# Package 'BLOQ'

October 12, 2022

**Type** Package

**Version** 0.1-1

**Date** 2020-06-03

**Title** Impute and Analyze Data with BLOQ Observations

**Maintainer** Vahid Nassiri <vahid.nassiri@openanalytics.eu>

**Description** It includes estimating the area under the concentrations versus time curve (AUC) and its standard error for data with Below the Limit of Quantification (BLOQ) observations. Two approaches are implemented: direct estimation using censored maximum likelihood, also by first imputing the BLOQ's using various methods, then compute AUC and its standard error using imputed data. Technical details can found in Barnett, Helen Yvette, Helena Geys, Tom Jacobs, and Thomas Jaki. ``Methods for Non-Compartmental Pharmacokinetic Analysis With Observations Below the Limit of Quantification." Statistics in Biopharmaceutical Research (2020): 1-12. (available online: <https://www.tandfonline.com/doi/full/10.1080/19466315.2019.1701546>).

**Imports** maxLik, mvtnorm,

**Suggests** testthat,

**License** GPL (>= 2)

**Encoding** UTF-8

**RoxygenNote** 7.1.0

**NeedsCompilation** no

**Author** Vahid Nassiri [cre],
Helen Barnett [aut],
Helena Geys [aut],
Tom Jacobs [aut],
Thomas Jaki [aut]

**Repository** CRAN

**Date/Publication** 2020-06-07 18:30:06 UTC

# R topics documented:



---

`estimateAUCandStdErr` *Estimate AUC and its standard error*

---

## Description

function to estimate AUC and compute standard error of this estimate

## Usage

```
estimateAUCandStdErr(
  imputedData,
  timePoints,
  isMultiplicative = FALSE,
  na.rm = FALSE
)
```

## Arguments

| | |
|---|---|
| `imputedData` | numeric matrix or data frame of size n by J (n the sample size and J the number of time points) |
| `timePoints` | vector of time points |
| `isMultiplicative` | logical variable indicating whether an additive error model (FALSE) or a multiplicative error model (TRUE) should be used |
| `na.rm` | logical variable indicating whether the rows with missing values should be ignored or not. |

## Value

vector of length 2 with estimated AUC and its standard error

## Author(s)

Vahid Nassiri, Helen Yvette Barnett

## Examples

```
# generate data from Beal model with only fixed effects
set.seed(111)
genDataFixedEffects <- simulateBealModelFixedEffects(10, 0.693,
+  1, 1, seq(0.5,3,0.5))
# Impute the data with BLOQ's with one of the provided methods,
# for example, here we use ROS
imputedDataROS <- imputeROS(genDataFixedEffects, 0.1)
# estimate AUC and its standard error
estimateAUCandStdErr(imputedDataROS,seq(0.5,3,0.5))
```

---

# estimateAUCwithCMLperTimePoint

*estimate AUC with censored maximum likelihood per time point*

---

## Description

function to estimate mean and standard error of each column of data with BLOQ's using a censored maximum likelihood (CML) approach, then use these estimates for estimating AUC and its standard error

## Usage

```
estimateAUCwithCMLperTimePoint(
  inputData,
  LOQ,
  timePoints,
  isMultiplicative = FALSE,
  onlyFitCML = FALSE,
  printCMLmessage = TRUE,
  optimizationMethod = NULL,
  CMLcontrol = NULL
)
```

## Arguments

| | |
|---|---|
| inputData | numeric matrix or data frame of the size n by J (n the sample size and J the number of time points) the input dataset |
| LOQ | scalar, limit of quantification value |
| timePoints | vector of time points |

`isMultiplicative`
: logical variable indicating whether an additive error model (FALSE) or a multiplicative error model (TRUE) should be used

`onlyFitCML`
: logical variable with FALSE as default, if TRUE only the censored maximum likelihood estimates will be calculated

`printCMLmessage`
: logical variable with TRUE as default, if TRUE then messages regarding the convergence status of censored log-likelihood maximization will be printed.

`optimizationMethod`
: single string specifying the method to be used for optimizing the log-likelihood, the default is NULL that allows the function to decide the about the best method. Otherwise, one can select among choices available via R package maxLik: "NR" (for Newton-Raphson), "BFGS" (for Broyden-Fletcher-Goldfarb-Shanno), "BFGSR" (for the BFGS algorithm implemented in R), "BHHH" (for Berndt-Hall-Hall-Hausman), "SANN" (for Simulated ANNealing), "CG" (for Conjugate Gradients), or "NM" (for Nelder-Mead). Lower-case letters (such as "nr" for Newton-Raphson) are allowed.

`CMLcontrol`
: list of arguments to control convergence of maximization algorithm. It is the same argument as control in the function maxLik in the R package maxLik

## Value

a list with three components: output of maxLik function, estimated parameters for each column using censored maximum likelihood, and estimated AUC and its standard error.

## Author(s)

Vahid Nassiri, Helen Yvette Barnett

## See Also

maxLik

## Examples

```
# generate data from Beal model with only fixed effects
set.seed(111)
genDataFixedEffects <- simulateBealModelFixedEffects(10, 0.693,
  1, 1, seq(0.5,3,0.5))
# Multiplicative error model
estimateAUCwithCMLperTimePoint(genDataFixedEffects, 0.1, seq(0.5,3,0.5), TRUE)
```

`estimateAUCwithFullCML` *estimate AUC with Full censored maximum likelihood*

## Description

function to estimate mean and and covariance matrix of censored data using a full censored maximum likelihood approach (with a special structure for the covariance matrix which only allows correlations between successive time points), then use these estimates for estimating AUC and its standard error

## Usage

```
estimateAUCwithFullCML(
  inputData,
  LOQ,
  timePoints,
  isMultiplicative = FALSE,
  onlyFitCML = FALSE,
  printCMLmessage = TRUE,
  optimizationMethod = NULL,
  CMLcontrol = NULL,
  na.rm = TRUE
)
```

## Arguments

| | |
|---|---|
| `inputData` | numeric matrix or data frame of the size n by J (n the sample size and J the number of time points) the input dataset |
| `LOQ` | scalar, limit of quantification value |
| `timePoints` | vector of time points |
| `isMultiplicative` | logical variable indicating whether an additive error model (FALSE) or a multiplicative error model (TRUE) should be used |
| `onlyFitCML` | logical variable with FALSE as default, if TRUE only the censored maximum likelihood estimates will be calculated |
| `printCMLmessage` | logical variable with TRUE as default, if TRUE then messages regarding the convergence status of censored log-likelihood maximization will be printed. |
| `optimizationMethod` | single string specifying the method to be used for optimizing the log-likelihood, the default is NULL that allows the function to decide the about the best method. Otherwise, one can select among choices available via R package maxLik: "NR" (for Newton-Raphson), "BFGS" (for Broyden-Fletcher-Goldfarb-Shanno), "BFGSR" |

(for the BFGS algorithm implemented in R), "BHHH" (for Berndt-Hall-Hall-Hausman), "SANN" (for Simulated ANNealing), "CG" (for Conjugate Gradients), or "NM" (for Nelder-Mead). Lower-case letters (such as "nr" for Newton-Raphson) are allowed.

`CMLcontrol` list of arguments to control convergence of maximization algorithm. It is the same argument as control in the function maxLik in the R package maxLik

`na.rm` logical variable indicating whether the lines with missing values should be ignored (TRUE, default) or not (FALSE).

## Value

a list with three components: output of maxLik function, estimated parameters (mean vector and the covariance matrix) using censored maximum likelihood, and estimated AUC and its standard error.

## Author(s)

Vahid Nassiri, Helen Yvette Barnett

## See Also

maxLik

## Examples

```
#' # generate data from Beal model with only fixed effects
set.seed(123)
genDataFixedEffects <- simulateBealModelFixedEffects(10, 0.693,
1, 1, seq(0.5,3,1.5))
estimateAUCwithFullCML(genDataFixedEffects, 0.15, seq(0.5,3,1.5))
```

---

# `estimateAUCwithMVNCML` *estimate AUC with multivariate normal censored maximum likelihood*

---

## Description

function to estimate mean and and covariance matrix of censored data using a full censored maximum likelihood approach (with a special structure for the covariance matrix which only allows correlations between successive time points), then use these estimates for estimating AUC and its standard error

## Usage

```
estimateAUCwithMVNCML(
  inputData,
  LOQ,
  timePoints,
  isMultiplicative = FALSE,
  onlyFitCML = FALSE,
  printCMLmessage = TRUE,
  optimizationMethod = NULL,
  CMLcontrol = NULL,
  na.rm = TRUE,
  isPairwise = FALSE
)
```

## Arguments

| | |
|---|---|
| inputData | numeric matrix or data frame of the size n by J (n the sample size and J the number of time points) the input dataset |
| LOQ | scalar, limit of quantification value |
| timePoints | vector of time points |
| isMultiplicative | logical variable indicating whether an additive error model (FALSE) or a multiplicative error model (TRUE) should be used |
| onlyFitCML | logical variable with FALSE as default, if TRUE only the censored maximum likelihood estimates will be calculated |
| printCMLmessage | logical variable with TRUE as default, if TRUE then messages regarding the convergence status of censored log-likelihood maximization will be printed. |
| optimizationMethod | single string specifying the method to be used for optimizing the log-likelihood, the default is NULL that allows the function to decide the about the best method. Otherwise, one can select among choices available via R package maxLik: "NR" (for Newton-Raphson), "BFGS" (for Broyden-Fletcher-Goldfarb-Shanno), "BFGSR" (for the BFGS algorithm implemented in R), "BHHH" (for Berndt-Hall-Hall-Hausman), "SANN" (for Simulated ANNealing), "CG" (for Conjugate Gradients), or "NM" (for Nelder-Mead). Lower-case letters (such as "nr" for Newton-Raphson) are allowed. |
| CMLcontrol | list of arguments to control convergence of maximization algorithm. It is the same argument as control in the function maxLik in the R package maxLik |
| na.rm | logical variable indicating whether the lines with missing values should be ignored (TRUE, default) or not (FALSE). |
| isPairwise | logical variable, if TRUE the unstructured covariance matrix will be estimated using pairwise approach, otherwise (FALSE, default) the full maximum likelihood will be used with a special structure imposed on the covariance matrix. |

## Value

a list with three components: output of maxLik function, estimated parameters (mean vector and the covariance matrix) using censored maximum likelihood, and estimated AUC and its standard error.

## Author(s)

Vahid Nassiri, Helen Yvette Barnett

## See Also

maxLik

## Examples

```
# generate data from Beal model with only fixed effects
set.seed(111)
genDataFixedEffects <- simulateBealModelFixedEffects(10, 0.693,
1, 1, seq(0.5,3,1.5))
estimateAUCwithMVNCML(genDataFixedEffects, 0.1, seq(0.5,3,1.5))
estimateAUCwithMVNCML(genDataFixedEffects, 0.1, seq(0.5,3,1.5),
isPairwise = TRUE)
```

---

# estimateAUCwithPairwiseCML

*estimate AUCwith pairwise censored maximum likelihood*

---

## Description

function to estimate mean and and covariance matrix of censored data using a full censored maximum likelihood approach via fitting all possible pairs, then use these estimates for estimating AUC and its standard error

## Usage

```
estimateAUCwithPairwiseCML(
  inputData,
  LOQ,
  timePoints,
  isMultiplicative = FALSE,
  onlyFitCML = FALSE,
  optimizationMethod = NULL,
  CMLcontrol = NULL,
  na.rm = TRUE
)
```

## Arguments

| | |
|---|---|
| inputData | numeric matrix or data frame of the size n by J (n the sample size and J the number of time points) the input dataset |
| LOQ | scalar, limit of quantification value |
| timePoints | vector of time points |
| isMultiplicative | logical variable indicating whether an additive error model (FALSE) or a multiplicative error model (TRUE) should be used |
| onlyFitCML | logical variable with FALSE as default, if TRUE only the censored maximum likelihood estimates will be calculated. |
| optimizationMethod | single string specifying the method to be used for optimizing the log-likelihood, the default is NULL that allows the function to decide the about the best method. Otherwise, one can select among choices available via R package maxLik: "NR" (for Newton-Raphson), "BFGS" (for Broyden-Fletcher-Goldfarb-Shanno), "BFGSR" (for the BFGS algorithm implemented in R), "BHHH" (for Berndt-Hall-Hall-Hausman), "SANN" (for Simulated ANNealing), "CG" (for Conjugate Gradients), or "NM" (for Nelder-Mead). Lower-case letters (such as "nr" for Newton-Raphson) are allowed. |
| CMLcontrol | list of arguments to control convergence of maximization algorithm. It is the same argument as control in the function maxLik in the R package maxLik |
| na.rm | logical variable indicating whether the lines with missing values should be ignored (TRUE, default) or not (FALSE). Note that, it will be applied for the sub-datasets regarding each pair. |

## Value

a list with three components: output of maxLik function, estimated parameters (mean vector and the covariance matrix) using censored maximum likelihood, and estimated AUC and its standard error.

## Author(s)

Vahid Nassiri, Helen Yvette Barnett

## See Also

maxLik

## Examples

```
# generate data from Beal model with only fixed effects
set.seed(111)
genDataFixedEffects <- simulateBealModelFixedEffects(10, 0.693,
1, 1, seq(0.5,3,1.5))
estimateAUCwithPairwiseCML(genDataFixedEffects, 0.1, seq(0.5,3,1.5))
```

imputeBLOQ *impute BLOQ's with various methods*

## Description

function to impute BLOQ's. The user can define column-specific methods to impute the BLOQ's.

## Usage

```
imputeBLOQ(inputData, LOQ, imputationMethod, progressPrint = FALSE, ...)
```

## Arguments

| | |
|---|---|
| inputData | numeric matrix or data frame of the size n by J (n the sample size and J the number of time points) the input dataset |
| LOQ | scalar, limit of quantification value |
| imputationMethod | could be a single string or a vector of strings with the same length as the number of time points (ncol(inputData)). If it is left blank, then the imputation is done using kernel density estimation method for the columns with at least one non-BLOQ component. For all the rest (only BLOQ) the constant imputation is used. The allowed values are "constant", "ros", "kernel", "cml" corresponding to constant imputation, imputing using regression on order statistics, imputing using kernel density estimator, and imputing using censored maximum likelihood, respectively. |
| progressPrint | logical variable indicating whether the imputation progress should be printed or not. |
| ... | any other argument which should be changed according to the input arguments regarding the functions corresponding to different imputation methods. |

## Value

a list with two components: imputed dataset, and the methods used to impute each column.

## Author(s)

Vahid Nassiri, Helen Yvette Barnett

## Examples

```
set.seed(111)
inputData <- simulateBealModelFixedEffects(10, 0.693,1, 1, seq(0.5,3,0.5))
LOQ = 0.125
imputeBLOQ(inputData, LOQ,
imputationMethod = c("cml", "ros", "kernel","constant", "constant", "constant"),
maxIter = 500, isMultiplicative = TRUE, constantValue = LOQ)
imputeBLOQ(inputData, LOQ, maxIter = 500, isMultiplicative = TRUE,
constantValue = LOQ/5, epsilon = 1e-04)
```

`imputeCML` *imputing BLOQ's using censored maximum likelihood*

## Description

function to impute BLOQ's using quantiles of a normal distribution with mean and standard error estimates using censored maximum likelihood

## Usage

```
imputeCML(
  inputData,
  LOQ,
  isMultiplicative = FALSE,
  useSeed = runif(1),
  printCMLmessage = TRUE,
  CMLcontrol = NULL
)
```

## Arguments

| | |
|---|---|
| `inputData` | numeric matrix or data frame of the size n by J (n the sample size and J the number of time points) the input dataset |
| `LOQ` | scalar, limit of quantification value |
| `isMultiplicative` | logical variable indicating whether an additive error model (FALSE) or a multiplicative error model (TRUE) should be used |
| `useSeed` | scalar, set a seed to make the results reproducible, default is runif(1), it is used to randomly order the first imputed column (if the first column has any BLOQ's) |
| `printCMLmessage` | logical variable with TRUE as default, if TRUE then messages regarding the convergence status of censored log-likelihood maximization will be printed. |
| `CMLcontrol` | list of arguments to control convergence of maximization algorithm. It is the same argument as control in the function maxLik in the R package maxLik |

## Value

the imputed dataset: a numeric matrix or data frame of the size n by J (n the sample size and J the number of time points)

## Author(s)

Vahid Nassiri, Helen Yvette Barnett

## See Also

maxLik

## Examples

```
# generate data from Beal model with only fixed effects
set.seed(111)
genDataFixedEffects <- simulateBealModelFixedEffects(10, 0.693,
+  1, 1, seq(0.5,3,0.5))
imputeCML(genDataFixedEffects, 0.1, FALSE, 1)
```

---

`imputeConstant` *imputing BLOQ's with a constant value*

---

## Description

function to impute BLOQ observations by replacing them with a constant value.

## Usage

```
imputeConstant(inputData, LOQ, constantValue)
```

## Arguments

| | |
|---|---|
| `inputData` | numeric matrix or data frame of the size n by J (n the sample size and J the number of time points) the input dataset |
| `LOQ` | scalar, limit of quantification value |
| `constantValue` | scalar, the constant value which replaces all BLOQ's, default is LOQ/2 |

## Value

the imputed dataset: a numeric matrix or data frame of the size n by J (n the sample size and J the number of time points)

## Author(s)

Vahid Nassiri, Helen Yvette Barnett

## Examples

```
# generate data from Beal model with only fixed effects
set.seed(111)
genDataFixedEffects <- simulateBealModelFixedEffects(10, 0.693,
+  1, 1, seq(0.5,3,0.5))
# replacing BLOQ's with LOQ/2
imputeConstant(genDataFixedEffects, 0.1, 0.1/2)
```

`imputeKernelDensityEstimation` *imputing BLOQ's using kernel density estimation*

## Description

function to impute BLOQ observations using kernel density estimation.

## Usage

```
imputeKernelDensityEstimation(
  inputData,
  LOQ,
  epsilon = 1e-07,
  maxIter = 1000,
  useSeed = runif(1)
)
```

## Arguments

| | |
|---|---|
| `inputData` | numeric matrix or data frame of the size n by J (n the sample size and J the number of time points) the input dataset |
| `LOQ` | scalar, limit of quantification value |
| `epsilon` | scalar with 1e-07 as default, the difference between two iterations which achieving it would stop the procedure (convergence). |
| `maxIter` | scalar, the maximum number of iterations with 1000 as default. |
| `useSeed` | scalar, set a seed to make the results reproducible, default is runif(1), it is used to randomly order the first imputed column (if the first column has any BLOQ's) |

## Value

the imputed dataset: a numeric matrix or data frame of the size n by J (n the sample size and J the number of time points)

## Author(s)

Vahid Nassiri, Helen Yvette Barnett

## Examples

```
# generate data from Beal model with only fixed effects
set.seed(111)
genDataFixedEffects <- simulateBealModelFixedEffects(10, 0.693,
+  1, 1, seq(0.5,3,0.5))
imputeKernelDensityEstimation(genDataFixedEffects, 0.1, epsilon = 1e-05)
```

imputeROS *imputing BLOQ's using regression on order statistics*

## Description

function to impute BLOQ's with regression on order statistics (ROS) approach.

## Usage

```
imputeROS(inputData, LOQ, isMultiplicative = FALSE, useSeed = runif(1))
```

## Arguments

| | |
|---|---|
| inputData | numeric matrix or data frame of the size n by J (n the sample size and J the number of time points) the input dataset |
| LOQ | scalar limit of quantification value |
| isMultiplicative | logical variable indicating whether an additive error model (FALSE) or a multiplicative model (TRUE) should be used |
| useSeed | scalar, set a seed to make the results reproducible, default is runif(1), it is used to randomly order the first imputed column (if the first column has any BLOQ's) |

## Value

the imputed dataset: a numeric matrix or data frame of the size n by J (n the sample size and J the number of time points)

## Author(s)

Vahid Nassiri, Helen Yvette Barnett

## Examples

```
# generate data from Beal model with only fixed effects
set.seed(111)
genDataFixedEffects <- simulateBealModelFixedEffects(10, 0.693,
+  1, 1, seq(0.5,3,0.5))
imputeROS(genDataFixedEffects, 0.1)
```

`simulateBealModelFixedEffects` *simulate data from Beal model with fixed effects*

## Description

function to generate data from a Beal model with fixed effects

## Usage

```
simulateBealModelFixedEffects(
  numSubjects,
  clearance,
  volumeOfDistribution,
  dose,
  timePoints
)
```

## Arguments

| | |
|---|---|
| `numSubjects` | scalar, number of subject which should be generated |
| `clearance` | scalar, clearance |
| `volumeOfDistribution` | scalar, volume of distribution |
| `dose` | scalar, dose |
| `timePoints` | vector of time points |

## Details

The model used to generate data at time t is as follows

$$y(t) = C(t)\exp(e(t)),$$

where $C(t)$, the PK-model, is defined as follows:

$$C(t) = \frac{\text{dose}}{V_d}\exp\left(CL.t\right),$$

with $V_d$ the volume of distribution and $CL$ as clearance. The error model is consdiered as $e(t) \sim N(0, h(t))$, with:

$$h(t) = 0.03 + 0.165\frac{C(t)^{-1}}{C(1.5)^{-1} + C(t)^{-1}}$$

## Value

generated sample with numSubjects as the number of rows and length of timePoints as the number of columns

## Author(s)

Vahid Nassiri, Helen Yvette Barnett

## See Also

Beal S. L., Ways to fit a PK model with some data below the quantification limit, Journal of Pharmacokinetics and Pharmacodynamics, 2001;28(**5**):481–504.

## Examples

```
set.seed(111)
simulateBealModelFixedEffects(10, 0.693,
+  1, 1, seq(0.5,3,0.5))
```

---

`simulateBealModelMixedEffects`
*simulate data from Beal model with fixed and random effects*

---

## Description

function to generate data from a Beal model with fixed effects

## Usage

```
simulateBealModelMixedEffects(
  numSubjects,
  clearance,
  volumeOfDistribution,
  dose,
  varCompClearance,
  varCompVolumeOfDistribution,
  timePoints
)
```

## Arguments

| | |
|---|---|
| `numSubjects` | scalar, number of subject which should be generated |
| `clearance` | scalar, clearance |
| `volumeOfDistribution` | scalar, volume of distribution |
| `dose` | scalar, dose |
| `varCompClearance` | scalar, standard error of the normal distribution generating clearance |
| `varCompVolumeOfDistribution` | scalar, standard error of the normal distribution generating volume of distribution |
| `timePoints` | vector of time points |

## Details

The model used to generate data at time t is as follows

$$y(t) = C(t)\exp(e(t)),$$

where $C(t)$, the PK-model, is defined as follows:

$$C(t) = \frac{\text{dose}}{V_d}\exp\left(CL.t\right),$$

with $V_d$ the volume of distribution and $CL$ as clearance. The error model is consdiered as $e(t) \sim N(0, h(t))$, with:

$$h(t) = 0.03 + 0.165\frac{C(t)^{-1}}{C(1.5)^{-1} + C(t)^{-1}}.$$

For the mixed effects model, $CL = \widetilde{CL}\exp\left(\eta_1\right)$, and $V_d = \widetilde{V_d}\exp\left(\eta_2\right)$, where $\eta_1 \sim N(0, w_1^2)$ and $\eta_1 \sim N(0, w_2^2)$. Note that $w_1$ and $w_2$ are specified by *varCompClearance*, and *varCompVolumeOfDistribution* in the arguments, respectively.

## Value

generated sample with numSubjects as the number of rows and length of timePoints as the number of columns

## Author(s)

Vahid Nassiri, Helen Yvette Barnett

## See Also

Beal S. L., Ways to fit a PK model with some data below the quantification limit, Journal of Pharmacokinetics and Pharmacodynamics, 2001;28(**5**):481–504.

## Examples

```
set.seed(111)
simulateBealModelMixedEffects(10, 0.693,
+  1, 1, 0.2,0.2, seq(0.5,3,0.5))
```

# Index