Creating Reports with ChainLadder Package Reserve Projections

Andy Merlino

Monday, February 23, 2015

Abstract

The 'ChainLadder' package can create some powerful loss reserving projections. The output from these reserving functions is a list containing all the relevant data an actuary would ever want. The 'exhibit' package extracts the most commonly desired 'ChainLadder' function outputs and returns a data frame ready for a report. This document displays how 'exhibit' can be used along with other R packages to quickly create reports. For more information on the 'exhibit' package download the package and enter 'browseVignettes("exhibit")' to see the introductory vignette.

Required Packages and Data

```r
# CRAN packages
library(ChainLadder) # reserving projections
library(xtable) # make pretty tables
options(xtable.comment = FALSE) # xtable option

# Ractuary packages. Install with devtools::install_github("merlinoa/*")
library(exhibit) # default exhibits from ChainLadder package
library(casdata) # load in the data
```

This document echos all of the code because it is intended as a reference document. In an actual actuarial report only the final table would be displayed.

The following examples use workers' compensation paid loss and ALAE from State Farm Mutual Group. This data is provided by the Casualty Actuarial Society and is made available in R through the `casdata` package.

```r
# filter the data to be used in our projections
calendar <- wkcomp$AccidentYear + wkcomp$DevelopmentLag
state <- wkcomp[wkcomp$GRCODE == 1767 & calendar < 1999, ]
```

`state` consists of State Farm Mutual Group workers' compensation paid losses, incurred losses, and other information from accident years 1988 through 1997.

Each code chuck operates as follows:

1. Use a `ChainLadder` function to make a reserve projection.
2. Use `exhibit()` to extract a data frame summary of that projection.
3. Use `xtable()` and its associated `xtable.print()` function to generate a LaTeX table.

Paid Loss & ALAE Triangle

```r
# create paid loss triangle
paid_tri <- as.triangle(state, origin = "AccidentYear",
                        dev = "DevelopmentLag", value = "CumPaidLoss_D")

# extracts a summary, format it, make it into a table
paid_tri_out <- exhibit(paid_tri)
paid_tri_out <- xtable(paid_tri_out, digits = 0)
print(paid_tri_out,
      format.args = list(big.mark = ","))
```

	1	2	3	4	5	6	7	8	9	10
1988	22,190	60,834	85,104	100,151	108,812	114,967	118,790	121,558	123,492	125,049
1989	26,542	77,798	106,407	122,422	133,359	138,599	143,029	145,712	147,358	
1990	32,977	100,494	134,886	157,758	168,991	178,065	182,787	187,760		
1991	38,604	114,428	157,103	181,322	197,411	208,804	213,396			
1992	42,466	125,820	164,776	189,045	204,377	213,904				
1993	46,447	116,764	154,897	179,419	193,676					
1994	41,368	100,344	132,021	151,081						
1995	35,719	83,216	111,268							
1996	28,746	66,033								
1997	25,265									

Paid Development Triangle

```
# create paid development triangle
paid_ata <- ata(paid_tri)

# format paid development triangle for presentation
paid_ata <- exhibit(paid_ata)
paid_ata_out <- xtable(paid_ata)
print(paid_ata_out)
```

	1-2	2-3	3-4	4-5	5-6	6-7	7-8	8-9	9-10
1988	2.74	1.40	1.18	1.09	1.06	1.03	1.02	1.02	1.01
1989	2.93	1.37	1.15	1.09	1.04	1.03	1.02	1.01	
1990	3.05	1.34	1.17	1.07	1.05	1.03	1.03		
1991	2.96	1.37	1.15	1.09	1.06	1.02			
1992	2.96	1.31	1.15	1.08	1.05				
1993	2.51	1.33	1.16	1.08					
1994	2.43	1.32	1.14						
1995	2.33	1.34							
1996	2.30								
Simple	2.69	1.35	1.16	1.08	1.05	1.03	1.02	1.01	1.01
Weighted	2.68	1.34	1.16	1.08	1.05	1.03	1.02	1.01	1.01

GLM Reserving model

Running the GLM model with default options is as easy as this:

```
glm_reserve <- glmReserve(paid_tri)

glm_reserve_out <- exhibit(glm_reserve)
glm_reserve_out <- xtable(glm_reserve_out,
                          digits = c(0, 0, 2, 0, 0, 0, 2))
print(glm_reserve_out)
```

	Latest	Dev.To.Date	Ultimate	IBNR	S.E	CV
1988	125049	1.00	125049	0		
1989	147358	0.99	149216	1858	1039	0.56
1990	187760	0.97	192674	4914	1667	0.34
1991	213396	0.95	224115	10719	2366	0.22
1992	213904	0.93	230811	16907	2858	0.17
1993	193676	0.88	219624	25948	3394	0.13
1994	151081	0.81	185415	34334	3806	0.11
1995	111268	0.70	157873	46605	4555	0.10
1996	66033	0.53	125746	59713	5770	0.10
1997	25265	0.20	129150	103885	12302	0.12
totals:	1434790	0.82	1739673	304882	19579	0.06

If we select to use the bootstrap method to estimate the mean squared error, the `glmReserve` function provides confidence levels.

```r
glm_boot <- glmReserve(paid_tri, mse.method = "boot")
glm_boot_out <- exhibit(glm_boot)

glm_boot_out <- xtable(glm_boot_out,
                  digits = c(0, 0, 2, 0, 0, 0, 2))
print(glm_boot_out)
```

	Latest	Dev.To.Date	Ultimate	IBNR	S.E	CV
1988	125049	1.00	125049	0		
1989	147358	0.99	149216	1858	1041	0.56
1990	187760	0.97	192674	4914	1686	0.34
1991	213396	0.95	224115	10719	2254	0.21
1992	213904	0.93	230811	16907	2808	0.17
1993	193676	0.88	219624	25948	3465	0.13
1994	151081	0.81	185415	34334	3796	0.11
1995	111268	0.70	157873	46605	4365	0.09
1996	66033	0.53	125746	59713	6042	0.10
1997	25265	0.20	129150	103885	12672	0.12
totals:	1434790	0.82	1739673	304882	19699	0.06

```r
pr <- as.data.frame(glm_boot$sims.reserve.pred)
qv <- c(0.5, 0.75, 0.9, 0.95, 0.975, 0.99)
res_q <- t(apply(pr, 2, quantile, qv))

res_q_out <- xtable(as.data.frame(res_q),
                    digits = 0)
print(res_q_out)
```

	50%	75%	90%	95%	97.5%	99%
1989	1785	2608	3311	3722	4203	4584
1990	4813	6006	7132	7903	8681	9281
1991	10505	12104	13608	14429	15467	16256
1992	16877	18675	20590	21466	22826	24129
1993	25879	28159	30537	31829	33476	34392
1994	34387	36906	39404	40759	42027	43800
1995	46282	49452	52331	53895	55171	56727
1996	59605	63857	67677	69953	72106	74361
1997	102869	112958	120369	125705	129349	133159

Mack Chain Ladder

```r
mack <- MackChainLadder(paid_tri)
mack_out <- exhibit(mack)

mack_out <- xtable(mack_out,
              digits = c(0, 0, 2, 0, 0, 0))
print(mack_out)
```

	Latest	Dev.To.Date	Ultimate	IBNR	Mack.S.E
1988	125049	1.00	125049	0	0
1989	147358	0.99	149216	1858	311
1990	187760	0.97	192674	4914	779
1991	213396	0.95	224115	10719	1308
1992	213904	0.93	230811	16907	1776
1993	193676	0.88	219624	25948	2333
1994	151081	0.81	185415	34334	2460
1995	111268	0.70	157873	46605	2831
1996	66033	0.53	125746	59713	4281
1997	25265	0.20	129150	103885	18207
Totals	1434790	0.82	1739672	304882	20364

Bootstrap Chain Ladder

```r
boot <- BootChainLadder(paid_tri)

boot_out <- exhibit(boot)

boot_out <- xtable(boot_out,
              digits = 0)
print(boot_out)
```

	Latest	Mean Ultimate	Mean IBNR	SD IBNR	IBNR 75%	IBNR 95%
1988	125049	125049	0	0	0	0
1989	147358	149201	1843	1025	2441	3808
1990	187760	192695	4935	1697	6020	7895
1991	213396	224115	10719	2376	12360	14728
1992	213904	230841	16937	2845	18717	21956
1993	193676	219667	25991	3461	28490	31692
1994	151081	185722	34641	3870	37264	41342
1995	111268	158283	47015	4514	49976	54701
1996	66033	125786	59753	5874	63731	69805
1997	25265	128691	103426	12254	111758	124408
Totals	1434790	1740048	305258	19577	318623	337325