Package ‘reserving’

March 30, 2006

Version 0.1-2

Date 2006-03-19

Title Actuarial tools for reserving analysis

Author Markus Gesmann

Maintainer Markus Gesmann <Markus.Gesmann@web.de>

Depends R (>= 2.2.0)

Suggests boot, Hmisc, lattice, MASS, grid

Description Tools for reserving analysis on insurance data, including standard chain ladder, Munich chain ladder and bootstrapping methods

License GPL Version 2 or later.

R topics documented:

BootReserve *Reserving Analysis of a Run Off Triangle by Bootstrapping*

Description

`BootReserve` relies on the `boot` function in the `boot` library and gives an estimated reserve, estimated standard error as well as a distribution of the reserve back.

Usage

```
BootReserve(triangle, R = 999, quarterly = FALSE,
            YOA = c(1:nrow(triangle)),
            tail = 1, usetail = FALSE, set.seed = 1, ...)
```

Arguments

`triangle`	`data.frame` of an accumulative quarterly or yearly development triangle of at least two underwriting years. The first column of this data frame may include the underwriting years, if its name is an element of the list: "UWY", "YOA", "YoA"
`R`	The number of bootstrap replicates.
`tail`	An optinal tail factor. The default is no tail factor, hence `tail=1`
`usetail`	logical. If `TRUE`, the setting of `tail` will be ignored and the function `tailfactor` will be called to estimate by loglinear regression a tail factor. If `FALSE` the value of `tail` will be used as tail factor.
`quarterly`	logical. `TRUE` if the triangle contains quarterly development data. `FALSE` if the triangle contains yearly development data.
`YOA`	An optional vector of underwriting years. If the first column of `object` is named UWY or YOA or YoA, this column will be used.
`set.seed`	if not NULL, this value will be used to set the random seed.
`...`	further parameters to boot.

Details

The implementation in `R` follwos the paper: Peter England and Richard Verrall, *Analytic and bootstrap estimates of prediction errors in claims reserving Insurance*, Mathematics and Economics **Vol. 25**, pp 281-293, 1999

Value

`BootReserve` gives an object of class "BootReserve" and "boot" back.

An object of class "BootReserve" is a list containing the following components:

`YOA`	Vector of underwriting years
`Triangle`	The original triangle
`Latest`	Vector of the latest available data for each UWY in the triangle
`Ultimate`	Vector of the bootstrap ultimates
`Reserve`	Vector of the bootstrap reserves

`Tail`	Tail factor
`BootSE`	Bootstrap standard error
`ReserveSE`	Prediction error of the reserve
`BootTotalSE`	Bootstrap standard error over all underwriting years
`ReserveTotalSE`	Prediction error of the reserve over all underwriting years
`ReserveTotalDist`	Reserve distribution over all underwriting years
`Call`	The original call to `ChainLadder`
`...`	Further objects of class "boot", see `boot`

Warning

Also it is possible to use a tail factor, "it should be noted that no allowance has been made for a tail factor in the bootstrap calculations. It is not obvious how uncertainty in predicetd values beyond the range of data observed should be taken into account. A fixed tail factor should not be included as this will increase the reserve estimates but leave the estimation variance unchanged, thus reducing the prediction error as a percentage of the reserve estimate. Extrapolating can only increase the uncertainty, not reduce it." Peter England and Richard Verrall (1999)

Author(s)

Markus Gesmann, ⟨Markus.Gesmann@web.de⟩

References

Peter England and Richard Verrall, *Analytic and bootstrap estimates of prediction errors in claims reserving Insurance*, Mathematics and Economics **Vol. 25**, pp 281-293, 1999

P.D.England and R.J.Verrall, *Stochastic Claims Reserving in General Insurance*, British Actuarial Journal, **Vol. 8**, pp443-544, 2002

See Also

See Also `ChainLadder`, `plot.BootReserve`

Examples

```
# See the example in Appendix A in Peter England and Richard Verrall (1999)

data(GenIns)
GenIns
bootG <- BootReserve(GenIns, quarterly=FALSE)
bootG

# See also the example in table 33 in England and Verrall (2002)

data(RAA)
RAA
bootR<-BootReserve(RAA, quarterly=FALSE)
bootR
plot(bootR)
#
plot(bootR, byYOA=TRUE)
```

`ChainLadder` *Chain Ladder Reserving Analysis of a Run Off Triangle*

Description

`ChainLadder` estimates the reserves and their standard errors of a given accumulative run off triangle by the chain ladder method. Quarterly and yearly development triangles are accepted.

Usage

```
ChainLadder(triangle, quarterly = FALSE, tail = 1,
            usetail = FALSE, YOA = c(1:nrow(triangle)))
```

Arguments

`triangle`	An accumulative triangle of quarterly or yearly development data stored in a `data.frame` of at least two underwriting years. The first column of this data frame may include the underwriting years, if its name is an element of the list: "UWY", "YOA", "YoA"
`quarterly`	logical. `TRUE` if the triangle contains quarterly development data. `FALSE` if the triangle contains yearly development data
`tail`	An optinal tail factor. The default is no tail factor, hence `tail=1`
`usetail`	logical. If `TRUE`, the setting of `tail` will be ignored and the function `tailfactor` will be called to estimate a tail factor by loglinear regression through the chain ladder ratios. If `FALSE` the value of `tail` will be used as tail factor.
`YOA`	An optional vector of underwriting years. If the first column of `object` is named "UWY" or "YOA" or "YoA", this column will be used.

Details

The implementation in `R` follwos the paper: The Standard Error of Chain Ladder Reserve Estimates: Recursive Calculation and Inclusion of a Tail Factor, Thomas Mack, 1999, *Astin Bulletin* **Vol. 29, No.2**, 361 - 266.

Value

`ChainLadder` returns an object of `class` "ChainLadder" and "data.frame". An object of class "ChainLadder" is a list containing the following components:

`YOA`	Vector of underwriting years
`Quarterly`	logical.
`Triangle`	The original triangle
`Latest`	Vector of the latest available data for each UWY in the triangle
`Ratios`	Vector of the chain ladder ratios
`Factors`	Vector of the chain ladder factors
`Dev`	Vector of the chain ladder development pattern
`Ultimate`	Vector of the chain ladder ultimate
`Reserve`	Vector of the chain ladder reserves

`Tail`	Tail factor
`FullTriangle`	Data frame of the filled triangle by chain ladder method
`sigma`	Estimation of the propotional constant of the variance
`mack.sef`	Estimation of the standard error of the chain ladder ratios
`mack.seF`	Estimation of the standard error of the individiual chain ladder ratioss
`MackSE`	Estimation of the standard error of the reserve and ultimate by underwriting year
`MackTotalSE`	Estimation of the standard error for the sum of all underwriting years.
`Call`	The original call to `ChainLadder`

Author(s)

Markus Gesmann, ⟨Markus.Gesmann@web.de⟩

Source

Distribution-free Calculation of the Standard Error of Chain Ladder Reserve Estimates, Thomas Mack, 1993, *ASTIN Bulletin* **23**, 213 - 225

References

The Standard Error of Chain Ladder Reserve Estimates: Recursive Calculation and Inclusion of a Tail Factor, Thomas Mack, 1999, *Astin Bulletin* **Vol. 29, No.2**, 361 - 266

P.D.England and R.J.Verrall, *Stochastic Claims Reserving in General Insurance*, British Actuarial Journal, **Vol. 8**, pp443-544, 2002

See Also

See Also: `plot.ChainLadder` and `plotCLRatios`, `plotCLResiduals` for plotting and `BootReserve` for the bootstrapping approach

Examples

```
# See also the first example in Mack (1993)
data(GenIns)
GenIns
G<-ChainLadder(GenIns, quarterly=FALSE, usetail=FALSE, YOA=c(1997:2006))
G
G$Ratios
G$sigma^2/1000
par(mfrow=c(1,2), oma=c(0,0,2,0))
# plot the development for each underwriting year
matplot(t(GenIns), t="o", main="Cum. vs Dev Year",
xlab="Development years", ylab="Cum. Amounts")
matplot(t(incTria(GenIns)), t="o", main="Inc. vs Dev Year",
xlab="Development years", ylab="Inc. Amounts")
title("Paid Development by UWY",out=TRUE)
par(mfrow=c(1,1))
# plot development of YOA 2000
plot(G, YOA=2000, main="Paid +/- MackSE")
# plot overview
plot(G)
# lattice plot by underwriting years
plot(G, byYOA=TRUE)
```

```
#  See also the second example in Mack (1993)
data(Mortgage)
Mortgage
M<-ChainLadder(Mortgage, quarterly=FALSE, usetail=FALSE)
M
M$Ratios
M$sigma^2/1000

# See also the example in Mack (1999)
MT<-ChainLadder(Mortgage, quarterly=FALSE, tail=1.05)
MT
MT$Ratios
# Beware of difference in sigma[9] comparing to Mack result
MT$sigma

# See also the example in table 13 in England and Verrall (2002)
data(RAA)
R<-ChainLadder(RAA, quarterly=FALSE, YOA=1997:2006)
R
# plot as ChainLadder object, cumulatvie vs cumulative
plotCLRatios(R)
# plot as original data.frame object, incremental vs cumulative
plotCLRatios(R, inc.vs.cum=TRUE)
# concentrate on the first two dev. periods
plotCLRatios(R, inc.vs.cum=FALSE, dev=1:2)
# plot the weighted residuals
plotCLResiduals(R)
# plot chain ladder result overview
plot(R)
# lattice plot by underwriting years
plot(R, byYOA=TRUE)
```

`GenIns` *Run off triangle (accumulated figures) from a portfolio of general insurance policies.*

Description

Run off triangle of accumulated claims data.

Usage

```
data(GenIns)
```

Format

A data frame with 10 underwriting years and 10 development years.

Source

Second moments of estimates of outstanding claims, G.C. Taylor & F.R. Ashe, *Journal of Econometrics*, **23**, pp 37-61

References

See Also: Distribution-free Calculation of the Standard Error of Chain Ladder Reserve Estimates, Thomas Mack, 1993, *ASTIN Bulletin* **23**, 213 - 225

Peter England and Richard Verrall, *Analytic and bootstrap estimates of prediction errors in claims reserving Insurance*, Mathematics and Economics **Vol. 25**, pp 281-293, 1999

P.D.England and R.J.Verrall, *Stochastic Claims Reserving in General Insurance*, British Actuarial Journal, **Vol. 8**, pp 443-544, 2002

Examples

```
data(GenIns)
matplot(t(GenIns), type="l")
```

`Latest` *Triangles*

Description

Get the latest development position for each YOA of a run off triangle.

Usage

```
Latest(triangle)
```

Arguments

`triangle`	a `data.frame` of a run off triangle

Value

`Latest` returns a vector of the latest development position.

Author(s)

Markus Gesmann, Markus.Gesmann@web.de

See Also

See Also `ChainLadder`

Examples

```
data(RAA)
RAA
Latest(RAA)
```

`Long.Triangle` *Triangles*

Description

`Long.Triangle` converts a table of three colums in a crosstab triangle, using `reshape`.

Usage

```
Long.Triangle(object)
```

Arguments

`object`	a `data.frame` of three columns, the first column will be interpreted as underwriting years, the second one as development period and the third one as a underwriting statistics like premium, paid, incurred, etc.

Value

`Long.Triangle` gives a `data.frame` back, with the underwriting year information in the first column and the underwriting statistic thereafter.

Author(s)

Markus Gesmann, Markus.Gesmann@web.de

See Also

See Also: `Triangle.Long`

Examples

```
data(RAA)
L <- Triangle.Long(RAA)
L
# convert back to triangle
Long.Triangle(L)
```

`Mortgage` *Development triangle (accumulated figures) of mortgage guarantee business*

Description

Run off triangle of claims data.

Usage

```
data(Mortgage)
```

Format

A data frame with 9 underwriting years and 9 development years.

Source

Competition Presented at a London Market Actuaries Dinner, D.E.A. Sanders, 1990

References

See also table 4 in: Distribution-free Calculation of the Standard Error of Chain Ladder Reserve Estimates, Thomas Mack, 1993, *ASTIN Bulletin* **23**, 213 - 225

Examples

```
data(Mortgage)
Mortgage
matplot(t(Mortgage), type="l")
```

`MunichChainLadder` *Munich Chain Ladder*

Description

A reserving method that reduces the gap between IBNR projections based on paid losses and IBNR projections based on incurred losses

Usage

```
MunichChainLadder(paid, incurred, quarterly = FALSE, tailP = 1,
                  tailI = 1, usetail = FALSE, sigmaP=NULL, sigmaI=NULL,
                  YOA = c(1:(nrow(paid))))
```

Arguments

`paid`	An accumulative paid triangle of quarterly or yearly development data stored in a `data.frame` of at least two underwriting years. The first column of this data frame may include the underwriting years, if its name is an element of the list: "UWY", "YOA", "YoA"
`incurred`	An accumulative incurred triangle of quarterly or yearly development data stored in a `data.frame` of at least two underwriting years.
`quarterly`	logical. `TRUE` if the triangle contains quarterly development data. `FALSE` if the triangle contains yearly development data
`tailP`	An optinal tail factor for the paid data. The default is no tail factor, hence `tail=1`
`tailI`	An optinal tail factor for the incurred data. The default is no tail factor, hence `tail=1`
`usetail`	logical. If `TRUE`, the setting of `tailP` and `tailI` will be ignored and the function `tailfactor` will be called to estimate tail factors by loglinear regression through the chain ladder ratios. If `FALSE` the value of `tailI` and `tailP` will be used as tail factors.

`YOA`	An optional vector of underwriting years. If the first column of `object` is named "UWY" or "YOA" or "YoA", this column will be used.
`sigmaP`	optional value for the latest sigma paid figures.
`sigmaI`	optional value for the latest sigma incurred figures.

Details

The Munich chain ladder (MCL) method combines the paid-loss (P) and incurred-loss (I) data types by taking Q=(P/I) ratios into account in projections.

Value

`MunichChainLadder` returns an object of `class` "MunichChainLadder" and "data.frame". An object of class "MunicgChainLadder" is a list containing the following components:

`Call`	Function call
`YOA`	Underwriting years
`Paid`	Input paid triangle
`Incurred`	Input incurred triangle
`Quarterly`	logical
`LatestP`	Latest paid position
`RatiosP`	Standard chain ladder ratios for the paid triangle
`FactorsP`	Standard chain ladder factors for the paid triangle
`UltimateP`	Ultimate based on standard chain ladder of the paid triangle
`TailP`	Tail factor used by standard chain ladder on the paid triangle
`FullTriangleP`	Full paid triangle based on standard chain ladder
`sigmaP`	Estimation of the propotional constant of the variance for the paid development triangle
`MackSEP`	Estimation of the standard error of the reserve and ultimate by underwriting year based on the paid triangle
`LatestI`	Latest incurred position
`RatiosI`	Standard chain ladder ratios for the incurred triangle
`FactorsI`	Standard chain ladder factors for the incurred triangle
`UltimateI`	Ultimate based on standard chain ladder of the incurred triangle
`TailI`	Tail factor used by standard chain ladder on the incurred triangle
`FullTriangleI`	Full incurred triangle based on standard chain ladder
`sigmaI`	Estimation of the propotional constant of the variance for the incurred development triangle
`MackSEI`	Estimation of the standard error of the reserve and ultimate by underwriting year based on the paid triangle
`RatiosQ`	Standard chain ladder ratios of Q=P/I
`rhoP`	sigma of P/I
`rhoI`	sigma of I/P

resP	Residuals of the paid development factors
resI	Residuals of the incurred development factors
resQ	Residuals of the paid/incurred development factors
resQinv	Residuals of the incurred/paid development factors
lambdaP	Paid correlation parameter
lambdaI	Incurred correlation parameter
MCLPaid	Ultimate based on Munich chain ladder of the paid triangle
MCLIncurred	Ultimate based on Munich chain ladder of the incurred triangle

Author(s)

Markus Gesmann, Markus.Gesmann@web.de

References

Munich Chain Ladder, Dr. Gerhard Quarg and Dr. Thomas Mack Corporate Actuarial Functions Munich Reinsurance Company

See Also

See Also ChainLadder

Examples

```
# data from Quarg's paper
data(MCLpaid)
MCLpaid
data(MCLincurred)
MCLincurred
M=MunichChainLadder(MCLpaid, MCLincurred, quarterly=FALSE)
M

# change sigmaP and sigmaI manually as in Quarg's paper
MCL=MunichChainLadder(MCLpaid, MCLincurred, quarterly=FALSE, sigmaP=0.1,sigmaI=0.1)
MCL
plot(MCL)
plot(MCL, plotResiduals=TRUE)
# get a second ultimate opinion
U <- MCL$UltimateI*(1+rnorm(7)/10)
plot(MCL, sndUlt=U)
```

RAA — *Run-off triangle of Automatic Factultative business in General Liability*

Description

Run off triangle of accumulated claims data.

Usage

```
data(RAA)
```

Format

A data frame with 10 underwriting years and 10 development years.

Source

Historical Loss Development, *Reinsurance Association of Ammerica (RAA)*, **1991**, p.96

References

See Also: Which Stochastik Model is Underlying the Chain Ladder Method?, Thomas Mack, 1994, *Insurance Mathematics and Economics*, **15, 2/3**, 133-138

P.D.England and R.J.Verrall, *Stochastic Claims Reserving in General Insurance*, British Actuarial Journal, **Vol. 8**, pp443-544, 2002

Examples

```
data(RAA)
RAA
matplot(t(RAA), type="l")
```

`Triangle.Long` *Triangles*

Description

`Triangle.Long` converts a cross table, e.g. a development triangle into a table structure, using `reshape`.

Usage

```
Triangle.Long(triangle, YOA = c(1:nrow(triangle)), na.rm=TRUE)
```

Arguments

`triangle`	a `data.frame`.
`YOA`	an optional `vector` of years of accounts
`na.rm`	na.rm: logical. Should missing values be removed?

Details

`Triangle.Long` is the inverse function of `Long.Triangle`.

Value

`Triangle.Long` returns an object of `class` "data.frame".

Author(s)

Markus Gesmann, Markus.Gesmann@web.de

See Also

See Also: Long.Triangle

Examples

```
data(RAA)
L <- Triangle.Long(RAA)
L
# convert back to triangle
Long.Triangle(L)
```

---

cumulativeTriangle *Triangles*

Description

`cumulativeTriangle` gives the running sum by columns back, e.g. `cumulativeTriangle` converts an incremental run off triangle into an accumulative run off triangle.

Usage

```
cumulativeTriangle(triangle)
```

Arguments

`triangle`	a `data.frame`, e.g. a triangle of incremental development underwriting data.

Author(s)

Markus Gesmann, Markus.Gesmann@web.de

See Also

See Also incrementalTriangle

Examples

```
data(RAA)
RAA
incRAA <- incrementalTriangle(RAA)
incRAA
cumulativeTriangle(incRAA)
```

`incrementalTriangle` *Triangles*

Description

`incrementalTriangle` gives the running difference by columns back, e.g. `incrementalTriangle` converts an accumulative triangle into an incremental triangle.

Usage

```
incrementalTriangle(triangle)
```

Arguments

`triangle`	a `data.frame`, e.g. a triangle of accumulative development underwriting data.

Author(s)

Markus Gesmann, Markus.Gesmann@web.de

See Also

See Also `cumulativeTriangle`

Examples

```
data(RAA)
RAA
incrementalTriangle(RAA)
```

`plot.BootReserve` *Plot Diagnostics for an BootReserve Object*

Description

Plotting method for objects inheriting from class "BootReserve".

Usage

```
plot.BootReserve(x, byYOA=FALSE, main="Bootstrap Reserve",...)
```

Arguments

`x`	`x` result of `ChainLadder`
`byYOA`	logical. If `FALSE` then `plot.BootReserve` plots the distribution of the cumulative reserve over all underwriting years plus a normal Q-Q plot. For `byYOA=TRUE` `plot.BootReserve` will use `histogram` of the `lattice` package to plot the bootstrap reserve for each underwriting year
`main`	Title for the plot
`...`	Further arguments to `plot` or `histogram`

Author(s)

Markus Gesmann, Markus.Gesmann@web.de

See Also

See Also `plot.default`, `histogram`, codeBootReserve

Examples

```
data(RAA)
RAA
R<-BootReserve(RAA, quarterly=FALSE)
plot(R)
plot(R, byYOA=TRUE)
```

`plot.ChainLadder` *Plot Diagnostics for a ChainLadder object*

Description

These are methods for objects of class "ChainLadder".

Usage

```
plot.ChainLadder(x, byYOA = FALSE, YOA = NULL, sndUlt = NULL,...)
```

Arguments

`x`	x result of `ChainLadder`
`byYOA`	logical. If `FALSE` and `YOA=NULL` then `plot.ChainLadder` produces a barplot of the latest amount plus reserve and plus/minus Mack's standard error. For `byYOA=TRUE` `plot.ChainLadder` will use `xyplot` of the `lattice` package to plot the chain ladder development of all underwriting years
`YOA`	if `YOA` is in the list of `object$YOA` and `byYOA=FALSE` `plot.ChainLadder` plots the chain ladder development of the given underwriting year
`sndUlt`	optional, a vector of a second opinion of ultimate, to plot against chain ladder ultimate
`...`	Further arguments to `plot` or `xyplot`

Author(s)

Markus Gesmann, Markus.Gesmann@web.de

See Also

See Also `plot.default`, `xyplot` and `ChainLadder`

Examples

```
data(RAA)
R<-ChainLadder(RAA, quarterly=FALSE, YOA=1995:2004)
plot(R)
plot(R, YOA=2000, main="Paid +/- MackSE")
plot(R, byYOA=TRUE)
# get a second ultimate opinion
U <- R$Ultimate*(1+rnorm(10)/10)
plot(R, byYOA=TRUE, sndUlt=U)
```

---

plot.MunichChainLadder

Plot Diagnostics for a MunichChainLadder object

Description

These are methods for objects of class "MunichChainLadder".

Usage

```
plot.MunichChainLadder(x, plotResiduals = FALSE, sndUlt = NULL, legend =
                  TRUE, main = "Munich Chain Ladder Results", xlab =
                  "YOA", col = c("lightgreen", "lightblue", "red"), ...)
```

Arguments

`x`	x result of `MunichChainLadder`
`plotResiduals`	logical. If `TRUE` then `plot.MunichChainLadder` will plot the paid and incurred development residuals together with regression lines. If `plotResiduals=FALSE` then `plot.MunichChainLadder` produces a barplot with two bars for each underwriting year representing the ultimate based on the paid and incurred data, respectively, using standard chain ladder plus/minus Mack's standard error. The Munich chain ladder ultimate will be marked with P and I respectively.
`sndUlt`	optional, a vector of a second opinion of ultimate, to plot against Munich chain ladder ultimate (only if `plotResiduals=FALSE`)
`legend`	logical, wether to draw a legend or not
`main`	title for the plot
`xlab`	lable for x axes
`col`	colours to be used
`...`	further arguments to `plot` or `barplot`

Author(s)

Markus Gesmann, Markus.Gesmann@web.de

References

Munich Chain Ladder, Dr. Gerhard Quarg and Dr. Thomas Mack Corporate Actuarial Functions Munich Reinsurance Company

See Also

See Also MunichChainLadder, plot.ChainLadder, barplot, par

Examples

```
# data from Quarg's paper
data(MCLpaid)
MCLpaid
data(MCLincurred)
MCLincurred
M=MunichChainLadder(MCLpaid, MCLincurred, quarterly=FALSE)
M

# change sigmaP and sigmaI manually as in Quarg's paper
MCL=MunichChainLadder(MCLpaid, MCLincurred, quarterly=FALSE,
sigmaP=0.1,sigmaI=0.1, YOA=1998:2004)
MCL
plot(MCL)
plot(MCL, plotResiduals=TRUE)
# get a second ultimate opinion
U <- MCL$UltimateI*(1+rnorm(7)/10)
plot(MCL, sndUlt=U)
```

`plotCLRatios` *Compare chain ladder ratios with a linear regression*

Description

Chain ladder ratios are compared with the result of a linear regression

Usage

```
plotCLRatios(triangle, quarterly = FALSE, dev = NULL, inc.vs.cum =
                  FALSE, xlab = "C[i,k]", ylab = ifelse(inc.vs.cum,
                  "C[i,k+1]-C[i,k]", "C[i,k+1]"), main =
                  ifelse(inc.vs.cum, "Chain Ladder Ratio Analysis\ninc. vs. cum."
col = c("red", "blue", "grey"), ...)
```

Arguments

`triangle`	Describe `triangle` here
`quarterly`	Describe `quarterly` here
`dev`	Describe `dev` here
`inc.vs.cum`	Describe `inc.vs.cum` here
`xlab`	Describe `xlab` here
`ylab`	Describe `ylab` here
`main`	Describe `main` here
`col`	Describe `col` here
`...`	Describe `...` here

Details

Value

Describe the value returned If it is a LIST, use

`comp1`	Description of 'comp1'
`comp2`	Description of 'comp2'

...

Author(s)

Markus.Gesmann@web.de

References

put references to the literature/web site here

See Also

See Also as `ChainLadder`

Examples

```
data(RAA)
R<-ChainLadder(RAA, quarterly=FALSE, YOA=1997:2006)
R
# plot as ChainLadder object, cumulatvie vs cumulative
plotCLRatios(R)
# plot as original data.frame object, incremental vs cumulative
plotCLRatios(R, inc.vs.cum=TRUE)
# concentrate on the first two dev. periods
plotCLRatios(R, inc.vs.cum=FALSE, dev=1:2)
```

`plotCLResiduals` *Plot Diagnostics for a ChainLadder object*

Description

Plot the residuals of the chain ladder ratios

Usage

```
plotCLResiduals(object, dev = NULL, col = "red", ...)
```

Arguments

`object`	Result of `ChainLadder`
`dev`	if `dev` not NULL, then plot only the specified development period
`col`	color
`...`	further arguments for `plot`

Details

plotCLResiduals plots the residuals of the chain ladder ratios.

Author(s)

Markus.Gesmann@web.de

See Also

See Also `ChainLadder`

Examples

```
data(RAA)
R <- ChainLadder(RAA, YOA=1997:2006)
plotCLResiduals(R)
```

`tailfactor` *Tail Factor for Chain Ladder Ratios*

Description

`tailfactor` uses a vector of chain ladder ratios to calculate an estimated tail factor via loglinear regression.

Usage

```
tailfactor(clratios)
```

Arguments

`clratios`	vector of chain ladder ratios.

Details

Assume $f_1, \ldots, f_{n-1} > 1$ are chain ladder ratios then a possible way to arrive for the tail factor is a linear extrapolation of $ln(f_k - 1)$ by a straight line $a * k + b, a < 0$, together with

$$f_{tail} = \prod_{k=n}^{\infty} f_k.$$

`taifactor` does this with $\infty = 100$.

Value

`tailfactor` gives an estimated tail factor back. It gives 1 back if the log linear regression estimates a tail factor > 2.

Note

`tailfactor` is called if the logical option `usetail` in `ChainLadder` or `BootReserve`, is set to `TRUE`

Author(s)

Markus Gesmann, ⟨Markus.Gesmann@web.de⟩

References

The Standard Error of Chain Ladder Reserve Estimates: Recursive Calculation and Inclusion of a Tail Factor, Thomas Mack, 1999, *Astin Bulletin* **Vol. 29, No.2**, 361 - 266

See Also

See Also `BootReserve`, `ChainLadder`

Examples

```
data(RAA)
# compare
ChainLadder(RAA, quarterly=FALSE, usetail=FALSE)
ChainLadder(RAA, quarterly=FALSE, usetail=TRUE)
```

Index