

# Package ‘RLumCarlo’

February 25, 2019

**Type** Package

**Title** Monte-Carlo Methods for Simulating Luminescence Phenomena

**Version** 0.1.0.14

**Date** 2019-02-25

**Author** Johannes Friedrich [aut, trl, cre] (<<https://orcid.org/0000-0002-0805-9547>>),  
Sebastian Kreutzer [aut] (<<https://orcid.org/0000-0002-0734-2199>>)

**Maintainer** Johannes Friedrich <[johannes.friedrich@uni-bayreuth.de](mailto:johannes.friedrich@uni-bayreuth.de)>

## Description

A collection of functions to simulate luminescence signals with Monte-Carlo methods in the mineral feldspar based on published models.

**Contact** Package Developer Team <[johannes.friedrich@uni-bayreuth.de](mailto:johannes.friedrich@uni-bayreuth.de)>

**License** GPL-3

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

**Depends** R (>= 3.3.0), utils, magrittr

**URL** <https://CRAN.R-project.org/package=RLumModel>

**LinkingTo** Rcpp, RcppProgress, RcppArmadillo

**Imports** abind, doParallel, foreach, parallel, methods, Rcpp

**Suggests** R.srp

**Encoding** UTF-8

**VignetteBuilder** R.srp

**RoxygenNote** 6.1.1

**NeedsCompilation** yes

## R topics documented:

RLumCarlo-package . . . . .	2
calc_RLumCarlo . . . . .	2
plot_RLumCarlo . . . . .	3
run_MC_CW_IRSL . . . . .	3
run_MC_ISO . . . . .	5
run_MC_LM_OSL . . . . .	6
run_MC_TL . . . . .	7
run_MC_TL_DELOC . . . . .	8

<b>Index</b>	<b>10</b>
--------------	-----------

---

RLumCarlo-package	<i>Modelling luminescence signals in feldspar</i>
-------------------	---

---

## Description

## Details

Package:	RLumCarlo
Type:	Package
Version:	0.0.2
Date:	2018-08-28
License:	GPL-3

## Author(s)

Johannes Friedrich (University of Bayreuth, Germany), Sebastian Kreutzer, IRAMAT-CRP2A, UMR 5060, CNRS-Université Bordeaux Montaigne (France)

---

calc_RLumCarlo	<i>Plot results from Monte-Carlo simulations with RLumCarlo</i>
----------------	---

---

## Description

Plot results from Monte-Carlo simulations with RLumCarlo

## Usage

```
calc_RLumCarlo(results)
```

## Arguments

results      [array](#):

## Value

This function returns a [data.frame](#)

## Function version

0.0.1 [2017-01-27]

## Author(s)

Johannes Friedrich, University of Bayreuth (Germany)

plot\_RLumCarlo

*Plot results from Monte-Carlo simulations with RLumCarlo***Description**

Plot results from Monte-Carlo simulations with RLumCarlo

**Usage**

```
plot_RLumCarlo(results, times = NULL, norm = FALSE, legend = FALSE,
  add = FALSE, ...)
```

**Arguments**

results	<a href="#">data.frame</a> ( <b>required</b> )
times	<a href="#">numeric</a> ( <i>optinal</i> ): Optional vector for the x-axis
norm	<a href="#">logical</a> ( <i>with default</i> ): Normalise curve to the highest intensity
legend	<a href="#">logical</a> ( <i>with default</i> ): Enable/disable legend
add	<a href="#">logical</a> ( <i>with default</i> ): allow overplotting of results
...	further arguments that can be passed to control the plot output. Currently supported are: xlab, xlim, ylim, main, lwd, type

**Value**

This function returns a graphical output

**Function version**

0.1.0

**Author(s)**

Johannes Friedrich, University of Bayreuth (Germany), Sebastian Kreutzer, IRAMAT-CRP2A, Université Bordeaux Montaigne (France)

run\_MC\_CW\_IRSL

*Run Monte-Carlo simulation for CW-IRSL***Description**

Run Monte-Carlo simulation for CW-IRSL

**Usage**

```
run_MC_CW_IRSL(A, rho, times, clusters = 10, r = NULL, N_e = 200,
  method = "seq", output = "signal", ...)
```

**Arguments**

A	numeric
rho	numeric
times	vector (with default)
clusters	numeric (with default):
r	numeric (with default)
N_e	numeric (with default):
method	character (with default):
output	character (with default):
...	further arguments

**Value**

This function returns a list.

**Function version**

0.0.2 [2017-01-31]

**Author(s)**

Johannes Friedrich, University of Bayreuth (Germany), Sebastian Kreutzer, IRAMAT-CRP2A, Université Bordeaux Montaigne (France)

**References**

Pagonis 2017

**Examples**

```
## Not run:

##=====##
## Example 1: Simulate CW-IRSL measurement
##=====##

run_MC_CW_IRSL(A = 0.12, rho = 0.003, times = 0:1000) %>%
  calc_RLumCarlo() %>%
  plot_RLumCarlo(norm = T, legend = T)

## End(Not run)
```

run\_MC\_ISO

*Run Monte-Carlo simulation for isothermal measurements***Description**

Run Monte-Carlo simulation for isothermal measurements

**Usage**

```
run_MC_ISO(A, rho, times, clusters = 10, r = NULL, N_e = 200,
  method = "par", output = "signal", ...)
```

**Arguments**

A	numeric
rho	numeric
times	vector (with default)
clusters	numeric (with default):
r	numeric (with default)
N_e	numeric (with default):
method	character (with default):
output	character (with default):
...	further arguments

**Value**

This function returns a list.

**Function version**

0.0.1 [2017-01-27]

**Author(s)**

Johannes Friedrich, University of Bayreuth (Germany)

**References**

Pagonis 2017

**Examples**

```
## Not run:
##=====##
## Example 1: Simulate isothermal measurement
##=====##

times <- seq(0, 500)
run_MC_ISO(A = 0.20,
  rho = 0.007,
```

```

        times = times) %>%
    calc_RLumCarlo() %>%
    plot_RLumCarlo(legend = T)

## End(Not run)

```

---

run\_MC\_LM\_OSL

*Run Monte-Carlo simulation for LM-OSL*


---

## Description

Run Monte-Carlo simulation for LM-OSL

## Usage

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

## Arguments

A	numeric
rho	numeric
times	vector (with default)
clusters	numeric (with default):
r	numeric (with default):
delta.r	numeric (with default):
N_e	numeric (with default):
method	character (with default):
output	character (with default):
...	further arguments

## Value

This function returns a list.

## Function version

0.0.1 [2017-01-27]

## Author(s)

Johannes Friedrich, University of Bayreuth (Germany)

## References

Pagonis 2017

## Examples

```
## Not run:

##TODO: Primary example, should be verified
run_MC_LM_OSL(A = 10000, rho = 0.0001, times = 1:100, clusters = 10, r = NULL,
  delta.r = 0.1,
  N_e = 200, method = "par", output = "signal") %>%
  calc_RLumCarlo() %>%
  plot_RLumCarlo(norm = T)

## End(Not run)
```

run\_MC\_TL

*Run Monte-Carlo simulation for TL*

## Description

Run Monte-Carlo simulation for TL

## Usage

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

## Arguments

s	list
E	numeric
rho	numeric
r_c	numeric (with default)
times	vector (with default)
clusters	numeric (with default):
N_e	numeric (with default):
delta.r	numeric (with default):
method	character (with default):
output	character (with default):
...	further arguments

## Value

This function returns an [array](#) with dimension length(times) x length(r) x clusters

## Function version

0.0.1 [2017-01-27]

**Author(s)**

Johannes Friedrich, University of Bayreuth (Germany)

**References**

Pagonis 2017

**Examples**

```
## Not run:
##=====##
## Example 1: Simulate TL measurement
##=====##

times <- seq(200, 500) # time = temperature

run_MC_TL(s = 3.5e12,
          E = 1.45,
          rho = 0.015,
          r_c = 0.85,
          times = times) %>%
  calc_RLumCarlo() %>%
  plot_RLumCarlo(legend = T)

## End(Not run)
```

---

run\_MC\_TL\_DELOC

---

*Run Monte-Carlo simulation for TL for GOT model*


---

**Description**

##TODO

**Usage**

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

**Arguments**

s	<a href="#">numeric</a> ( <i>required</i> )
E	<a href="#">numeric</a> ( <i>required</i> )
times	<a href="#">numeric</a> (with default)
clusters	<a href="#">numeric</a> (with default):
N_e	<a href="#">integer</a> (with default)
n_filled	<a href="#">integer</a> (with default)
R	<a href="#">numeric</a> (with default):
method	<a href="#">character</a> (with default):
output	<a href="#">character</a> (with default):
...	further arguments



**Details**

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

**Value**

This function returns an [array](#) with dimension length(times) x length(r) x clusters

**Function version**

0.0.1

**Author(s)**

Sebastian Kreutzer, IRAMAT-CRP2A, UMR 5060, CNRS - Université Bordeaux Montaigne (France)

**References**

##TODO

**Examples**

```
##=====##
## Example 1: Simulate TL
##=====##
## Not run:
run_MC_TL_DELOC(
  s = 3.5e12,
  E = 1.45,
  R = 1,
  times = 100:450) %>%
  calc_RLumCarlo() %>%
  plot_RLumCarlo(legend = T)

## End(Not run)
```

# Index

array, [2](#), [7](#), [9](#)

calc\_RLumCarlo, [2](#)

character, [4–8](#)

data.frame, [2](#), [3](#)

integer, [8](#)

list, [7](#)

logical, [3](#)

numeric, [3–8](#)

plot\_RLumCarlo, [3](#)

RLumCarlo-package, [2](#)

run\_MC\_CW\_IRSL, [3](#)

run\_MC\_ISO, [5](#)

run\_MC\_LM\_OSL, [6](#)

run\_MC\_TL, [7](#)

run\_MC\_TL\_DELOC, [8](#)

vector, [4–7](#)