Ancient TL

www.ancienttl.org · ISSN: 2693-0935

Mercier, N., Kreutzer, S., Christophe, C., Guérin, G., Guibert, P., Lahaye, C., Lanos, P., Philippe, A. and Tribolo, C., 2016. *Bayesian statistics in luminescence dating: The 'baSAR'-model and its implementation in the R package 'Luminescence'.* Ancient TL 34(2): 14-21.
https://doi.org/10.26034/la.atl.2016.504

Ancient TL

Bayesian statistics in luminescence dating: The 'baSAR'-model and its implementation in the R package 'Luminescence'

Norbert Mercier,[1*] Sebastian Kreutzer,[1] Claire Christophe,[1,2]
Guillaume Guérin,[1] Pierre Guibert,[1] Christelle Lahaye,[1] Philippe Lanos,[1]
Anne Philippe,[2,3] and Chantal Tribolo[1]

[1]IRAMAT-CRP2A, UMR 5060 CNRS- Université Bordeaux Montaigne, Pessac, France
[2]Laboratoire de mathématiques Jean Leray, UMR 6629 CNRS, Université de Nantes, Nantes, France
[3]ANJA INRIA, Campus universitaire de Beaulieu, Rennes, France

[*]Corresponding Author: norbert.mercier@u-bordeaux-montaigne.fr
Received: July 6, 2016; in final form: September 13, 2016

Abstract

A function named `analyse_baSAR()` was written using the statistical programming language R and its code is now available within the R package 'Luminescence'. The function allows the application of the Bayesian hierarchical model 'baSAR' proposed by Combès et al. (2015) and comes with additional features to analyse luminescence data in a straight forward way. Example scripts are provided showing the possible numerical and graphical outputs.

Keywords: R, Bayesian statistics, Luminescence dating, Hierarchical model, SAR

1. Introduction

Analysing the distribution of obtained equivalent doses (D_e), to estimate a D_e and its standard error ($se(D_e)$) best representing the data after applying the single aliquot regenerative-dose (SAR) protocol (Murray & Wintle, 2000), either on single or multi-grain aliquots, is a vital step of the luminescence dating process. A summary of commonly applied statistical approaches in the luminescence dating community with their individual limitations is given by Galbraith & Roberts (2012).

Additionally, several differing suggestions have been made in the past to use Bayesian statistics (Bayes 1773; cf., e.g., Buck et al. 1996; Gelman et al. 2013 for a general introduction) for particular problems while analysing luminescence data and best estimate the D_e (or age) of particular interest, with regard to the underlying geochronological problem, e.g., Rhodes et al. (2003); Huntriss (2008); Peng & Dong (2014); Cunningham et al. (2015); Zink (2015).

It is out of the scope of this paper to compare or repeat details of the so far presented approaches, but we will rather focus on the work presented by Combès et al. (2015). They proposed an alternative to the central age (equivalent dose) model (usually termed: CAM) by Galbraith et al. (1999) using Bayesian statistics. The model by Combès et al. (2015) comprises a hierarchical structure in the sense that their central equivalent dose (in analogy to Combès et al. 2015 hereafter $\mathcal{D}$) is not directly deduced from the individual D_e values and their associated standard errors, as it is the case in the CAM model, but (a) from the normalised luminescence signal ratios (L_x/T_x) of each aliquot or grain leading to these individual values, and (b) from the type of function representing the distribution of the D_e values; the model is termed 'baSAR' henceforth. Combès et al. (2015) have chosen a Cauchy distribution to describe the dispersion around $\mathcal{D}$, as it *"[...] has the advantage of being close to a Gaussian near the mode and of having heavy tails [..]"* (Combès et al., 2015, p. 67) and with this allowing more spread in the data than a log-normal (or normal) distribution and ensures a greater statistical robustness. The $\mathcal{D}$ value defines the mode of the distribution and another parameter, $\sigma_{\mathcal{D}}$, measures the dispersion of the individual doses around $\mathcal{D}$.

Nevertheless, though the proposed baSAR-model, originally implemented using the language BUGS, has been tested on a series of samples (Guérin et al., 2015) and proved some advantages over the CAM model, its use by the luminescence community was so far limited due to a lack of any available user-friendly and flexible code. Here, we present the function `analyse_baSAR()` available in the **R** (R Development Core Team, 2016) package 'Luminescence' (version

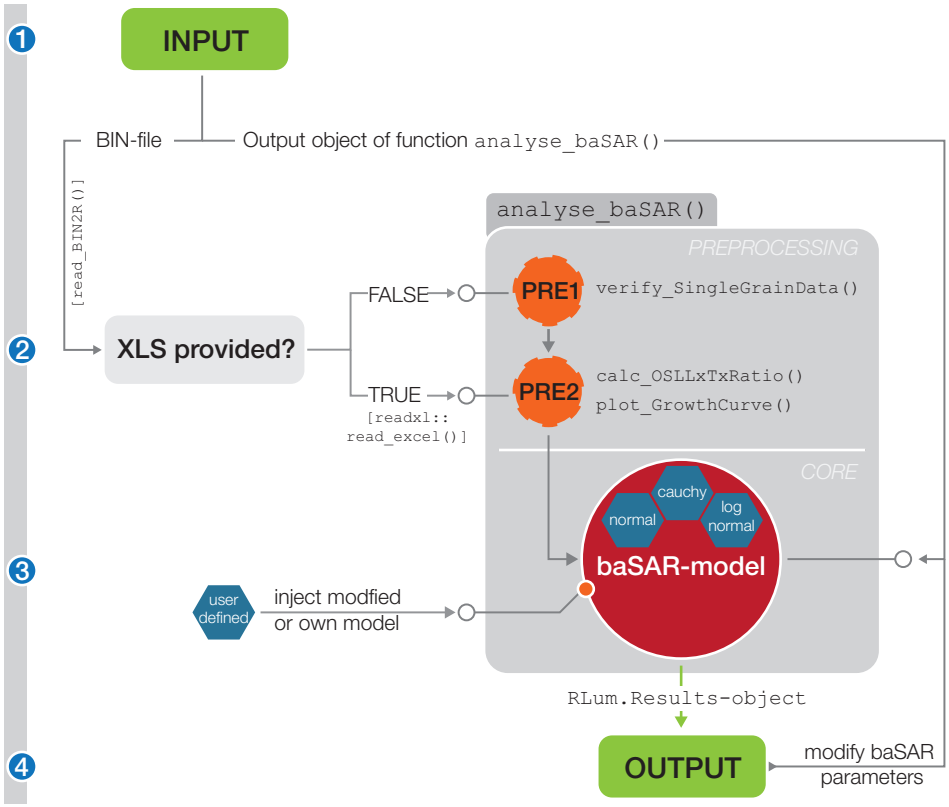

Figure 1. Possible input and output scenarios implemented in the function `analyse_baSAR()`. Alternatively of a BIN/BINX-file that is automatically imported using the function `read_BIN2R()` the user can provide the `RisoeBINfileData-object` produced by this function. This avoids potentially time consuming repeated imports of the measurement data in scenarios where the user wants to play with the function arguments. Furthermore, instead of an XLS-file a `data.frame` of similar structure is accepted as input. For further details see main text.

>= 0.6.4, Kreutzer et al. 2012 and Kreutzer et al. 2016, see also Dietze et al. 2013 and Fuchs et al. 2015 for guides and introductions). The function is not a copy & paste implementation of the proposed model, but a consequent enhancement of the published model, combined with the data processing features by the **R** package 'Luminescence'.

With our contribution we provide technical details on the implementation of the `analyse_baSAR()` function and running examples that can directly be applied by the user on own data. The code is provided under General Public Licence (GPL-3) conditions.

Below **R** code snippets are given as separated listings and typed in `monospace` letters. If not stated otherwise mentioned **R** `functions()` are part of the **R** package 'Luminescence'. Function calls from other **R** packages are indicated by `::`, e.g., `readxl::read_excel()`.

2. Enhancement and implementation

2.1. Enhancing the 'baSAR'-model

The function `analyse_baSAR()` includes all functionalities developed in the baSAR-model and enhances it as suggested by Combès et al. (2015) with regard to additional fitting functions and supported probability distributions.

- While the mathematical function fitting the dose response curve to pairs of L_x/T_x ratios and dose values in the original baSAR-model was limited to (I) a single saturating exponential + linear term

$$f_{\Theta=(a,b,c,d)} : x \rightarrow d + (c \cdot x) + \left(a(1 - exp(\frac{-x}{b}))\right) \quad (1)$$

with its curve parameters a, b, c and d, the function `analyse_baSAR()` includes a further two mathematical functions to describe the dose-response curve:

(II) a linear function

$$f_{\Theta=(c,d)} : x \rightarrow d + (c \cdot x) \quad (2)$$

(III) a single exponential function

$$f_{\Theta=(a,b,d)} : x \rightarrow d + \left(a(1 - exp(\frac{-x}{b}))\right) \quad (3)$$

- furthermore, the user now has the option to include the recycling point(s) in the calculation, and to force the dose response curve through the origin,
- and finally, in order to improve the application of the Bayesian statistics to the luminescence data, the function supports, in addition of the Cauchy distribution, a

Gaussian (normal) distribution and a log-normal distribution which can be chosen to characterise the dispersion of the individual D_e values.

2.2. Implementing `analyse_baSAR()`

From a technical point of view, the `analyse_baSAR()` function uses the software *JAGS* (Just Another Gibbs Sampler; Plummer 2003) available via the **R** interface 'rjags' (Plummer et al., 2016), which is a tool for the analysis of Bayesian hierarchical models using a Markov chain Monte Carlo (MCMC) simulation. The software *JAGS* needs to be installed separately and additionally to the **R** environment. Internally the function is separated in two parts: (I) a Bayesian core, i.e., the implementation of the baSAR-model and (II) a data preprocessing part (cf. Fig. 1).

This separation allows rather complex input/output scenarios and flexible data handling. Fig. 1 drafts a generalised view of possible workflow scenarios. Running examples with its function arguments are given in Sec. 3 and in the supplement. The subsequent numbering was chosen in accordance with the one (numbers in the blue circle) in Fig. 1.

1. The function distinguishes between two different input scenarios: (A) raw measurement data, e.g., a BIN/BINX-file, which is the standard output of Risø TL/OSL readers and (B) an output object produced by the function itself. If the latter one is provided the function automatically starts with the Bayesian calculation and the data preprocessing is skipped. If measurement data are provided (BIN-file), the measurement data are imported into the **R** session. For BIN/BINX-files this is done using the function `read_BIN2R()`.

2. Along with the measurement data (A) an *MS Excel*^{TM} (file ending either *.XLS or *.XLSX)[1] can be provided to limit the measurement data to the aliquots specified in the table (see screenshot in Fig. 2), i.e. the data processing will be continued with a reduced, previously selected dataset. If no XLS-file is provided (B) the data will be piped to the function `verify_SingleGrainData()`[2] to remove dim aliquots (not curves!), as such aliquots would bias the output. Removing dim or zero light grains (aliquots) is a usual task while dealing with single grain data.

   Once the data had been selected with either the one (A) or other (B) approach, L_x/T_x ratios are calculated from the single curves using the function `calc_OSLLxTxRatio()`. The data would be now ready for the baSAR-model, but they will be first piped to the function `plot_GrowthCurve()` to calculate D_e and D_0 values and, if wanted, it allows a visual feedback of the data. Nevertheless, these values (D_e and D_0) are not taken into account for the subsequent modelling, but are returned and can be used for further data subsetting, e.g., sorting grains by D_0 values.

3. After the data preprocessing is finished the ordinary Bayesian modelling starts as described by Combès et al. (2015) internally using the package 'rjags' and the software *JAGS*. The result is a comprehensive object of type `RLum.Results` (see supplement for examples and more details).

4. The results of the modelling can be used for further data processing or directly piped for another run into the function `analyse_baSAR()` itself. As written above, in the latter case the entire data preprocessing is skipped and the function jumps into the baSAR-model core, remembering the previous set function arguments, but the user can modify parameters on request for the Bayesian calculation, e.g., number of MCMC runs.

	A	B	C
1	**BIN_FILE NAME**	**DISC**	**GRAIN**
2	BINfile 1	4	27
3			
4	BINfile 2	11	89
5	BINfile 3	6	51
6			
7	BINfile 3	13	6
8	BINfile 3	3	22

Figure 2. Screenshot of an example *MS Excel*^{TM} sheet that can be provided as input to limit the number of aliquots according to the disc and grain number. As shown in the figure empty rows are allowed to structure the table and they will be ignored during the import.

The injection of own and/or modified models is possible in every scenario (see below). The current implementation is limited to BIN/BINX-files only.

3. Working example

In this section a simplified example of the function input and output is given for the current implementation. The details given for specific function arguments are intentionally vague. They may change in the future due to a continuous development process and would here remain of limited use for the reader. An always up-to-date and detailed description of the function arguments can be found by typing `?analyse_baSAR` in the **R** terminal.

3.1. Constructing the function call

In the following function call, the user provides a list of BIN-files (argument `object`) that were measured on different readers or at different dates.

```
results <- analyse_baSAR(
    object <- list(
        "Bin1.bin",
        "Bin2.binx",
        "Bin3.bin"),
```

[1] The import is realised using the function `readxl::read_excel()` (Wickham et al., 2016)

[2] Type `?verify_SingleGrainData` in the **R** terminal for further information.

As usually irradiations are carried out as durations and not as doses, the dose rate and its standard error of the irradiation source (argument `source_doserate`) needs to be provided[3] for each BIN-file. The standard error of the source dose rate is considered to be systematic and is therefore only added at the end to the standard error of the $\mathcal{D}$.

```
    source_doserate = list(
          c(0.10, 0.001),
          c(0.13, 0.002),
          c(0.12, 0.001)),
```

As stated above, to increase the flexibility of the calculation, the user can pass a list of discs/grains, each pair defining an aliquot, to be included in the calculation. This is possible either in the form of a `data.frame` or in using an *MS Excel*^{TM} sheet. The user might also prefer to use the `verify_SingleGrainData()` function, included in the `analyse_baSAR()` function, which automatically provides a list of grains exhibiting luminescence signals significantly higher than a pre-selected threshold (see `?verify_SingleGrainData`). Regardless of this option, in this example an XLS-file (argument `XLS_file`) is passed to the function comprising a sheet (`sheet`) with a pair list of discs and grains. This selection is further limited to the aliquots 1 to 30 by `aliquot_range`. Please note that currently the argument `aliquot_range` works only if an XLS-file is provided or the output of the function itself `analyse_baSAR()` is provided as input.

```
    XLS_file = "~/Bayesian/Site/Sample.xls",
    sheet = "Disc-Grain-list",
    aliquot_range = c(1:30),
```

In the next step signal and background integration limits are set and additional uncertainty (`sig0`) is added to each resulting L_x/T_x value, the over-dispersion of the count distribution (`sigmab`) is set to 0 in this example.

```
    signal.integral = c(5:10),
    signal.integral.Tx = c(5:10),
    background.integral = c(40:60),
    background.integral.Tx = c(40:60),
    sigmab = 0,
    sig0 = 0.025,
```

Controlling the Bayesian modelling is an important option and several arguments are provided to control the process. Here, namely the chosen probability `distribution` (`'cauchy'`, `'normal'` or `'log_normal'`), the number of Markov chain Monte Carlo runs (`n.MCMC`; default 100,000) and the applied fitting function (`fit.method`) and its options (`fit.force_through_origin` and `fit.includingRecyclingPoints`). The fitting arguments chosen here are also used during the data processing.

Deeper control of the modelling process is granted via argument provided via the `method_control`. In the example the number of used MCMC is set.

```
    distribution = "normal",
    n.MCMC = 100000,
    fit.method = "EXP",
    fit.force_through_origin = TRUE,
    fit.includingRecyclingPoints = TRUE,
    method_control = list(
     n.chains = 3),
```

The last arguments to be set control various terminal (`verbose`) and plot output (`plot`, `output.plot`, `plot_reduced`) options.

```
    plot = TRUE,
    output.plot = TRUE,
    plot_reduced = TRUE,
    verbose = TRUE
  )
```

The complete function call (putting the single snippets together) becomes:

Listing 1. Example combined function call

```
results <- analyse_baSAR(
    object <- list(
          "Bin1.bin",
          "Bin2.binx",
          "Bin3.bin"),
    source_doserate = list(
          c(0.10, 0.001),
          c(0.13, 0.002),
          c(0.12, 0.001)),
    XLS_file = "~/Bayesian/Site/Sample.xls",
    sheet = "Disc-Grain-list",
    aliquot_range = c(1:30),
    signal.integral = c(5:10),
    signal.integral.Tx = c(5:10),
    background.integral = c(40:60),
    background.integral.Tx = c(40:60),
    sigmab = 0,
    sig0 = 0.025,
    distribution = "normal",
    n.MCMC = 100000,
    fit.method = "EXP",
    fit.force_through_origin = TRUE,
    fit.includingRecyclingPoints = TRUE,
    method_control = list(
     n.chains = 3),
    plot = TRUE,
    output.plot = TRUE,
    plot_reduced = TRUE,
    verbose = TRUE
)
```

The code line numbers are similar to the one of the code snippets before as there are extracted from the combined call. The example function call appears rather complex and the number of arguments might be confusing, but most of them are preset and can be modified on request.

3.2. Graphical and terminal output

The (reduced) graphical feedback of the function is shown in Fig. 3, 4 and 5 providing useful information regarding the convergence process (Fig. 3) of the Bayesian analysis (three Markov chains are created by the code): convergence is observed if the slope of the trend of the red, green and black lines is almost zero, i.e., the posterior distributions (here three chains) had converged, the individual doses (Fig. 4), the dose response curve and the conventional D_e with the $\mathcal{D}$ marked within (Fig. 5).

[3] `source_doserate` is a required argument; leaving this argument empty will stop the function from running.

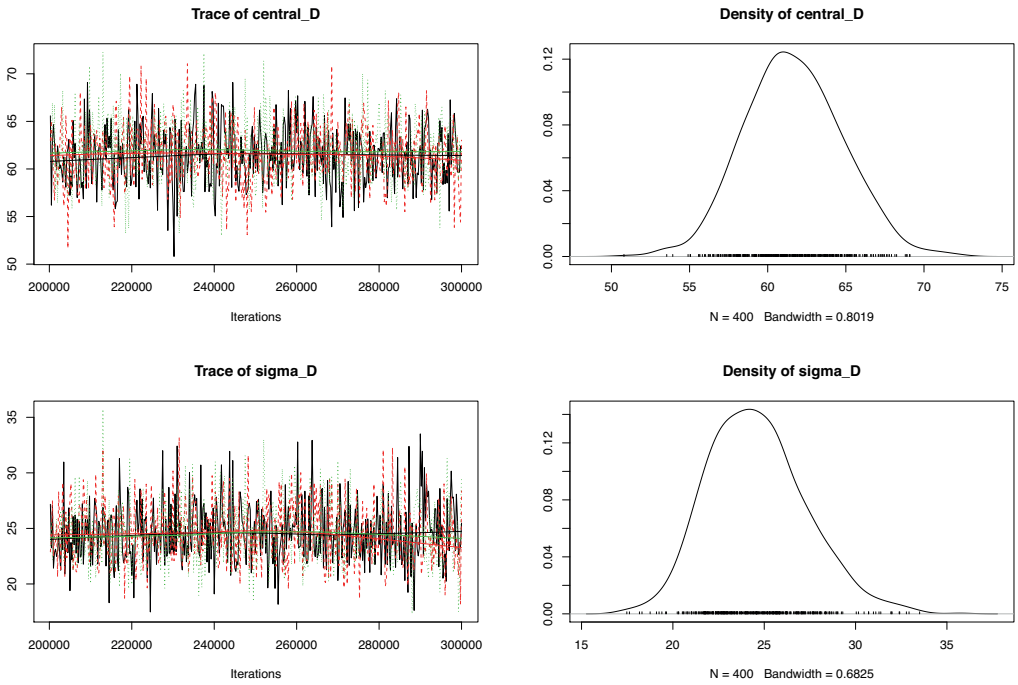

Figure 3. Exemplary plot output of the function `analyse_baSAR()`. The lefthand figures show three Markov chains, each one being a sample of the posterior distribution of $\mathcal{D}$ (top) and $\sigma_{\mathcal{D}}$ (bottom). The righthand figures are estimates of the posterior density functions for these two variables.

The corresponding **R** terminal output may look like the example given below.[4]

Listing 2. Exemplary **R** terminal output

```
[analyse_baSAR()] ------ RESULTS ------

-----------------------------------------------
Used distribution:                  normal
Number of aliquots used:            128/128
Considered fitting method:          EXP
Number MCMC iterations:             100000
-----------------------------------------------

                        mean   sd    HPD
>> Central dose:        61.24  2.82  [58.46;63.95]**
                                     [55.81;66.96]***
>> sigma_D:              23.9   2.39  [21.56;26.21]**
                                     [19.52;29.06]***
>> Final central De:    61.24  2.82      -     -
-----------------------------------------------

(systematic error contribution
to final De: 6.28741e-06 %)
** 68 % level | *** 95 % level
```

The first four lines return the applied parameters used by the baSAR-model. The following three lines report on the estimated $\mathcal{D}$ and $\sigma_{\mathcal{D}}$ and the final $\mathcal{D}$ including the systematic uncertainty provided via `source_doserate`. For $\mathcal{D}$ and $\sigma_{\mathcal{D}}$ the mean, the standard deviation, as well as the highest posterior densities (HPD) at their 68 % and 95 % confidence levels are provided. For an interpretation of the numerical output, the reader is referred to Combès et al. (2015).

The output of the function can now be piped again to the function but with modified parameters, e.g., the distribution is set to `'cauchy'` instead of `'normal'`. In this case the previously produced object `results` is now set as input (`object`) for the new function run.

Listing 3. Use previous output as input

```
results_new <- analyse_baSAR(
        object = results,
        distribution = 'cauchy')
```

Finally the obtained $\mathcal{D}$ is combined and plotted along with the D_e values in an abanico plot (Dietze et al., 2016) in Fig. 5.

3.3. Additional remarks

3.3.1 Selecting records

The function is written to deal with data measured using the SAR protocol only, i.e., the function is searching for OSL/IRSL curves following the pattern proposed by Murray & Wintle (2000). Any additional curve / measurement steps not belonging to this original protocol are not expected and need to be excluded from the input data set. Unfortunately it is not possible to account for all potential types of minor protocol modifications commonly applied in a particular laboratory. Therefore, if a BIN-file is provided, the function `analyse_baSAR()` respects the record selection made previously. This selection was made either with the software *Analyst* (Duller, 2015) or the base **R** function `subset()`. If the package 'Luminescence' is attached (the normal case if the `analyse_baSAR()` function is to be used) the function `subset` can be used in combination with objects produced by the function `read_BIN2R()`. In the example below only

[4] The output has been modified for a correct typesetting and the appearance in the **R** terminal may be different.

OSL curves are selected from the input data set.

Listing 4. Reduce BIN-file record to OSL curves

```
BIN_file_selection <- subset(
        x = BIN_file,
        subset = LTYPE == 'OSL')
```

The available list of selection criteria is defined by the corresponding file version of the BIN-file itself[5]. See the supplement for further examples of subsetting.

3.3.2 Additional parameters

Typing `?analyse_baSAR` in the **R** terminal reveals that the function `analyse_baSAR()` has an argument represented by three dots (`...`). This placeholder allows additional arguments to be passed to lower-level functions. These arguments are not listed as explicit function arguments and not necessary to run the function (usually because they have a meaningful default value). For example: an argument `skip` is passed to the function `read_excel::readxl()` and tells it to ignore the specified number of rows in the *Excel*^{TM} while importing the data. Please see the manual of the function for further information (`?analyse_baSAR`).

[5]It equals the list of columns in the software *Analyst*

3.3.3 User-defined model

In their paper, Combès et al. (2015) had chosen a Cauchy distribution, where the mode is defined by $\mathcal{D}$, and even though the function `analyse_baSAR()` allows normal and log-normal distributions as well, the user has the possibility to define their own model; in such cases, the model has to be passed to the function as a simple string of characters following the example below:

```
my_model <- "model {
  central_D ~ dunif(lower_centralD,upper_centralD)
  precision_D ~ dt(0, pow(0.16*central_D, -2), 1)T(0, )
  sigma_D <- 1/sqrt(precision_D)

 for (i in 1:Nb_aliquots) {
  #Priors
  a[i] ~  dnorm(6.5, 1/(9.2^2) ) T(0, )
  b[i] ~  dnorm(50, 1/(1000^2) )  T(0, )
  c[i] ~  dnorm(1.002, 1/(0.9^2) ) T(0, )
  g[i] ~  dnorm(0.5, 1/(2.5^2) ) I(-a[i], )
  sigma_f[i]  ~  dexp (20)

 #Cauchy distribution
 D[i] ~ dt ( central_D , precision_D, 1)

 #Likelihood
 S_y[1,i] <- 1/(sLum[1,i]^2 + sigma_f[i]^2)
 Lum[1,i] ~ dnorm ( Q[1,i] , S_y[1,i])
 Q[1,i]  <-
```

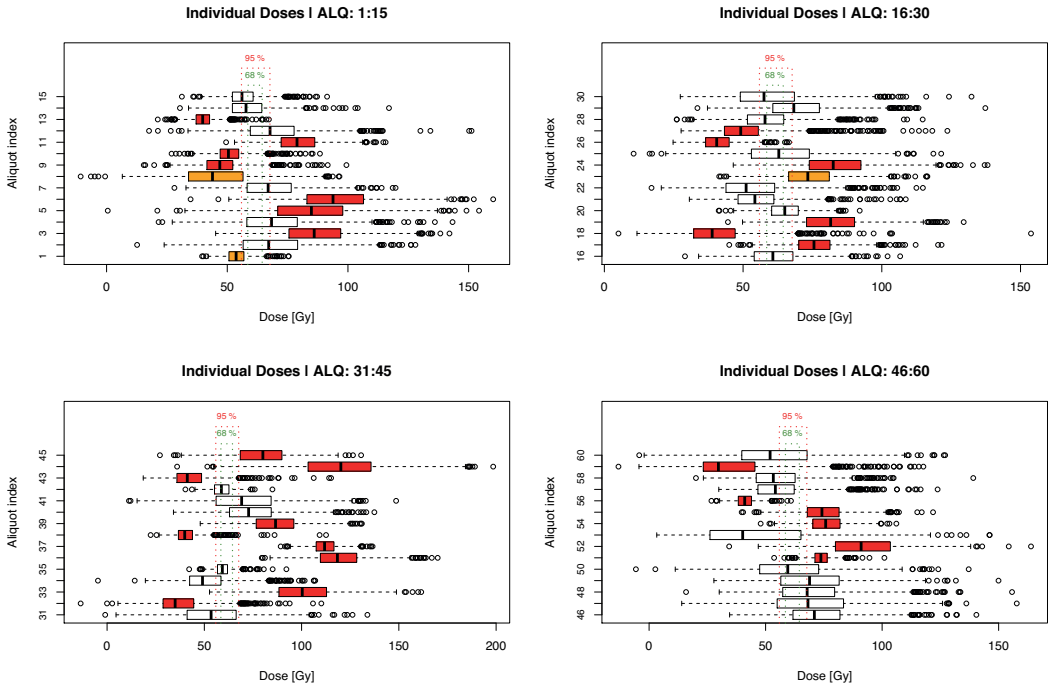

Figure 4. Boxplots of individual doses obtained during the bayesian calculation. Each box represents 50 % (interquartile range, IQR) of the data, the whiskers extend to 1.5 times this range. The HPD at its 68 % and 95 % level is indicated by the dashed lines (green and red). The plots may help to identify extreme values that might be worth a 2nd look. Box colours indicate distances of the IQR of the aliquots from the HPD. Chosen colour code: IQR outside of the HPD - 68 %: orange, IQR outside of the HPD - 95 %: red. All other boxes are coloured white. The aliquot index is indicated on the y-axis.

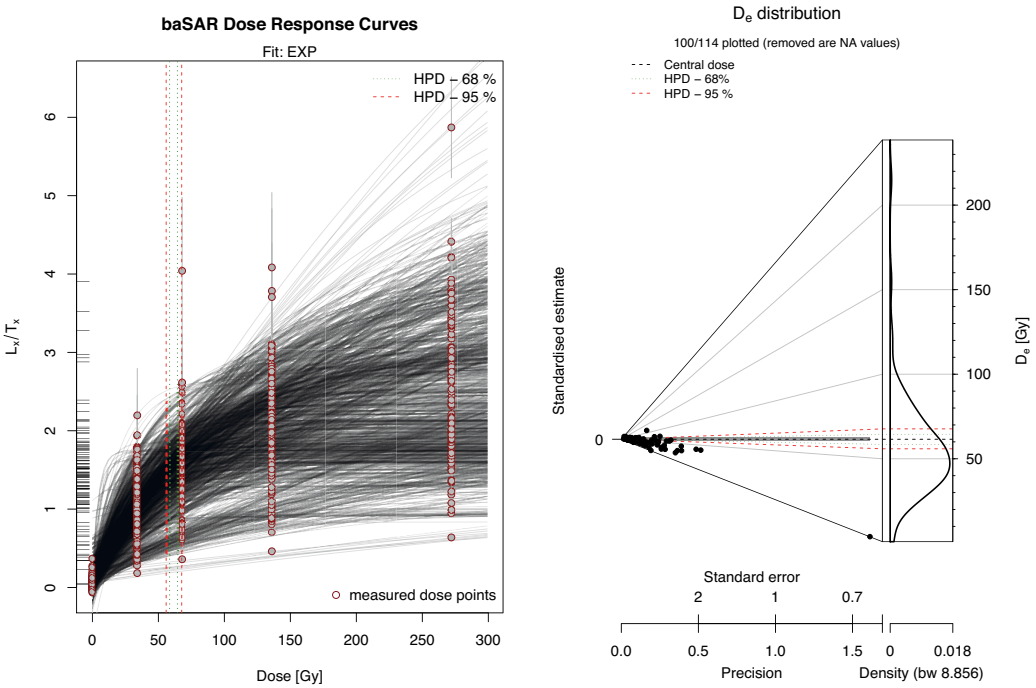

Figure 5. Dose response curves obtained using the baSAR-model (left plot) and dose distribution plot (right plot, here abanico plot; (Dietze et al., 2016)). Left plot: the individual dose response curves are plotted with the L_x/T_x values (measured dose points) used as input for the baSAR-model. For graphical reasons the maximum number of curves is limited to 1000 (randomly chosen). The plot allows to evaluate the general succession of the analysis. The right plot presents the D_e (not individual dose!) distribution. These D_e values are calculated during the preprocessing and can be considered as the 'conventional' approach of the data analysis. By contrast, the HPDs and the $\mathcal{D}$ (central dose) are indicated within the plot (dashed lines).

```
  GC_Origin * g[i] + LinGC * (c[i] * D[i] ) +
  ExpoGC * (a[i] * (1 - exp (-D[i] /b[i])))

  for (m in 2:Limited_cycles[i]) {
   S_y[m,i] <-  1/(sLum[m,i]^2 + sigma_f[i]^2)
   Lum[m,i] ~ dnorm( Q[m,i] , S_y[m,i] )
   Q[m,i]  <-
    GC_Origin * g[i] + LinGC*(c[i]*Dose[m,i]) +
    ExpoGC*(a[i] *(1 - exp(-Dose[m,i]/b[i])))
  }
 }
}"
```

For example, changing the numerical values for the first prior requires a modification of the *JAGS* code itself, i.e. the lines:

```
  #Priors
  a[i] ~  dnorm(6.5, 1/(9.2^2) ) T(0, )
```

might become

```
  #Priors
  a[i] ~  dnorm(10, 1/(100^2) ) T(0, )
```

However, it is worth mentioning that for any change of the priors or underlying assumption of the model **a rigour scientific justification is indispensable**.

Run a user-defined model is simply done by adding in the list parameters the following line:

Listing 5. Run a user-defined model

```
results <- analyse_baSAR(
 ...,
baSAR_model = my_model,
 ...
)
```

Please note that in cases where a new or a modified model is provided, the previously set of variables need to be respected, e.g., the variable `precision_D` (cf. model example above) must not be renamed, otherwise the function will crash.

4. Conclusions

An implementation and enhancement of the central dose model (baSAR) proposed by Combès et al. (2015) for the programming language **R** and the **R** package 'Luminescence' were presented along with examples. The baSAR-model can applied on single grain and multi grain aliquots measured with the SAR (Murray & Wintle, 2000) protocol. For the Bayesian modelling the software *JAGS* (Plummer, 2003) and the **R** package 'rjags' (Plummer et al., 2016) as interface is used.

This contribution did not present or discuss the underlying statistical assumptions, for this the user is referred to Combès et al. (2015). Finally it should be stressed that the availability of an easy to use software solution does not free

the user from carefully checking its own data and verify underlying assumptions.

Nota bene: For easy copy & paste code snippets and running examples the reader is referred to the supplement.

Acknowledgments

The authors gratefully acknowledge financial support by the LaScArBx. LaScArBx is a research programme supported by the ANR (ANR-10-LABX-52). SK is also thankful for the invaluable support by the **R** 'Luminescence' team and the cooperation and personal exchange funded by the DFG (SCHM 3051/3-1) in the framework of the programme "Scientific Networks", project title: "RLum.Network: Ein Wissenschaftsnetzwerk zur Analyse von Lumineszenzdaten mit **R**".

References

Bayes, T. *An Essay Toward Solving a Problem in the Doctrine of Chances*. Philosophical Transactions of the Royal Society of London, 53: 370–418, 1773.

Buck, C. E., Cavanagh, W. G., and Litton, C. *Bayesian Approach to Intrepreting Archaeological Data*. Wiley, 1996.

Combès, B., Philippe, A., Lanos, P., Mercier, N., Tribolo, C., Guérin, G., Guibert, P., and Lahaye, C. *A Bayesian central equivalent dose model for optically stimulated luminescence dating*. Quaternary Geochronology, 28: 62–70, 2015.

Cunningham, A. C., Evans, M., and Knight, J. *Quantifying bleaching for zero-age fluvial sediment: A Bayesian approach*. Radiation Measurements, 2015.

Dietze, M., Kreutzer, S., Fuchs, M. C., Burow, C., Fischer, M., and Schmidt, C. *A practical guide to the R package Luminescence*. Ancient TL, 31(1): 11–18, 2013.

Dietze, M., Kreutzer, S., Burow, C., Fuchs, M. C., Fischer, M., and Schmidt, C. *The abanico plot: visualising chronometric data with individual standard errors*. Quaternary Geochronology, 31: 12–18, 2016.

Duller, G. A. T. *The Analyst software package for luminescence data: overview and recent improvements*. Ancient TL, 33(1): 35–42, 2015.

Fuchs, M. C., Kreutzer, S., Burow, C., Dietze, M., Fischer, M., Schmidt, C., and Fuchs, M. *Data processing in luminescence dating analysis: An exemplary workflow using the R package 'Luminescence'*. Quaternary International, 362: 8–13, 2015.

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

Galbraith, R. F., Roberts, R. G., Laslett, G. M., Yoshida, H., and Olley, J. M. *Optical dating of single and multiple grains of Quartz from Jinmium Rock Shelter, Northern Australia: Part I, Experimental design and statistical models*. Archaeometry, 41(2): 339–364, 1999.

Gelman, A., Carlin, J. B., Stern, H. S., Dunson, D. B., Vehtari, A., and Rubin, D. B. *Bayesian Data Analysis, Third Edition*. CRC Press, 2013.

Guérin, G., Combès, B., Lahaye, C., Thomsen, K. J., Tribolo, C., Urbanova, P., Guibert, P., Mercier, N., and Valladas, H. *Testing the accuracy of a Bayesian central-dose model for single-grain OSL, using known-age samples*. Radiation Measurements, 2015.

Huntriss, A. *A Bayesian analysis of luminescence dating*. PhD thesis, University of Durham, 2008.

Kreutzer, S., Schmidt, C., Fuchs, M. C., Dietze, M., Fischer, M., and Fuchs, M. *Introducing an R package for luminescence dating analysis*. Ancient TL, 30(1): 1–8, 2012.

Kreutzer, S., Dietze, M., Burow, C., Fuchs, M. C., Schmidt, C., Fischer, M., Friedrich, J., Mercier, N., Smedley, R. K., Durcan, J., and King, G. *Luminescence: Comprehensive Luminescence Dating Data Analysis*, 2016. URL https://CRAN.R-project.org/package=Luminescence. R package version 0.6.4.

Murray, A. S. and Wintle, A. G. *Luminescence dating of quartz using an improved single-aliquot regenerative-dose protocol*. Radiation Measurements, 32(1): 57–73, 2000.

Peng, J. and Dong, Z. *A simple Bayesian method for assessing the standard error of equivalent dose estimates*. Ancient TL, 32(2): 17–23, 2014.

Plummer, M. *JAGS: A Program for Analysis of Bayesian Graphical Models using Gibbs Sampling*. DSC 2003 Working Papers, 124, April 2003. URL http://mcmc-jags.sourceforge.net.

Plummer, M., Stukalov, A., and Denwood, M. *Interface to the JAGS MCMC library*, 2016. URL https://CRAN.R-project.org/package=rjags. R package version 4-6.

R Development Core Team. *R: A Language and Environment for Statistical Computing*. Vienna, Austria, 2016.

Rhodes, E. J., Bronk R., C., Outram, Z., Batt, C., Willis, L., Dockrill, S., and Bond, J. *Bayesian methods applied to the interpretation of multiple OSL dates: high precision sediment ages from Old Scatness Broch excavations, Shetland Isles*. Quaternary Science Reviews, 22(10-13): 1231–1244, 2003.

Wickham, H., RStudio, Kalicinski, M., Valery, K., Leitienne, C., Colbert, B., and Hoerl, D. *Import excel files into R. Supports '.xls' via the embedded 'libxls' C library (http://sourceforge.net/projects/libxls/) and '.xlsx' via the embedded 'RapidXML' C++ library*, 2016. URL https://CRAN.R-project.org/package=readxl. R package version 0.1.1.

Zink, A. J. C. *Bayesian analysis of luminescence measurements*. Radiation Measurements, 2015.

Reviewer

Alastair Cunningham