Package ‘ChainLadder’

November 24, 2009

Type Package

Title Mack-, Bootstrap and Munich-chain-ladder methods for insurance claims reserving

Version 0.1.2-13

Date 2009-10-18

Author Markus Gesmann

Maintainer Markus Gesmann <markus.gesmann@gmail.com>

Description The package provides Mack-, Munich-, and Bootstrap chain-ladder methods. These methods are typically used in insurance claims reserving.

Depends Hmisc, lattice, stats

License GPL (>= 2)

URL http://code.google.com/p/chainladder/

LazyData yes

Repository CRAN

Date/Publication 2009-11-24 10:11:18

R topics documented:

`ChainLadder-package` *Various chain-ladder methods for claims reserving*

Description

The ChainLadder-package grew out of presentations the author gave at the Stochastic Reserving Seminar at the Institute of Actuaries in November 2007 and 2008. This package has currently implementations for the Mack-, Munich- and Bootstrap-chain-ladder methods. The package offers also some utility functions to convert quickly tables into triangles, triangles into tables, cumulative into incremental and incremental into cumulative triangles.

The ChainLadder-package comes with an example spreadsheet which demonstrates how to use the `ChainLadder` functions in Excel. The spreadsheet is located in the Excel folder of the package. The R command `searchpaths()[grep('ChainLadder', searchpaths())]` will tell you the exact path to the directory. To use the spreadsheet you will need to have the RExcel-Addin, see `http://sunsite.univie.ac.at/rcom/` for more details.

More information is available on the project web site `http://code.google.com/p/chainladder/`

If you are also interested in loss distributions modeling, risk theory (including ruin theory), simulation of compound hierarchical models and credibility theory check out the `actuar` package by C. Dutang, V. Goulet and M. Pigeon.

For more financial packages see also CRAN Task View 'Emperical Finance' at `http://cran.r-project.org/web/views/Finance.html`.

Details

Package:	ChainLadder
Type:	Package
Version:	0.1.2-13
Date:	2008-10-12
License:	GPL version 2 or later

Author(s)

Markus Gesmann

Maintainer: Markus Gesmann <markus.gesmann@gmail.com>

References

Thomas Mack. Distribution-free calculation of the standard error of chain ladder reserve estimates. Astin Bulletin. *Vol. 23. No 2. 1993. pp.213:225*

Thomas Mack. The standard error of chain ladder reserve estimates: Recursive calculation and inclusion of a tail factor. Astin Bulletin. *Vol. 29. No 2. 1999. pp.361:366*

Gerhard Quarg and Thomas Mack. Munich Chain Ladder. Blatter DGVFM 26. *Munich. 2004.*

England, PD and Verrall, RJ. Stochastic Claims Reserving in General Insurance (with discussion). British Actuarial Journal 8. *III. 2002*

B. Zehnwirth and G. Barnett. Best Estimates for Reserves. Proceedings of the CAS. *Volume LXXXVII. Number 167.November 2000.*

Examples

```
# Example triangle
RAA
plot(RAA)
plot(RAA, lattice=TRUE)
MCL=MackChainLadder(RAA, est.sigma="Mack")
MCL
plot(MCL)
# plot developments by origin period
plot(MCL, lattice=TRUE)

# BootChainLadder
B <- BootChainLadder(RAA, R=999, process.distr="gamma")
B
plot(B)
# fit a log-normal distribution
library(MASS)
# fit a log-normal distribution
fit <-  fitdistr(B$IBNR.Totals, "lognormal")
fit
plot(ecdf(B$IBNR.Totals))
curve(plnorm(x,fit$estimate["meanlog"], fit$estimate["sdlog"]), col="red", add=TRUE)

# Munich Chain Ladder
MCLpaid
MCLincurred

MCL = MunichChainLadder(MCLpaid, MCLincurred)
MCL
```

```
plot(MCL)

# Working with triangles
# RAA is a matrix with additional class attribute 'triangle'
class(RAA)
# default plot for a triangle
plot(RAA)
# plot developments by origin period
plot(RAA, lattice=TRUE)

# Triangles can easily be converted into data.frames via
X=as.data.frame(RAA)
X
# Tables can also be converted into triangles
triangle <- as.triangle(X, origin="origin", dev="dev", value="value")
triangle


## More for a laugh - 3d plot of a triangle and MackChainLadder output
if(require(rgl)){ #provides interactive 3d plotting functions
MCL=MackChainLadder(GenIns/1e6, est.sigma="Mack")
FT <- MCL$FullTriangle
FTpSE <- FT+MCL$Mack.S.E
FTpSE[which(MCL$Mack.S.E==0, arr.ind=TRUE)] <- NA
FTmSE <- FT-MCL$Mack.S.E
FTmSE[which(MCL$Mack.S.E==0, arr.ind=TRUE)] <- NA
zr <- round(FT/FT[1,10]*100)
zlim <- range(zr, na.rm=TRUE)
zlen <- zlim[2] - zlim[1] + 1
colorlut <- terrain.colors(zlen) # height color lookup table
cols <- colorlut[ zr -zlim[1]+1 ] # assign colors to heights for each point
x <- as.numeric(dimnames(FT)$origin)
y <- as.numeric(dimnames(FT)$dev)
persp3d(x, y=y,
        z=(FT), col=cols, xlab="origin", ylab="dev", zlab="loss",back="lines")
mSE <- data.frame(as.table(FTmSE))
points3d(xyz.coords(x=as.numeric(as.character(mSE$origin)),
    y=as.numeric(as.character(mSE$dev)),z=mSE$Freq), size=2)
pSE <- data.frame(as.table(FTpSE))
points3d(xyz.coords(x=as.numeric(as.character(pSE$origin)),
    y=as.numeric(as.character(pSE$dev)),z=pSE$Freq), size=2)
}
```

ABC — *Run off triangle of accumulated claims data*

Description

Run-off triangle of a worker's compensation portfolio of a large company

Usage

```
data(ABC)
```

Format

A matrix with 11 accident years and 11 development years.

Source

B. Zehnwirth and G. Barnett. Best Estimates for Reserves. Proceedings of the CAS. *Volume LXXXVII. Number 167. November 2000.*

Examples

```
ABC
plot(ABC)
plot(ABC, lattice=TRUE)
```

BootChainLadder   *Bootstrap-Chain-Ladder Model*

Description

The `BootChainLadder` procedure provides a predictive distribution of reserves or IBNRs for a cumulative claims development triangle.

Usage

```
BootChainLadder(Triangle, R = 999, process.distr=c("gamma", "od.pois"))
```

Arguments

`Triangle` cumulative claims triangle. Assume columns are the development period, use transpose otherwise. A (mxn)-matrix C_{ik} which is filled for $k \leq n+1-i; i = 1, \ldots, m; m \geq n$. See `qpaid` for how to use (mxn)-development triangles with m<n, say higher development period frequency (e.g quarterly) than origin period frequency (e.g accident years).

`R` the number of bootstrap replicates.

`process.distr` character string indicating which process distribution to be assumed. One of "gamma" (default), or "od.pois" (over-dispersed Poisson), can be abbreviated

Details

The `BootChainLadder` function uses a two-stage bootstrapping/simulation approach. In the first stage an ordinary chain-ladder methods is applied to the cumulative claims triangle. From this we calculate the scaled Pearson residuals which we bootstrap R times to forecast future incremental claims payments via the standard chain-ladder method. In the second stage we simulate the process error with the bootstrap value as the mean and using the process distribution assumed. The set of reserves obtained in this way forms the predictive distribution, from which summary statistics such as mean, prediction error or quantiles can be derived.

Value

BootChainLadder gives a list with the following elements back:

- `call` matched call
- `Triangle` input triangle
- `f` chain-ladder factors
- `simClaims` array of dimension `c(m,n,R)` with the simulated claims
- `IBNR.ByOrigin` array of dimension `c(m,1,R)` with the modeled IBNRs by origin period
- `IBNR.Triangles` array of dimension `c(m,n,R)` with the modeled IBNR development triangles
- `IBNR.Totals` vector of R samples of the total IBNRs
- `ChainLadder.Residuals` adjusted Pearson chain-ladder residuals
- `process.distr` assumed process distribution
- `R` the number of bootstrap replicates

Note

The implementation of `BootChainLadder` follows closely the discussion of the bootstrap model in section 8 and appendix 3 of the paper by England and Verrall (2002).

Author(s)

Markus Gesmann, `<markus.gesmann@gmail.com>`

References

England, PD and Verrall, RJ. Stochastic Claims Reserving in General Insurance (with discussion), British Actuarial Journal *8, III. 2002*

Barnett and Zehnwirth. The need for diagnostic assessment of bootstrap predictive models, Insureware technical report. *2007*

See Also

See also `summary.BootChainLadder`, `plot.BootChainLadder`

Examples

```
# See also the example in section 8 of England & Verrall (2002) on page 55.

B <- BootChainLadder(RAA, R=999, process.distr="gamma")
B
plot(B)
# Compare to MackChainLadder
MackChainLadder(RAA)
quantile(B, c(0.75,0.95,0.99, 0.995))

# fit a distribution to the IBNR
library(MASS)
plot(ecdf(B$IBNR.Totals))
# fit a log-normal distribution
fit <- fitdistr(B$IBNR.Totals[B$IBNR.Totals>0], "lognormal")
fit
curve(plnorm(x,fit$estimate["meanlog"], fit$estimate["sdlog"]), col="red", add=TRUE)

# See also the ABC example in  Barnett and Zehnwirth (2007)
A <- BootChainLadder(ABC, R=999, process.distr="gamma")
A
plot(A, log=TRUE)
```

---

chainladder *Estimate age-to-age factors*

Description

Basic chain ladder function to estimate age-to-age factors for a given cumulative run-off triangle. This function is used by Mack- and MunichChainLadder.

Usage

```
chainladder(Triangle, weights = 1, delta = 1)
```

Arguments

`Triangle` cumulative claims triangle. A (mxn)-matrix C_{ik} which is filled for $k \leq n+1-i; i = 1, \ldots, m; m \geq n$, see `qpaid` for how to use (mxn)-development triangles with m<n, say higher development period frequency (e.g quarterly) than origin period frequency (e.g accident years).

`weights` weights. Default: 1, which sets the weights for all triangle entries to 1. Otherwise specify weights as a matrix of the same dimension as `Triangle` with all weight entries in [0; 1] `delta` 'weighting' parameters, either 0,1 or 2. Default: 1; delta=1 gives the historical chain ladder age-to-age factors, delta=0 gives the straight average of the observed individual development factors and delta=2 is the result of an ordinary regression of $C_{i,k+1}$ against $C_{i,k}$ with intercept 0, see Barnett & Zehnwirth (2000);. Please note that Mack (1999) used the notation of alphas, with alpha=2-delta.

Details

The key idea is to see the chain ladder algorithm as a weighted linear regression through the origin applied for each development period.

Suppose `y` is the vector of cumulative claims at development period `i+1`, and `x` at development period `i`, `w` are weighting factors and `F` the individual age-to-age factors `F=y/x`, than we get the various age-to-age factors for different deltas (alphas) as:

```
sum(w*x^alpha*F)/sum(w*x^alpha) # Mack (1999) notation

delta <- 2-alpha

lm(y~x + 0 ,weights=w/x^delta) # Barnett & Zehnwirth (2000) notation
```

Value

chainladder returns a list with the following elements:

`Models`	linear regression models for each development period
`Triangle`	input triangle of cumulative claims
`weights`	weights used
`delta`	deltas used

Author(s)

Markus Gesmann <markus.gesmann@gmail.com>

References

Thomas Mack. The standard error of chain ladder reserve estimates: Recursive calculation and inclusion of a tail factor. Astin Bulletin. *Vol. 29. No 2. 1999. pp.361:366*

G. Barnett and B. Zehnwirth. Best Estimates for Reserves. Proceedings of the CAS. *Volume LXXXVII. Number 167. November 2000.*

See Also

See also `predict.ChainLadder` `MackChainLadder`,

Examples

```
## Concept of different chain ladder age-to-age factors.
## Compare Mack's and Barnett & Zehnwirth's papers.
x <- RAA[1:9,1]
y <- RAA[1:9,2]

weights <- RAA
weights[!is.na(weights)] <- 1
w <- weights[1:9,1]

F <- y/x
## wtd. average chain ladder age-to-age factors
alpha <- 1
delta <- 2-alpha

sum(w*x^alpha*F)/sum(w*x^alpha)
lm(y~x + 0 ,weights=w/x^delta)
summary(chainladder(RAA, weights=weights, delta=delta)$Models[[1]])$coef

## straight average age-to-age factors
alpha <- 0
delta <- 2 - alpha
sum(w*x^alpha*F)/sum(w*x^alpha)
lm(y~x + 0 ,weights=w/x^(2-alpha))
summary(chainladder(RAA, weights=weights, delta=delta)$Models[[1]])$coef

## ordinary regression age-to-age factors
alpha=2
delta <- 2-alpha
sum(w*x^alpha*F)/sum(w*x^alpha)
lm(y~x + 0 ,weights=w/x^delta)
summary(chainladder(RAA, weights=weights, delta=delta)$Models[[1]])$coef

## Change weights

weights[2,1] <- 0.5
w <- weights[1:9,1]

## wtd. average chain ladder age-to-age factors
alpha <- 1
delta <- 2-alpha
sum(w*x^alpha*F)/sum(w*x^alpha)
lm(y~x + 0 ,weights=w/x^delta)
summary(chainladder(RAA, weights=weights, delta=delta)$Models[[1]])$coef

## straight average age-to-age factors
alpha <- 0
delta <- 2 - alpha
sum(w*x^alpha*F)/sum(w*x^alpha)
lm(y~x + 0 ,weights=w/x^(2-alpha))
summary(chainladder(RAA, weights=weights, delta=delta)$Models[[1]])$coef
```

```
## ordinary regression age-to-age factors
alpha=2
delta <- 2-alpha
sum(w*x^alpha*F)/sum(w*x^alpha)
lm(y~x + 0 ,weights=w/x^delta)
summary(chainladder(RAA, weights=weights, delta=delta)$Models[[1]])$coef

## Model review
CL0 <- chainladder(RAA, weights=weights, delta=0)
## age-to-age factors
sapply(CL0$Models, function(x) summary(x)$coef["x","Estimate"])
## f.se
sapply(CL0$Models, function(x) summary(x)$coef["x","Std. Error"])
## sigma
sapply(CL0$Models, function(x) summary(x)$sigma)

CL1 <- chainladder(RAA, weights=weights, delta=1)
## age-to-age factors
sapply(CL1$Models, function(x) summary(x)$coef["x","Estimate"])
## f.se
sapply(CL1$Models, function(x) summary(x)$coef["x","Std. Error"])
## sigma
sapply(CL1$Models, function(x) summary(x)$sigma)

CL2 <- chainladder(RAA, weights=weights, delta=2)
## age-to-age factors
sapply(CL2$Models, function(x) summary(x)$coef["x","Estimate"])
## f.se
sapply(CL2$Models, function(x) summary(x)$coef["x","Std. Error"])
## sigma
sapply(CL2$Models, function(x) summary(x)$sigma)


## Forecasting

predict(CL0)
predict(CL1)
predict(CL2)
```

Cumulative and incremental triangles

Cumulative and incremental triangles

Description

Functions to convert between cumulative and incremental triangles

Usage

```
incr2cum(Triangle, na.rm=FALSE)
cum2incr(Triangle)
```

Arguments

`Triangle`	triangle. Assume columns are the development period, use transpose otherwise.
`na.rm`	logical. Should missing values be removed?

Details

`incr2cum` transforms an incremental triangle into a cumulative triangle, `cum2incr` provides the reserve operation.

Value

Both functions return a `triangle`.

Author(s)

Markus Gesmann, Christophe Dutang

See Also

See also `as.triangle`

Examples

```
# See the Taylor/Ashe example in Mack's 1993 paper

#original triangle
GenIns

#incremental triangle
cum2incr(GenIns)

#original triangle
incr2cum(cum2incr(GenIns))

# See the example in Mack's 1999 paper

#original triangle
Mortgage
incMortgage <- cum2incr(Mortgage)
#add missing values
incMortgage[1,1] <- NA
incMortgage[2,1] <- NA
incMortgage[1,2] <- NA

#with missing values argument
```

```
incr2cum(incMortgage, na.rm=TRUE)

#compared to
incr2cum(Mortgage)
```

GenIns *Run off triangle of claims data.*

Description

Run off triangle of accumulated general insurance claims data. `GenInsLong` provides the same data in a 'long' format.

Usage

```
GenIns
```

Format

A matrix with 10 accident years and 10 development years.

Source

TAYLOR, G.C. and ASHE, F.R. (1983) Second Moments of Estimates of Outstanding Claims. Journal of Econometrics ***23**, 37-61.*

References

See table 1 in: *Distribution-free Calculation of the Standard Error of Chain Ladder Reserve Estimates, Thomas Mack, 1993*, ASTIN Bulletin ***23**, 213 - 225*

Examples

```
GenIns
plot(GenIns)

plot(GenIns, lattice=TRUE)


head(GenInsLong)

## Convert long format into triangle
## Triangles are usually stored as 'long' tables in data bases
as.triangle(GenInsLong, origin="accyear", dev="devyear", "incurred claims")
```

M3IR5 — *Run off triangle of claims data*

Description

Run off triangle of simulated incremental claims data

Usage

```
data(M3IR5)
```

Format

A matrix with simulated incremental claims of 14 accident years and 14 development years.

Source

G. Barnett and B. Zehnwirth. Best Estimates for Reserves. Proceedings of the CAS. *Volume LXXXVII. Number 167. November 2000.*

Examples

```
M3IR5
plot(M3IR5)
plot(incr2cum(M3IR5), lattice=TRUE)
```

MackChainLadder — *Mack-Chain-Ladder Model*

Description

The Mack-chain-ladder model forecasts future claims developments based on a historical cumulative claims development triangle and estimates the standard error around those.

Usage

```
MackChainLadder(Triangle, weights = 1, alpha=1, est.sigma="log-linear",
tail=FALSE, tail.se=NULL, tail.sigma=NULL)
```

Arguments

`Triangle` cumulative claims triangle. Assume columns are the development period, use transpose otherwise. A (mxn)-matrix C_{ik} which is filled for $k \leq n+1-i; i = 1, \ldots, m; m \geq n$, see `qpaid` for how to use (mxn)-development triangles with m<n, say higher development period frequency (e.g quarterly) than origin period frequency (e.g accident years).

`weights` weights. Default: 1, which sets the weights for all triangle entries to 1. Otherwise specify weights as a matrix of the same dimension as `Triangle` with all weight entries in [0; 1]

`alpha` 'weighting' parameter. Default: 1 for all development periods; alpha=1 gives the historical chain ladder age-to-age factors, alpha=0 gives the straight average of the observed individual development factors and alpha=2 is the result of an ordinary regression of $C_{i,k+1}$ against $C_{i,k}$ with intercept 0, see also Mack's 1999 paper and `chainladder`

`est.sigma` defines how to estimate $sigma_{n-1}$, the variability of the individual age-to-age factors at development time $n-1$. Default is "log-linear" for a log-linear regression, "Mack" for Mack's approximation from his 1999 paper. Alternatively the user can provide a numeric value. If the log-linear model appears to be inappropriate (p-value > 0.05) the 'Mack' method will be used instead and a warning message printed.

`tail` can be logical or a numeric value. If `tail=FALSE` no tail factor will be applied, if `tail=TRUE` a tail factor will be estimated via a linear extrapolation of $log(chainladderfactors - 1)$, if `tail` is a numeric value than this value will be used instead.

`tail.se` defines how the standard error of the tail factor is estimated. Only needed if a tail factor > 1 is provided. Default is `NULL`. If `tail.se` is `NULL`, `tail.se` is estimated via "log-linear" regression, if `tail.se` is a numeric value than this value will be used instead.

`tail.sigma` defines how to estimate individual tail variability. Only needed if a tail factor > 1 is provided. Default is `NULL`. If `tail.sigma` is `NULL`, `tail.sigma` is estimated via "log-linear" regression, if `tail.sigma` is a numeric value than this value will be used instead

Details

Following Mack's 1999 paper let C_{ik} denote the cumulative loss amounts of origin period (e.g. accident year) $i = 1, \ldots, m$, with losses known for development period (e.g. development year) $k \leq n+1-i$. In order to forecast the amounts C_{ik} for $k > n+1-i$ the Mack chain-ladder-model assumes:

$$\text{CL1: } E[F_{ik}|C_{i1}, C_{i2}, \ldots, C_{ik}] = f_k \text{ with } F_{ik} = \frac{C_{i,k+1}}{C_{ik}}$$

$$\text{CL2: } Var(\frac{C_{i,k+1}}{C_{ik}}|C_{i1}, C_{i2}, \ldots, C_{ik}) = \frac{\sigma_k^2}{w_{ik} C_{ik}^{\alpha}}$$

$$\text{CL3: } \{C_{i1}, \ldots, C_{in}\}, \{C_{j1}, \ldots, C_{jn}\}, \text{ are independent for origin period } i \neq j$$

with $w_{ik} \in [0;1]$, $\alpha \in \{0,1,2\}$. If these assumptions are hold, the Mack-chain-ladder-model gives an unbiased estimator for IBNR (Incurred But Not Reported) claims.

The Mack-chain-ladder model can be regarded as a weighted linear regression through the origin for each development period: `lm(y ~ x + 0, weights=w/x^(2-alpha))`, where `y` is the vector of claims at development period $k+1$ and `x` is the vector of claims at development period k.

Value

MackChainLadder returns a list with the following elements

- `call` matched call
- `Triangle` input triangle of cumulative claims
- `FullTriangle` forecasted full triangle
- `Models` linear regression models for each development period
- `f` chain-ladder age-to-age factors
- `f.se` standard errors of the chain-ladder age-to-age factors f (assumption CL1)
- `F.se` standard errors of the true chain-ladder age-to-age factors F_{ik} (square root of the variance in assumption CL2)
- `sigma` sigma parameter in CL2
- `Mack.ProcessRisk` variability in the projection of future losses not explained by the variability of the link ratio estimators (unexplained variation)
- `Mack.ParameterRisk` variability in the projection of future losses explained by the variability of the link-ratio estimators alone (explained variation)
- `Mack.S.E` total variability in the projection of future losses by the chain ladder method; the square root of the mean square error of the chain ladder estimate: Mack.S.E.$^2 =$ Mack.ProcessRisk$^2 +$ Mack.ParameterRisk^2
- `Total.Mack.S.E` total variability of projected loss for all origin years combined
- `weights` weights used.
- `alpha` alphas used.
- `tail` tail factor used. If tail was set to TRUE the output will include the linear model used to estimate the tail factor

Note

Additional references for further reading:

England, PD and Verrall, RJ. Stochastic Claims Reserving in General Insurance (with discussion), British Actuarial Journal *8, III. 2002*

Murphy, Daniel M. Unbiased Loss Development Factors. Proceedings of the Casualty Actuarial Society Casualty Actuarial Society - *Arlington, Virginia 1994: LXXXI 154-222.*

Barnett and Zehnwirth. Best estimates for reserves. Proceedings of the CAS, LXXXVI I(167), November 2000.

Author(s)

Markus Gesmann <markus.gesmann@gmail.com>

References

Thomas Mack. Distribution-free calculation of the standard error of chain ladder reserve estimates. Astin Bulletin. *Vol. 23. No 2. 1993. pp.213:225*

Thomas Mack. The standard error of chain ladder reserve estimates: Recursive calculation and inclusion of a tail factor. Astin Bulletin. *Vol. 29. No 2. 1999. pp.361:366*

See Also

See also qpaid, chainladder, summary.MackChainLadder, plot.MackChainLadder, residuals.MackChainLadder, MunichChainLadder, BootChainLadder,

Examples

```
# See the Taylor/Ashe example in Mack's 1993 paper
GenIns
plot(GenIns)
plot(GenIns, lattice=TRUE)
GNI <- MackChainLadder(GenIns, est.sigma="Mack")
GNI$f
GNI$sigma^2
GNI # compare to table 2 and 3 in Mack's 1993 paper
plot(GNI)
plot(GNI, lattice=TRUE)

# Different weights
# Using alpha=0 will use straight average age-to-age factors
MackChainLadder(GenIns, alpha=0)$f
# You get the same result via:
apply(GenIns[,-1]/GenIns[,-10],2, mean, na.rm=TRUE)

# See the example in Mack's 1999 paper
Mortgage
plot(Mortgage)
MRT <- MackChainLadder(Mortgage, tail=1.05, tail.sigma=71, tail.se=0.02, est.sigma="Mack")
MRT
plot(MRT, lattice=TRUE)
# Table 1 in the above paper
f <- c(11.10, 4.092, 1.708, 1.276, 1.139, 1.069, 1.026, 1.023, 1.05)
f.se <- c(2.24, 0.517, 0.122, 0.051, 0.042, 0.023, 0.015, 0.012, 0.02)
F.se3 <- c(7.38, 1.89, 0.357, 0.116, 0.078, 0.033, 0.015, 0.007, 0.03)
sig <- c(1337, 988.5, 440.1, 207, 164.2, 74.60, 35.49, 16.89,71)
# test output from MackChainLadder
MRT$f
MRT$f.se
MRT$F.se[3,]
MRT$sigma
```

```
plot(MRT) # We observe trends along calendar years.

# Table 2 in the above paper
MRT$FullTriangle[,9]/1000 ## C_{i9}
MRT$FullTriangle[,10]/1000 ## C_{i,ult}
MRT$Mack.S.E[,9]/1000 ## s.e.(C_{i9})

# Access process risk error
MRT$Mack.ProcessRisk

# Access parameter risk error
MRT$Mack.ParameterRisk

# Total risk
MRT$Mack.S.E

op <- par(mfrow=c(2,1))
plot(with(summary(MRT)$ByOrigin, Mack.S.E/Ultimate),t="l",
ylab="CV(Ultimate)", xlab="origin period")
plot(summary(MRT)$ByOrigin[["CV(IBNR)"]], t="l", ylab="CV(IBNR)",
xlab="origin period")
par(op)


# This data set is discussed in many papers, e.g. England and Verrall (2000),
# see Table 1 just there
RAA
plot(RAA)
R <- MackChainLadder(RAA)
R
plot(R)
plot(R, lattice=TRUE)
# Table 12 in England and Verrall (2000)
R$f
R$sigma^2
# Table 13 in England and Verrall (2000)
# Please note the different indexing of sigma
MackChainLadder(RAA, est.sigma=R$sigma[7])
# Table 14 in England and Verrall (2000)
MackChainLadder(RAA, est.sigma=R$sigma[8])

# Let's investigate the Mack model in more detail
R[["Models"]][[1]]   # Model for first development period
summary( R[["Models"]][[1]]) # Look at the model stats
op <- par(mfrow=c(2,2)) # plot residuals
  plot( R[["Models"]][[1]])
par(op)

# Let's include an intercept in our model
newModel <- update(R[["Models"]][[1]], y ~ x+1,
             weights=1/R[["Triangle"]][1:9,1],
             data=data.frame(x=R[["Triangle"]][1:9,1],
```

```
                              y=R[["Triangle"]][1:9,2])
               )

# View the new model
summary(newModel)
op <- par(mfrow=c(2,2))
  plot( newModel )
par(op)

# Change the model for dev. period one to the newModel
R2 <- R
R2[["Models"]][[1]] <- newModel
predict(R2) # predict the full triangle with the new model
#(only the last origin year will be affected)

R2[["FullTriangle"]] <-  predict(R2)
R2[["FullTriangle"]]
R2   # Std. Errors have not been re-estimated!
# Plot the result

plot(R2, title="Changed R Model")

## Suppose you have a long table with claims development by line of
## business and would like to apply the MackChainLadder on all triangles
## in one go.

## First lets create a table similar to what you would get from a 'real' data base

myList <- list("General Liabilty" = RAA/1e3,
               "General Insurance" = GenIns/1e3,
               "Workers Comp"=ABC/1e3,
               "Mortgage Guarantee"=Mortgage/1e3)

myData <- do.call("rbind" , lapply(names(myList),
                            function(x) as.data.frame(myList[[x]],lob=x ,na.rm=TRUE)))

## Let's plot a nice summary, but first lets normalise the origin years
myData <- do.call("rbind",
    by(myData, list(lob=myData$lob),
    function(x) {org=as.numeric(as.character(x$origin))
      x$origin <- org-min(org)+2000;x}
    ))
rownames(myData) <- NULL

head(myData)  ## Does this look familiar? ;-)
xyplot(value ~ dev | lob, groups=factor(origin), data=myData, t="l",
      scales="free", auto.key=list(space="right", points=FALSE, lines=TRUE))

## Lets create triangles again and apply MackChainLadder for each lob:

myResults <- by(myData, list(lob=myData$lob), function(x)
                MackChainLadder(as.triangle(x), est.sigma="Mack"))
## That's it, lets look at the output
```

```
myResults

## Summarise all results by origin period in one data frame:
by.origin <- function(x){
              data.frame(lob=x,
              origin=dimnames(myResults[[x]]$Triangle)$origin,
              summary(myResults[[x]])$ByOrigin)
             }

ByOrigin <-do.call("rbind", lapply(names(myResults) , by.origin))
ByOrigin

## Similar for the totals
Totals <- do.call("rbind", lapply(names(myResults) ,
            function(x) data.frame(LOB=x, t(summary(myResults[[x]])$Totals))))
Totals

require(lattice)
barchart(Latest + IBNR ~ factor(origin) | lob, stack=TRUE, data=ByOrigin,
        scale="free", auto.key=TRUE, as.table=TRUE, xlab="origin")
```

MCLpaid *Run off triangles of accumulated paid and incurred claims data.*

Description

Run-off triangles based on a fire portfolio

Usage

```
data(MCLpaid)
data(MCLincurred)
```

Format

A matrix with 7 origin years and 7 development years.

Source

Gerhard Quarg and Thomas Mack. Munich Chain Ladder. Blatter DGVFM. *26, Munich, 2004.*

Examples

```
MCLpaid
MCLincurred
op=par(mfrow=c(2,1))
plot(MCLpaid)
plot(MCLincurred)
par(op)
```

Mortgage *Run off triangle of accumulated claims data*

Description

Development triangle of a mortgage guarantee business

Usage

```
data(Mortgage)
```

Format

A matrix with 9 accident years and 9 development years.

Source

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

References

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

Examples

```
Mortgage
Mortgage
plot(Mortgage)
plot(Mortgage, lattice=TRUE)
```

MunichChainLadder *Munich-chain-ladder Model*

Description

The Munich-chain-ladder model forecasts ultimate claims based on a cumulative paid and incurred claims triangle. The model assumes that the Mack-chain-ladder model is applicable to the paid and incurred claims triangle, see `MackChainLadder`.

Usage

```
MunichChainLadder(Paid, Incurred,
                  est.sigmaP = "log-linear", est.sigmaI = "log-linear",
                  tailP=FALSE, tailI=FALSE)
```

Arguments

- `Paid` — cumulative paid claims triangle. Assume columns are the development period, use transpose otherwise. A (mxn)-matrix P_{ik} which is filled for $k \leq n+1-i; i = 1, \ldots, m; m \geq n$
- `Incurred` — cumulative incurred claims triangle. Assume columns are the development period, use transpose otherwise. A (mxn)-matrix I_{ik} which is filled for $k \leq n+1-i; i = 1, \ldots, m, m \geq n$
- `est.sigmaP` — defines how $sigma_{n-1}$ for the Paid triangle is estimated, see `est.sigma` in `MackChainLadder` for more details, as `est.sigmaP` gets passed on to `MackChainLadder`
- `est.sigmaI` — defines how $sigma_{n-1}$ for the Incurred triangle is estimated, see `est.sigma` in `MackChainLadder` for more details, as `est.sigmaI` is passed on to `MackChainLadder`
- `tailP` — defines how the tail of the `Paid` triangle is estimated and is passed on to `MackChainLadder`, see `tail` just there.
- `tailI` — defines how the tail of the `Incurred` triangle is estimated and is passed on to `MackChainLadder`, see `tail` just there.

Value

MunichChainLadder returns a list with the following elements

- `call` — matched call
- `Paid` — input paid triangle
- `Incurred` — input incurred triangle
- `MCLPaid` — Munich-chain-ladder forecasted full triangle on paid data
- `MCLIncurred` — Munich-chain-ladder forecasted full triangle on incurred data
- `MackPaid` — Mack-chain-ladder output of the paid triangle
- `MackIncurred` — Mack-chain-ladder output of the incurred triangle
- `PaidResiduals` — paid residuals
- `IncurredResiduals` — incurred residuals
- `QResiduals` — paid/incurred residuals
- `QinverseResiduals` — incurred/paid residuals
- `lambdaP` — dependency coefficient between paid chain ladder age-to-age factors and incurred/paid age-to-age factors
- `lambdaI` — dependency coefficient between incurred chain ladder ratios and paid/incurred ratios
- `qinverse.f` — chain-ladder-link age-to-age factors of the incurred/paid triangle
- `rhoP.sigma` — estimated conditional deviation around the paid/incurred age-to-age factors
- `q.f` — chain-ladder age-to-age factors of the paid/incurred triangle
- `rhoI.sigma` — estimated conditional deviation around the incurred/paid age-to-age factors

Author(s)

Markus Gesmann <markus.gesmann@gmail.com>

References

Gerhard Quarg and Thomas Mack. Munich Chain Ladder. Blatter DGVFM *26, Munich, 2004.*

See Also

See also summary.MunichChainLadder, plot.MunichChainLadder , MackChainLadder

Examples

```
MCLpaid
MCLincurred
op <- par(mfrow=c(1,2))
plot(MCLpaid)
plot(MCLincurred)
par(op)

# Following the example in Quarg's (2004) paper:
MCL <- MunichChainLadder(MCLpaid, MCLincurred, est.sigmaP=0.1, est.sigmaI=0.1)
MCL
plot(MCL)
# You can access the standard chain ladder (Mack) output via
MCL$MackPaid
MCL$MackIncurred

# Input triangles section 3.3.1
MCL$Paid
MCL$Incurred
# Parameters from section 3.3.2
# Standard chain ladder age-to-age factors
MCL$MackPaid$f
MCL$MackIncurred$f
MCL$MackPaid$sigma
MCL$MackIncurred$sigma
# Check Mack's assumptions graphically
plot(MCL$MackPaid)
plot(MCL$MackIncurred)

MCL$q.f
MCL$rhoP.sigma
MCL$rhoI.sigma

MCL$PaidResiduals
MCL$IncurredResiduals

MCL$QinverseResiduals
MCL$QResiduals
```

```
MCL$lambdaP
MCL$lambdaI
# Section 3.3.3 Results
MCL$MCLPaid
MCL$MCLIncurred
```

---

plot.BootChainLadder

Plot method for a BootChainLadder object

Description

`plot.BootChainLadder`, a method to plot the output of `BootChainLadder`. It is designed to give a quick overview of a `BootChainLadder` object and to check the model assumptions.

Usage

```
## S3 method for class 'BootChainLadder':
plot(x, mfrow=c(2,2), title=NULL, log=FALSE, ...)
```

Arguments

`x`	output from `BootChainLadder`
`mfrow`	see `par`
`title`	see `title`
`log`	logical. If `TRUE` the y-axes of the 'latest incremental actual vs. simulated' plot will be on a log-scale
`...`	optional arguments. See `plot.default` for more details.

Details

`plot.BootChainLadder` shows four graphs, starting with a histogram of the total simulated IBNRs over all origin periods, including a rug plot; a plot of the empirical cumulative distribution of the total IBNRs over all origin periods; a box-whisker plot of simulated ultimate claims costs against origin periods; and a box-whisker plot of simulated incremental claims cost for the latest available calendar period against actual incremental claims of the same period. In the last plot the simulated data should follow the same trend as the actual data, otherwise the original data might have some intrinsic trends which are not reflected in the model.

Note

The box-whisker plot of latest actual incremental claims against simulated claims follows is based on ideas from Barnett and Zehnwirth in: *Barnett and Zehnwirth. The need for diagnostic assessment of bootstrap predictive models,* Insureware technical report. *2007*

Author(s)

Markus Gesmann

See Also

See also BootChainLadder

Examples

```
B <- BootChainLadder(RAA)
plot(B)
plot(B, log=TRUE)
```

---

plot.MackChainLadder

Plot method for a MackChainLadder object

Description

plot.MackChainLadder, a method to plot the output of MackChainLadder. It is designed to give a quick overview of a MackChainLadder object and to check Mack's model assumptions.

Usage

```
## S3 method for class 'MackChainLadder':
plot(x, mfrow=c(3,2), title=NULL, lattice=FALSE,...)
```

Arguments

x	output from MackChainLadder
mfrow	see par
title	see title
lattice	logical. Default is set to FALSE and plots as described in the details section are produced. If lattice=TRUE, the function xyplot of the lattice package is used to plot developments by origin period in different panels, plus Mack's S.E.
...	optional arguments. See plot.default for more details.

Details

plot.MackChainLadder shows six graphs, starting from the top left with a stacked bar-chart of the latest claims position plus IBNR and Mack's standard error by origin period; next right to it is a plot of the forecasted development patterns for all origin periods (numbered, starting with 1 for the oldest origin period), and 4 residual plots. The residual plots show the standardised residuals against fitted values, origin period, calendar period and development period. All residual plot should show no patterns or directions for Mack's method to be applicable. Pattern in any direction can be the result of trends and should be further investigated, see *Barnett and Zehnwirth. Best estimates for reserves. Proceedings of the CAS, LXXXVI I(167), November 2000.* for more details on trends.

Author(s)

Markus Gesmann

See Also

See Also MackChainLadder, residuals.MackChainLadder

Examples

```
plot(MackChainLadder(RAA))
```

`plot.MunichChainLadder` *Plot method for a MunichChainLadder object*

Description

`plot.MunichChainLadder`, a method to plot the output of `MunichChainLadder` object. It is designed to give a quick overview of a `MunichChainLadder` object and to check the correlation between the paid and incurred residuals.

Usage

```
## S3 method for class 'MunichChainLadder':
plot(x,  mfrow=c(2,2), title=NULL, ...)
```

Arguments

`x`	output from `MunichChainLadder`
`mfrow`	see `par`
`title`	see `title`
`...`	optional arguments. See `plot.default` for more details.

Details

`plot.MunichChainLadder` shows four plots, starting from the top left with a barchart of forecasted ultimate claims costs by Munich-chain-ladder (MCL) on paid and incurred data by origin period; the barchart next to it compares the ratio of forecasted ultimate claims cost on paid and incurred data based on the Mack-chain-ladder and Munich-chain-ladder methods; the two residual plots at the bottom show the correlation of (incurred/paid)-chain-ladder factors against the paid-chain-ladder factors and the correlation of (paid/incurred)-chain-ladder factors against the incurred-chain-ladder factors.

Note

The design of the plots follows those in Quarg's (2004) paper: *Gerhard Quarg and Thomas Mack. Munich Chain Ladder. Blatter DGVFM 26, Munich, 2004.*

Author(s)

Markus Gesmann

See Also

See also `MunichChainLadder`

Examples

```
M <- MunichChainLadder(MCLpaid, MCLincurred)
plot(M)
```

---

predict.TriangleModel — *Prediction of a claims triangle*

Description

The function is internally used by `MackChainLadder` to forecast future claims.

Usage

```
## S3 method for class 'TriangleModel':
predict(object,...)
## S3 method for class 'ChainLadder':
predict(object,...)
```

Arguments

`object`	a list with two items: `Models`, `Triangle` `Models` list of linear models for each development period `Triangle` input triangle to forecast
`...`	not in use

Value

`FullTriangle` forecasted claims triangle

Author(s)

Markus Gesmann

See Also

See also chainladder, MackChainLadder

Examples

```
RAA

CL <- chainladder(RAA)
CL
predict(CL)
```

---

qpaid — *Quarterly run off triangle of accumulated claims data*

Description

Sample data to demonstrate how to work with triangles with a higher development period frequency than origin period frequency

Usage

```
data(qpaid); data(qincurred)
```

Format

A matrix with 12 accident years and 45 development quarters of claims costs.

Source

Made up data for testing purpose

Examples

```
dim(qpaid)
dim(qincurred)
op=par(mfrow=c(1,2))
ymax <- max(c(qpaid,qincurred),na.rm=TRUE)*1.05
matplot(t(qpaid), type="l", main="Paid development",
  xlab="Dev. quarter", ylab="$", ylim=c(0,ymax))
matplot(t(qincurred), type="l", main="Incurred development",
      xlab="Dev. quarter", ylab="$", ylim=c(0,ymax))
par(op)
## MackChainLadder expects a quadratic matrix so let's expand
## the triangle to a quarterly origin period.
n <- ncol(qpaid)
Paid <- matrix(NA, n, n)
Paid[seq(1,n,4),] <- qpaid
M <- MackChainLadder(Paid)
```

```
plot(M)

# We expand the incurred triangle in the same way
Incurred <- matrix(NA, n, n)
Incurred[seq(1,n,4),] <- qincurred

# With the expanded triangles we can apply MunichChainLadder
MunichChainLadder(Paid, Incurred)

# In the same way we can apply BootChainLadder
# We reduce the size of bootstrap replicates R from the default of 999 to 99 purely to reduc
BootChainLadder(Paid, R=99)
```

RAA *Run off triangle of accumulated claims data*

Description

Run-off triangle of Automatic Factultative business in General Liability

Usage

```
data(RAA)
```

Format

A matrix with 10 accident years and 10 development years.

Source

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

References

See Also: *Which Stochastic Model is Underlying the Chain Ladder Method?*, *Thomas Mack*, Insurance Mathematics and Economics, ***15, 2/3***, *pp133-138*, *1994*

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

Examples

```
RAA
plot(RAA)
plot(RAA, lattice=TRUE)
```

`residuals.MackChainLadder`
Extract residuals of a MackChainLadder model

Description

Extract residuals of a `MackChainLadder` model by origin-, calendar- and development period.

Usage

```
## S3 method for class 'MackChainLadder':
residuals(object, ...)
```

Arguments

`object`	output of `MackChainLadder`
`...`	not in use

Value

The function returns a `data.frame` of residuals and standardised residuals by origin-, calendar- and development period.

Author(s)

Markus Gesmann

See Also

See Also `MackChainLadder`

Examples

```
RAA
MCL=MackChainLadder(RAA)
MCL

residuals(MCL)
```

summary.BootChainLadder *Methods for BootChainLadder objects*

Description

summary, print, mean, and quantile methods for BootChainLadder objects

Usage

```
## S3 method for class 'BootChainLadder':
summary(object, probs=c(0.75,0.95), ...)

## S3 method for class 'BootChainLadder':
print(x, probs=c(0.75,0.95), ...)

## S3 method for class 'BootChainLadder':
quantile(x, probs=c(0.75, 0.95), na.rm = FALSE,
               names = TRUE, type = 7,...)

## S3 method for class 'BootChainLadder':
mean(x, ...)

## S3 method for class 'BootChainLadder':
residuals(object, ...)
```

Arguments

x, object	output from BootChainLadder
probs	numeric vector of probabilities with values in [0,1], see quantile for more help
na.rm	logical; if true, any NA and NaN's are removed from 'x' before the quantiles are computed, see quantile for more help
names	logical; if true, the result has a names attribute. Set to FALSE for speedup with many 'probs', see quantile for more help
type	an integer between 1 and 9 selecting one of the nine quantile algorithms detailed below to be used, see quantile
...	further arguments passed to or from other methods

Details

print.BootChainLadder calls summary.BootChainLadder and prints a formatted version of the summary. residuals.BootChainLadder gives the residual triangle of the expected chain-ladder minus the actual triangle back.

Value

`summary.BootChainLadder`, `mean.BootChainLadder`, and `quantile.BootChainLadder`, give a list with two elements back:

`ByOrigin`	data frame with summary/mean/quantile statistics by origin period
`Totals`	data frame with total summary/mean/quantile statistics for all origin period

Author(s)

Markus Gesmann

See Also

See also `BootChainLadder`

Examples

```
B <- BootChainLadder(RAA, R=999, process.distr="gamma")
B
summary(B)
mean(B)
quantile(B, c(0.75,0.95,0.99, 0.995))
```

`summary.MackChainLadder`

Summary and print function for Mack-chain-ladder

Description

`summary` and `print` methods for a `MackChainLadder` object

Usage

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

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

Arguments

`x, object`	object of class `"MackChainLadder"`
`...`	optional arguments to `print` or `summary` methods

Details

`print.MackChainLadder` calls `summary.MackChainLadder` and prints a formatted version of the summary.

Value

`summary.MackChainLadder` gives a list of two elements back

`ByOrigin`	data frame with `Latest` (latest actual claims costs), `Dev.To.Date` (chain-ladder development to date), `Ultimate` (estimated ultimate claims cost), `IBNR` (estimated IBNR), `Mack.S.E` (Mack's estimation of the standard error of the IBNR), and `CV(IBNR)` (Coefficient of Variance=Mack.S.E/IBNR)
`Totals`	data frame of totals over all origin periods. The items follow the same naming convention as in `ByOrigin` above

Author(s)

Markus Gesmann

See Also

See also `MackChainLadder`, `plot.MackChainLadder`

Examples

```
R <- MackChainLadder(RAA)
R
summary(R)
summary(R)$ByOrigin$Ultimate
```

`summary.MunichChainLadder`

Summary and print function for Munich-chain-ladder

Description

`summary` and `print` methods for a `MunichChainLadder` object

Usage

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

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

Arguments

`x, object`	object of class "`MunichChainLadder`"
`...`	optional arguments to `print` or `summary` methods

Details

`print.MunichChainLadder` calls `summary.MunichChainLadder` and prints a formatted version of the summary.

Value

`summary.MunichChainLadder` gives a list of two elements back

`ByOrigin`	data frame with *Latest Paid* (latest actual paid claims costs), *Latest Incurred* (latest actual incurred claims position), *Latest P/I Ratio* (ratio of latest paid/incurred claims), *Ult. Paid* (estimate ultimate claims cost based on the paid triangle), *Ult. Incurred* (estimate ultimate claims cost based on the incurred triangle),*Ult. P/I Ratio* (ratio of ultimate paid forecast / ultimate incurred forecast)
`Totals`	data frame of totals over all origin periods. The items follow the same naming convention as in `ByOrigin` above

Author(s)

Markus Gesmann

See Also

See also `MunichChainLadder`, `plot.MunichChainLadder`

Examples

```
M <- MunichChainLadder(MCLpaid, MCLincurred)
M
summary(M)
summary(M)$ByOrigin
```

`triangle S3 Methods` *Generic functions for triangles*

Description

Functions to ease the work with triangle shaped matrix data. A 'triangle' is a matrix with some generic functions. Triangles are usually stored in a 'long' format in data bases. The function `as.triangle` can transform a `data.frame` into a triangle shape.

Usage

```
## S3 method for class 'matrix':
as.triangle(Triangle,origin="origin", dev="dev", value="value",...)
## S3 method for class 'data.frame':
as.triangle(Triangle, origin="origin", dev="dev", value="value",...)
## S3 method for class 'triangle':
as.data.frame(x, row.names=NULL, optional, lob=NULL, na.rm=FALSE, ...)
as.triangle(Triangle, origin="origin", dev="dev", value="value",...)
## S3 method for class 'triangle':
plot(x, t = "b", xlab = "dev. period", ylab = NULL, lattice=FALSE, ...)
```

Arguments

`Triangle`	a triangle
`origin`	name of the origin period, default is `"origin"`.
`dev`	name of the development period, default is `"dev"`.
`value`	name of the value, default is `"value"`.
`row.names`	default is set to `NULL` an will merge origin and dev. period to create row names.
`lob`	default is `NULL`. The idea is to use `lob` (line of business) as an additional column to label a triangle in a long format, see the examples for more details.
`optional`	not used
`na.rm`	logical. Remove missing values?
`x`	a matrix of class 'triangle'
`xlab`	a label for the x axis, defaults to 'dev. period'
`ylab`	a label for the y axis, defaults to NULL
`lattice`	logical. If `FALSE` the function `matplot` is used to plot the developments of the triangle in one graph, otherwise the `xyplot` function of the lattice package is used, to plot developments of each origin period in a different panel.
`t`	type, see `plot.default`
`...`	arguments to be passed to other methods

Author(s)

Markus Gesmann

Examples

```
GenIns
plot(GenIns)
plot(GenIns, lattice=TRUE)


## Convert long format into triangle
## Triangles are usually stored as 'long' tables in data bases
head(GenInsLong)
```

```
as.triangle(GenInsLong, origin="accyear", dev="devyear", "incurred claims")

X <- as.data.frame(RAA)
head(X)

Y <- as.data.frame(RAA, lob="General Liability")
head(Y)
```

Index