN) model. Tunnelling refers to quantum mechanical tunnelling processes from the excited state of the trapped charge, into a recombination centre.

Usage

```
run_MC_LM_OSL_TUN(
  A,
  rho,
  times,
  clusters = 10,
  r_c = 0,
  delta.r = 0.1,
  N_e = 200,
  method = "par",
  output = "signal",
  ...
)
```

Arguments

`A`	numeric (**required**): The effective optical excitation rate for the tunnelling process
`rho`	numeric (**required**): The dimensionless density of recombination centres (defined as ρ' in Huntley 2006) (dimensionless)
`times`	numeric (**required**): The sequence of time steps within the simulation (s)
`clusters`	numeric (*with default*): The number of MC runs
`r_c`	numeric (*with default*): Critical distance (>0) that is to be used if the sample has 1 been thermally and/or optically pretreated. This parameter expresses the fact that electron-hole pairs within a critical radius `r_c` have already been recombined.
`delta.r`	numeric (*with default*): Increments of dimensionless distance r'
`N_e`	numeric (*width default*): The total number of electron traps available (dimensionless). Can be a vector of `length(clusters)`, shorter values are recycled.

method	character (*with default*): Sequential 'seq' or parallel 'par'processing. In the parallel mode the function tries to run the simulation on multiple CPU cores (if available) with a positive effect on the computation time.
output	character (*with default*): output is either the 'signal' (the default) or 'remaining_e' (the remaining charges, electrons, in the trap)
...	further arguments, such as cores to control the number of used CPU cores or verbose to silence the terminal

Details

The model

$$I_{TUN}(r',t) = -dn/dt = (A * t/P) * exp(-(\rho')^{-1/3} * r') * n(r',t)$$

Where in the function:
A := the optical excitation rate for the tunnelling process (s^-1)
t := time (s)
P := maximum stimulation time (s)
r' := the dimensionless tunnelling radius
ρ := rho the dimensionless density of recombination centres see Huntley (2006)
n := the instantaneous number of electrons corresponding to the radius r'

Value

This function returns an object of class RLumCarlo_Model_Output which is a list consisting of an array with dimension length(times) x length(r) x clusters and a numeric time vector.

Function version

0.1.0

How to cite

Friedrich, J., Kreutzer, S., 2025. run_MC_LM_OSL_TUN(): Run Monte-Carlo Simulation for LM-OSL (tunnelling transitions). Function version 0.1.0. In: Friedrich, J., Kreutzer, S., Pagonis, V., Schmidt, C., 2025. RLumCarlo: Monte-Carlo Methods for Simulating Luminescence Phenomena. R package version 0.1.10. https://CRAN.R-project.org/package=RLumCarlo

Author(s)

Johannes Friedrich, University of Bayreuth (Germany), Sebastian Kreutzer, Institute of Geography, Heidelberg University (Germany)

References

Huntley, D.J., 2006. An explanation of the power-law decay of luminescence. Journal of Physics: Condensed Matter, 18(4), 1359.

Pagonis, V. and Kulp, C., 2017. Monte Carlo simulations of tunneling phenomena and nearest neighbor hopping mechanism in feldspars. Journal of Luminescence 181, 114–120. doi:10.1016/j.jlumin.2016.09.014

Pagonis, V., Friedrich, J., Discher, M., Müller-Kirschbaum, A., Schlosser, V., Kreutzer, S., Chen, R. and Schmidt, C., 2019. Excited state luminescence signals from a random distribution of defects:

A new Monte Carlo simulation approach for feldspar. Journal of Luminescence 207, 266–272. doi:10.1016/j.jlumin.2018.11.024

Further reading Aitken, M.J., 1985. Thermoluminescence dating. Academic Press.

Jain, M., Guralnik, B., Andersen, M.T., 2012. Stimulated luminescence emission from localized recombination in randomly distributed defects. Journal of Physics: Condensed Matter 24, 385402.

Examples

```
##the short example
run_MC_LM_OSL_TUN(
 A = 1,
 rho = 1e-3,
 times = 0:100,
 clusters = 10,
 N_e = 100,
 r_c = 0.1,
 delta.r = 1e-1,
 method = "seq",
 output = "signal") %>%
plot_RLumCarlo(norm = TRUE)

## Not run:
## the long (meaningful) example
results <- run_MC_LM_OSL_TUN(
 A = 1,
 rho = 1e-3,
 times = 0:1000,
 clusters = 30,
 N_e = 100,
 r_c = 0.1,
 delta.r = 1e-1,
 method = "par",
 output = "signal")

plot_RLumCarlo(results, norm = TRUE)

## End(Not run)
```

---

run_MC_TL_DELOC *Run Monte-Carlo Simulation for TL (delocalized transitions)*

Description

Runs a Monte-Carlo (MC) simulation of thermoluminescence (TL) using the one trap one recombination centre (OTOR) model. Delocalised refers to involvement of the conduction band. The heating rate in this function is assumed to be 1 K/s.

Usage

```
run_MC_TL_DELOC(
  s,
  E,
```

```
  times,
  b = 1,
  clusters = 10,
  N_e = 200,
  n_filled = N_e,
  R = 1,
  method = "par",
  output = "signal",
  ...
)
```

Arguments

`s`	numeric (**required**): The frequency factor of the trap (s^-1)
`E`	numeric (**required**): Thermal activation energy of the trap (eV)
`times`	numeric (**required**): The sequence of temperature steps within the simulation (s). The default heating rate is set to 1 K/s. The final temperature is `max(times) * b`
`b`	numeric (*with default*): the heating rate in K/s
`clusters`	numeric (*with default*): The number of created clusters for the MC runs. The input can be the output of create_ClusterSystem. In that case `n_filled` indicate absolute numbers of a system.
`N_e`	integer (*with default*): The total number of electron traps available (dimensionless). Can be a vector of `length(clusters)`, shorter values are recycled.
`n_filled`	integer (*with default*): The number of filled electron traps at the beginning of the simulation (dimensionless). Can be a vector of `length(clusters)`, shorter values are recycled.
`R`	numeric (*with default*): Re-trapping ratio for delocalized transitions
`method`	character (*with default*): Sequential `'seq'` or parallel `'par'`processing. In the parallel mode the function tries to run the simulation on multiple CPU cores (if available) with a positive effect on the computation time.
`output`	character (*with default*): output is either the `'signal'` (the default) or `'remaining_e'` (the remaining charges/electrons in the trap)
`...`	further arguments, such as `cores` to control the number of used CPU cores or `verbose` to silence the terminal

Details

The model

$$I_{DELOC}(t) = -dn/dt = (s * exp(-E/(k_B * T))) * (n^2/(N * R + n(1 - R))))$$

Where in the function:
E := the thermal activation energy (eV)
s := the frequency factor in (s^-1)
t := time (s)
k_B := Boltzmann constant (8.617 x 10^-5 eV K^-1)
T := temperature (°C)
R := Delocalised retrapping ratio
n := `n_filled`, the instantaneous number of electrons

N := N_e, the total number of electron traps available (dimensionless)

Why times and b instead of temperature?

The parameter to control the temperature is a function of the stimulation times (the parameter times) and the heating rate (b). Thus, the final temperature is max(times) * b. For a heating rate (b = 1) the final temperature is max(times). While this might be a little bit confusing, it also allows you to control the time resolution of the simulation, i.e. you can simulate more points per second.

Value

This function returns an object of class RLumCarlo_Model_Output which is a list consisting of an array with dimension length(times) x clusters and a numeric time vector.

Function version

0.1.0

How to cite

Kreutzer, S., 2025. run_MC_TL_DELOC(): Run Monte-Carlo Simulation for TL (delocalized transitions). Function version 0.1.0. In: Friedrich, J., Kreutzer, S., Pagonis, V., Schmidt, C., 2025. RLumCarlo: Monte-Carlo Methods for Simulating Luminescence Phenomena. R package version 0.1.10. https://CRAN.R-project.org/package=RLumCarlo

Author(s)

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

References

Pagonis, V., Friedrich, J., Discher, M., Müller-Kirschbaum, A., Schlosser, V., Kreutzer, S., Chen, R. and Schmidt, C., 2019. Excited state luminescence signals from a random distribution of defects: A new Monte Carlo simulation approach for feldspar. Journal of Luminescence 207, 266–272. doi:10.1016/j.jlumin.2018.11.024

Further reading

Chen, R., McKeever, S.W.S., 1997. Theory of Thermoluminescence and Related Phenomena. WORLD SCIENTIFIC. doi:10.1142/2781

Examples

```
## the short example
run_MC_TL_DELOC(
 s = 3.5e12,
 E = 1.45,
 R = 0.1,
 method = 'seq',
 clusters = 100,
 times = 150:350) %>%
plot_RLumCarlo(legend = TRUE)

## Not run:
## the long (meaningful) example
# define your parameters
```

```
times <- seq(100, 450, 1)
s <- rep(3.5e12, 4)
E <- rep(1.45, 4)
R <- c(0.7e-6, 1e-6, 0.01, 0.1)
clusters <- 300
N_e <- c(400, 500, 700, 400)
n_filled <- c(400, 500, 300, 70)
method <- "par"
output <- "signal"
col <- c(1, 2, 3, 4) # different colours for the individual curves
plot_uncertainty <- c(TRUE, TRUE, TRUE, TRUE)  # do you want to see the uncertainty?
add_TF <- c(FALSE, rep(TRUE, (length(R) - 1)))

# loop to plot different curves into one plot
for (u in 1:length(R)){
 results <- run_MC_TL_DELOC(
  times=times,
  s = s[u],
  E = E[u],
  clusters = clusters,
  N_e = N_e[u],
  n_filled = n_filled[u],
  R = R[u],
  method = method,
  output = output)

plot_RLumCarlo(
 results,
 add = add_TF[u],
 legend = FALSE,
 col=col[u],
 main = " your plot",
 ylim=c(0,20))
}
#add your legend with your parameters
legend("topright",
  ncol = 5,
  cex = 0.55,
  bty = "n",
  title = "parameters",
  legend = c(
   paste0("E = ", E),
   paste0("s = ", s),
   paste0("n_filled = ", n_filled),
   paste0("N_e = ", N_e), paste0("R = ", R)),
   text.col = col)

## End(Not run)
```

---

run_MC_TL_LOC *Run Monte-Carlo Simulation for TL (localized transitions)*

Description

Runs a Monte-Carlo (MC) simulation of thermoluminescence (TL) using the generalized one trap (GOT) model. Localized transitions refer to transitions which do not involve the conduction or valence band. These transitions take place between the ground state and an excited state of the trapped charge, and also involve an energy state of the recombination centre. The heating rate in this function is assumed to be 1 K/s.

Usage

```
run_MC_TL_LOC(
  s,
  E,
  times,
  b = 1,
  clusters = 10,
  n_filled = 100,
  r,
  method = "par",
  output = "signal",
  ...
)
```

Arguments

`s`	numeric (**required**): The frequency factor of the trap (s^-1)
`E`	numeric (**required**): Thermal activation energy of the trap (eV)
`times`	numeric (**required**): The sequence of temperature steps within the simulation (s). The default heating rate is set to 1 K/s. The final temperature is `max(times) * b`
`b`	numeric (*with default*): the heating rate in K/s
`clusters`	numeric (*with default*): The number of created clusters for the MC runs. The input can be the output of create_ClusterSystem. In that case `n_filled` indicate absolute numbers of a system.
`n_filled`	integer (*with default*): The number of filled electron traps at the beginning of the simulation (dimensionless). Can be a vector of `length(clusters)`, shorter values are recycled.
`r`	numeric (**required**): The localized retrapping ratio (dimensionless)
`method`	character (*with default*): Sequential `'seq'` or parallel `'par'`processing. In the parallel mode the function tries to run the simulation on multiple CPU cores (if available) with a positive effect on the computation time.
`output`	character (*with default*): output is either the `'signal'` (the default) or `'remaining_e'` (the remaining charges/electrons in the trap)
`...`	further arguments, such as `cores` to control the number of used CPU cores or `verbose` to silence the terminal

Details

The model

$$I_{LOC}(t) = -dn/dt = (s * exp(-E/(k_B * T))) * (n^2/(r+n))$$

Where in the function:
E := the thermal activation energy (eV)
s := the frequency factor for the trap (s^-1)
t := time (s)
k_B := Boltzmann constant (8.617 x 10^-5 eV K^-1)
T := temperature (°C)
n := the instantaneous number of electrons
r := the retrapping ratio for localized transitions

Value

This function returns an object of class `RLumCarlo_Model_Output` which is a list consisting of an array with dimension length(times) x clusters and a numeric time vector.

Function version

0.1.0

How to cite

Kreutzer, S., 2025. run_MC_TL_LOC(): Run Monte-Carlo Simulation for TL (localized transitions). Function version 0.1.0. In: Friedrich, J., Kreutzer, S., Pagonis, V., Schmidt, C., 2025. RLumCarlo: Monte-Carlo Methods for Simulating Luminescence Phenomena. R package version 0.1.10. https://CRAN.R-project.org/package=RLumCarlo

Author(s)

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

References

Pagonis, V., Friedrich, J., Discher, M., Müller-Kirschbaum, A., Schlosser, V., Kreutzer, S., Chen, R. and Schmidt, C., 2019. Excited state luminescence signals from a random distribution of defects: A new Monte Carlo simulation approach for feldspar. Journal of Luminescence 207, 266–272. doi:10.1016/j.jlumin.2018.11.024

Examples

```
## the short example
run_MC_TL_LOC(
 s = 1e14,
 E = 0.9,
 times = 50:100,
 b = 1,
 method = "seq",
 clusters = 30,
 r = 1) %>%
plot_RLumCarlo()

## Not run:
## the long (meaningful) example
results <- run_MC_TL_LOC(
 s = 1e14,
 E = 0.9,
 times = 50:100,
```

```
 method = "par",
 clusters = 100,
 r = 1)

## plot
plot_RLumCarlo(results)


## End(Not run)
```

`run_MC_TL_TUN` *Run Monte-Carlo Simulation for TL (tunnelling transitions)*

Description

Runs a Monte-Carlo (MC) simulation of thermoluminescence (TL) caused by tunnelling (TUN) transitions. Tunnelling refers to quantum mechanical tunnelling processes from the excited state of the trap into a recombination centre. The heating rate in this function is assumed to be 1 K/s.

Usage

```
run_MC_TL_TUN(
  s,
  E,
  rho,
  r_c = 0,
  times,
  b = 1,
  clusters = 10,
  N_e = 200,
  delta.r = 0.1,
  method = "par",
  output = "signal",
  ...
)
```

Arguments

`s`	list (**required**): The effective frequency factor for the tunnelling process (s^-1)
`E`	numeric (**required**): Thermal activation energy of the trap (eV)
`rho`	numeric (**required**): The dimensionless density of recombination centres (defined as ρ' in Huntley 2006)
`r_c`	numeric (*with default*): Critical distance (>0) that is to be used if the sample has been thermally and/or optically pretreated. This parameter expresses the fact that electron-hole pairs within a critical radius `r_c` have already recombined.
`times`	numeric (**required**): The sequence of temperature steps within the simulation (s). The default heating rate is set to 1 K/s. The final temperature is `max(times) * b`
`b`	numeric (*with default*): the heating rate in K/s

`clusters`	numeric (*with default*): The number of created clusters for the MC runs. The input can be the output of create_ClusterSystem. In that case `n_filled` indicate absolute numbers of a system.
`N_e`	numeric (*with default*): The total number of electron traps available (dimensionless). Can be a vector of `length(clusters)`, shorter values are recycled.
`delta.r`	numeric (*with default*): The increments of the dimensionless distance r'
`method`	character (*with default*): Sequential `'seq'` or parallel `'par'`processing. In the parallel mode the function tries to run the simulation on multiple CPU cores (if available) with a positive effect on the computation time.
`output`	character (*with default*): output is either the `'signal'` (the default) or `'remaining_e'` (the remaining charges/electrons in the trap)
`...`	further arguments, such as `cores` to control the number of used CPU cores or `verbose` to silence the terminal

Details

The model

$$I_{TUN}(r',t) = -dn/dt = (s * exp(-E/(k_B * T))) * exp(-(\rho')^{-1/3} * r') * n(r',t)$$

Where in the function:
s := frequency for the tunnelling process (s^-1)
E := thermal activation energy (eV)
k_B := Boltzmann constant (8.617 x 10^-5 eV K^-1)
T := temperature (°C)
r' := the dimensionless tunnelling radius
ρ' := `rho'`, the dimensionless density of recombination centres (see Huntley (2006))
t := time (s)
n := the instantaneous number of electrons at distance r'

Value

This function returns an object of class `RLumCarlo_Model_Output` which is a list consisting of an array with dimension length(times) x length(r) x clusters and a numeric time vector.

Function version

0.1.0

How to cite

Friedrich, J., Kreutzer, S., 2025. run_MC_TL_TUN(): Run Monte-Carlo Simulation for TL (tunnelling transitions). Function version 0.1.0. In: Friedrich, J., Kreutzer, S., Pagonis, V., Schmidt, C., 2025. RLumCarlo: Monte-Carlo Methods for Simulating Luminescence Phenomena. R package version 0.1.10. https://CRAN.R-project.org/package=RLumCarlo

Author(s)

Johannes Friedrich, University of Bayreuth (Germany), Sebastian Kreutzer, Geography & Earth Sciences, Aberystwyth University (United Kingdom)

References

Huntley, D.J., 2006. An explanation of the power-law decay of luminescence. Journal of Physics: Condensed Matter, 18(4), 1359.

Pagonis, V. and Kulp, C., 2017. Monte Carlo simulations of tunneling phenomena and nearest neighbor hopping mechanism in feldspars. Journal of Luminescence 181, 114–120. doi:10.1016/j.jlumin.2016.09.014

Pagonis, V., Friedrich, J., Discher, M., Müller-Kirschbaum, A., Schlosser, V., Kreutzer, S., Chen, R. and Schmidt, C., 2019. Excited state luminescence signals from a random distribution of defects: A new Monte Carlo simulation approach for feldspar. Journal of Luminescence 207, 266–272. doi:10.1016/j.jlumin.2018.11.024

Further reading

Aitken, M.J., 1985. Thermoluminescence dating. Academic Press.

Jain, M., Guralnik, B., Andersen, M.T., 2012. Stimulated luminescence emission from localized recombination in randomly distributed defects. Journal of Physics: Condensed Matter 24, 385402.

Examples

```
## the short example
run_MC_TL_TUN(
 s = 1e12,
 E = 0.9,
 rho = 1,
 r_c = 0.1,
 times = 80:120,
 b = 1,
 clusters = 50,
 method = 'seq',
 delta.r = 1e-1) %>%
plot_RLumCarlo()

## Not run:
## the long (meaningful example)
results <- run_MC_TL_TUN(
 s = 1e12,
 E = 0.9,
 rho = 0.01,
 r_c = 0.1,
 times = 80:220,
 clusters = 100,
 method = 'par',
 delta.r = 1e-1)

## plot
plot_RLumCarlo(results)

## End(Not run)
```

Index