

# Package ‘MLEcens’

April 15, 2010

**Version** 0.1-3

**Date** 2010-02-15

**Title** Computation of the MLE for bivariate (interval) censored data

**Author** Marloes Maathuis <maathuis@stat.math.ethz.ch>

**Maintainer** Marloes Maathuis <maathuis@stat.math.ethz.ch>

## Depends

**Description** This package contains functions to compute the nonparametric maximum likelihood estimator (MLE) for the bivariate distribution of  $(X, Y)$ , when realizations of  $(X, Y)$  cannot be observed directly. To be more precise, we consider the situation where we observe a set of rectangles (that we call ‘observation rectangles’) that are known to contain the unobservable realizations of  $(X, Y)$ . We compute the MLE based on such a set of rectangles. The methods can also be used for univariate censored data (see data set ‘cosmesis’), and for censored data with competing risks (see data set ‘menopause’). We also provide functions to visualize the observed data and the MLE. (This package contains the functionality of the R-package ‘biceduc’, which will no longer be maintained.)

**License** GPL (>= 2)

**URL** <http://stat.ethz.ch/~maathuis/>

**Repository** CRAN

**Date/Publication** 2010-04-15 13:49:21

## R topics documented:

actg181 . . . . .	2
actg181Mod . . . . .	4
canon2real . . . . .	5
computeMLE . . . . .	7
cosmesis . . . . .	10
ex . . . . .	12
menopause . . . . .	12

menopauseMod . . . . .	14
plotCDF1 . . . . .	15
plotCDF2 . . . . .	17
plotCM . . . . .	19
plotDens1 . . . . .	20
plotDens2 . . . . .	22
plotHM . . . . .	24
plotRects . . . . .	26
real2canon . . . . .	28
reduc . . . . .	29

## Index 32

actg181

*Data from the Aids Clinical Trials Group protocol ACTG 181*

### Description

An example data set with bivariate interval censored data. The data come from the AIDS Clinical Trials Group protocol ACTG 181, and contain information on the time (in months) to shedding of cytomegalovirus (CMV) in the urine and blood and the time (in months) to colonization of mycobacterium avium complex (MAC) in the sputum and stool (Betensky and Finkelstein, 1999).

### Usage

```
data(actg181)
```

### Format

A matrix containing 204 rows and 4 columns. Each row  $(x_1, x_2, y_1, y_2)$  corresponds to a subject in the study, and represents the rectangle that is known to contain the unobservable times of CMV shedding ( $x$ ) and MAC colonization ( $y$ ) of this person:  $x_1 \leq x \leq x_2$  and  $y_1 \leq y \leq y_2$ . The times are given in months. We use the values  $\pm 100$  to represent  $\pm$  infinity.

### Details

Extracted from Betensky and Finkelstein (1999): The data describe 204 of the 232 subjects in the study who were tested for CMV shedding and MAC colonization at least once during the trial, and did not have a prior CMV or MAC diagnosis. Tests were performed during clinic visits, scheduled at regular monthly intervals. For patients who did not miss any clinic visits, the time of event was recorded as the month that the first positive test occurred, resulting in discrete failure time data. For patients who missed some visits, and who were detected to be positive directly following one or more missed visits, the event time was recorded as having occurred in a time interval, resulting in discrete interval censored failure time data. All visit times were rounded to the closest quarter.

One should use closed boundaries ( $B=c(1,1,1,1)$ ) in order to reproduce the results of Betensky and Finkelstein (1999). In that case the probability masses of the MLE that we find are exactly equal to those given in Table IV of Betensky and Finkelstein, but there are some discrepancies in the maximal intersections (compare rows 1, 2, 7, 8 and 10 of their table IV).

## Source

Betensky and Finkelstein (1999). A non-parametric maximum likelihood estimator for bivariate interval censored data. *Statistics in Medicine* **18** 3089-3100.

## See Also

[actg181Mod](#)

## Examples

```
# Load the data
data(actg181)

# Compute the MLE
mle <- computeMLE(R=actg181, B=c(1,1,1,1))

# Create CDF plots of the MLE:
# (Maximal intersections are denoted in red)
par(mfrow=c(2,2))

# Lower bound for bivariate CDF
plotCDF2(mle, bound="l", xlim=c(-1,101), ylim=c(-1,101),
  n.key=5, main="Bivariate CDF (lower bound)",
  xlab="time to CMV shedding (months)",
  ylab="time to MAC colonization (months)")
plotRects(mle$rects, border="red", add=TRUE)

# Upper bound for bivariate CDF
plotCDF2(mle, bound="u", xlim=c(-1,101), ylim=c(-1,101),
  n.key=5, main="Bivariate CDF (upper bound)",
  xlab="time to CMV shedding (months)",
  ylab="time to MAC colonization (months)")
plotRects(mle$rects, border="red", add=TRUE)

# Marginal CDF for X
plotCDF1(mle, margin=1, xlim=c(0,90),
  main="CDF for time to CMV shedding",
  xlab="t (months)", ylab="P(time to CMV shedding <= t)")

# Marginal CDF for Y
plotCDF1(mle, margin=2, xlim=c(0,90),
  main="CDF for time to MAC colonization",
  xlab="t (months)", ylab="P(time to MAC colonization <= t)")

# Note that the difference between the upper and lower bound
# of the MLE (because of representational non-uniqueness)
# is large, especially for the time to MAC colonization.
```

---

`actg181Mod`*Modified data from the Aids Clinical Trials Group protocol ACTG 181*

---

### Description

An example data set with bivariate interval censored data. The data come from the AIDS Clinical Trials Group protocol ACTG 181, and contain information on the time (in months) to shedding of cytomegalovirus (CMV) in the urine and blood and the time (in months) to colonization of mycobacterium avium complex (MAC) in the sputum and stool (Betensky and Finkelstein, 1999).

The format of the data has been modified to allow for easy plotting with the functions `plotHM`, `plotDens1` and `plotDens2` (see section 'Format').

### Usage

```
data(actg181Mod)
```

### Format

A matrix containing 204 rows and 4 columns. Each row  $(x_1, x_2, y_1, y_2)$  corresponds to a subject in the study, and represents the rectangle that is known to contain the unobservable times of CMV shedding ( $x$ ) and MAC colonization ( $y$ ) of this person:  $x_1 \leq x \leq x_2$  and  $y_1 \leq y \leq y_2$ . The times are given in months. We use the values  $\pm 100$  to represent  $\pm$  infinity.

In order to allow easy plotting with `plotHM`, `plotDens1` and `plotDens2`, the  $x$ - and  $y$ - intervals were modified as follows:  $[x_1, x_2]$  was changed into  $[x_1 - 0.5, x_2 + 0.5]$  and  $[y_1, y_2]$  was changed into  $[y_1 - 0.5, y_2 + 0.5]$ .

### Details

Extracted from Betensky and Finkelstein (1999): The data describe 204 of the 232 subjects in the study who were tested for CMV shedding and MAC colonization at least once during the trial, and did not have a prior CMV or MAC diagnosis. Tests were performed during clinic visits, scheduled at regular monthly intervals. For patients who did not miss any clinic visits, the time of event was recorded as the month that the first positive test occurred, resulting in discrete failure time data. For patients who missed some visits, and who were detected to be positive directly following one or more missed visits, the event time was recorded as having occurred in a time interval, resulting in discrete interval censored failure time data. All visit times were rounded to the closest quarter.

One should use closed boundaries ( $B=c(1,1,1,1)$ ) in order to reproduce the results of Betensky and Finkelstein (1999). In that case the probability masses of the MLE that we find are exactly equal to those given in Table IV of Betensky and Finkelstein, but there are some discrepancies in the maximal intersections (compare rows 1, 2, 7, 8 and 10 of their table IV).

### Source

Betensky and Finkelstein (1999). A non-parametric maximum likelihood estimator for bivariate interval censored data. *Statistics in Medicine* **18** 3089-3100.

**See Also**

[actg181](#)

**Examples**

```
# Load the data
data(actg181Mod)

# Compute the MLE
mle <- computeMLE(R=actg181Mod, B=c(1,1,1,1))

# Create density plots
par(mfrow=c(2,2))

# Bivariate density plot
plotDens2(mle, main="Bivariate density",
  xlab="time to CMV shedding (months)",
  ylab="time to MAC colonization (months)")

# Marginal density plot for time to MAC colonization
plotDens1(mle, margin=2, main="Density for time
to MAC colonization", xlab="t (months)",
  ylab="density")

# Marginal density plot for time to CMV shedding
plotDens1(mle, margin=1, main="Density for time
to CMV shedding", xlab="t (months)",
  ylab="density")

# Note that many maximal intersections extend to
# infinity, and hence the value of the density is
# not very meaningful.
```

---

canon2real

*Transform (intersections of) canonical rectangles back to their original coordinates*

---

**Description**

This function transforms a set of (intersections of) canonical rectangles (see [real2canon](#) for a definition) back to their original coordinates. It performs the inverse operation of the function [real2canon](#).

**Usage**

```
canon2real(Rcanon, R, B = c(0,1))
```

### Arguments

Rcanon	A $m \times 4$ matrix of (intersections of) canonical rectangles that are to be transformed back to their original coordinates. Each row corresponds to a rectangle, represented as $(x1, x2, y1, y2)$ . The point $(x1, y1)$ is the lower left corner of the rectangle and $(x2, y2)$ is the upper right corner of the rectangle.
R	A $n \times 4$ matrix with the coordinates of the original rectangles. Each row corresponds to a rectangle, represented as $(x1, x2, y1, y2)$ .
B	This describes the boundaries of the original rectangles (0=open or 1=closed). It can be specified in three ways: (1) A $n \times 4$ matrix containing 0's and 1's. Each row corresponds to a rectangle and is denoted as $(cx1, cx2, cy1, cy2)$ , where $cx1$ denotes the boundary type of $x1$ , $cx2$ denotes the boundary type of $x2$ , etc. (2) A vector $(cx1, cx2, cy1, cy2)$ containing 0's and 1's. This representation can be used if all rectangles have the same type of boundaries. (3) A vector $(c1, c2)$ containing 0's and 1's. This representation can be used if all $x$ and $y$ intervals have the same type of boundaries. $c1$ denotes the boundary type of $x1$ and $y1$ , and $c2$ denotes the boundary type of $x2$ and $y2$ .

### Details

The functions `real2canon` and `canon2real` are carried out automatically in C-code as part of the functions `reduc` and `computeMLE`. We chose to make the functions available separately as well, in order to illustrate our algorithm for computing the MLE.

As a first step in the computation of the MLE, we transform rectangles into canonical rectangles, using `real2canon`). This is useful for two reasons. Firstly, it forces us in the very beginning to deal with possible ties and with the fact whether endpoints are open or closed. As a consequence, we do not have to account for ties and open or closed endpoints in the actual computation of the MLE. Secondly, it is convenient to work with the integer coordinates of the canonical rectangles in the computation of the MLE. After all computations are done, we transform the canonical rectangles back to their original coordinates, using `canon2real`. For more details, see Maathuis (2005, Section 2.1).

### Value

A list with the following elements:

<code>rects</code>	A $m \times 4$ matrix giving the original coordinates of the input rectangles. Each row $(x1, x2, y1, y2)$ represents a rectangle.
<code>bounds</code>	This describes the boundaries <code>rects</code> . It is given in the same format as <code>B</code> .

### Author(s)

Marloes Maathuis: <maathuis@stat.math.ethz.ch>

### References

M.H. Maathuis (2005). Reduction algorithm for the NPML for the distribution function of bivariate interval censored data. *Journal of Computational and Graphical Statistics* **14** 252–262.

**See Also**[real2canon](#)**Examples**

```

# An example with 3 arbitrarily chosen observation rectangles
R <- rbind(c(3.5, 4.2, 3.3, 9.1), # first rectangle
          c(4.2, 4.9, 3, 4.5), # second rectangle
          c(3.8, 5.1, 8.1, 9.5)) # third rectangle

# Plot the rectangles
par(mfrow=c(2,2))
plotRects(R, lwd=2, main="Original rectangles")

# Transform rectangles into canonical rectangles
res1 <- real2canon(R, c(0,1))
plotRects(res1, grid=TRUE, lwd=2, main="Canonical rectangles")

# Transform canonical rectangles back to original coordinates
res2 <- canon2real(res1, R, c(0,1))
plotRects(res2$rects, lwd=2, main="Original rectangles")

# Only transform rectangle (2,3)x(4,5), which is the
# the intersection of the canonical rectangles R1 and R3.
# The result is the intersection of the original rectangles R1 and R3.
R.1.3 <- matrix(c(2,3,4,5),nrow=1)
res3 <- canon2real(R.1.3, R, c(0,1))
res3$rects

# Note that the algorithm keeps track of the boundaries of the rectangles:
B <- rbind(c(1,0,1,0),
          c(1,1,1,1),
          c(0,1,0,1))
res4 <- canon2real(R.1.3, R, B)
res4$bounds

```

computeMLE

*Compute the MLE for bivariate censored data***Description**

This function computes the MLE for bivariate censored data. To be more precise, we compute the MLE for the bivariate distribution of  $(X,Y)$  in the following situation: realizations of  $(X,Y)$  cannot be observed directly; instead, we observe a set of rectangles (that we call 'observation rectangles') that are known to contain the unobservable realizations of  $(X,Y)$ .

**Usage**

```
computeMLE(R, B=c(0,1), max.inner=10, max.outer=1000, tol=1e-10)
```

## Arguments

<code>R</code>	A nx4 matrix of observation rectangles. Each row corresponds to a rectangle, represented as (x1,x2,y1,y2). The point (x1,y1) is the lower left corner of the rectangle and the point (x2,y2) is the upper right corner of the rectangle.
<code>B</code>	This describes the boundaries of the rectangles (0=open or 1=closed). It can be specified in three ways: (1) A nx4 matrix containing 0's and 1's. Each row corresponds to a rectangle and is denoted as (cx1, cx2, cy1, cy2), where cx1 denotes the boundary type of x1, cx2 denotes the boundary type of x2, etc. (2) A vector (cx1, cx2, cy1, cy2) containing 0's and 1's. This representation can be used if all rectangles have the same type of boundaries. (3) A vector (c1, c2) containing 0's and 1's. This representation can be used if all x and y intervals have the same type of boundaries. c1 denotes the boundary type of x1 and y1, and c2 denotes the boundary type of x2 and y2. The default value is c(0,1).
<code>max.inner</code>	Maximum number of iterations for the inner iteration loop (see section 'Details').
<code>max.outer</code>	Maximum number of iterations for the outer loop (see section 'Details').
<code>tol</code>	Tolerance up to which the necessary and sufficient conditions of the MLE must be satisfied (see section 'Details').

## Details

Let  $PF(R_i)$  be the probability mass in observation rectangle  $R_i$  under distribution  $F$ . Then the MLE maximizes  $\log PF(R_1) + \dots + \log PF(R_n)$  over the space of all bivariate distribution functions  $F$ . The MLE can only assign mass to a finite number of disjoint sets, called maximal intersections, and the MLE is indifferent to the distribution of mass within these sets. Hence, the computation of the MLE can be split into two steps: a reduction step and an optimization step. In the reduction step, the maximal intersections are computed. Next, in the optimization step, it is determined how much probability mass should be assigned to each of these areas.

The function `computeMLE` uses the height map algorithm of Maathuis (2005) for the reduction step (see `reduc`) For the optimization step, it uses a combination of sequential quadratic programming (outer iteration loop) and the support reduction algorithm of Groeneboom, Jongbloed and Wellner (2007) (inner iteration loop). It terminates when the necessary and sufficient conditions for the MLE (see Maathuis (2003, page 49, eq (5.4))) are satisfied up to a tolerance `tol`, or when the maximum number of iterations is reached, whichever comes first. We have found that it works well in practice to set `max.inner` low; hence it's default value is 10.

The MLE is typically non-unique, in two ways: (1) The MLE is indifferent to the distribution of mass within the maximal intersections (called representational non-uniqueness by Gentleman and Vandal (2002)); (2) The amounts of mass assigned to the maximal intersections may be non-unique (called mixture non-uniqueness by Gentleman and Vandal (2002)).

Hence, the algorithm `computeMLE` returns a MLE. One can deal with representational non-uniqueness by creating an upper bound (assign all mass to the lower left corners of the maximal intersections) and a lower bound (assign all mass to the upper right corners of the maximal intersections) of the MLE (see also `plotCDF1`, `plotCDF2`, `plotDens1`, `plotDens2`). The algorithm does not (yet) allow to account for mixture non-uniqueness.

**Value**

A list containing the following elements:

<code>p</code>	Vector of length <code>m</code> with positive probability masses.
<code>rects</code>	A <code>m</code> x4 matrix containing the maximal intersections that correspond to the positive probability masses <code>p</code> .
<code>bounds</code>	Boundaries of <code>rects</code> , specified in the same way as <code>B</code> .
<code>conv</code>	Boolean indicating if the algorithm has converged.
<code>llh</code>	Value of the log likelihood $\log(P(R_1))+\dots+\log(P(R_n))$ .

**Author(s)**

Marloes Maathuis: <maathuis@stat.math.ethz.ch>. Part of the code for the optimization step is adapted from code that was written by Piet Groeneboom.

**References**

Groeneboom, Jongbloed and Wellner (2007). The support reduction algorithm for computing non-parametric function estimates in mixture models. Submitted.

Gentleman and Vandal (2002). Nonparametric estimation of the bivariate CDF for arbitrarily censored data. *Canadian Journal of Statistics* **30** 557-571.

M.H. Maathuis (2003). Nonparametric maximum likelihood estimation for bivariate censored data. Master's thesis, Delft University of Technology, The Netherlands. Available at <http://stat.ethz.ch/~maathuis/papers/>.

M.H. Maathuis (2005). Reduction algorithm for the NPML for the distribution function of bivariate interval censored data. *Journal of Computational and Graphical Statistics* **14** 252–262.

**See Also**

[reduc](#), [plotCDF1](#), [plotCDF2](#), [plotDens1](#), [plotDens2](#)

**Examples**

```
# Load example data:
data(ex)
mle <- computeMLE(ex)
par(mfrow=c(2,2))

#### Bivariate density plots of the MLE:

# The colors represent the density=p/(area of maximal intersection)
plotDens2(mle, xlim=range(ex[,1:2]), ylim=range(ex[,3:4]),
  main="Bivariate density plot of the MLE")
plotRects(ex, add=TRUE)

# Alternative: numbers represent the mass p in the maximal intersections
plotDens2(mle, xlim=range(ex[,1:2]), ylim=range(ex[,3:4]),
  col="lightgray", main="Bivariate density plot of the MLE",
  key=FALSE, numbers=TRUE)
```

```

plotRects(ex, add=TRUE)

#### Univariate density plots of the MLE:

# Plot univariate density for X
plotDens1(mle, margin=1, xlim=range(ex[,1:2]),
  main="Marginal density plot,
  x-margin", xlab="x", ylab=expression(f[X](x)))

# Plot univariate density for Y
plotDens1(mle, margin=2, xlim=range(ex[,3:4]),
  main="Marginal density plot,
  y-margin", xlab="y", ylab=expression(f[Y](y)))

### Bivariate CDF plots of the MLE:

# Plot lower bound for representational non-uniqueness
plotCDF2(mle, xlim=c(min(ex[,1])-1,max(ex[,2])+1),
  ylim=c(min(ex[,3])-1, max(ex[,4])+1), bound="l", n.key=4,
  main="Bivariate CDF plot of the MLE,
  lower bound")

# Add observation rectangles and shaded maximal intersections
plotRects(ex, add=TRUE)
plotRects(mle$rects, density=20, border=NA, add=TRUE)

# Plot upper bound for representational non-uniqueness
plotCDF2(mle, xlim=c(min(ex[,1])-1,max(ex[,2])+1),
  ylim=c(min(ex[,3])-1, max(ex[,4])+1), bound="u", n.key=4,
  main="Bivariate CDF plot of the MLE,
  upper bound")

# Add observation rectangles and shaded maximal intersections
plotRects(ex, add=TRUE)
plotRects(mle$rects, density=20, border=NA, add=TRUE)

### Marginal CDF plots of the MLE:

# Plot marginal CDF for X
plotCDF1(mle, margin=1, xlim=c(min(ex[,1])-1,max(ex[,2])+1),
  bound="b", xlab="x", ylab="P(X<=x)", main="MLE for P(X<=x)")

# Plot marginal CDF for Y
plotCDF1(mle, margin=2, xlim=c(min(ex[,3])-1,max(ex[,4])+1),
  bound="b", xlab="y", ylab="P(Y<=y)", main="MLE for P(Y<=y)")

```

**Description**

An example data set with univariate interval censored data (and one covariate). The data come from a retrospective study that compared the cosmetic effect of two types of treatments for early breast cancer patients: radiotherapy alone, and radiotherapy with adjuvant chemotherapy. The time of interest was the time to breast retraction (in months).

**Usage**

```
data(cosmesis)
```

**Format**

A matrix containing 94 rows and 3 columns. Each row ( $x_1, x_2, tr$ ) corresponds to a subject in the study. The interval  $(x_1, x_2]$  contains the unobservable time of breast retraction (in months). The variable  $tr$  indicates the treatment:  $tr=0$  for patients who were treated with radiotherapy alone (RT alone), and  $tr=1$  for patients who were treated with radiotherapy and chemotherapy (RT+CT). We use the value 100 to represent infinity.

**Source**

Finkelstein and Wolfe (1985). A semiparametric model for regression analysis of interval-censored failure time data. *Biometrics* **41** 933-945.

**Examples**

```
data(cosmesis)

# Split data according to treatment group
cosmesis0 <- cosmesis[cosmesis[,3]==0, 1:2]
cosmesis1 <- cosmesis[cosmesis[,3]==1, 1:2]
n0 <- nrow(cosmesis0)
n1 <- nrow(cosmesis1)

# Add dummy y-intervals (0,1)
cosmesis0 <- cbind(cosmesis0, rep(0,times=n0), rep(1,times=n0))
cosmesis1 <- cbind(cosmesis1, rep(0,times=n1), rep(1,times=n1))

# Compute MLEs in both treatment groups
mle0 <- computeMLE(cosmesis0)
mle1 <- computeMLE(cosmesis1)

# Plot MLEs
par(mfrow=c(2,2))

# Density for women who were treated with radio therapy alone
plotDens1(mle0, margin=1, col="black", main="Density for time to breast
retraction (RT alone)", xlab="time (months)", ylab="density")

# Density for women who were treated with radio therapy + chemo therapy
plotDens1(mle1, margin=1, col="red", main="Density for time to breast
retraction (RT+CT)", xlab="time (months)", ylab="density")
```

```
# Survival functions for both groups, plus legend
plotCDF1(mle0, margin=1, surv=TRUE, col="black",
  main="Survival functions", xlab="time (months)", ylab="probability")
plotCDF1(mle1, margin=1, surv=TRUE, col="red", add=TRUE)
legend(3, .3, c("RT alone", "RT+CT"), lty=1, col=c("black", "red"))
```

---

ex

*Example data set (artificial)*

---

### Description

Example data set with artificial bivariate interval censored data. These data are used to illustrate various functions of this R-package.

### Usage

```
data(ex)
```

### Format

A matrix containing 6 rows and 4 columns. Each row (x1,x2,y1,y2) represents a rectangle that is known to contain the unobservable realization of the variables of interest (X,Y). The point (x1,y1) is the lower left corner of the rectangle and (x2,y2) is the upper right corner of the rectangle.

### Examples

```
# Load the data
data(ex)

# Plot the rectangles
par(mfrow=c(1,1))
plotRects(ex)
```

---

menopause

*Menopause data*

---

### Description

An example data set with interval censored data and competing risks. The data come from Cycle I of the Health Examination Survey of the National Center for Health Statistics, and contain information on the menopausal status of 2423 women (MacMahon and Worcester, 1966).

### Usage

```
data(menopause)
```

**Format**

A matrix containing 2423 rows and 4 columns. Each row  $(x_1, x_2, y_1, y_2)$  corresponds to a subject in the study. The interval  $(x_1, x_2]$  contains the unobservable age of menopause  $X$ . The interval  $[y_1, y_2]$  contains the type of menopause  $Y$ , where  $Y=1$  represents operative menopause and  $Y=2$  represents natural menopause. We use the value 100 to represent infinity.

**Details**

The Health Examination Survey used a nationwide probability sample of people between age 18 and 79 from the United States civilian, noninstitutional population. The participants were asked to complete a self-administered questionnaire. The sample contained 4211 females, of whom 3581 completed the questionnaire. We restrict attention to the age range 25-59 years. Furthermore, seven women who were less than 35 years of age and reported having had a natural menopause were excluded as being an error or abnormal. The remaining data set contains information on 2423 women.

MacMahon and Worcester (1966) found that there was marked terminal digit clustering in the response of this question, especially for women who had a natural menopause. Therefore, Krailo and Pike (1983) decided to only consider the menopausal status of women at the time of the questionnaire, yielding current status data on the age of menopause with two competing risks: operative menopause and natural menopause.

**Source**

MacMahon and Worcester (1966). Age at menopause, United States 1960 - 1962. *National Center for Health Statistics. Vital and Health Statistics*, volume 11, number 19.

**References**

Krailo and Pike (1983). Estimation of the distribution of age at natural menopause from prevalence data. *American Journal of Epidemiology* **117** 356-361.

**See Also**

[menopauseMod](#)

**Examples**

```
# Load the data
data(menopause)

# Compute the MLE
mle <- computeMLE(R=menopause, B=c(0,1,1,1))

# Plot first sub-distribution function  $P(X \leq x, 0.5 < Y \leq 1.5) = P(X \leq x, Y=1)$ 
par(mfrow=c(1,1))
plotCDF1(mle, margin=1, bound="b", int=c(0.5,1.5), col="red", ylim=c(0,1),
  xlab="x", main="P(X<=x, Y=k), k=1,2")

# Plot second sub-distribution function  $P(X \leq x, 1.5 < Y \leq 2.5) = P(X \leq x, Y=2)$ 
plotCDF1(mle, margin=1, bound="b", int=c(1.5,2.5), col="black", add=TRUE)
```

```
# Add legend
legend(0,1,c("k=1: operative","k=2: natural"), col=c("red","black"), lty=1)

# Plot marginal distribution of the failure cause Y
plotCDF1(mle, margin=2, bound="u", col="black", xlim=c(0,3),
  xlab="y", main="P(Y<=y)")
```

---

menopauseMod

*Modified menopause data*


---

## Description

Example data set with interval censored data and competing risks. The data come from Cycle I of the Health Examination Survey of the National Center for Health Statistics, and contain information on the menopausal status of 2423 women (MacMahon and Worcester, 1966).

The format of the data has been modified to allow for easy plotting with the functions [plotHM](#) and [plotDens2](#) (see section 'Format').

## Usage

```
data(menopauseMod)
```

## Format

A matrix containing 2423 rows and 4 columns. Each row ( $x_1, x_2, y_1, y_2$ ) corresponds to a subject in the study. The interval  $(x_1, x_2]$  contains the unobservable age of menopause  $X$ . The interval  $[y_1, y_2]$  contains the type of menopause  $Y$ , where  $Y=1$  represents operative menopause and  $Y=2$  represents natural menopause. We use the value 100 to represent infinity.

In order to allow easy plotting with [plotHM](#) and [plotDens2](#), the  $y$ -intervals were modified as follows:  $[y_1, y_2]$  was changed into  $[y_1 - 0.25, y_2 + 0.25]$ .

## Details

The Health Examination Survey used a nationwide probability sample of people between age 18 and 79 from the United States civilian, noninstitutional population. The participants were asked to complete a self-administered questionnaire. The sample contained 4211 females, of whom 3581 completed the questionnaire. We restrict attention to the age range 25-59 years. Furthermore, seven women who were less than 35 years of age and reported having had a natural menopause were excluded as being an error or abnormal. The remaining data set contains information on 2423 women.

MacMahon and Worcester (1966) found that there was marked terminal digit clustering in the response of this question, especially for women who had a natural menopause. Therefore, Krailo and Pike (1983) decided to only consider the menopausal status of women at the time of the questionnaire, yielding current status data on the time of menopause with two competing risks: operative menopause and natural menopause.

**Source**

MacMahon and Worcester (1966). Age at menopause, United States 1960 - 1962. *National Center for Health Statistics. Vital and Health Statistics*, volume 11, number 19.

**References**

Krailo and Pike (1983). Estimation of the distribution of age at natural menopause from prevalence data. *American Journal of Epidemiology* **117** 356-361.

**See Also**

[menopause](#)

**Examples**

```
# Load the data
data(menopauseMod)

# Compute the MLE
mle <- computeMLE(menopauseMod)

# Create density plot
par(mfrow=c(1,1))
plotDens2(mle, xlim=c(0,100), border="black", xlab="age in years",
  ylab="cause of menopause (1=operative, 2=natural)",
  main="Density plot of the MLE for the menopause data")
```

---

plotCDF1

*Create a marginal CDF (or survival function) plot of the MLE*

---

**Description**

This function plots the MLE for a marginal (sub)-CDF (or survival function) for one of the two variables of interest. To be precise, it can plot the MLE for  $P(X \leq x, a < Y \leq b)$  (or  $1 - P(X \leq x, a < Y \leq b)$ ) as a function of  $x$ , and the MLE for  $P(a < X \leq b, Y \leq y)$  (or  $1 - P(a < X \leq b, Y \leq y)$ ) as a function of  $y$ , where  $a$  and  $b$  may take the values  $-\infty$  and  $\infty$ . The values of these estimates are computed by summing all probability mass of the MLE that falls in the regions  $(-\infty, x] \times (a, b]$  and  $(a, b] \times (-\infty, y]$ , respectively.

**Usage**

```
plotCDF1(mle, margin, bound="b", int=NULL, surv=FALSE,
  add=FALSE, col=1, lty=1, xlim=NULL,
  ylim=NULL, xlab="", ylab="", main="", sub="")
```

**Arguments**

<code>mle</code>	List with elements 'p' and 'rects', as outputted by <a href="#">computeMLE</a> .
<code>margin</code>	Indicates which margin should be plotted: 1 = x-margin, 2 = y-margin. So if <code>margin=1</code> , the MLE for $P(X \leq x, a < Y \leq b)$ is plotted, and if <code>margin=2</code> , then the MLE for $P(a < X \leq b, Y \leq y)$ is plotted.
<code>bound</code>	Parameter taking the values "u", "l" and "b". It indicates how representational non-uniqueness of the MLE should be handled. Option "u" (upper) indicates an upper bound, obtained by assigning all mass to the lower left corners of the maximal intersections. Option "l" (lower) indicates a lower bound, obtained by assigning all mass to the upper right corners of the maximal intersections. Option "b" (both) indicates that both the upper and the lower bound should be plotted. The default value is "b".
<code>int</code>	This indicates the range of interest of the variable that was <i>not</i> chosen in <code>margin</code> . If <code>int</code> is specified, it should be of the form $c(a,b)$ , with $a < b$ . If <code>margin=1</code> , the MLE for $P(X \leq x, a < Y \leq b)$ is plotted as a function of $x$ . If <code>margin=2</code> , the MLE for $P(a < X \leq b, Y \leq y)$ is plotted as a function of $y$ . This parameter defaults to $(-\infty, \infty)$ , yielding plots of the estimates for $P(X \leq x)$ and $P(Y \leq y)$ .
<code>surv</code>	Logical. The default value is FALSE. If TRUE, the function $1 - P(X \leq x, a < Y \leq b)$ is plotted instead of $P(X \leq x, a < Y \leq b)$ , and the function $1 - P(a < X \leq b, Y \leq y)$ is plotted instead of $P(a < X \leq b, Y \leq y)$ .
<code>add</code>	Logical, indicating if the lines should be added to an existing plot. The default value is FALSE.
<code>col</code>	Line color. The default value is <code>l="black"</code> .
<code>lty</code>	Line type. The default value is <code>l="solid"</code> .
<code>xlim</code>	Range for the horizontal axis, defaulting to the range of x-coordinates (if <code>margin=1</code> ) or y-coordinates (if <code>margin=2</code> ) of the relevant corners of maximal intersections.
<code>ylim</code>	Range for the vertical axis, defaulting to the range of values of the estimate.
<code>xlab, ylab</code>	Labels of the x- and y-axis. The default values are empty.
<code>main</code>	Title of the plot.
<code>sub</code>	Sub title of the plot.

**Value**

No value is returned.

**Author(s)**

Marloes Maathuis: <maathuis@stat.math.ethz.ch>

**See Also**

[computeMLE](#)

**Examples**

```

# Load example data:
data(ex)

# Compute the MLE:
mle <- computeMLE(ex)

# Plot marginal CDF for X
par(mfrow=c(2,2))
plotCDF1(mle, margin=1, xlim=c(min(ex[,1])-1,max(ex[,2])+1),
  bound="b", xlab="x", ylab="P(X<=x)", main="MLE for P(X<=x)")

# Plot marginal survival function for X
plotCDF1(mle, margin=1, surv=TRUE, xlim=c(min(ex[,1])-1,max(ex[,2])+1),
  bound="b", xlab="x", ylab="P(X>x)", main="MLE for P(X>x)")

# Plot marginal CDF for Y
plotCDF1(mle, margin=2, xlim=c(min(ex[,3])-1,max(ex[,4])+1),
  bound="b", xlab="y", ylab="P(Y<=y)", main="MLE for P(Y<=y)")

# Plot marginal survival function for Y
plotCDF1(mle, margin=2, surv=TRUE, xlim=c(min(ex[,3])-1,max(ex[,4])+1),
  bound="b", xlab="y", ylab="P(Y>y)", main="MLE for P(Y>y)")

```

plotCDF2

*Create a bivariate CDF (or survival function) plot of the MLE***Description**

This function plots the MLE for the bivariate CDF of (X,Y) (or the bivariate survival function). The value of the estimate at the point (x,y) is computed by summing all probability mass of the MLE that falls in the region  $(-\infty, x] \times (-\infty, y]$ . The plot uses colors/shades to represent the value of the MLE, and is generated using the function [image](#).

**Usage**

```

plotCDF2(mle, bound, col=gray(seq(0.9,0.3,len=30)), surv=FALSE,
  key=TRUE, n.key=10, round.key=2, cex.key=0.6, xlim=NULL,
  ylim=NULL, zlim=NULL, breaks=NULL, xlab="", ylab="",
  main="", sub="")

```

**Arguments**

mle	List with elements 'p' and 'rects', as outputted by <a href="#">computeMLE</a> .
bound	Parameter taking the values "u" and "l". It indicates how representational non-uniqueness of the MLE should be handled. Option "u" (upper) indicates an upper bound, obtained by assigning all mass to the lower left corners of the maximal intersections. Option "l" (lower) indicates a lower bound, obtained by assigning all mass to the upper right corners of the maximal intersections.

<code>col</code>	Color vector used to represent the values of the MLE. The default value is <code>gray(seq(0.9, 0.3, len=30))</code> .
<code>surv</code>	Logical. If FALSE, the bivariate CDF $P(X \leq x, Y \leq y)$ is plotted. If TRUE, the bivariate survival function $1 - P(X \leq x, Y \leq y)$ is plotted. The default value is FALSE.
<code>key</code>	Logical, indicating if a color key should be drawn. The default value is TRUE.
<code>n.key</code>	Approximate number of tickmarks for the color key. The default value is 10.
<code>round.key</code>	Number of decimals used for the labels of the color key. The default value is 2.
<code>cex.key</code>	Numerical value giving the amount by which text in the key should be scaled relative to the default. The default value is 0.6.
<code>xlim, ylim</code>	Ranges for the plotted x and y values, defaulting to the ranges of the x- and y-coordinates of the relevant corners of the maximal intersections.
<code>zlim</code>	The minimum and maximum values of the estimate for which colors should be plotted, defaulting to the range of the finite values of the estimate. Each of the given colors will be used to color an equispaced interval of this range. The midpoints of the intervals cover the range, so that values just outside the range will be plotted (see the documentation of <a href="#">image</a> ). This parameter is not used if <code>breaks</code> is specified.
<code>breaks</code>	Numeric vector with break points for the colors, satisfying <code>length(breaks)=length(col)+1</code> . This parameter overrides <code>zlim</code> .
<code>xlab, ylab</code>	Labels for the x- and y-axis. The default values are empty.
<code>main</code>	Title of the plot. The default value is empty.
<code>sub</code>	Sub title of the plot. The default value is empty.

**Value**

No value is returned.

**Author(s)**

Marloes Maathuis: <maathuis@stat.math.ethz.ch>

**See Also**

[computeMLE](#)

**Examples**

```
# Load example data:
data(ex)

# Compute the MLE:
mle <- computeMLE(ex)

### Bivariate CDF plot of the MLE

# Plot lower bound for representational non-uniqueness
```

```

par(mfrow=c(1,1))
plotCDF2(mle, xlim=c(min(ex[,1])-1,max(ex[,2])+1),
  ylim=c(min(ex[,3])-1, max(ex[,4])+1), bound="l", n.key=4,
  main="Bivariate CDF plot of the MLE,
  lower bound")

# Add observation rectangles and shaded maximal intersections
plotRects(ex, add=TRUE)
plotRects(mle$rects, density=20, border=NA, add=TRUE)

# Plot upper bound for representational non-uniqueness
plotCDF2(mle, xlim=c(min(ex[,1])-1,max(ex[,2])+1),
  ylim=c(min(ex[,3])-1, max(ex[,4])+1), bound="u", n.key=4,
  main="Bivariate CDF plot of the MLE,
  upper bound")

# Add observation rectangles and shaded maximal intersections
plotRects(ex, add=TRUE)
plotRects(mle$rects, density=20, border=NA, add=TRUE)

```

---

plotCM

*Plot a clique matrix*


---

### Description

This function can be used to make an image of a clique matrix (or any other 0-1 matrix). It is basically just the function `image`, with some pre-defined settings.

### Usage

```

plotCM(cm, col=c("white","black"), at.x=NULL, at.y=NULL,
  xlab="Observation rectangles", ylab="Maximal intersections",
  main="", sub="")

```

### Arguments

<code>cm</code>	A $m \times n$ clique matrix, where $m$ is the number of maximal intersections, and $n$ is the number of observation rectangles. The $(i,j)$ th element is 1 if the $i$ th maximal intersection is contained in the $j$ th observation rectangle, and it is 0 otherwise.
<code>col</code>	Colors to be used. The default value is <code>c("white","black")</code> .
<code>at.x</code>	The points at which tick-marks are to be drawn along the x-axis. The default value is <code>NULL</code> , meaning that tickmark locations are computed automatically. See also the documentation of <code>axis</code> .
<code>at.y</code>	The points at which tick-marks are to be drawn along the y-axis. The default value is <code>NULL</code> , meaning that tickmark locations are computed automatically. See also the documentation of <code>axis</code> .
<code>xlab</code>	Label for the x-axis. The default value is "Observation rectangles".

ylab	Label for the y-axis. The default value is "Maximal intersections".
main	Title of the plot. The default value is empty.
sub	Sub-title of the plot. The default value is empty.

**Value**

No value is returned.

**Author(s)**

Marloes Maathuis: <maathuis@stat.math.ethz.ch>

**See Also**

[reduc](#)

**Examples**

```
# Load example data and plot observation rectangles
data(ex)
par(mfrow=c(2,1))
plotRects(ex,main="Rectangles and maximal intersections")

# Perform reduction step and plot maximal intersections (shaded)
res<-reduc(ex, cm=TRUE)
plotRects(res$rects, density=15, border=NA, add=TRUE)

# Plot clique matrix
plotCM(res$cm, main="Clique matrix")
```

---

plotDens1

*Create a univariate density plot of the MLE*

---

**Description**

This function creates a univariate density plot of the MLE. To be precise, it can plot the density  $(d/dx) P(X \leq x, a < Y \leq b)$  as a function of  $x$ , and  $(d/dy) P(a < X \leq b, Y \leq y)$  as a function of  $y$ , under the assumption that the mass is distributed uniformly over the maximal intersections.

**Usage**

```
plotDens1(mle, margin, int=NULL, col=1, lty=1, add=FALSE,
          xlim=NULL, ylim=NULL, xlab="", ylab="", main="", sub="")
```

**Arguments**

<code>mle</code>	List with elements 'p' and 'rects', as outputted by <code>computeMLE</code> .
<code>margin</code>	Indicates which margin should be plotted: 1 = x-margin, 2 = y-margin. So if <code>margin=1</code> , the MLE for $(d/dx) P(X \leq x, a < Y \leq b)$ is plotted, and if <code>margin=2</code> , then the MLE for $(d/dy) P(a < X \leq b, Y \leq y)$ is plotted.
<code>int</code>	This indicates the range of interest of the variable that was <i>not</i> chosen in <code>margin</code> . If <code>int</code> is specified, it should be of the form $c(a,b)$ , with $a < b$ . If <code>margin=1</code> , the MLE for $(d/dx) P(X \leq x, a < Y \leq b)$ is plotted as a function of $x$ . If <code>margin=2</code> , the MLE for $(d/dy) P(a < X \leq b, Y \leq y)$ is plotted as a function of $y$ . This parameter defaults to $(-\infty, \infty)$ , yielding plots of the marginal density of $X$ and $Y$ .
<code>col</code>	Line color. The default value is <code>1="black"</code> .
<code>lty</code>	Line type. The default value is <code>1="solid"</code> .
<code>add</code>	Logical, indicating if the lines should be added to an existing plot. The default value is <code>FALSE</code> .
<code>xlim</code>	Range for the horizontal axis, defaulting to the range of $x$ -coordinates (if <code>margin=1</code> ) or $y$ -coordinates (if <code>margin=2</code> ) of the relevant corners of maximal intersections.
<code>ylim</code>	Range for the vertical axis, defaulting to the range of values of the estimate.
<code>xlab, ylab</code>	Labels of the $x$ - and $y$ -axis. The default values are empty.
<code>main</code>	Title of the plot.
<code>sub</code>	Sub title of the plot.

**Details**

In many cases we assign specific values to represent  $\pm$  infinity and (see, e.g., [actg181](#)). Note that these values determine the size of maximal intersections that extend to  $\pm$  infinity, and hence they also determine the value of the density at such maximal intersections. The value of the density at such maximal intersections is therefore meaningless.

**Value**

No value is returned.

**Author(s)**

Marloes Maathuis: <maathuis@stat.math.ethz.ch>

**See Also**

[computeMLE](#)

## Examples

```
# Load example data:
data(ex)

# Compute the MLE:
mle <- computeMLE(ex)

# Bivariate density plot of the MLE:
# (Numbers represent the mass p in the maximal intersections)
par(mfrow=c(2,2))
plotDens2(mle, xlim=range(ex[,1:2]), ylim=range(ex[,3:4]),
  col="lightgray", main="Bivariate density plot of the MLE",
  key=FALSE, numbers=TRUE)
plotRects(ex, add=TRUE)

### Univariate density plots of the MLE:

# Plot of the marginal density of Y
plotDens1(mle, margin=2, xlim=range(ex[,3:4]),
  main="Marginal density plot,
  y-margin", xlab="y", ylab=expression(f[Y](y)))

# Plot of the marginal density of X
plotDens1(mle, margin=1, xlim=range(ex[,1:2]),
  main="Marginal density plot,
  x-margin", xlab="x", ylab=expression(f[X](x)))
```

---

plotDens2

*Create a bivariate density plot of the MLE*

---

## Description

This function creates a bivariate density plot of the MLE. It plots the maximal intersections that get positive mass under the MLE, filled with colors that represent the estimated density of  $F$  under the assumption that the mass is distributed uniformly over the maximal intersections. In other words, if the MLE assigns mass  $p$  to a maximal intersection, then the color represents  $p/(\text{area of maximal intersection})$ .

## Usage

```
plotDens2(mle, col=gray(seq(.9,.3,len=30)), border=NA,
  key=TRUE, n.key=10, round.key=2,
  numbers=FALSE, round.numbers=2, cex.numbers=0.6,
  xlim=NULL, ylim=NULL, zlim=NULL, breaks=NULL,
  xlab="", ylab="", main="", sub="")
```

**Arguments**

<code>mle</code>	List with elements 'p' and 'rects', as outputted by <code>computeMLE</code> .
<code>col</code>	Color vector used to represent the values of the density, and to fill the maximal intersections. The default value is <code>col=gray(seq(.9, .3, len=30))</code> .
<code>border</code>	Color for rectangle borders of the maximal intersections. The default value NA means that the borders are omitted.
<code>key</code>	Logical, indicating if a color key should be drawn. The default value is TRUE.
<code>n.key</code>	Approximate number of tickmarks for the color key. The default value is 10.
<code>round.key</code>	Number of decimals used for the labels of the color key. The default value is 2.
<code>numbers</code>	Logical, indicating if the total amounts of mass <code>mle\$p</code> should be printed in the maximal intersections. The default value is FALSE.
<code>round.numbers</code>	Number of decimals used for <code>numbers</code> . The default value is 2.
<code>cex.numbers</code>	Numerical value giving the amount by which text size of <code>numbers</code> should be scaled relative to the default. The default value is 0.6.
<code>xlim, ylim</code>	Ranges for the plotted x and y values, defaulting to the ranges of the x- and y-coordinates of the relevant corners of the maximal intersections.
<code>zlim</code>	The minimum and maximum values of the density for which colors should be plotted, defaulting to the range of the finite values of the density. Each of the given colors will be used to color an equispaced interval of this range. The midpoints of the intervals cover the range, so that values just outside the range will be plotted (see the documentation of <code>image</code> ). This parameter is not used if <code>breaks</code> is specified.
<code>breaks</code>	Numeric vector with break points for the colors, satisfying <code>length(breaks)=length(col)+1</code> . This parameter overrides <code>zlim</code> .
<code>xlab, ylab</code>	Labels for the x- and y-axis. The default values are empty.
<code>main</code>	Title of the plot. The default value is empty.
<code>sub</code>	Sub title of the plot. The default value is empty.

**Details**

In many cases we assign specific values to represent +/- infinity and (see, e.g., [actg181](#)). Note that these values determine the size of maximal intersections that extend to +/- infinity, and hence they also determine the value of the density at such maximal intersections. The value of the density at such maximal intersections is therefore meaningless.

**Value**

No value is returned.

**Author(s)**

Marloes Maathuis: <maathuis@stat.math.ethz.ch>

**See Also**[computeMLE](#)**Examples**

```
# Load example data:
data(ex)

# Compute the MLE:
mle <- computeMLE(ex)

### Bivariate density plots of the MLE:

# (The colors represent the density=p/(area of maximal intersection))
par(mfrow=c(1,1))
plotDens2(mle, xlim=range(ex[,1:2]), ylim=range(ex[,3:4]),
  main="Plot of the MLE. Colors represent the density.")
plotRects(ex, add=TRUE)

# Alternative: numbers show the amount of mass in each maximal intersection
plotDens2(mle, col="lightgray", xlim=range(ex[,1:2]),
  ylim=range(ex[,3:4]), numbers=TRUE, key=FALSE,
  main="Plot of the MLE")
plotRects(ex, add=TRUE)
```

plotHM

*Plot a height map***Description**

This function can be used to plot a 'height map' of a set of rectangles in a new plot, or to add it to an existing plot. The value of the heightmap at a point equals the number of rectangles that overlap at this point.

**Usage**

```
plotHM(hm, R, grid=TRUE, grid.lty=3, grid.col="lightgray",
  key=TRUE, n.key=10, cex.key=0.6, numbers=FALSE,
  col=terrain.colors(max(hm)+1), xlim=NULL, ylim=NULL,
  xlab="", ylab="", main="", sub="")
```

**Arguments**

hm	A $(2n+1) \times (2n+1)$ matrix with the values of the heightmap, outputted by the function <a href="#">reduc</a> .
R	A $n \times 4$ matrix with the real or canonical observation rectangles. Each row corresponds to a rectangle, represented as $(x1, x2, y1, y2)$ . The point $(x1, y1)$ is the lower left corner of the rectangle and the point $(x2, y2)$ is the upper right corner of the rectangle.

<code>grid</code>	Logical, indicating if a grid should be drawn. The default value is TRUE.
<code>grid.lty</code>	Line type of the grid lines. The default value is 3=dotted.
<code>grid.col</code>	Color of the grid lines. The default value is light gray.
<code>key</code>	Logical, indicating if a color key should be drawn in the right margin of the plot. The default value is TRUE.
<code>n.key</code>	Approximate number of tickmarks for the color key. The default value is 10.
<code>cex.key</code>	Numerical value giving the amount by which text in the key should be scaled relative to the default. The default value is 0.6.
<code>numbers</code>	Logical, indicating if numbers should be plotted in the grid cells that indicate the values of the height map.
<code>col</code>	Color vector used to represent the values of the height map. The length of <code>col</code> must equal <code>max(hm)+1</code> .
<code>xlim</code>	Range for the plotted x values, defaulting to <code>c(min(x-coordinates)-1, max(x-coordinates)+1)</code> .
<code>ylim</code>	Range for the plotted y values, defaulting to <code>c(min(x-coordinates)-1, max(x-coordinates)+1)</code> .
<code>xlab, ylab</code>	Labels of the x and y axis. The default values are empty.
<code>main</code>	Title of the plot. Default value is empty.
<code>sub</code>	Sub title of the plot. Default value is empty.

### Details

We chose to create a thin color key that fits in the margin of the plot. In this way, the plotting margins and plotting region do not have to be adjusted, so that other elements can be easily added to the plot later on (like observation rectangles or maximal intersections).

### Value

No value is returned.

### Author(s)

Marloes Maathuis: <maathuis@stat.math.ethz.ch>

### See Also

[reduc](#), [plotRects](#), [palette](#)

### Examples

```
# Load example data
data(ex)

# Perform reduction step
res <- reduc(ex, hm=TRUE)
```

```

# Plot the height map:
par(mfrow=c(1,1))
plotHM(res$hm, ex, main="Height map")

# Add observation rectangles in black:
plotRects(ex, add=TRUE, border="black")

# Add shaded maximal intersections:
plotRects(res$rects, add=TRUE, border=NA, density=15)

# Compute heightmap of the canonical rectangles:
canon <- real2canon(ex)
res2 <- reduc(canon, hm=TRUE)
# Note that res$hm and res2$hm are identical. So we only need to change
# the x- and y-coordinates of the height map.

# Plot height map of the canonical rectangles
plotHM(res$hm, canon, key=FALSE, numbers=TRUE, main="Canonical height map")

# Add canonical rectangles in black:
plotRects(canon, add=TRUE, border="black")

# Add canonical maximal intersections (local maxima of height map) in red:
plotRects(res2$rects, add=TRUE, border="red")

```

---

plotRects

*Plot a set of rectangles*


---

## Description

This function can be used to plot a set of rectangles in a new plot, or to add them to an existing plot. It is basically the function `rect`, with some pre-defined settings.

## Usage

```

plotRects(R, grid=FALSE, grid.lty=3, grid.col="lightgray",
          density=NULL, angle=45, col=NA, border=rainbow(nrow(R)),
          lty=1, lwd=1, add=FALSE, xlim=NULL, ylim=NULL, xlab="",
          ylab="", main="", sub="")

```

## Arguments

R	A nx4 matrix of rectangles. Each row corresponds to an rectangle, represented as (x1,x2,y1,y2). The point (x1,y1) is the lower left corner of the rectangle and the point (x2,y2) is the upper right corner of the rectangle.
grid	Logical, indicating if a grid should be drawn. The default value is FALSE.
grid.lty	Line type of the grid lines. The default value is 3=dotted.
grid.col	Line color of the grid lines. The default value is light gray.

density	Density of shading lines for the rectangles, in lines per inch. The default value is NULL, meaning that no shading lines are drawn. A zero value of density means no shading lines whereas negative values (and NA) suppress shading (and so allow color filling).
angle	Angle (in degrees) of the shading lines.
col	Color(s) to fill or shade the rectangles with. The default NA (or also NULL) means do not fill, i.e., draw transparent rectangles, unless density is specified.
border	Color for rectangle borders. Use border = NA to omit borders. If there are shading lines, border = TRUE means use the same color for the border as for the shading lines. The default value is rainbow(n): a vector of n contiguous colors from the palette <a href="#">rainbow</a> .
lty	Line type for borders and shading of the rectangles. The default value is l="solid".
lwd	Line width for borders and shading of the rectangles. The default value is 1.
add	Logical, indicating if the rectangles should be added to an existing plot. The default value is FALSE.
xlim	Range of the x-axis. The default value is the range of x-coordinates of the rectangles.
ylim	Range of the y-axis. The default value is the range of y-coordinates of the rectangles.
xlab, ylab	Labels of the x and y axis. The default values are empty.
main	Title of the plot. The default value is empty.
sub	Sub title of the plot. The default value is empty.

**Value**

No value is returned.

**Author(s)**

Marloes Maathuis: <maathuis@stat.math.ethz.ch>

**See Also**

[rect](#), [palette](#)

**Examples**

```
n <- 10
x <- c(0:(n-1))
R <- cbind(x, x+3, x, x+3) # first rectangle is (0,3)x(0,3),
                          # second rectangle is (1,4)x(1,4), etc...

par(mfrow=c(1,1))
plotRects(R, main="Example")
```

---

real2canon

*Transform a set of rectangles into canonical rectangles*


---

### Description

This function transforms a set of rectangles  $R_1, \dots, R_n$  into canonical rectangles  $R_1', \dots, R_n'$  with the following properties: (1)  $R_1', \dots, R_n'$  have the same intersection structure as the original rectangles, i.e.,  $R_i'$  and  $R_j'$  intersect if and only if  $R_i$  and  $R_j$  intersect. (2) All x-coordinates of  $R_1', \dots, R_n'$  are distinct, and take values in  $1, \dots, 2n$ . (3) All y-coordinates of  $R_1', \dots, R_n'$  are distinct, and take values in  $1, \dots, 2n$ .

The function `real2canon` performs the inverse operation of the function `canon2real`.

### Usage

```
real2canon(R, B=c(0, 1))
```

### Arguments

- |   |   |
|---|---|
| R | A $n \times 4$ real matrix of rectangles. Each row corresponds to a rectangle, represented as $(x_1, x_2, y_1, y_2)$ . The point $(x_1, y_1)$ is the lower left corner of the rectangle and the point $(x_2, y_2)$ is the upper right corner of the rectangle.  |
| B | This describes the boundaries of the rectangles (0=open or 1=closed). It can be specified in three ways: (1) A $n \times 4$ matrix containing 0's and 1's. Each row corresponds to a rectangle, and is denoted as $(cx_1, cx_2, cy_1, cy_2)$ . Here $cx_1$ denotes the boundary type of $x_1$ , $cx_2$ denotes the boundary type of $x_2$ , etc. (2) A vector $(cx_1, cx_2, cy_1, cy_2)$ containing 0's and 1's. This representation can be used if all rectangles have the same type of boundaries. (3) A vector $(c_1, c_2)$ containing 0's and 1's. This representation can be used if all x and y intervals have the same type of boundaries. $c_1$ denotes the boundary type of $x_1$ and $y_1$ , and $c_2$ denotes the boundary type of $x_2$ and $y_2$ . The default value is $c(0,1)$ . |

### Details

The functions `real2canon` and `canon2real` are carried out automatically in C-code as part of the functions `reduc` and `computeMLE`. We chose to make the functions available separately as well, in order to illustrate our algorithm for computing the MLE.

As a first step in the computation of the MLE, we transform rectangles into canonical rectangles, using `real2canon`). This is useful for two reasons. Firstly, it forces us in the very beginning to deal with possible ties and with the fact whether endpoints are open or closed. As a consequence, we do not have to account for ties and open or closed endpoints in the actual computation of the MLE. Secondly, it is convenient to work with the integer coordinates of the canonical rectangles in the computation of the MLE. After all computations are done, we transform the canonical rectangles back to their original coordinates, using `canon2real`. For more details, see Maathuis (2005, Section 2.1).

**Value**

A nx4 matrix of canonical observation rectangles. Each row (x1,x2,y1,y2) represents a rectangle.

**Author(s)**

Marloes Maathuis: <maathuis@stat.math.ethz.ch>

**References**

M.H. Maathuis (2005). Reduction algorithm for the NPMLE for the distribution function of bivariate interval censored data. *Journal of Computational and Graphical Statistics* **14** 252–262.

**See Also**

[canon2real](#)

**Examples**

```
# An example with 3 arbitrarily chosen observation rectangles
R <- rbind(c(3.5, 4.2, 3.3, 9.1), # first rectangle
          c(4.2, 4.9, 3, 4.5), # second rectangle
          c(3.8, 5.1, 8.1, 9.5)) # third rectangle

# Plot the rectangles
par(mfrow=c(2,2))
plotRects(R, lwd=2, main="Original rectangles")

# Transform rectangles to canonical rectangles. Since the
# boundaries of R1 and R2 coincide, it matters which boundaries
# we define to be open or closed.

# With boundary structure c(0,1), R1 and R2 do *not* overlap:
res1 <- real2canon(R, c(0,1))
plotRects(res1, grid=TRUE, lwd=2, main="Canonical rectangles
boundary c(0,1)")

# But with boundary structure c(1,1), R1 and R2 *do* overlap:
res2<-real2canon(R, c(1,1))
plotRects(res2, grid=TRUE, lwd=2, main="Canonical rectangles
boundary c(1,1)")
```

---

reduc

*Determine areas of possible mass support of the MLE*


---

**Description**

The MLE for censored data can only assign mass to a finite set distinct regions, called maximal intersections. The function `reduc` computes these areas, using the height map algorithm described in Maathuis (2005). In addition to the maximal intersections, the function can also output the height map and the clique matrix.

**Usage**

```
reduc(R, B=c(0, 1), hm=FALSE, cm=FALSE)
```

**Arguments**

- R** A  $n \times 4$  matrix containing the observation rectangles. Each row corresponds to a rectangle, represented as  $(x_1, x_2, y_1, y_2)$ . The point  $(x_1, y_1)$  is the lower left corner of the rectangle and the point  $(x_2, y_2)$  is the upper right corner of the rectangle.
- B** This describes the boundaries of the rectangles (0=open or 1=closed). It can be specified in three ways: (1) A  $n \times 4$  matrix containing 0's and 1's. Each row corresponds to a rectangle, and is denoted as  $(cx_1, cx_2, cy_1, cy_2)$ . Here  $cx_1$  denotes the boundary type of  $x_1$ ,  $cx_2$  denotes the boundary type of  $x_2$ , etc. (2) A vector  $(cx_1, cx_2, cy_1, cy_2)$  containing 0's and 1's. This representation can be used if all rectangles have the same type of boundaries. (3) A vector  $(c_1, c_2)$  containing 0's and 1's. This representation can be used if all  $x$  and  $y$  intervals have the same type of boundaries.  $c_1$  denotes the boundary type of  $x_1$  and  $y_1$ , and  $c_2$  denotes the boundary type of  $x_2$  and  $y_2$ .  
The default value is  $c(0,1)$ .
- hm** Logical, indicating if the heightmap must be outputted. The default value is FALSE. The height map is a  $(2n+1) \times (2n+1)$  matrix, where  $n$  is the number of observation rectangles. Its values give the number of rectangles that overlap at any given point.
- cm** Logical, indicating if the clique matrix must be outputted. The default value is FALSE. The clique matrix is a  $m \times n$  matrix, where  $m$  is the number of maximal intersections and  $n$  is the number of observation rectangles. The  $(i,j)$ th element of the matrix is 1 if the  $i$ th maximal intersection is contained in the  $j$ th observation rectangle, and it is 0 otherwise.

**Details**

The computation of the MLE for censored data can be split into two steps: a reduction step and an optimization step. In the reduction step, the areas of possible mass support are computed. Next, in the optimization step, it is determined how much probability mass should be assigned to each of these areas. The function `reduc` can be used for the reduction step. It is carried out automatically as part of the function `computeMLE`.

The time and space complexity of the function `reduc` depend on the parameters `hm` and `cm`. If `hm=FALSE` and `cm=FALSE`, then the algorithm is  $O(n^2)$  in time and  $O(n)$  in memory space. If `hm=TRUE` and `cm=FALSE`, then the algorithm is  $O(n^2)$  in both time and space. If `cm=TRUE`, then the algorithm is  $O(n^3)$  in time and space.

The function `reduc` uses the height map algorithm of Maathuis (2005). It first converts the observation rectangles to canonical rectangles. Next, it computes the local maxima of the height map, and then it convert these back to the original coordinates. This process can be mimicked by hand as follows: (1) use `real2canon` to convert the rectangles to canonical rectangles; (2) use `reduc` to find the canonical maximal intersections (local maxima of the height map of the canonical rectangles); (3) use `canon2real` to convert the canonical maximal intersections back to the original coordinates.

**Value**

A list containing the following elements:

<code>rects</code>	A $m \times 4$ matrix of maximal intersections. Each row $(x_1, x_2, y_1, y_2)$ represents a maximal intersection, i.e., an area where the MLE can possibly assign mass.
<code>bounds</code>	This describes the boundaries of <code>rects</code> . It is given in the same format as B.
<code>hm</code>	(Optional) A $(2n) \times (2n)$ matrix containing the height map. Its values represent the number of rectangles that overlap at any given point.
<code>cm</code>	(Optional) A $m \times n$ matrix containing the clique matrix. The $(i, j)$ th element of the matrix is 1 if the $i$ th maximal intersection is contained in the $j$ th observation rectangle, and it is 0 otherwise.

**Author(s)**

Marloes Maathuis: <maathuis@stat.math.ethz.ch>

**References**

M.H. Maathuis (2005). Reduction algorithm for the NPMLE for the distribution function of bivariate interval censored data. *Journal of Computational and Graphical Statistics* **14** 252–262.

**See Also**

[real2canon](#), [canon2real](#), [computeMLE](#), [plotCM](#), [plotHM](#)

**Examples**

```
# Load example data:
data(ex)
par(mfrow=c(1,1))

# Plot the observation rectangles
plotRects(ex, main="Example")

# Perform the reduction step
res<-reduc(ex, hm=TRUE, cm=TRUE)

# Shade the maximal intersections
plotRects(res$rects, density=15, add=TRUE, border=NA)

# Plot the height map, together with the observation
# rectangles (in black) and the maximal intersections (shaded)
plotHM(res$hm, ex)
plotRects(ex, add=TRUE, border="black")
plotRects(res$rects, add=TRUE, border=NA, density=15)

# Print the clique matrix
res$cm

# Make a plot of the clique matrix (useful for large data sets)
plotCM(res$cm)
```

# Index

- \*Topic **aplot**
  - plotCDF1, 15
  - plotDens1, 20
  - plotRects, 26
- \*Topic **datasets**
  - actg181, 2
  - actg181Mod, 3
  - cosmesis, 10
  - ex, 11
  - menopause, 12
  - menopauseMod, 13
- \*Topic **dplot**
  - plotCDF1, 15
  - plotCDF2, 17
  - plotDens1, 20
  - plotDens2, 22
- \*Topic **hplot**
  - plotCDF1, 15
  - plotCDF2, 17
  - plotCM, 19
  - plotDens1, 20
  - plotDens2, 22
  - plotHM, 24
  - plotRects, 26
- \*Topic **iteration**
  - computeMLE, 7
- \*Topic **manip**
  - canon2real, 5
  - real2canon, 27
  - reduc, 29
- \*Topic **nonparametric**
  - computeMLE, 7
  - reduc, 29
- \*Topic **optimize**
  - computeMLE, 7
  - reduc, 29
- \*Topic **survival**
  - computeMLE, 7
  - reduc, 29
- actg181, 2, 4, 21, 23
- actg181Mod, 2, 3
- axis, 19
- canon2real, 5, 6, 27, 28, 30
- computeMLE, 6, 7, 8, 15–18, 20–23, 28, 30
- cosmesis, 10
- ex, 11
- image, 17, 19, 22
- menopause, 12, 14
- menopauseMod, 13, 13
- palette, 25, 27
- plotCDF1, 8, 9, 15
- plotCDF2, 8, 9, 17
- plotCM, 19, 30
- plotDens1, 3, 4, 8, 9, 20
- plotDens2, 3, 4, 8, 9, 13, 14, 22
- plotHM, 3, 4, 13, 14, 24, 30
- plotRects, 25, 26
- rainbow, 26
- real2canon, 5, 6, 27, 27, 28, 30
- rect, 26, 27
- reduc, 6, 8, 9, 19, 24, 25, 28, 29, 29, 30