Package 'ProfileLadder'

July 23, 2025

Type Package

Title Functional Profile Chain Ladder for Claims Reserving

Version 0.1.3

Description Functional claims reserving methods based on aggregated chain-ladder data, also known as the run-off triangle (functional) development profiles, implemented in three nonparametric algorithms (PARALLAX, REACT, and MACRAME) proposed in Maciak, Mizera, and Pešta (2022) <doi:10.1017/asb.2022.4>.

Depends R (>= 4.0.0)

License GPL (>= 2)

Encoding UTF-8

Imports ChainLadder (>= 0.2.12), raw (>= 0.1.8)

LazyData true

Config/testthat/edition 3

RoxygenNote 7.3.2

Maintainer Matúš Maciak <maciak@karlin.mff.cuni.cz>

NeedsCompilation no

Author Matúš Maciak [aut, cre],
Rastislav Matúš [ctb],
Ivan Mizera [aut],
Michal Pešta [aut]

Repository CRAN

Date/Publication 2025-07-23 17:20:07 UTC

Contents

1

---

as.profileLadder   *S3 Method Class* profileLadder

Description

A function to make the work with the functional development profiles within run-off triangles more easy and straightforward (particularly when vizualizing the functional profiles—observed, completed, or both—in a plot)

Usage

```
as.profileLadder(x)
```

Arguments

x   an object of the class matrix or triangle

Value

an object of the class profileLadder which is a list with the following elements:

reserve   basic summary of the run-off triangle and the true/estimated reserve (if it is available otherwise NA values are provided instead)

method   type of the printed triangle (either a run-off triangle itself if no estimation method is applied or the completed triangle where the missing fragments are imputed by one of the algorithm, PARALLAX, REACT, or MACRAME)

completed	completed development profiles estimated by using one of the estimation algorithm (i.e., PARALLAX, REACT, or MACRAME)—if applied—value NA provided otherwise
inputTriangle	standard (triangular shaped) run-off triangle
trueComplete	true completed development profiles of the run-off triangle (if available) or NA returned otherwise

See Also

[parallelReserve()](#), [mcReserve()](#), [permuteReserve()](#), [plot.profileLadder()](#)

Examples

```
data(CameronMutual)
print(CameronMutual)

x <- as.profileLadder(CameronMutual)

print(x)
plot(x)
```

CameronMutual	*Cameron Mutual Insurance Company Data*

Description

An illustrative dataset—a matrix (of the dimensions 10x10) with ten completed years of claims payment developments of the Cameron Mutual Insurance company from the period 1988 – 1997. The data matrix contains ten origin/occurrence years (in rows where the first row represents the incident year 1988) with ten consecutive development periods/years (in columns).

Usage

```
data(CameronMutual)
```

Format

CameronMutual:

A simple 10x10 matrix of a class triangle with ten origin years (rows) each being fully developed within ten consecutive development periods/years (columns)

origin matrix rows with the occurrence year (origin)

dev matrix columns with the development period (development)

Details

The run-off triangle (the upper-left triangular part of the data matrix) contains only positive increments making the triangle suitable for the standard modelling approach—the over-dispersed Poisson model (GLM regression model).

In practice, the upper-left triangle (the run-off triangle) is typically observed (known) while the bottom-right triangular part of the data matrix is treated as a future payments outcome (an "unknown" truth) that should be estimated/predicted. The Cameron Mutual Insurance data matrix is fully observed to allow for some goodness-of-fit evaluations.

Source

https://www.casact.org/publications-research/research/research-resources
(PP Auto Data Set, NAIC group code: 5320)

References

Meyers, G. G. and P. Shi (2011). Loss reserving data pulled from NAIC Schedule P. Available from https://www.casact.org/publications-research/research/research-resources

Maciak, M., Mizera, I., and Pešta, M. (2022). Functional Profile Techniques for Claims Reserving. ASTIN Bulletin, 52(2), 449-482. DOI:10.1017/asb.2022.4 (Portfolio #1)

---

covid19CZ   *First Occurrences of Covid-19 Cases in the Czech Republic*

Description

An illustrative dataset—a matrix (of the dimensions 4x8) with the cumulative counts of the first reported cases of the Covid-19 pandemic in the Czech Republic. Four cohorts are defined by the Czech counties where the first reported case occurred during the period March 1st – 7th, 2020 (e.g., Prague, Vsetín, or Děčín), March 8th – March 14th (e.g, Brno, České Budějovice, Kladno, Mladá Boleslav, Plzeň), March 15th – March 21st (e.g., Chomutov, Český Krumlov, Písek, Tábor), and, finally, during the week in March 22nd – March 28th, 2020 (e.g., Jindřichův Hradec, Klatovy, Teplice).

Usage

```
data(covid19CZ)
```

Format

covid19CZ:

A simple 4x8 matrix of a class `triangle` with four cohorts (rows) consecutively observed for 8 weeks (starting in March 1st 2020 with the first case in the first cohort (first row) reported in March 1st)

Details

The cumulative reported cases are provided in the table for 8 consecutive development periods (where the periods represent seven consecutive days) starting in March 1st, 2020.

Source

Institute of Health Information and Statistics of the Czech Republic https://www.uzis.cz:443/

GFCIB   *Guarantee Fund of the Czech Insurers' Bureau Data*

Description

Illustrative datasets provided by the Guarantee Fund of the Czech Insurers' Bureau (GFCIB) for the mandatory car insurance in the Czech Republic. The quarterly based payments are aggregated in four run-off triangles with the paid amounts for four separate lines of business: bodily injury, material damage, technical provision, and annuities.

Usage

    data(GFCIB)

Format

GFCIB:

Four data matrices of the dimensions `60x60` of a class `triangle` with 15 origin years (provided on a quarterly basis in individual rows) and `60` development periods/quartals (columns)

origin matrix rows with the occurrence quartal (origin)
dev matrix columns with the development period (development)

Details

The data are structured in the the list object `GCCIB` with four elements—one for each line of business: \$bodilyInjury, \$materialDamage, \$provisions, and \$annuity. The run-off triangles are all aggregated over the period from the first quartal of 2008 (`Q1`) till the last quartal of 2022 (`Q4`).

Source

The Czech Insurers' Bureau https://www.ckp.cz

---

incrExplor   *Exploratory Function for Run-Off Triangle Increments*

Description

The function takes a cumulative or incremental run-off triangle (partially or completely observed) and provides some basic exploratory and graphical inspection of the observed incremental payments. The function serves as a useful tool for a user-based insight when manualy defining the states of the Markov Chain that is used to drive the reserve prediction in the MACRAME algorithm implemented in the function mcReserve().

Usage

```
incrExplor(
  triangle,
  method = c("median", "mean", "max", "min"),
  out = 1,
  states = NULL,
  breaks = NULL
)
```

Arguments

triangle	cumulative or incremental run-off triangle (an object of the class `triangle` or `matrix`) specified in terms of a partially observed (run-off triangle) or a fully observed (completed triangle) matrix. Only the upper-left triangular part (run-off trangle) is used to provide the output analysis of the incremental payments and the underlying Markov chain setting options
method	method form c("median", "mean", "max", "min") used to summarize the run-off triangle increments within the given set of bins. Each bin with the increments is represented by a corresponding Markov state value (obtained by the `method` choice with `median` being the DEFAULT option)
out	integer value (or a vector of integers) to indicate which columns of the run-off triangle should be excluded from the exploratory analysis of the increments. By DEFAULT, the first incremental payments—i.e., the first column of the run-off triangle—are not considered (out = 1). No colums are exluded for out = 0 and the whole run-off triangle is analyzed by incrExplor(). To specify multiple columns that should be excluded, one can use out = c(1,2,3) which will exlude the first three columns from the exploratory analysis
states	either an integer value to indicate an explicit number of the Markov chain states to be used or a vector of explicit Markov chain states can be provided. The DEFAULT option (states = NULL) provides a fully data-driven (automatic) definition of the Markov chain states as proposed in Maciak, Mizera, and Pešta (2022)
breaks	numeric vector of explicit (unique and monotonously increasing) break points to define the bins for the run-off triangle increments. If states is equal to some integer number (i.e., the explicit number of the Markov chain states is requested

by `states`) then the value of `breaks` is ignored. If both `states` and `breaks` are specified (i.e., numeric vectors are provided for both) then the set of states in `states` must be given in a way that exactly one state value belongs to exactly one bin defined by the break points specified by `breaks`

Value

An object of the class `mcSetup` with the following elements:

`incrTriangle`	an object of the class `triangle` with the incremental run-off triangle
`triangleType`	type of the input run-off triangle provided for the input object `triangle` (cumulative or incremental)
`defaultStates`	the data-driven set of explicit states as used (by DEFAULT) by the `mcReserve()` function – the MACRAME prediction algorithm
`defaultBreaks`	the set of explicit data-driven breaks as used (by DEFAULT) by the `mcReserve()` function – the MACRAME prediction algorithm
`increments`	table with basic empirical characteristics of the increments of the input run-off triangle (without the first origin payments—the values in the first column of the run-off triangle). Two sets of increments are provided: the raw incremental payments in the first row of the table and the standardized increments (i.e., row incremental payments divided by the maximum payment within the row (while not considering the columns specified by the out parameter)
`userDefined`	a list with all information regarding the USER modified input (numeric vector `increments` with the increments being analyzed; numeric value in `outColumns` denoting the excluded columns in the run-off triangle; `method` used to summarize the increments within the bins; numeric vector with the resulting Markov chain states in `states` and the corresponding numeric vector with the break points in `breaks` defining the bins for the run-off triangle increments)

References

Maciak, M., Mizera, I., and Pešta, M. (2022). Functional Profile Techniques for Claims Reserving. ASTIN Bulletin, 52(2), 449-482. DOI:10.1017/asb.2022.4

See Also

[mcReserve()](#), [permuteReserve()](#)

Examples

```
data(CameronMutual)

## default Markov Chain states with (roughly) equally occupied bins
incrExplor(CameronMutual)

## five Markov Chain states (with roughly equally occupied bins)
incrExplor(CameronMutual, states = 5)

## explicitly defined breaks for five increment bins while the Markov states
```

```
## are obtained as medians of the increments within each bin
incrExplor(CameronMutual, breaks = c(20, 500, 1000, 2000))

## explicitly defined breaks for five bins and the Markov states
## are given as the maximum increments within each bin
incrExplor(CameronMutual, breaks = c(20, 500, 1000, 2000), method = "max")

## manually defined breaks for the bins and the corresponding states
## exactly one state must be within each break
incrExplor(CameronMutual, breaks = c(20, 500, 1000),
                          states = c(10, 250, 800, 1500))
```

---

mcBreaks   *Access Markov Chain Breaks for Run-Off Triangle Increments*

Description

Retrieves the Markov chain components from the profileLadder object returned from the function mcReserve() or the mcSetup object returned from the function incrExplor(). In particular, the function returns the set of breaks used to define the bins for the incremental run-off triangle increments.

Usage

```
mcBreaks(object)
```

Arguments

object   An object of class profileLadder.

Value

The vector of the break points that define the set bins for the run-off triangle increments.

See Also

[mcReserve()](), [mcStates()](), [mcTrans()]()

Examples

```
## MACRAME reserve prediction with the DEFAULT Markov chain setup
output <- mcReserve(CameronMutual)

## Extracting the corresponding break points
mcBreaks(output)
```

| mcReserve | *MACRAME Based Development Profile Reserve* |

Description

The function takes a cumulative (or incremental) run-off triangle (partially or completely observed) and returns the reserve estimate obtained by the MACRAME algorithm (see Maciak, Mizera, and Pešta (2022) for further details).

Usage

```
mcReserve(
  chainLadder,
  cum = TRUE,
  residuals = FALSE,
  states = NULL,
  breaks = NULL
)
```

Arguments

chainLadder	a cumulative or incremental run-off triangle (the triangle must be of the class `triangle` or `matrix`) in terms of a square matrix with a fully observed upper-left triangular part. If the lower-right part is also provided the function will also return standard residuals but only the upper-left (run-off) triangle is be used for the reserve estimation purposes
cum	logical to indicate the time of the input triangle provided (DEFAULT value is `TRUE` for the cumulative triangle, `FALSE` if `chainLadder` is of the incremental type)
residuals	logical to indicate whether (incremental) residuals should be provided in output or not. If the run-off triangle is completely observed then the residuals are obtained in terms of the true increments minus the predicted ones. If the bottom-right triangle is not provided (`NA` values) then the residuals are obtained in terms of a back-fitting approach (see Maciak, Mizera, and Pešta (2022) for further details). However, the back-fitted residuals are only calculated when no user specification of the states (in `states`) and breaks (in `breaks`) is provided
states	numeric value to provide either the number of the Markov states to be used or it can provide an explicit set of the states to be used. The default setting (`states = NULL`) provides the set of states in a fully data-driven manner as proposed in Maciak, Mizera, and Pešta (2022) while any choice of `breaks` is ignored. If the number of states is specified by `states`, the states are obtained analogously as in Maciak, Mizera, and Pešta (2022), however, the number of actual states for the estimation is adjusted and the parameter `breaks` is again ignored
	If parameter `states` provides an explicit vector of Markov chain states (the smallest state should be larger than the smallest observed increment in the run-off triangle and, similarly, the largest state should be smaller than largest

observed increment) then the corresponding bins (breaks) for the run-off triangle increments are defined automatically by the midpoints between the provided states (with `breaks` being set to `NULL DEFAULT`)

breaks   vector parameter which provides explicit (unique and monotonly increasing) break points (disjoint bins) for the run-off triangle incremenets. Each bin should be represented by the corresponding Markov chain state—either the values given in `states` or provided automatically if `states` is not a valid vector of the Markov states. If the breaks are provided as `breaks = c(-Inf, ... , Inf)` defining k bins all together then `states` should be a vector of the same length k. Alternatively, the breaks can be also specified by a set of finite numbers defining again k bins—in such cases, the parameter `states` should be of the length `length(states) = k + 1`. Each value in `states` should represent one bin defined by `breaks`

Value

An object of the type `list` with with the following elements:

reserve   numeric vector with four values: Total paid amount (i.e., the sum of the last observed diagonal in a cumulative run-off triangle); Total estimated amount (i.e., the sum of the last column in the completed cumulative triangle); Estimated reserve (i.e., the sum of the last column in the completed cumulative triangle minus the sum of the last observed diagonal in `chainLadder`); True reserve if a completed `chainLadder` is provided in the output (i.e., the sum of the last column in `chainLadder` minus the sum of the last diagonal in `chainLadder`)

method   algorithm used for the reserve estimation

fullTriangle   completed run-off triangle (the upper-left triangular part is identical with the input triangle in `chainLadder` and the lower-right trianglular part is completed by the MACRAME algorithm)

inputTriangle   the input run-off triangle provided in `chainLadder`

trueCompleted   true completed triangle (if available) where the upper-left part is used by the MACRAME algorithm to estimate the reserve and the lower-right part is provided for some evaluation purposes. If the full triangle is not available `NA` is returned instead

residuals   a triangle with the corresponding residuals (for `residuals = TRUE`). The residuals are either provided in the upper-left triangle (so-called back-fitted incremental residuals if true completed triangle is not available) or the residuals are given in the lower-right triangle (i,e., standard incremental residuals—if the true completed triangle is given)

References

Maciak, M., Mizera, I., and Pešta, M. (2022). Functional Profile Techniques for Claims Reserving. ASTIN Bulletin, 52(2), 449-482. DOI:10.1017/asb.2022.4

See Also

[incrExplor()](), [permuteReserve()](), [mcBreaks()](), [mcStates()](), [mcTrans()]()

Examples

```
## run-off (upper-left) triangle with NA values
if (requireNamespace("ChainLadder")) {
data(MW2014, package = "ChainLadder")
print(MW2014)

## MACRAME reserve prediction with the DEFAULT Markov chain setting
mcReserve(MW2014, residuals = TRUE)}

## complete run-off triangle with 'unknown' truth (lower-bottom run-off triangle)
## with incremental residuals (true increments minus predicted ones)
data(CameronMutual)
mcReserve(CameronMutual, residuals = TRUE)

## the same output in terms of the reserve prediction but back-fitted residuals
## are provided instead (as the run-off triangle only is provided)
data(observed(CameronMutual))
mcReserve(observed(CameronMutual), residuals = TRUE)

## MACRAME reserve prediction with the underlying Markov chain with five
## explicit Markov chain states
mcReserve(CameronMutual, residuals = TRUE, states = c(200, 600, 1000))
```

---

mcStates   *Access Markov Chain States in the MACRAME Algorithm*

Description

Retrieves the Markov chain components from a `profileLadder` object returned from the function `mcReserve()` – in particular, the function returns the vector of states used by the underlying Markov Chain utilized in the MACRAME reserve prediction.

Usage

```
mcStates(object)
```

Arguments

object   An object of class `profileLadder`.

Value

The vector of the Markov chain states that are used by the MACRAME algorithm.

See Also

[mcReserve()](), [mcBreaks()](), [mcTrans()]()

Examples

```
## MACRAME reserve prediction with the DEFAULT Markov chain setup
output <- mcReserve(CameronMutual)

## Extracting the corresponding break points
mcStates(output)
```

---

mcTrans   *Access Markov Chain Transition Matrix in the MACRAME Algorithm*

Description

Retrieves the Markov chain components from a `profileLadder` object returned from the function `mcReserve()` – in particular, the function returns the matrix of the estimated transition probabilities used by the underlying Markov Chain to provide the reserve prediction.

Usage

```
mcTrans(object)
```

Arguments

object   An object of class `profileLadder`.

Value

The matrix of the estimated Markov chain transition probabilities

See Also

[mcReserve()](#), [mcBreaks()](#), [mcStates()](#)

Examples

```
## MACRAME reserve prediction with the DEFAULT Markov chain setup
output <- mcReserve(CameronMutual)

## Extracting the corresponding break points
mcTrans(output)
```

MidwestMutual   *Midwest Family Mutual Insurance Company Data*

Description

An illustrative dataset—a matrix (of the dimensions 10x10) with ten completed years of claims payment developments of the Midwest Family Mutual Insurance company from the period 1988 – 1997. The data matrix contains ten origin/occurrence years (with the first row representing the incident year 1988) and ten consecutive development periods/years (in columns).

Usage

```
data(MidwestMutual)
```

Format

MidwestMutual:

A simple `10x10` matrix of a class `triangle` with ten origin years (rows) each being fully developed within ten consecutive development periods/years (columns)

origin matrix rows with the occurrence year (origin)

dev matrix columns with the development period (development)

Details

The run-off triangle (the upper-left triangular part of the data matrix) contains only positive increments making the triangle suitable for the standard modelling approach—the over-dispersed Poisson model (GLM approach).

In practice, the run-off triangle only (the upper triangular part) of the data matrix is known while the bottom-right triangular part is treated as a future outcome (an "unknown" truth) that should be estimated/predicted. The Midwest Family Mutual Insurance data matrix is fully observed to allow for some retrospective goodness-of-fit evaluations.

Source

https://www.casact.org/publications-research/research/research-resources
(Other Liability Data Set, NAIC group code: 23574)

References

Meyers, G. G. and P. Shi (2011). Loss reserving data pulled from NAIC Schedule P. Available from https://www.casact.org/publications-research/research/research-resources

Maciak, M., Mizera, I., and Pešta, M. (2022). Functional Profile Techniques for Claims Reserving. ASTIN Bulletin, 52(2), 449-482. DOI:10.1017/asb.2022.4 (Portfolio #2)

NevadaGeneral	*Nevada General Insurance Company Data*

Description

An illustrative dataset—a matrix (of the dimensions 10x10) with ten completed years of claims payment developments of the Nevada General Insurance company from the period 1988 – 1997. However, the data matrix only contains four non-zero origin/occurrence years (from the period 1994 – 1997) all being fully developed for ten consecutive development periods/years (in columns). The remaining matrix rows are all zeros. The resulting run-off triangle (the upper-left triangular part of the data matrix) is, therefore, sparse and very uninformative.

Usage

```
data(NevadaGeneral)
```

Format

NevadaGeneral:

A simple `10x10` matrix of a class `triangle` with ten origin years (rows) each being fully developed within ten consecutive development periods (columns). However, only for development profiles are nonzero and standard (parametric) reserving techniques (e.g. the ODP model) are not applicable

origin matrix rows with the occurrence year (origin)

dev matrix columns with the development period (development)

Details

In practice, the reserve for such sparse run-off triangles is not estimated by any stochastic model but, instead, an expert judgement is used to declare the reserve. Nevertheless, the nonparametric estimation performed by PARALLAX, REACT, or MACRAME can still provide resonable reserve estimates

Source

https://www.casact.org/publications-research/research/research-resources
(PP Auto Data Set, NAIC group code: 10007)

References

Meyers, G. G. and P. Shi (2011). Loss reserving data pulled from NAIC Schedule P. Available from https://www.casact.org/publications-research/research/research-resources

---

observed   *Observed Run-Off Triangle Layout vs. Predicted (Unknown) Layout*

Description

Simple layout function to make work with (cumulative or incremental) run-off triangles more easy and straightforward.

Usage

```
observed(object, cum = TRUE)
```

Arguments

object	either an integer value to denote the dimension of the run-off triangle layout (i.e., the value that represents the number of origins (rows) and also the number of the development periods (columns)). Alternatively, a cumulatige or incremental run-off triangle (i.e, an object of the class matrix or triangle) can be provided in object. In such case the output returns the standard run-off triangle with NA values in the lower-right triangular part of the matrix (regardless of wheter the input triangle in object forms a run-off triangle or it is a fully observed triangle—data matrix)
cum	logical to indicate whether the output run-off triangle is supposed to be of a cumulative type (DEFAULT) or an incremental type (cum = FALSE). If the input in object is an integer value (i.e., the dimension of the run-off triangle) then the choice of the cum parameter is ignored

Value

If object is an integer value then the function returns a TRUE/FALSE layout matrix with the TRUE values for the observed (known) part of the run-off triangle (the upper-left triangular part of the matrix) and values FALSE otherwise. If object is a matrix (an object of the class matrix or triangle) then the function returns the observed (known) part of the run-off triangle with NA values elsewhere. Depending on the choice of the cum parameter, either a cumulative (DEFAULT) or incremental (cum = FALSE) run-off triangle is returned

See Also

[plot.profileLadder()](), [parallelReserve()](), [mcReserve()]()

Examples

```
print(observed(5))
print(!observed(5))

data(CameronMutual)
observed(CameronMutual)
observed(CameronMutual, cum = FALSE)
```

---

parallelReserve   *Parallel Based Development Profile Reserve*

Description

The function takes a cumulative (or incremental) run-off triangle (partially or completely observed) and returns the reserve estimate obtained by the PARALLAX or REACT algorithm (see Maciak, Mizera, and Pešta (2022) for more details). If the full square is provided as the input then the algorithms still rely only on the partially observed data—run-off triangle only (i.e., the top-left triangular part of the data)—when estimating the underlying reserve but, in addition, incremental residuals (true increments minus predicted increments) are returned for retrospective validation purposes. If the run-off triangle is provided,then algorithm caclulates back-fitted (incremental) residuals instead (see Maciak, Mizera, and Pešta (2022) for details).

Usage

```
parallelReserve(
  chainLadder,
  method = "parallax",
  cum = TRUE,
  residuals = FALSE
)
```

Arguments

chainLadder	cumulative or incremental run-off triangle (the triangle must be of the class `triangle` or `matrix`) in terms of a square matrix (i.e., a fully observed run-off triangle) or a standard run-off triangle instead (i.e, the top-left triangular part of the matrix
method	prediction method to be used: PARALLAX (DEFAULT method = "parallax") or REACT (method = "react")
cum	logical (TRUE for a cumulative triangle and FALSE for an incremental triangle)
residuals	logical to indicate whether incremental residuals should be provided or not. If the run-off triangle is complete then the residuals are obtained in terms of true increments minus the predicted increments. If the bottom-right part of the triangle is not available the residuals are provided in terms of the backfitting approach (see Maciak, Mizera, and Pesta (2022) for further details)

Value

An object of the class `list` with with the following elements:

reserve	numeric vector with four values summarizing the reserve: Total paid amount (i.e., the sum of the last observed diagonal in a cumulative run-off triangle); Total estimated amount (i.e., the sum of the last column in the completed cumulative triangle); Estimated reserve (i.e., the sum of the last column in the completed cumulative triangle minus the sum of the last observed diagonal in

chainLadder); True reserve—if the completed (true) `chainLadder` is provided in the input (i.e., the sum of the last column in `chainLadder` minus the sum of the last diagonal in `chainLadder`)

method	algorithm used for the reserve estimation (PARALLAX or REACT)
completed	completed functional development profiles (the lower-right triangular part in `completed`) estimated by the PARALLAX algorithm or the REACT algorithm
inputTriangle	the run-off triangle considered as the input for the underlying estimation algorithm (PARALLAX or REACT)
trueCompleted	true (complete) run-off triangle (if available) and NA value provided otherwise
residuals	a triangle with the corresponding residuals (for `residuals = TRUE`). The residuals are either provided in the upper-left triangle (so-called back-fitted incremental residuals if true completed triangle is not available) or the residuals are given in the lower-right triangle (i,e., standard incremental residuals—if the true completed triangle is given)

References

Maciak, M., Mizera, I., and Pešta, M. (2022). Functional Profile Techniques for Claims Reserving. ASTIN Bulletin, 52(2), 449-482. DOI:10.1017/asb.2022.4

See Also

[permuteReserve()](#), [mcReserve()](#)

Examples

```
## run-off (upper-left) triangle with NA values (bottom-right part)
if (requireNamespace("ChainLadder")) {
data(MW2014, package = "ChainLadder")
print(MW2014)
parallelReserve(MW2014, residuals = TRUE)}

## completed run-off triangle with 'unknown' truth (lower-bottom part)
## for the estimation purposes only the upper-left triangle is used
data(CameronMutual)
parallelReserve(CameronMutual, residuals = TRUE)

## the previous output is identical (in term of the reserve prediction)
## but back-fitted residuals are provided in the output instead
print(observed(CameronMutual))
parallelReserve(observed(CameronMutual), residuals = TRUE)
```

---

permuteReserve   *Permutation Bootstrap Reserve (PARALLAX, REACT, MACRAME)*

Description

The function takes the output from the function `parallelReserve()` or `mcReserve` and estimates the overall reserve distribution in terms of the permutation bootstrap approach proposed in Maciak, Mizera, and Pešta (2022).

Usage

```
permuteReserve(object, B = 500, std = TRUE, quantile = 0.995)
```

Arguments

object	an object of the class `profileLadder` (output from `parallelReserve()` or `mcReserve()` respectively)
B	number of permutations to be performed (DEFAULT B = 500)
std	logical to indicate whether the run-off triangle should be standardized by the first column increments (DEFAULT) or not (`std = FALSE`). For more details about the triangle standardization, see Maciak, Mizera, and Pešta (2022)
quantile	quantile level for the `BootVar.` characteristic of the bootstrapped distribution (the DEFAULT choice `quantile = 0.995` is explicitly required by the Solvency II principle used by actuaries in practice)

Value

An object of the class `permutedReserve` which is a list with the following elements:

eSummary	numeric vector with four values summarizing the estimated reserve: Paid amount (i.e., the sum of the last observed diagonal in the given cumulative run-off triangle); Estimated ultimate (i.e., the sum of the last column in the completed cumulative triangle); Estimated reserve (i.e., the sum of the last column in the completed cumulative triangle minus the sum of the last observed diagonal); True reserve if a completed (true) run-off triangle is available
pSummary	numeric vector with four values summarizing the overall reserve distribution: `Boot.Mean` gives the verage of B permutation bootstrap reserves; `Std.Er.` provides the corresponding standard error of B permutation bootstrap reserves; The value of `BootCov%` stands for a percentage proportion between the standard error and the average; Finally, `BootVar.995` provides the estimated 0.995 quantile (by DEFAULT) of the bootstrap reserve distribution (for `quantile = 0.995` and, otherwise, it is modified acordingly) given relatively with respect to the permutation bootstrapped mean reserve
pReserves	a numeric vector of the length B with the estimated (permuted) reserves for each row-permuted run-off triangle in B independent Monte Carlo simulation runs

pUltimates	A matrix of the dimensions B x n (where n stands for the number of the origin/development periods) with B simulated ultimate payments – the last column in the completed run-off triangle
pLatest	A matrix of the dimensions B x n (where n again stands for the number of the origin/development periods) with B simulated incremental diagonals
pLatestCum	A matrix of the dimensions B x n (n being the number of the origin/development periods) with B simulated cumulative diagonals
inputTriangle	The input run-off triangle
completed	The completed run-off triangle by using one of the PARALLAX, REACT, or MACRAME estimation method
trueComplete	The true complete run-off triangle (if available) and NA value otherwise
info	a numeric vector summarizing the bootstrap compuational efficiency: In particular, the OS/Architecture type, the number of permutations (B), the input run-off triangle dimension (n) and the computation time needed for the permutation bootstrap calculations

References

Maciak, M., Mizera, I., and Pešta, M. (2022). Functional Profile Techniques for Claims Reserving. ASTIN Bulletin, 52(2), 449-482. DOI:10.1017/asb.2022.4

European Parliament and Council (2009). Directive 2009/138/EC of the European Parliament and of the Council of 25 November 2009 on the taking-up and pursuit of the business of Insurance and Reinsurance (Solvency II). Official Journal of the European Union, 1–155.
https://data.europa.eu/eli/dir/2009/138/oj

See Also

parallelReserve(), mcReserve(), plot.permutedReserve()

Examples

```
## REACT algorithm and the permutation bootstrap reserve
data(CameronMutual)
output <- parallelReserve(CameronMutual, method = "react")
permuteReserve(output, B = 100)

## MACRAME algorithm with a pre-specified number of states
output <- mcReserve(CameronMutual, states = 5)
permuteReserve(output, B = 100)
```

plot.mcSetup *Plotting the Run-Off Triangle Increments by the MC Setting*

Description

The function provides a graphical visualization of the results obtained from the function incrExplor().
In particular, the considered run-off triangle increments are distributed into the bins according the
given Markov chain breaks or states.

Usage

```
## S3 method for class 'mcSetup'
plot(x, ...)
```

Arguments

x	an object of the class mcSetup – i.e., the output of the incrExplor() function
...	other graphical parameters to plot

Value

The function returns a layout with two plots. ...

See Also

[incrExplor()](), [mcReserve()]()

Examples

```
## run-off triangle increments within the default bins
x <- incrExplor(CameronMutual)
plot(x)

## run-off triangle increments and user-defined number of bins
x <- incrExplor(CameronMutual, states = 5)
plot(x)

## run-off triangle increments within the user-specified bins
x <- incrExplor(CameronMutual, breaks = c(500, 1000, 1500))
plot(x)
```

plot.permutedReserve   *Plotting the Output of the Permutation Bootstrap*

Description

The function provides a graphical visualization of the results obtained from the permutation boot-strap (see Maciak, Mizera, and Pesta (2022) for further details) applied to the output of one of the nonparametric functional based estimation algorithm—PARALLAX or REACT implemented in the `parallelReserve()` function or MACRAME implemented in the `mcReserve()` function.

Usage

```
## S3 method for class 'permutedReserve'
plot(x, ...)
```

Arguments

x	an object of the class `permutedReserve` – i.e., the output of the `permuteReserve()` function
...	other graphical parameters to plot

Value

The function returns a layout for four plots. The first panel shows a simple barplot type visualization of the estimated reserve, the estimated ultimate and the true reserve (if available). The second panel provides a histogram for (permuted) bootstrapped reserves with a nonparametric estimate of the corresponding density. The third panel provides a detailed inspection of the bootstrapped ultimates (with true ultimates if provided) and, finaly, the last panel shows the observed diagonal vs. simulated ones.

See Also

[permuteReserve()](#)

Examples

```
## reserve estimated by MACRAME and the corresponding visualization
x <- mcReserve(CameronMutual)
plot(permuteReserve(x, B = 100))
```

---

plot.profileLadder   *Plotting Development Profiles*

Description

The function provides a graphical representation of the completed functional profiles estimated by
the PARALLAX, REACT, or MACRAME algorithm (see Maciak, Mizera, and Pesta (2022) for
further details). The function takes an object of the class `profileLadder` which is the output of the
`parallelReserve()` function or the `mcReserve()` function. Alternatively, the function can be also
applied to visualise the run-off triangle itself—if the triangle is of the class `profileLadder`.

Usage

```
## S3 method for class 'profileLadder'
plot(x, xlab = "Development Year", ylab = "Cumulative Claims", main = "", ...)
```

Arguments

x	an object of the class `profileLadder` (output form `parallelReserve()`, `mcReserve()`, or `as.profileLadder()`
xlab	label for the x axis
ylab	label for the y axis
main	title of the plot
...	other graphical parameters to plot

Value

A graph with the observed functional development profiles from the input run-off triangle, the
estimated/predicted functional segments (i.e., functional profile completion provided by the cor-
responding estimation method—PARALLAX, REACT, or MACRAME) the and the true future
profiles (if these are available)

See Also

`as.profileLadder()`, `parallelReserve()`, `mcReserve()`

Examples

```
## completed run-off triangle with the 'unknown' (future) payments
print(triangle <- GFCIB$bodilyInjury[1:15, 1:15])
plot(mcReserve(triangle))

## completed run-off triangle with unknown future
print(observed(triangle))
plot(mcReserve(observed(triangle)))

## the run-off triangle with future payments without MACRAME completion
```

```
plot(as.profileLadder(triangle))
```

---

print.mcSetup   *Print Objects of the S3 Class* mcSetup

Description

Function to organize and print the output provided by the function incrExplor()

Usage

```
## S3 method for class 'mcSetup'
print(x, ...)
```

Arguments

x   an object of the class mcSetup resulting from a call of the incrExplor() func-
            tion

...   further arguments passed to print

Value

displays information resulting from a call of the incrExplor() function

See Also

[incrExplor()](), [mcReserve()](), [mcBreaks()](), [mcStates()]()

Examples

```
data(CameronMutual)
x <- incrExplor(CameronMutual)
print(x)
```

---

print.permutedReserve   *Print Objects of the S3 Class* permutedReserve

---

Description

Function to organize and print the output provided by the permutation bootstrap method implemented in the function permuteReserve()

Usage

```
## S3 method for class 'permutedReserve'
print(x, ...)
```

Arguments

x	an object of the class permutedReserve resulting from a call of the functions permuteReserve()
...	further arguments passed to print

Value

Displays information about the estimated reserve (by one of the estimation algorithms – PARALLAX, REACT, or MACRAME) and the overall reserve distribution resulting from a call of the permuteReserve() function

See Also

permuteReserve()

Examples

```
## reserve point prediction by the PARALLAX method
output <- parallelReserve(CameronMutual)

## reserve distribution prediction by the permutation bootstrap
x <- permuteReserve(output, B = 100)

## summary of the results
print(x)
```

---

print.profileLadder   *Print Objects of the S3 Class* profileLadder

---

Description

Function to organize and print the outputs provided by the function parallelReserve() and the function mcReserve

Usage

```
## S3 method for class 'profileLadder'
print(x, ...)
```

Arguments

x	an object of the class profileLadder resulting from a call of one of the functions parallelReserve(), mcReserve, or as.profileLadder()
...	further arguments passed to print

Value

displays information resulting from a call of the parallelReserve() function or the mcReserve function

See Also

[as.profileLadder()](), [parallelReserve()](), [mcReserve()]()

Examples

```
data(CameronMutual)
x <- as.profileLadder(CameronMutual)
print(x)
```

---

summary.mcSetup   *Summary Method for the S3 Class Object* mcSetup

---

Description

The function provides an overall summary of the output from the function incrExplor()

Usage

```
## S3 method for class 'mcSetup'
summary(object, ...)
```

Arguments

object	an object of the class `mcSetup` – the output from the `incrExplor()` function
...	not used

Value

Returns a standard summary table (with basic description characteristics) for raw run-off triangle increments and their standardized (by using the maximum increment) counterparts. The function also returns the corresponding bins for the increments and their representations in terms of the Markov chain states.

See Also

[incrExplor()](), [mcBreaks()](), [mcStates()]()

Examples

```
data(CameronMutual)
summary(CameronMutual)

## default summary output
summary(incrExplor(CameronMutual))

## summary output for user-modified settings
summary(incrExplor(CameronMutual, states = 5, method = "mean"))
```

---

summary.permutedReserve

Summary Method for Objects of the S3 Class Method
`permutedReserve`

Description

The function provides an overall summary of the output from the functions permuteReserve() (i.e., the summary of the object of the class permutedReserve)

Usage

```
## S3 method for class 'permutedReserve'
summary(object, ...)
```

Arguments

object	an object of the class `permutedReserve` – i.e., the output form the `permuteReserve()` functions
...	not used

Value

Summary of the completed functional profiles (provided by one of the functions `parallelReserve()` or `mcReserve()`) and the overall reserve distribution obtained in terms of the permutation bootstrap – the function `permuteReserve()`. The output is a list with the following items:

origins   a matrix with the row-specific summary of the completed functional profiles (except the first fully developed profile—i.e., the first row in the run-off triangle). The first column of the matrix (`First`) gives the first origin payments; The second column (`Latest`) gives the last available (cumulative) payments (i.e., values from the last running diagonal in the run-off triangle); The third column (`Dev.To.Date`) gives a relative proportion of the paid amount (`Latest`) with respect to the estimated ultimate (`Ultimate`) given in the fourth column; The column denoted as `IBNR` gives the estimated amount still left to pay (Incurred But Not Reported)

; The sixth column provides the estimated standard errors (`S.E.`) of IBNR obtained from the permutation bootstrap; The last column returns the corresponding coefficients of variation (`CV`).

overall   Table with the summary of the true/estimated reserve: `Paid amount` represents the sum of the last running diagonal; `Estimated reserve` gives the reserve estimate provided by one of the estimation algorithm (PARALLAX, REACT, or MACRAME); `True reserve` is given as a sum of the last column (if available, NA otherwise); Finally, some Accuracy in terms of `Reserve%` is given as a percentage of the estimated reserve with respect to the true reserve (see Maciak, Mizera, and Pešta (2022) and `Dev.To.Date` gives the proportion of the overall estimatd ultimate and the overall paid amount

dist   Table with basic empirical characteristics of the overall reserve distribution provided by the permutation bootstrap: `Boot.Mean` stands for the empirical mean of the bootstrap distribution; `Std.Er.` gives the corresponding standard error of the bootstrap distribution; `BootCov%` stands for a percentage proportion between the standard error and the empirical mean of the bootstrap distribution; Finally, `BootVar.xxx` provides the estimated quantile of the bootstrap reserve distribution (0.995 by DEFAULT).

See Also

[parallelReserve()](), [mcReserve()](), [permuteReserve()]()

Examples

```
data(CameronMutual)
summary(CameronMutual)

## summary for the point reserve prediction
summary(parallelReserve(CameronMutual))

## summary for the overall reserve distribution
summary(permuteReserve(parallelReserve(CameronMutual)))
```

---

summary.profileLadder   *Summary Method for Objects of the S3 Class Method* `profileLadder`

Description

The function provides an overall summary of the output from the functions parallelReserve() and mcReserve() (summary of the object of the class `profileLadder`)

Usage

```
## S3 method for class 'profileLadder'
summary(object, plotOption = FALSE, ...)
```

Arguments

`object`	an object of the class `profileLadder` – i.e., either a run-off triangle itself or the output form the `parallelReserve()` or `mcReserve()` functions
`plotOption`	logical to indicate whether a graphical output should be also provided (set by DEFAULT to `FALSE`). If the incremental residuals (standard or back-fitted) are provided within the object x the plot provides a summary of the residuals (otherwise a simple barplot summarizing the estimated reserve is given)
`...`	not used

Value

Summary of the completed functional profiles and the estimated reserve (provided by the function `parallelReserve()` or `mcReserve()`). Summary of the incremental residuals (standard or back-fitted) is also provided if the residuals are available. The output is a list with the following items:

`origins`	a matrix with the row-specific summary of the completed functional profiles (except the first fully developed profile—i.e., the first row in the run-off triangle). The first column of the matrix (`First`) gives the first origin payments; The second column (`Latest`) gives the last available (cumulative) payments (i.e., values from the last running diagonal in the run-off triangle); The third column (`Dev.To.Date`) gives a relative proportion of the paid amount (`Latest`) with respect to the estimated ultimate (`Ultimate`) given in the fourth column; Finally, the last column (`IBNR`) gives the estimated amount still left to pay (Incurred But Not Reported)
`overall`	Table with the summary of the true/estimated reserve: `Paid amount` represents the sum of the last running diagonal; `Estimated reserve` gives the reserve estimate provided by one of the estimation algorithm (PARALLAX, REACT, or MACRAME); `True reserve` is given as a sum of the last column (if available, `NA` otherwise); Finally, some Accuracy in terms of `Reserve%` is given as a percentage of the estimated reserve with respect to the true reserve (see Maciak, Mizera, and Pešta (2022) and `Dev.To.Date` gives the proportion of the overall estimatd ultimate and the overall paid amount
`resids`	Table with basic empirical description characteristics of the residuals (standard or back-fitted) if the residuals are provided in x

See Also

as.profileLadder(), parallelReserve(), mcReserve()

Examples

```
data(CameronMutual)
summary(CameronMutual)

## standard summary output
summary(mcReserve(CameronMutual))

## summary output with plotOption = TRUE
summary(mcReserve(CameronMutual), plotOption = TRUE)

## summary output with (standard) residuals and plotOption = TRUE
summary(mcReserve(CameronMutual, residuals = TRUE), plotOption = TRUE)

## summary output with (back-fitted) residuals and plotOption = TRUE
summary(mcReserve(observed(CameronMutual), residuals = TRUE), plotOption = TRUE)
```

---

xNetSubscribe   *Internet Provider Monthly Income Data*

Description

An illustrative dataset—a matrix (of the dimensions 12x12) with a monthly-based income (in EUR) of a local internet data provider with the income structured by the customers subscribing within the given month (in 2023) reported in the rows and monthly-based payments reported in columns. The data matrix represents the incremental type of the run-off triangle.

Usage

```
data(xNetSubscribe)
```

Format

xNetSubscribe:

A simple 12x12 (trangular) matrix of the class triangle with twelve consecutive months (January 2023 – December 2023) when new customers subscribed to the stream service (rows) and monthly-based payments (columns)

Index