用 R 也能做精算—actuar 包学习笔记

李 皞

中国人民大学统计学院
风险管理与精算
Email: kimbooland@gmail.com

Contents

1 引言 **2**

2 数据描述 **3**

2.1 构造分组数据对象 3

2.2 分组数据分布图 6

2.3 计算数据经验矩 7

3 损失分布 **11**

3.1 损失分布种类 11

3.2 损失分布的估计 13

3.3 损失随机变量的修正 18

4 风险理论 **22**

4.1 复合分布 22

4.2 连续分布的离散化 22

4.3 复合分布的计算 24

4.4 VaR 和 TVaR 28

5 保单组合的模拟 **29**

5.1 复合层次模型的模拟 29

5.2 模拟结果的处理 32

6 信度理论 **37**

6.1 信度理论简述 37

6.2 层次信度模型 38

6.3 信度回归模型 44

Chapter 1

引言

本文是对 R 中精算学专用包 actuar 使用的一个简单教程。actuar 项目开始于 2005 年，在 2006 年 2 月首次提供公开下载，其目的就是将一些常用的精算功能引入 R 系统，截止到目前，最新的版本是 1.1-1，且该包仍在不断完善中。actuar 是一个集成化的精算函数系统，虽然其他 R 包中的很多函数可以供精算师使用，但是为了达到某个目的而寻找某个包的某个函数是一个费时费力的过程，因此，actuar 将精算建模中常用的函数汇集到一个包中，方便了人们的使用。目前，该包提供的函数主要涉及风险理论，损失分布和信度理论，特别是为非寿险研究提供了很多方便的工具。

如题所示，本文是我在学习 actuar 包过程中的学习笔记，主要涉及这个包中一些函数的使用方法和细节，对一些方法的结论也有稍许探讨，因此能简略的地方简略，而讨论的地方可能讲的会比较详细。文章主要是针对 R 语言的初学者，因此每种函数或数据的结构进行了尽可能直白的描述，以便于理解，如有描述不清或者错漏之处，敬请各位指正。闲话少提，下面就正式开始咯！

Chapter 2

数据描述

2.1 构造分组数据对象

损失数据的类型主要分为分组数据和非分组数据。对于非分组数据的描述方法大家会比较熟悉，无论是数量上，还是图形上的，比如均值、方差、直方图、柱形图还有核密度估计等。因此下文的某些部分只介绍如何处理分组数据。

分组数据是精算研究中经常见到的数据类型，虽然原始的损失数据比分组数据包含有更多的信息，但是某些情况下受条件所限，只能获得某个损失所在的范围。与此同时，将数据分组也是处理原始数据的基本方法，通过将数据分到不同的组中，我们可以看到各组中数据的相对频数，有助于对数据形成直观的印象 (比如我们对连续变量绘制直方图)；而且在生存函数的估计中，数据量经常成千上万，一种处理方法是选定合适的时间或损失额度间隔，对数据进行分组，然后再使用分组数据进行生存函数的估计，这样可以有效减小计算量。现在假设我们要把一组连续变量分为 r 组： $(c_0, c_1], (c_1, c_2], \ldots, (c_{r-1}, c_r]$，那么就需要定义 $r+1$ 个边界 $c_0, c_1, \ldots, c_r$。实际中的损失数据或生存数据都是取非负值，因此 c_0 经常取 0。

对于分组数据来说，只需要知道每个组的数值范围及落在该组的观测频数，因此要构造一个完整的分组数据只需要提供上面两个信息即可。下面是分组数据的构造函数，注意这个函数是构造一个分组数据的结构，而非对现有连续数据进行分组，该函数返回一个分组数据的对象 (grouped data object)。

函数语法：

```
gouped.data(Group=c(…),freq1=c(…),freq2=c(….),…
,right=TRUE,row.names=NULL)
```

使用说明：

1. Group 定义的是分组的边界值，freq1 和 freq2(还可以定义 freq3 及更多) 是每条分组数据的频数。Group，freq1 和 freq2 可以随意命名，比如我们可以将 Group 改为“分数档”，将 freq1，freq2 改为“一班”，“二班”等，那么“一班”，“二班”就是两条分组数据，有着共同的分组数据边界值。要注意，一定要把边界值向量放在第一个参数的位置！
2. Group 向量要比 freq 向量多出一个长度 (边界数比组数多 1)。
3. 默认分组区间是左开右闭，如果想变为左闭右开可以设置 right=FALSE。row.names 可以自定义行的名称。
4. 返回的是一个数据框。特别要注意对第一列的处理，见下面例子。

例子：

```
> library(actuar)
> options(digits = 4)
> x = grouped.data(Group = c(0, 25, 50, 100, 150, 250, 500), Line.1 = c(30,
+     31, 57, 42, 65, 84), Line.2 = c(26, 33, 31, 19, 16, 11))
> x
```

```
       Group Line.1 Line.2
1   (0,  25]     30     26
2  (25,  50]     31     33
3  (50, 100]     57     31
4 (100, 150]     42     19
5 (150, 250]     65     16
6 (250, 500]     84     11
```

改成左闭右开区间，并自定义行名称。

```
> x1 = grouped.data(Group = c(0, 25, 50, 100, 150, 250, 500), Line.1 = c(30,
+     31, 57, 42, 65, 84), Line.2 = c(26, 33, 31, 19, 16, 11),
+     right = F, row.names = LETTERS[1:6])
> x1
```

```
       Group Line.1 Line.2
A   [0,  25)     30     26
B  [25,  50)     31     33
C  [50, 100)     57     31
D [100, 150)     42     19
E [150, 250)     65     16
F [250, 500)     84     11
```

为避免修改原始数据 x，以下我们将 x 赋值到 x2，对 x2 进行操作。

```
> x2 = x
```

提取某个组的数据 (某行)。

```
> x2[1, ]
```

```
     Group Line.1 Line.2
1 (0,  25]     30     26
```

提取第一条分组数据 (某列)。

```
> x2[, 2]
```

```
[1] 30 31 57 42 65 84
```

提取各组的边界值。如果引用第一列你期待会出现什么结果呢？

```
> x2[, 1]
```

```
[1]   0  25  50 100 150 250 500
```

如同任何对数据框的操作，也可以对数据框中的数据进行修改。特别需要注意的是对第一列的修改，一定要同时指定分组区间的左右的边界值。比如下面这条命令将第一组的右边界由 25 改为 20，同时第二组的左边界也同时变为 20：

```
> (x2[1, 1] = c(0, 20))
```

```
[1]  0 20
```

体会这样修改波及的范围。

```
> (x2[c(3, 4), 1] = c(55, 110, 160))
```

```
[1]  55 110 160
```

如果只指定一个边界的话，那就默认左右边界值相同，所以不要这样做。下面这条命令将会导致第一组的左右边界都变为 10。

```
> (x2[1, 1] = 10)
```

```
[1] 10
```

到这里可能有人会问，如果现在我手中只有一组连续数据，如何实现对数据的分组并统计数据落在每组中的频数呢？答案就是使用 cut 函数。

例子：

生成 100 个服从均值为 5 的指数分布的随机数。

```
> set.seed(5)
> z = rexp(100, rate = 0.2)
```

指定边界点，也是划分点。

```
> break.points = c(0, 1, 4, 8, 14, Inf)
> (tz = table(cut(z, breaks = break.points)))
```

```
   (0,1]    (1,4]    (4,8]   (8,14] (14,Inf]
      16       35       26       18        5
```

用汇总结果直接构造分组数据对象。

```
> grouped.data(Group = break.points, freq = as.matrix(tz))
```

```
      Group freq
1  (0,   1]   16
2  (1,   4]   35
3  (4,   8]   26
4  (8,  14]   18
5 (14, Inf]    5
```

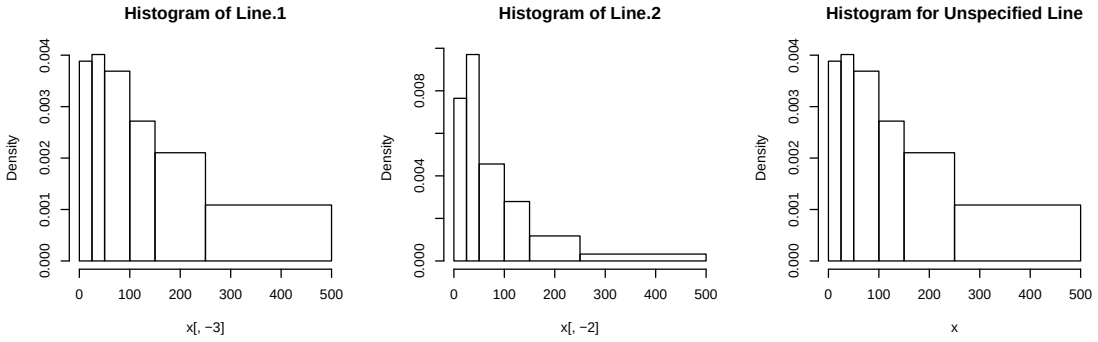
2.2 分组数据分布图

有了 grouped.data 对象，我们就可以对该对象进行一系列操作。首先是绘制分组数据的经验密度函数——直方图，和经验分布函数——拱形图。

1）绘制直方图。由于数据已经被划分好组别，因此 R 会应用分组数据对象 x 的第一列划分组距并绘制直方图，这在绘制非等距直方图时是十分方便的。由于每次只能绘制一组频率，因此绘图时需要指定频率所在的列，如果不指定，默认绘制第一组频率 (数据框的第二列)。

例子：

```
> layout(matrix(1:3, 1, 3))
> hist(x[, -3], main = "Histogram of Line.1")
> hist(x[, -2], main = "Histogram of Line.2")
> hist(x, main = "Histogram for Unspecified Line")
```



2）绘制拱形图。如同对连续的随机变量可以绘制经验分布函数图一样，对于分组数据也可以绘制“拱形图”(ogive)，也就是令分组临界点的函数值等于累计频率，对临界点间的函数值使用线性插值的方法构造的一条曲线。累计频率曲线的公式如下：

$$\tilde{F}_n(x) = \begin{cases} 0 & x \le c_0 \\ \frac{(c_j - x)F_n(c_{j-1}) + (x - c_{j-1})F_n(c_j)}{c_j - c_{j-1}} & c_{j-1} < x \le c_j \\ 1 & x > c_r \end{cases} \tag{2.1}$$

函数 ogive(x) 输入的是分组数据对象 x，返回的是一个阶梯函数对象 (Step Function Class)，也就是说实现了分组数据对象向阶梯函数对象的转换。如果给定函数的横坐标，就可以返回相对应的函数值，这点和 ecdf 是相同的。我们可以通过 konts 返回阶梯函数对象的临界点/间断点，通过 plot 绘制阶梯函数。

例子：

得到一个阶梯函数。

```
> Fnt = ogive(x)
```

返回临界点。

```
> knots(Fnt)
```

```
[1]   0  25  50 100 150 250 500
```

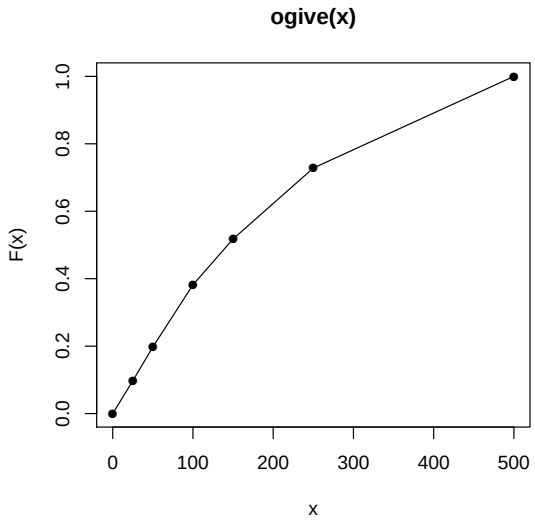
返回临界点对应的累积频率值。

```
> Fnt(knots(Fnt))
```

```
[1] 0.00000 0.09709 0.19741 0.38188 0.51780 0.72816 1.00000
```

对函数作图，得到 ogive 曲线。

```
> plot(Fnt)
```



2.3 计算数据经验矩

首先是计算经验[1]一阶矩。函数 mean 是一个泛型函数 (generic function)[2]，除可以作用于通常的非分组数据向量对象 (vector) 以外，还可以作用于分组数据对象 (grouped.data)，计算分组数据的均值。非分组数据的经验均值就是将所有数据算术平均，分组数据的经验均值定义为：

$$\frac{1}{n}\sum_{j=1}^{r} n_j\left(\frac{c_{j-1}+c_j}{2}\right) \tag{2.2}$$

该公式使用每组观测数乘以两端边界点的均值，得到该组的总值，将所有 r 个组的总值相加再除以样本量 n，就得到每个观测均值的估计。该公式假设每组内的观测值分布是均匀的。

例子：

以上文中的分组数据对象 x 为例。

```
> x
```

```
       Group Line.1 Line.2
1   (0,  25]     30     26
2  (25,  50]     31     33
3  (50, 100]     57     31
```

[1]文中经常会提到“经验 **”，比如经验分布函数，经验一阶矩等，个人理解所谓经验就是将样本当成总体去对待，比如经验方差是除以样本量 n，而样本方差是除以 n-1，或者说在 bootstrap 方法中，使用经验分布函数去代替总体。

[2]又译作类函数，泛型函数将作用对象的属性也作为一个参数输入，对于不同的对象类别采用不同的方法，并得到不同的输出。actuar 包中使得 mean 函数可以作用于 aggregateDist 和 grouped.data 对象，在加载 actuar 包后，可以通过命令 methods(mean) 查看 mean 函数的所有方法。

```
4 (100, 150]     42     19
5 (150, 250]     65     16
6 (250, 500]     84     11
```

```
> mean(x)
```

```
Line.1 Line.2
 179.8   99.9
```

如果直接用公式(2.2)计算，Line.1 的均值等价于：

```
> ((0 + 25)/2 * 30 + (25 + 50)/2 * 31 + (50 + 100)/2 * 57 + (100 +
+     150)/2 * 42 + +(150 + 250)/2 * 65 + (250 + 500)/2 * 84)/sum(x[,
+     2])
```

```
[1] 179.8
```

如果说均值函数 mean() 只能计算一阶矩，那么 emm 函数则可以计算任意阶的经验原点矩。首先引入 actuar 包中的两个数据集。其中 dental 是非分组数据，gdental 是分组数据。

```
> data(dental)
> dental
```

```
 [1]  141   16   46   40  351  259  317 1511  107  567
```

```
> data(gdental)
> gdental
```

```
             cj nj
1      (0,  25] 30
2    ( 25,  50] 31
3    ( 50, 100] 57
4    (100, 150] 42
5    (150, 250] 65
6    (250, 500] 84
7   (500, 1000] 45
8  (1000, 1500] 10
9  (1500, 2500] 11
10 (2500, 4000]  3
```

emm 函数可以计算任意阶经验原点矩，其使用方法是这样的:

```
emm(x,order=1)
```

其中，order 是阶数，可以赋值给它一个向量，这样就能一次性计算多个原点矩。x 可以是数据向量或者是矩阵，对于矩阵，emm 将每一列视为一条数据。

非分组数据 k 阶经验矩的计算公式为：

$$\frac{1}{n}\sum_{j=1}^{n} x_j^k \tag{2.3}$$

例子：

非分组数据向量形式。

```
> emm(dental, 2)

[1] 293068
```

数据矩阵形式。

```
> xx = matrix(1:9, 3, 3)
> xx

     [,1] [,2] [,3]
[1,]    1    4    7
[2,]    2    5    8
[3,]    3    6    9
```

下面代码的输出结果：第一行是 xx 第一列的均值和二阶矩，第二行是 xx 第二列的均值和二阶矩，依此类推。

```
> emm(xx, 1:2)

     [,1]   [,2]
[1,]    2  4.667
[2,]    5 25.667
[3,]    8 64.667
```

如果是分组数据，x 也可以是由 grouped.data() 生成的分组数据对象。分组数据 k 阶经验矩的计算公式为：

$$\frac{1}{n}\sum_{j=1}^{r} n_j \frac{(c_j^{k+1} - c_{j-1}^{k+1})}{(k+1)\cdot(c_j - c_{j-1})} \tag{2.4}$$

例子：

```
> emm(gdental, 1:3)

[1] 3.533e+02 3.577e+05 6.586e+08
```

有时候，损失数据会有保单限额 u 的存在 (有关保单限额的介绍参见 2.3 节)，超过 u 的损失额度被强制定义为 u。elev 函数可以计算经验有限期望值 (empirical limited expected value)，经验有限期望值都是一阶矩。使用方法是：

```
elev(x)
```

其中 x 可以是非分组数据，也可以是分组数据。

对于非分组数据，经验有限期望值的公式为：

$$\hat{E}[X \wedge u] = f(u) = \frac{1}{n}\sum_{j=1}^{n} \min(x_j, u) \tag{2.5}$$

对于分组数据，还要考虑 u 是否是位于分组的边界值上，因此经验有限期望公式比较复杂，在此略去，有兴趣的同学可以参考帮助文档。

我们注意到，有限期望值是上限值 u 的函数，不同的 u 计算出的经验有限期望值是不一样的。elev 直接返回一个函数对象，如果要具体计算某一个 u 的经验有限期望值，只需要特别指定 u 即可。

例子：

非分组数据，返回的 lev 是上限 u 的函数。

```
> lev = elev(dental)
```

这里将保单限额 u 设为 200。

```
> lev(200)
```

```
[1] 135
```

返回 lev 函数的拐点，注意到拐点都发生在数据点，而观察下图可知拐点间的函数都是线性的，你能从数学上解释这个现象吗？

```
> knots(lev)
```

```
 [1]   16   40   46  107  141  259  317  351  567 1511
```

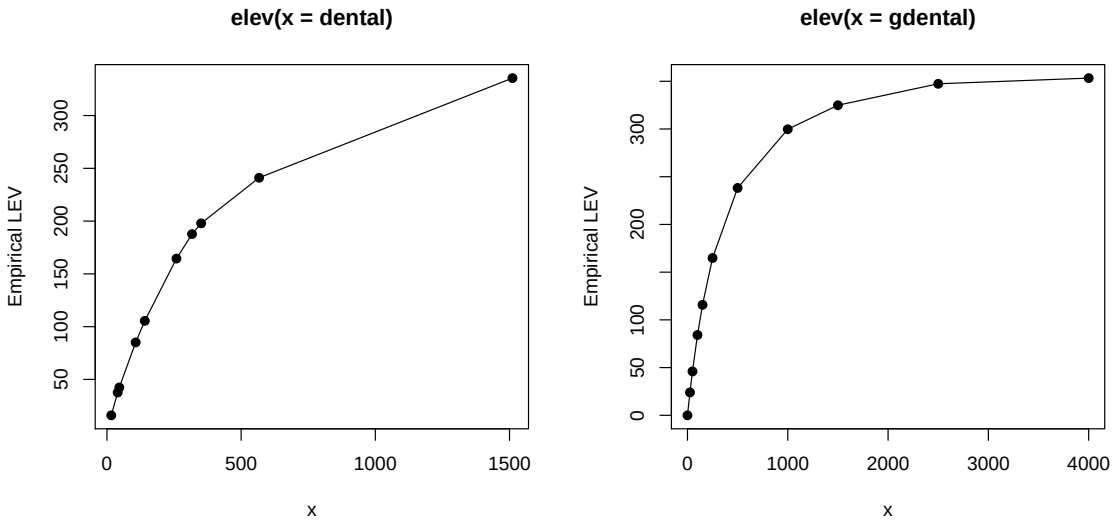
分组数据。

```
> lev2 = elev(gdental)
> lev2(200)
```

```
[1] 142.5
```

分别对非分组数据和分组数据的有限期望函数作图。

```
> par(mfrow = c(1, 2))
> plot(lev, type = "o", pch = 19)
> plot(lev2, type = "o", pch = 19)
```



Chapter 3

损失分布

3.1 损失分布种类

根据损失额的特征，损失分布常选用具有非负支集 (密度函数 $f(x)$ 的支集指的是使得 $f(x) \neq 0$ 的 x 的集合) 的连续分布。R 中对于一些分布提供了 d，p，q，r 四种函数，分别是密度函数、分布函数、分布函数的反函数 (分位数) 和生成该分布的随机数。actuar 包提供了与 [5] 的附录 A 中所列示的连续分布族相配套的这四种函数 (除去逆高斯和对数 t 分布，但包括对数 Gamma 分布)，这些分布中 R 的基础包 stats 中并不自带，但有些分布在精算研究中却很重要 (比如常用的后尾分布 pareto 分布)。此外，actuar 包还对这些连续分布提供了 m、lev 和 mgf 三种函数，m 是计算理论原点矩，lev 是计算有限期望值，mgf 是计算矩母函数。密度函数、分布函数、原点矩、有限期望值及其 k 次方都可以通过查询该附录得到。

对于经验数据，如上面所介绍的，actuar 包中提供了 emm 和 elev 来计算经验原点矩和经验有限期望值 (这两个函数的前缀都是 empirical)。

需要注意的是，这些分布有的需要指定 rate 参数或 scale 参数，scale=1/rate，因此两者在本质上是等价的，[5] 中使用的是 scale 参数，在指定参数时千万不要弄混。

例子：

这里以双参数 pareto 分布为例。

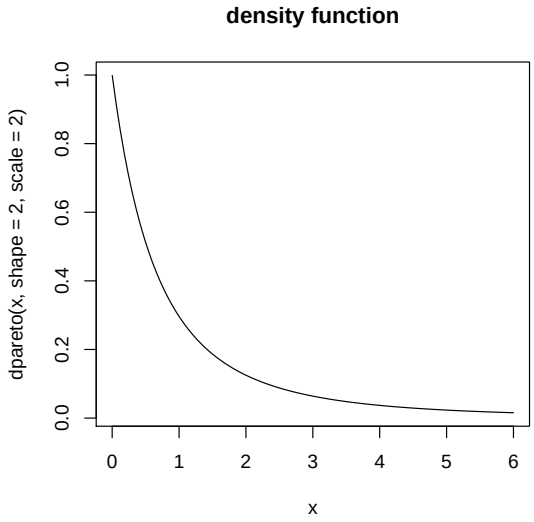
```
> par(mfrow = c(1, 2))
```

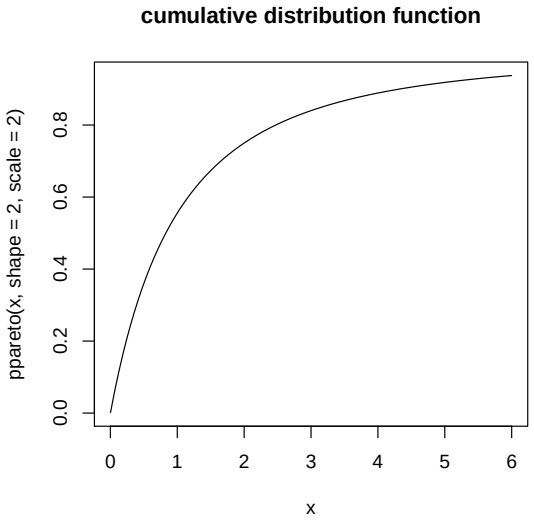
绘制密度函数曲线。

```
> curve(dpareto(x, shape = 2, scale = 2), from = 0.001, to = 6,
+     main = "density function")
```

绘制分布函数曲线。

```
> curve(ppareto(x, shape = 2, scale = 2), from = 0.001, to = 6,
+     main = "cumulative distribution function")
```





求 pareto 分布中位数。

```
> qpareto(0.5, shape = 2, scale = 2)

[1] 0.8284
```

生成 5 个 pareto 分布随机数。

```
> rpareto(5, shape = 2, scale = 2)

[1] 7.91025 0.09817 0.18824 0.48281 0.85600
```

求 $E(X^{1.5})$，注意 pareto 分布是厚尾分布，其 k 阶矩要求 $-1 < k < \alpha$， α 是 shape 参数。

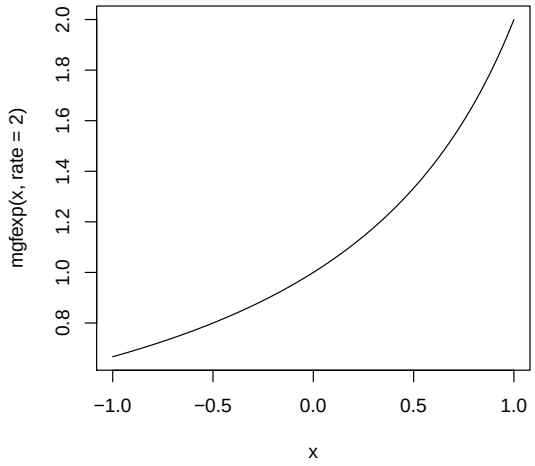
```
> mpareto(order = 1.5, shape = 2, scale = 2)

[1] 6.664
```

求 $E[(X \wedge 5)^{1.5}]$ ，注意同样要求 $-1 < k < \alpha$。

```
> levpareto(limit = 5, shape = 2, scale = 2, order = 1.5)

[1] 2.355
```

指数分布矩母函数，均值 $=1/2$。矩母函数形式为 $M_X(t) = \mu/(\mu - t)$， μ 为 rate 参数。

```
> curve(mgfexp(x, rate = 2), -1, 1)
```



3.2 损失分布的估计

矩估计和极大似然估计是分布参数估计的基本方法。在 R 中，MASS 包中的 fitdistr 函数可以进行极大似然估计。在 actuar 包中，mde 函数则提供了三种距离最小化的分布拟合方法 (minium distance estimates)。

1) Cramér-von Mises 方法 (CvM) 最小化理论分布函数和经验分布函数 (对于分组数据是 ogive) 的距离。

未分组数据：

$$d(\theta) = \sum_{j=1}^{r} w_j (F(x_j;\theta) - F_n(x_j))^2 \tag{3.1}$$

分组数据：

$$d(\theta) = \sum_{j=1}^{r} w_j (F(c_j;\theta) - \tilde{F}_n(c_j))^2 \tag{3.2}$$

在这里， $F(x;\theta)$ 是理论分布函数， θ 是其参数； $F_n(x)$ 是经验分布函数 ecdf； $\tilde{F}_n(x)$ 是分组数据的经验分布函数 ogive； w_j 是赋予每个观测或组别的权重，默认都取 1。

2) 修正卡方法仅应用于分组数据，通过最小化各组期望频数与实际观测频数的平方误差得到。

$$d(\theta) = \sum_{j=1}^{r} w_j [n(F(c_j;\theta) - F(c_{j-1};\theta)) - n_j]^2 \tag{3.3}$$

其中 $n = \sum_{j=1}^{n} n_j$， w_j 默认情况下为 n_j^{-1}。

3)LAS 法 (layer average severity) 也仅应用于分组数据。通过最小化各组内的理论和经验有限期望函数的平方误差得到。

$$d(\theta) = \sum_{j=1}^{r} w_j (LAS(c_{j-1}, c_j;\theta) - \tilde{LAS}_n(c_{j-1}, c_j))^2 \tag{3.4}$$

其中 $LAS(x,y) = E(X \wedge y) - E(X \wedge x)$， $\tilde{LAS}_n(x,y) = \tilde{E}_n[X \wedge y] - \tilde{E}_n[X \wedge x]$， $E()$ 是

理论分布的有限期望函数，而 $\tilde{E}_n$ 是经验分布的有限期望函数。 w_j 默认情况下为 n_j^{-1}。

该函数调用 stats 包中的 optim 函数做最优化。

函数语法：

```
mde(x,fun,start,measure=c("CvM","chi-square","LAS"),weights=NULL,...)
```

使用说明：

1. x 是分组数据对象的或未分组的数据。
2. fun 是待拟合的分布，CvM 法和修正卡方法需要指定分布函数：p**。LAS 法需要指定理论有限期望函数 lev**。
3. start 指定参数初始值。形式必须以列表的形式，形式可以见例子，有几个参数就要指定几个初始值。
4. measure 是指定方法。weight 指定权重，否则采用默认权重。
5. …是其他参数，可以指定 optim 函数中的参数，比如使用 L-BFGS-B 方法进行优化可以添加参数 method=“L-BFGS-B”。

mde 输出的结果是一个列表 (list)，rate 是参数估计结果，distance 是最小化后的距离。我们可以对上面的 gdental 数据进行分布拟合，这是一个分组数据，在为参数估计赋初始值时，假设数据来自均值为 200 的指数分布。。

例子：

首先观察一下数据的分布。

```
> hist(gdental)
```

CvM 法。

```
> (mde.est1 = mde(gdental, pexp, start = list(rate = 1/200), measure = "CvM"))

   rate
  0.003551


 distance
  0.002842

> mu1 = mde.est1[[1]]
```

修正卡方法。

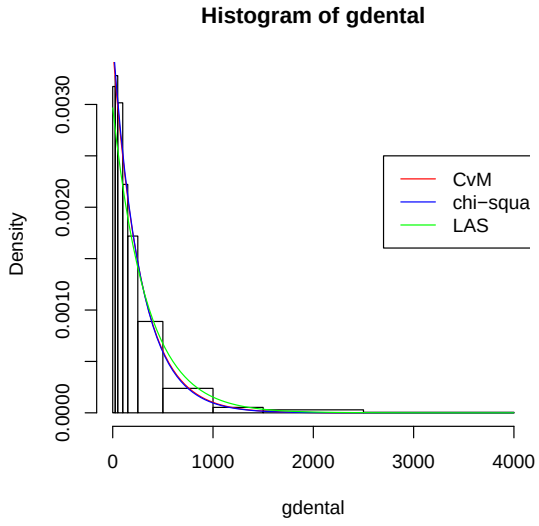
```
> (mde.est2 = mde(gdental, pexp, start = list(rate = 1/200), measure = "chi-square"))

   rate
  0.00364


 distance
     13.54

> mu2 = mde.est2[[1]]
```

LAS 法。

```
> (mde.est3 = mde(gdental, levexp, start = list(rate = 1/200),
+     measure = "LAS"))

   rate
  0.002966

 distance
     694.5

> mu3 = mde.est3[[1]]
```

做图，可以看出 CvM 法和修正卡方法估计结果相似，而 LAS 法损失分布尾部拟合效果较好。

```
> curve(mu1 * exp(1)^(-mu1 * x), from = 0, to = 4000, add = T,
+     col = "red")
> curve(mu2 * exp(1)^(-mu2 * x), from = 0, to = 4000, add = T,
+     col = "blue")
> curve(mu3 * exp(1)^(-mu3 * x), from = 0, to = 4000, add = T,
+     col = "green")
> legend(2700, 0.0025, legend = c("CvM", "chi-square", "LAS"),
+     col = c("red", "blue", "green"), lty = 1)
```



我们还可以对非分组数据进行分布拟合，下面的一个例子是一个混合分布。

例子：

首先生成 400 个随机数，其中 200 个来自 $Gamma(\alpha = 2, \theta = 2)$，200 个来自 $Gamma(\alpha = 10, \theta = 2)$。

```
> set.seed(3)
> dat = c(rgamma(200, shape = 2, scale = 2), rgamma(200, shape = 10,
+     scale = 2))
```

混合分布密度：

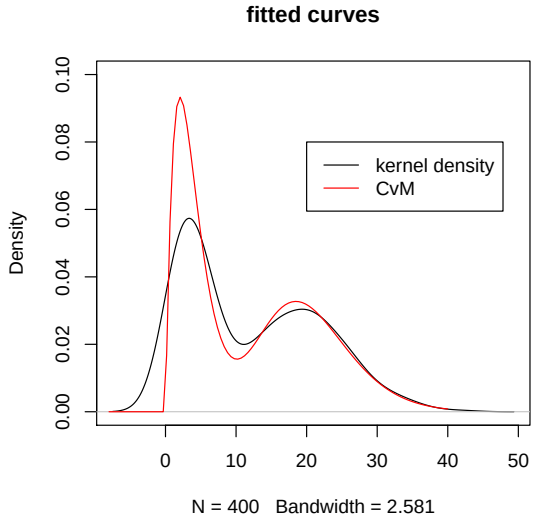
```
> dfn = function(x, a, alpha1, alpha2, theta) {
+     a * dgamma(x, shape = alpha1, scale = theta) + (1 - a) *
+         dgamma(x, shape = alpha2, scale = theta)
+ }
```

混合分布函数：

```
> pfn = function(x, a, alpha1, alpha2, theta) {
+     a * pgamma(x, shape = alpha1, scale = theta) + (1 - a) *
+         pgamma(x, shape = alpha2, scale = theta)
+ }
```

使用 mde 估计混合分布的参数，对于非分组数据只能使用 CvM 法。

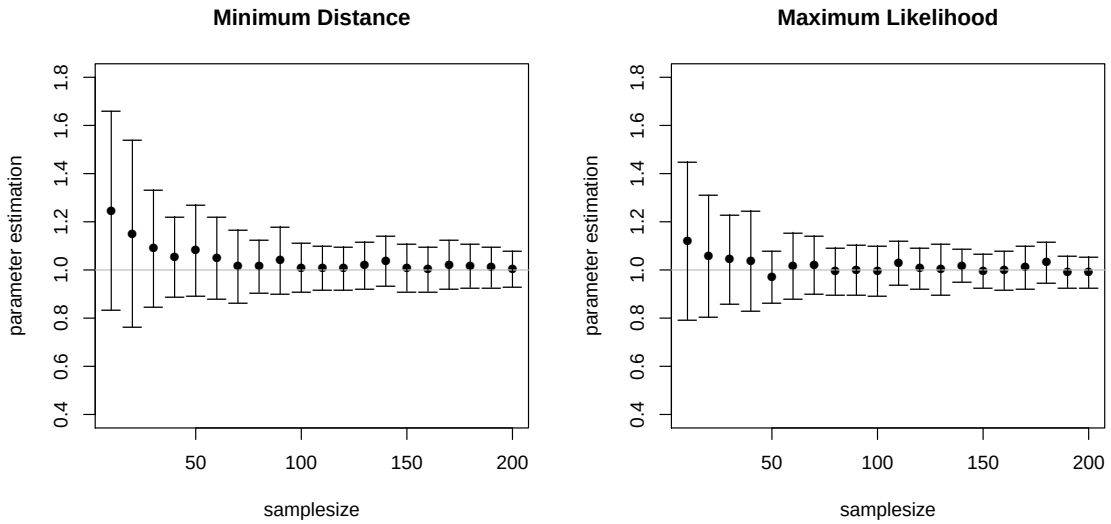
```
> mde.est4 = mde(dat, pfn, start = list(a = 0.4, alpha1 = 1, alpha2 = 8,
+     theta = 2.5), measure = "CvM")
> (para = mde.est4[[1]])

      a  alpha1  alpha2   theta
 0.5053  2.0552 10.5406  1.9478

> plot(density(dat), ylim = c(0, 0.1), main = "fitted curves")
> curve(dfn(x, a = para[1], alpha1 = para[2], alpha2 = para[3],
+     theta = para[4]), from = -8, to = 40, col = "red", add = T)
> legend(20, 0.08, legend = c("kernel density", "CvM"), col = c("black",
+     "red"), lty = 1)
```



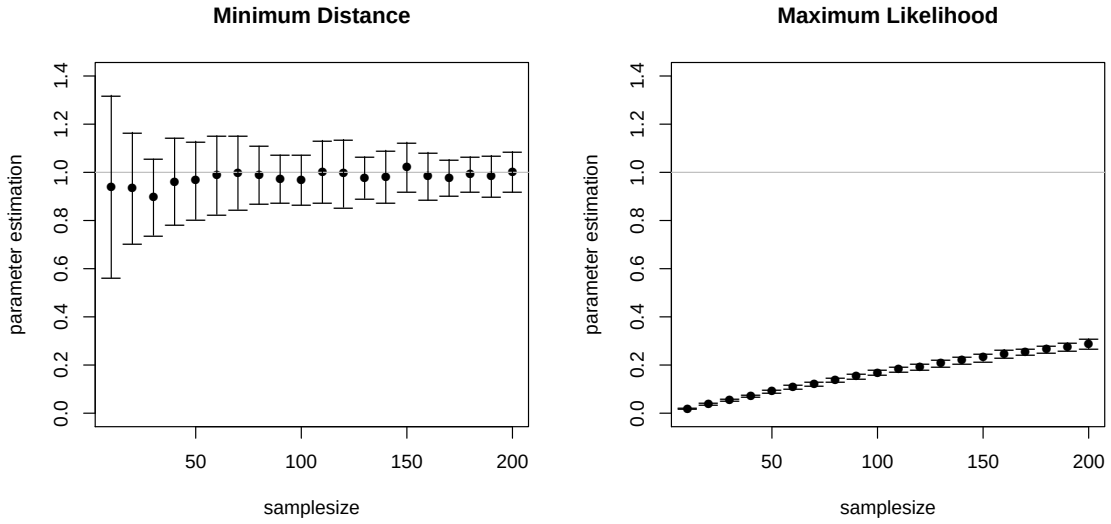
在此，我们感兴趣的是将最小距离法与极大似然法的参数估计效果进行以下对比。因此不妨做一个实验：

简单起见，先从单参数拟合问题开始，这是一个一维优化问题。首先生成 50 组来自于 rate=1 的指数分布随机数，每组的个数都为 10。然后，对于每一组随机数，分别用基于距离的估计方法和极大似然估计进行参数估计，将 50 次模拟结果的均值和标准差记录下来。之后，将

随机数的个数由 10 增加到 20，30⋯200。不断增大样本量，并重复之前的过程。最终得到的结果如下图：



可以看出，在单参数估计中，尤其是在样本量较大时，两种方法估计的结果相差不大，而且极大似然估计的方差要最小距离法的估计方差略小，因此极大似然估计的稳健性要优于最小距离估计。那么，两种方法对于异常值的稳健性如何呢？在上面生成的所有组随机数中，先剔除两个指数分布随机数，再混入两个来自 $[200, 300]$ 均匀分布的随机数，再重新对参数进行估计，结果很明显，最小距离法估计的结果很稳定，而极大似然法估计的参数结果受异常值的影响很大！因此，如果损失数据存在有少量极端损失，使用最小距离法往往更加稳健。



对于两参数的估计，由于某些分布要求参数恒为正，使用 mde 函数经常会报错，通常的解决办法是估计参数的对数形式：

$$\theta = \exp(\tau) \tag{3.5}$$

当算法在迭代时，让 τ 在 $(-\infty, \infty)$ 范围内变动，同时 θ 恒为正。当估计出 τ 的值后，再取指数进行还原。这样可以有效的降低优化失败的次数。

例子：

```
> pgammalog = function(x, logshape, logscale) {
+     pgamma(x, exp(logshape), exp(logscale))
+ }
> aa = rgamma(200, shape = 3, scale = 1)
> estlog = mde(aa, pgammalog, start = list(logshape = 1.3, logscale = 0.2),
+     measure = "CvM", method = "L-BFGS-B", lower = c(0.5, -0.5),
+     upper = c(1, 5, 0.5))$estimate
> exp(estlog)

logshape logscale
  2.7183   0.9288
```

3.3 损失随机变量的修正

我们知道，由于某些保险条款规则的存在，保险实际赔付额往往和实际损失数额是不相等的。假定我们定义实际损失数额为随机变量 X，实际赔付额为 Y，那么 $Y = f(X)$。具体说来，f 包括以下几种情况：

1) 免赔额 (deductible)：损失额超过某个数额，则赔付超出的部分。具体可以分为一般免陪额 (ordinary deductible) 和绝对免赔额 (franchise deductible)。

一般免赔额的数学形式：

$$Y = (X - d)_+ = \begin{cases} 0 & X \le d \\ X - d & X > d \end{cases} \tag{3.6}$$

绝对免赔额的数学形式：

$$Y = \begin{cases} 0 & X \le d \\ X & X > d \end{cases} \tag{3.7}$$

2) 最大保障损失 (maximum covered loss)：是保险人对于单个损失支付的最大赔付额。无论是否有免赔额，只要损失额超过最大保障损失就以最大保障损失封顶。

数学形式：

$$Y = X \wedge u = \begin{cases} X & X \le u \\ u & X > u \end{cases} \tag{3.8}$$

其中随机变量 $X \wedge u$ 称为有限损失随机变量，其期望 $E(X \wedge u)$ 称为有限期望值 (也就是前面介绍的 lev)。前面提到过保单限额的概念，在那里也用符号 u 表示。保单限额 (policy limit) 与最大保障损失的区别在于：当存在免赔额时，保单限额 = 最大保障损失 -免赔额。因此当不存在免赔额时，两者是等价的。

3) 通货膨胀和共同保险 (coinsurance)：通货膨胀是指未来的赔付额等于当前损失额乘以一个通胀因子， $Y = (1 + r)X$；而共同保险是指对每一次损失，保险公司只赔付一定的比例 $Y = \alpha X (0 < \alpha < 1)$，其余部分则由投保人自行承担。之所以将这两者放在一起，是因为它们都

是对原始损失变量乘以一个常量得到赔付额，在没有免赔额和赔偿限额的情况下，两者数学形式上是相同的。当存在赔额和赔偿限额时，通货膨胀的数学形式为：

$$Y=\begin{cases}0 & (1+r)X\le d\\ (1+r)X-d & d<(1+r)X\le u\\ u-d & (1+r)X>u\end{cases} \tag{3.9}$$

而共同保险的数学形式为：

$$Y=\begin{cases}0 & X\le d\\ \alpha(X-d) & d<X\le u\\ \alpha(u-d) & u<X\end{cases} \tag{3.10}$$

可以看出，通货膨胀首先对随机变量 X 进行通胀修正，再进行免赔额和赔偿限额的修正；共同保险首先对随机变量 X 进行免赔额和赔偿限额的修正，再进行共保修正，两者的顺序是不同的。

严格的说， Y 是可以进一步区分为 cost per loss(Y_L) 和 cost per payment(Y_P)。两者的区别在于， Y_L 是指每次损失带来保险公司的赔付额，而 Y_P 是指每次赔付带来保险公司的赔付额。如果没有免赔额，那么两者自然是等价的，但是由于免赔额的存在，每次损失保险公司不一定要赔偿， Y_L 可以等于 0，而每次赔付保险公司必然有正的赔付额， Y_P 严格大于 0。因此当损失额 X 大于免赔额， $Y_P=Y_L$，否则 $Y_L=0$ 而 Y_P 没有定义。以一般免赔额为例：

$$Y_L=(X-d)_+=\begin{cases}0 & X\le d\\ X-d & X>d\end{cases} \tag{3.11}$$

$$Y_P=\begin{cases}\text{无定义} & X\le d\\ X-d & X>d\end{cases} \tag{3.12}$$

为什么会有 Y_L 和 Y_P 的区分？通常保险公司只能获得实际赔付额的数据，由于免赔额的存在，当损失额小于免赔额 d 时，被保险人可能根本不会去保险公司报案，因此小于 d 的损失数据要么不完全，要么干脆无法获得，因此精算师索性将这部分损失数据忽略，只考虑导致正的赔付额的损失。如果只考虑正的赔付额那么 Y_P 就是一个条件随机变量，也就是以 $X>d$ 为条件，也就是：

$$Y_P=Y_L|X>d \tag{3.13}$$

这样 Y_P 的分布就是一个条件分布，而且 Y_P 的取值恒大于 0。

当精算师需要根据赔付数据 Y 对损失额 X 进行建模时，就需要建立两者之间的联系。什么样的联系呢？就是根据两个随机变量间的变换关系得到两个随机变量间分布密度和分布函数间的关系。大体的建模过程可以表述为：

1. 假定 X 的原始分布形式，然后根据变换关系得到 Y 的修正分布形式。在这个过程中，原始分布的参数也同时传递到了修正分布中。

2. 将实际赔付数据带入修正分布中，使用极大似然或其他估计方法估计得到修正分布的参数。

3. 根据原始分布和修正分布的关系得到原始分布的参数。

actuar 包中，coverage 这个函数可以完成将原始分布变换为修正分布的工作。coverage 输出的是一个函数对象。

函数语法：

```
coverage(pdf,cdf,deductible = 0,franchise = FALSE,limit = Inf,
coinsurance = 1,inflation = 0,per.loss = FALSE)
```

使用说明：

1. 如果 pdf 和 cdf 同时指定，那么输出是修正后的 pdf，如果只指定 cdf，那么输出的是修正后的 cdf。特别注意的是如果存在 deductible 或 limit，那么 cdf 必须指定。
2. deductible 设置免赔额 d，franchise 控制是绝对免赔额还是一般免赔额，默认为 FALSE，即一般免赔额。
3. limit 设置最大保障损失 u，默认为无上限。
4. coinsurance 是共保因子 α，取值为 0-1 之间的数。
5. inflation 是通胀率 r，取值为 0-1 之间的数。
6. per.loss 控制是采用 Y_P 还是 Y_L，默认采用 Y_P。
7. coverage 返回的是一个函数对象，如果存在重概率点 (probability mass)，那么这个函数对象在不同的点返回的值的意义是不同的。由于此时分布密度函数 pdf 并不是完全连续，在 probability mass 点，其“分布密度”的取值其实是一个概率值，而其他点的取值则是密度值，在绘制对应的分布密度图像时，应该对 probability 进行强调。

例子：

假设原始损失服从形状参数 shape=3，尺度参数 scale=1 的 Gamma 分布。首先计算修正后的密度函数，然后分别作出 Y_P 和 Y_L 的分布密度函数和分布函数。在作图时，有以下两点需要注意：1. 修正后的密度函数是连续分布与离散分布的混合，因此重概率点用加粗的点标出。2. 在出现重概率点时，对应的分布函数存在跳跃。

```
> par(mfrow = c(2, 2))
```

Y_P 的免赔额 =1，限额 =7。首先绘制原始分布的密度曲线，然后绘制 Y_P 的密度曲线。

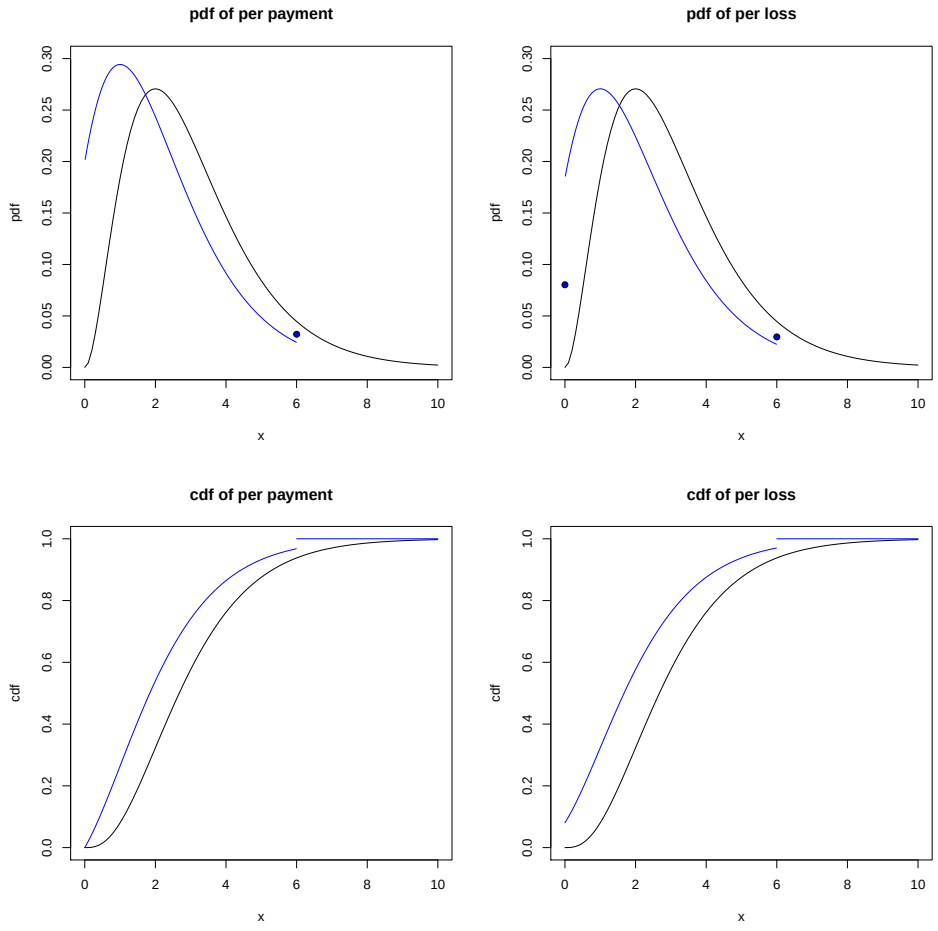
```
> f = coverage(pdf = dgamma, cdf = pgamma, deductible = 1, limit = 7)
> curve(dgamma(x, 3, 1), xlim = c(0, 10), ylim = c(0, 0.3), ylab = "pdf",
+     main = "pdf of per payment")
> curve(f(x, 3, 1), xlim = c(0.01, 5.99), col = 4, add = TRUE)
> points(6, f(6, 3, 1), pch = 21, bg = 4)
```

Y_L 的免赔额 =1，限额 =7。首先绘制原始分布的密度曲线，然后绘制 Y_L 的密度曲线。

```
> f1 = coverage(pdf = dgamma, cdf = pgamma, deductible = 1, limit = 7,
+     per.loss = T)
> curve(dgamma(x, 3, 1), xlim = c(0, 10), ylim = c(0, 0.3), ylab = "pdf",
+     main = "pdf of per loss")
> curve(f1(x, 3, 1), xlim = c(0.01, 5.99), col = 4, add = TRUE)
> points(6, f1(6, 3, 1), pch = 21, bg = 4)
> points(0, f1(0, 3, 1), pch = 21, bg = 4)
```

下面是 Y_P、 Y_L 和原始分布的分布函数曲线。

```
> F = coverage(cdf = pgamma, deductible = 1, limit = 7)
> curve(pgamma(x, 3, 1), xlim = c(0, 10), ylim = c(0, 1), ylab = "cdf",
+     main = "cdf of per payment")
> curve(F(x, 3, 1), xlim = c(0, 5.99), col = 4, add = TRUE)
> curve(F(x, 3, 1), xlim = c(6, 10), col = 4, add = TRUE)
> F1 = coverage(cdf = pgamma, deductible = 1, limit = 7, per.loss = T)
> curve(pgamma(x, 3, 1), xlim = c(0, 10), ylim = c(0, 1), ylab = "cdf",
+     main = "cdf of per loss")
> curve(F1(x, 3, 1), xlim = c(0, 5.99), col = 4, add = TRUE)
> curve(F1(x, 3, 1), xlim = c(6, 10), col = 4, add = TRUE)
```



Chapter 4

风险理论

4.1 复合分布

本部分主要介绍风险理论中的聚合风险模型。在机动车保险中，对于一辆或一批机动车，其每年发生的事故次数 N 服从一个离散分布，每次事故的损失金额 X 服从一个连续分布。那么，这一年总的损失额 S 可以表示为：

$$S = X_1 + X_2 + \ldots + X_N \tag{4.1}$$

可以看出 S 是一个随机和，我们把事故次数 N 的分布称作索赔频率分布 (frequency distribution)，每次损失额 X 的分布称作索赔强度分布 (severity distribution)， S 的分布称为复合分布 (compound distribution)。

上一小节讲如何估计分布的参数，假设我们已经将频率分布和强度分布的参数估计出来了，那么现在的问题就是如何得到总损失额 S 的分布，事实上，就保险公司的整体运营来讲，精算师可能更关心这个分布。对于 S 的分布，我们有：

$$F_S(x) = P(S \leq x) = \sum_{n=0}^{\infty} P(S \leq x | N = n) p_n = \sum_{n=0}^{\infty} F_X^{*n}(x) p_n \tag{4.2}$$

其中， $p_n = P(N = n)$ 是频率分布， $F_X(x)$ 是强度分布， $F_X^{*n}(x)$ 是强度分布的 n 重卷积。如果随机变量 X 仅在 0，1，2…取值，那么 n 重卷积的计算方法如下：

$$F_X^{*n}(x) = \begin{cases} I(x \geq 0) & n = 0 \\ F_X(x) & n = 1 \\ \sum_{y=0}^{x} F_X^{*(n-1)}(x-y) f_X(y) & n = 2, 3, \ldots \end{cases} \tag{4.3}$$

4.2 连续分布的离散化

为什么要对连续分布进行离散化？通常我们假设索赔强度分布是连续分布，如果要得到总损失 S 的分布，根据式(4.2)，需要求得 X 的 n 充卷积，这就需要进行多重积分，如果被积函数的形式比较复杂，那么这将变成一个十分艰难的工作。因此实际操作中通常对连续的索赔强度分布进行离散化处理，采用数值迭代方法计算总损失额的分布，这样就能在保证足够精度的前提下显著提高计算速度。从某种程度上来将，对索赔强度的离散化更加接近实际，因为损失

额度通常是货币单位的整数倍。

所谓离散化就是将连续分布的支集区域划分为若干小区域，然后以这个区域中的某一个点代替原来连续分布在这片小区域的取值概率。这个“代表点”可以是区域的左右端点，也可以是区域中点。此外，我们通常只对分布的“主体”进行离散化。什么叫做分布的“主体”？以正态分布为例，其分布的支集为 $(-\infty,\infty)$，显然不可能对其全部取值范围进行离散化，由于正态分布在两侧的取值概率很小，可以忽略不计，我们于是可以以均值为中心，以若干倍标准差为半径划定一个区域，在这个区域上进行离散化，这个区域上的分布函数就是该分布的“主体”，区域的大小则依赖于研究的精确程度。定义 $F(x)$ 为连续分布函数， f_x 为离散化后的概率函数。目前，actuar 包中的 discretize 函数支持四种离散化方法。

1) 上端离散化，或者说对 $F(x)$ 向前微分。

$$f_x = F(x+h) - F(x) \tag{4.4}$$

对于 $x = a, a+h, \ldots, b-h$，离散化后的 cdf 总是在原 cdf 之上。

2) 下端离散化，或者说对 $F(x)$ 向后微分。

$$f_x = \begin{cases} F(a) & x = a \\ F(x) - F(x-h) & x = a+h, \ldots, b \end{cases} \tag{4.5}$$

离散化后的 cdf 总是在原 cdf 之下。

3) 中点离散化。

$$f_x = \begin{cases} F(a+h/2) & x = a \\ F(x+h/2) - F(x-h/2) & x = a+h, \ldots, b-h \end{cases} \tag{4.6}$$

原 cdf 正好从中间穿过离散化后的 cdf。

4) 无偏离散化，或者说是局部一阶矩匹配。

$$f_x = \begin{cases} \frac{E(X\wedge a) - E(X\wedge a+h)}{h} + 1 - F(a) & x = a \\ \frac{2E(X\wedge x) - E(X\wedge x-h) - E(X\wedge x+h)}{h} & a < x < b \\ \frac{E(X\wedge b) - E[X\wedge b-h]}{h} - 1 + F(b) & x = b \end{cases} \tag{4.7}$$

离散后的分布和原分布在区间 $[a,b]$ 内有相同的取值概率和期望值。

discretize 函数返回的是一串 f_x 概率值，如果要对其进行做图需要进行特殊处理。

函数语法：

```
discretize(cdf, from, to, step = 1, method = c("upper", "lower",
"rounding","unbiased"), lev, by = step, xlim = NULL)
```

使用说明：

1. cdf 必须是一个含 x 的表达式。
2. from 和 to 分别指定 a 和 b，也就是分布主体的范围，step 指定步长 h。
3. lev 只在 method=“unbiased”时才指定。
4. by 和 xlim 与前面参数等价，详见帮助文档。

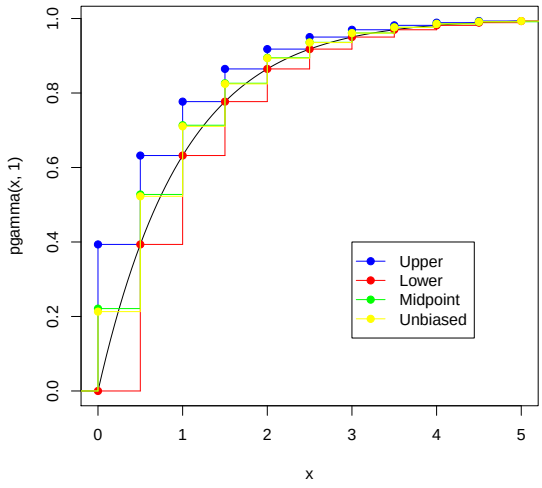
例子：

假设要对分布 $Gamma(1,1)$ 进行离散化。fu，fl，fr 和 fb 分别对应上端离散化，下端离散化，中点离散化和无偏离散化。

```
> fu = discretize(pgamma(x, 1), method = "upper", from = 0, to = 5,
+     step = 0.5)
> fl = discretize(pgamma(x, 1), method = "lower", from = 0, to = 5,
+     step = 0.5)
> fr = discretize(pgamma(x, 1), method = "rounding", from = 0,
+     to = 5, step = 0.5)
> fb = discretize(pgamma(x, 1), method = "unbiased", lev = levgamma(x,
+     1), from = 0, to = 5, step = 0.5)
```

作出离散化后的函数图像。函数 stepfun 返回一个阶梯函数，diffinv 是差分的逆，具体使用方法参见在线帮助。

```
> curve(pgamma(x, 1), xlim = c(0, 5))
> x = seq(0, 5, 0.5)
> plot(stepfun(head(x, -1), diffinv(fu)), pch = 19, , col = "blue",
+     add = TRUE)
> plot(stepfun(x, diffinv(fl)), pch = 19, , col = "red", add = TRUE)
> plot(stepfun(head(x, -1), diffinv(fr)), pch = 19, , col = "green",
+     add = TRUE)
> plot(stepfun(x, diffinv(fb)), pch = 19, , col = "yellow", add = TRUE)
> legend(3, 0.4, legend = c("Upper", "Lower", "Midpoint", "Unbiased"),
+     col = c("blue", "red", "green", "yellow"), pch = 19, lty = 1)
```



4.3 复合分布的计算

当我们对索赔强度分布离散化后，我们可以使用一些算法计算复合分布，当然有些方法并不需要对索赔强度的分布进行离散化而只是利用其前 k 阶矩的信息。函数 aggregateDist 提供了五种方法，下面仅进行简单介绍，对细节感兴趣的读者可以参考帮助文档：

1)Panjer 递推法。索赔频率分布仅支持 $(a,b,0)$ 分布族和 $(a,b,1)$ 分布族[1]，索赔强度分布需要输入离散后的分布。

2) 卷积法。使用(4.2)和(4.3)进行计算，频率分布可以是任何离散分布，强度分布需要输入离散化后的分布。这种方法只能解决小型问题。

3) 正态近似法。给定频率分布和强度分布就可以利用公式

$$\mu_S = E(S) = E(N) * E(X) \tag{4.8}$$

$$\sigma_S^2 = Var(S) = E(N)Var(X) + Var(N)E(X)^2 \tag{4.9}$$

计算复合分布的均值和方差，再利用

$$F_S(x) \approx \Phi(\frac{x-\mu_S}{\sigma_S}) \tag{4.10}$$

计算复合分布的分布函数。这种方法仅利用了前二阶矩的信息，对于大样本近似效果较好。

4) 正态幂近似法 (Normal Power approximation)

$$F_S(x) = \Phi(-\frac{3}{\gamma_S} + \sqrt{\frac{9}{\gamma_S^2} + 1 + \frac{6}{\gamma_S}\frac{x-\mu_S}{\sigma_S}}) \tag{4.11}$$

其中 γ_S 是偏度系数。正态幂近似法的原理是对标准化后的随机变量 S 泰勒展开为标准正态随机变量及其 2 次幂的线性组合，也就是令 $S \approx g(Y)$， Y 服从标准正态分布。这种近似法在 $x > \mu_S$ 时可以进行，在 $\gamma_S < 1$ 时效果较好。因此可以对分布的右尾进行近似。

5) 随机模拟法。通过生成 N 和 X 的随机数并计算得到 S，再用 S 的经验分布近似 $F_S(x)$。这种方法通过调用 simul 函数 (后面会详细讲) 对频率分布和强度分布进行模拟，适用于复杂系统的建模。

aggregateDist 的参数依据所选方法的不同而不同。从下面函数的语法可以看出，该函数是先定方法，再选参数，因此在使用说明中，只介绍和此种方法有关的参数。

函数语法：

```
aggregateDist(method = c("recursive", "convolution", "normal", "npower",
"simulation"), model.freq = NULL, model.sev = NULL, p0 = NULL, x.scale = 1,
moments, nb.simul, ..., tol = 1e-06, maxit = 500, echo = FALSE)
```

使用说明：

1. 递推法中：method=“recursive”，model.freq 必须是”binomial”, ”geometric”, ”negative binomial”, ”poisson” 中的一种，分布参数可以在省略号处指定，但是参数名称必须要与这四个分布规定的一致。model.sev 是一个向量，向量的每个元素依次是 X 取 0，1，2... 个货币单位的概率，注意这个向量的第一个元素必须是 X 取 0 的概率，如果 X 不能取到某个值 (比如 2)，那么向量的对应位置 (这里是第三个元素) 一定要写成 0，通常 model.sev 可以直接调用函数 discretize 的结果。P0 是频数分布在 $N = 0$ 时的概率，也就是 $(a,b,0)$ 分布族零调整的概率。x.scale 指定 X 的货币单位，比如 1 元，100 元等。

[1] $(a,b,0)$ 分布族和 $(a,b,1)$ 分布族是一类满足递推关系分布的总称，包括泊松，二项，几何和负二项分布及其在 0 点概率调整后的分布，对该分布族的具体说明可以参阅 [5]。

2. 卷积法中：method=”convolution”，model.freq 是一个向量，向量的每个元素依次是 N 取 0，1，2... 的概率，第一个元素必须是 N 取 0 的概率。model.sev 的使用方法与递推法相同。x.scale 指定 X 的货币单位，比如 1 元，100 元等。

3. 正态近似法和正态幂近似法：method=”normal” 或”npower”，moments 是一个向量，分别是 S 的均值、方差和偏度系数，moments=c(μ_S， σ_S^2， γ_S)，其中正态近似只需指定前两个元素即可。注意这种近似法需要满足 $x > \mu_S$ 且 $\gamma_S < 1$，否则会发生报错。

4. 模拟法中：method=”simulation”，具体使用方法可以参考后文中对 simul 函数的介绍。nb.simul 是模拟的次数。

5. 函数返回的是一个 aggregateDist 对象，可以对其进行五数总括 (summary)，输出结果 (print)，求均值 (mean)，求分位数 (quantile)，做图 (plot), 求节点 (konts) 等操作。

例子：

首先是卷积法，假设：强度分布，货币单位数 X 从 0 到 10，频率分布， N 从 0 到 8，货币单位数设为 25。通过作图观察 S 的最大值是不是 $10 \times 25 \times 8 = 2000$?

```
> par(mfrow = c(2, 2))
> fx1 = c(0, 0.15, 0.2, 0.25, 0.125, 0.075, 0.05, 0.05, 0.05, 0.025,
+     0.025)
> pn1 = c(0.05, 0.1, 0.15, 0.2, 0.25, 0.15, 0.06, 0.03, 0.01)
> Fs1 = aggregateDist("convolution", model.freq = pn1, model.sev = fx1,
+     x.scale = 25)
> plot(Fs1)
```

递推法，首先对 Gamma 分布进行离散化得到强度分布，频数分布选用 poisson 分布, 特别指定 poisson 分布参数 lambda=10。

```
> fx2 = discretize(pgamma(x, 2, 1), from = 0, to = 22, step = 0.5,
+     method = "unbiased", lev = levgamma(x, 2, 1))
> Fs2 = aggregateDist("recursive", model.freq = "poisson", model.sev = fx2,
+     lambda = 10, x.scale = 0.5)
> plot(Fs2)
```

正态近似和正态幂近似，注意正态幂近似的有效范围。

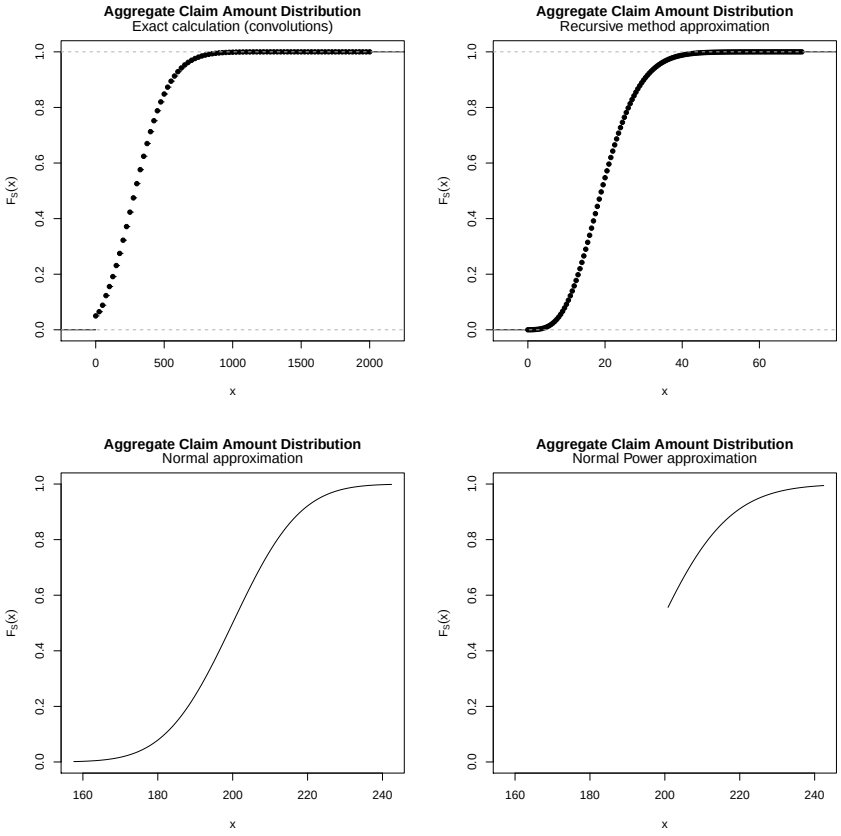
```
> Fs3 = aggregateDist("normal", moments = c(200, 200))
> plot(Fs3)
> Fs4 = aggregateDist("npower", moments = c(200, 200, 0.5))
> plot(Fs4)
```

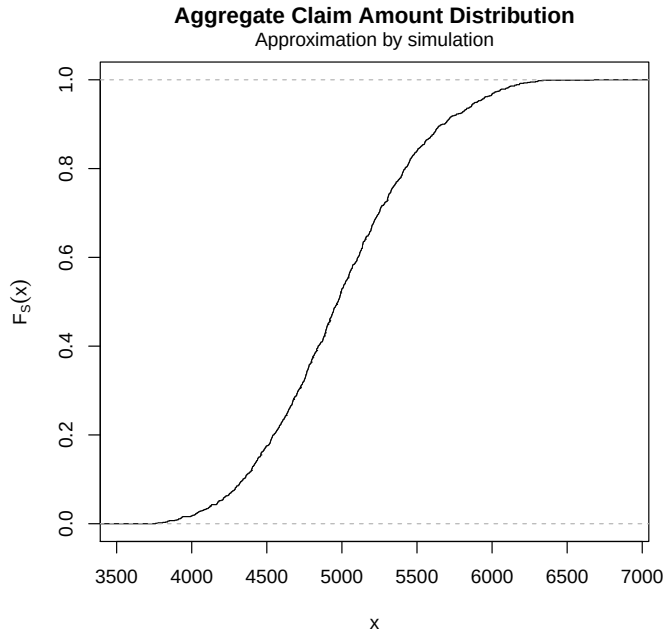
模拟法。

```
> model.freq = expression(data = rpois(100))
> model.sev = expression(data = rgamma(100, 2))
> Fs5 = aggregateDist("simulation", nb.simul = 1000, model.freq,
+     model.sev)
> summary(Fs5)
```

```
Aggregate Claim Amount Empirical CDF:
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
   3759    4637    4974    4992    5327    6675
```

```
> plot(Fs5)
```





4.4 VaR 和 TVaR

计算出复合分布 (也就是总索赔额的分布) 后，我们可以求得该分布的 VaR 和 TVaR，VaR 和 TVaR 是常见的风险度量指标，是衡量金融机构偿付能力的重要手段，其定义分别为：

VaR(value-at-risk):

$$\mathrm{VaR}_{\mathrm{p}} = \inf\{x : P(S > x) \leq p\} \tag{4.12}$$

TVaR(tail-value-at-risk):

$$\mathrm{TVaR}_{\mathrm{p}} = E(S|S > \mathrm{VaR}_{\mathrm{p}}) \tag{4.13}$$

函数 VaR 和 TVaR 用来计算这两个指标，TVaR 也可写作 CTE(Conditional Tail Expectation)，这两个函数是等价的。

函数语法：

```
VaR(x, conf.level = c(0.9, 0.95, 0.99), names = TRUE, ...)
TVaR(x, conf.level = c(0.9, 0.95, 0.99), names = TRUE, ...)
```

使用说明：

1. x 目前仅支持 aggregateDist 对象，依据计算复合分布时采用的不同方法，计算 VaR 和 TVaR 的方法也有所区别，具体请参阅帮助文档。
2. conf.level 指定置信水平，默认值是 0.9，0.95 和 0.99。
3. names 控制输出是否包含名称，默认是 TURE。

例子：

考虑上文模拟法得到的结果 Fs5，首先计算其 VaR。

```
> VaR(Fs5, names = FALSE)

[1] 5668 5901 6183
```

然后计算其 98% 的 TVaR。

```
> TVaR(Fs5, conf.level = 0.98)

 98%
6214
```

Chapter 5

保单组合的模拟

5.1 复合层次模型的模拟

在信度理论的研究中，方法的比较通常需要数据去支持，数据一般有两个来源：实际数据和模拟数据。实际数据如果拿得到当然好，但当实际数据不足够或者无法获得时，常常需要对损失数据进行模拟。

根据(4.1)可知，如果要得到某个保单组合在一段时期内的总赔付额，需要同时模拟索赔频率分布和索赔强度分布。回到上文模拟复合分布的例子，如果认为某保单组合 2010 年赔付的次数服从均值为 100 的 Poisson 分布，每次赔付的赔付额服从参数为 $(100, 2)$ 的 Gamma 分布。在一次模拟中，可以首先生成一个 Poisson 随机数 N_0，然后再生成 N_0 个 Gamma 分布随机数，加总起来，就得到一个总赔付额的模拟值。如此模拟 1000 次，就可以得到总赔付额的经验分布。

上面的简单例子说明，如果要模拟一个保单组合的总赔付额，我们首先要模拟索赔频率分布，然后根据模拟出的索赔频率，再对每一次索赔模拟出索赔额，也就是索赔强度。但是有时候，无论是索赔频率还是索赔强度都会受到客观环境 (外生变量) 的影响，比如不同的风险类别 (class)，同一类别中的不同保单 (contract)，以及同一保单不同的事故年度 (year)，这些外生变量通过决定分布的参数进而影响分布。因此索赔频率和索赔强度的模拟可以通过复合层次模型 (compound hierarchical model) 来完成。

例如，考虑如下三层复合层次模型：

$$S_{ijt} = X_{ijt1} + \cdots + X_{ijtN_{ijt}} \tag{5.1}$$

其中 $i = 1, \ldots, I$ ， $j = 1, \ldots, J_i$， $t = 1, \ldots, n_{ij}$，同时有

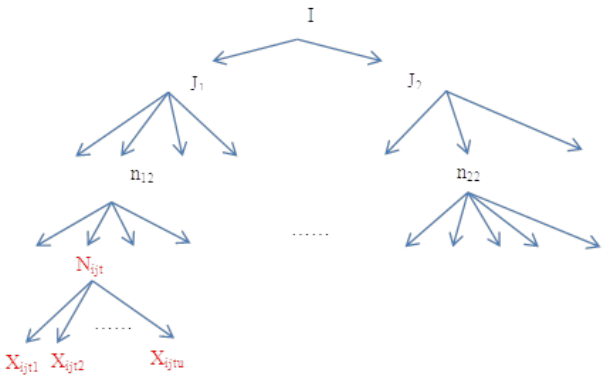
$$\begin{aligned} N_{ijt}|\Lambda_{ij}, \Phi_i &\sim Poisson(w_{ijt}\Lambda_{ij}) \\ \Lambda_{ij}|\Phi_i &\sim Gamma(\Phi_i, 1) \\ \Phi_i &\sim Exp(2) \end{aligned} \tag{5.2}$$

$$\begin{aligned} X_{ijtu}|\Theta_{ij}, \Psi_i &\sim Lognormal(\Theta_{ij}, 1) \\ \Theta_{ij}|\Psi_i &\sim N(\Psi_i, 1) \\ \Psi_i &\sim N(2, 0.1). \end{aligned} \tag{5.3}$$

随机变量 Φ_i， Λ_{ij}， Ψ_i 和 Θ_{ij} 在精算文献中通常被称作风险参数； w_{ijt} 是先验的权重。用以损失频率 N_{ijt} 的分布为例， Φ_i 是第一层分布，在 Φ_i 确定后，通过参数的传递，可以确定第二层分布 $\Lambda_{ij}|\Phi_i$，当 Φ_i 和 $\Lambda_{ij}|\Phi_i$ 都确定后，就可以得到第三层分布也就是 N_{ijt} 的分布。

在这里需要对权重 w_{ijt} 进行特别说明。它反映了保险合同的风险基础 (exposure base) 的大小，风险基础是非寿险精算的一个基本概念，其大小通常用风险单位数来衡量。举例来说：考虑承保一辆汽车，保单期限为一年，那么年初投保的话这辆汽车在整个一年内都将面临着发生事故的风险，因此风险单位数为 1 车年，如果是在年中投保，那么截止至年末的评估日，风险单位数只有 0.5 车年；考虑每份保单承保一个车队，年初投保，保单期限也为一年，那么车队的规模越大，其平均发生的损失次数也越多，如果某年车队的车辆数为 100，其风险单位数就等于 100 车年。在模型(5.2)中，索赔频率分布服从 Poisson 分布，其参数的形式是 $w_{ijt}\Lambda_{ij}$， w_{ijt} 反映了在特定的风险类别下 (i,j) 各年 (t) 风险单位数的大小， Λ_{ij} 是不同风险类别 (i,j) 所导致的对基本风险单位的调整因子。风险单位数越大，调整因子越大，那么对应的 Poisson 分布的均值参数也越大。

通常， I， J， t 都是已经确定好的数值，比如令 $I=2$， $J_1=4$， $J_2=3$， $n_{11}=\ldots=n_{14}=4$， $n_{21}=n_{22}=n_{23}=5$，如果我们把 I， J， t 分别与前面的风险类别、保单和事故年度相对应，那么上面的赋值就可以解读为 (参见下面的树形示意图)：我们要模拟一个保单组合，这个组合中包含 2 个风险类别，类别 1 中有 4 份保单，类别 2 中有 3 份保单，类别 1 中的每份保单保障时间都为 4 年，类别 2 中的每份保单的保障年度都为 5 年，因此我们总共需要模拟 $4\times4+3\times5=31$ 次索赔频率 (每份保单每一年都要进行模拟)，而且对于每一次索赔，还要模拟该次索赔的索赔强度。索赔频率和强度的模拟可以分别使用模型(5.2)和(5.3)。在下图中，标红字的部分属于对损失数据实际模拟的过程，并不包含在层次模型的结构范畴内，画出来只是为了帮助读者对整个流程有一个总体的把握。如果仅看示意图的上半部分，也就是模型部分，每个分叉处我们称之为节点 (nodes)，也就是对应的 I， J， t，所谓层次模型就是通过对参数的层层递归，从高层逐步将随机生成的参数传递到低层，来反映风险类别、保单和事故年度的影响。



函数 simpf(simulation portfolio，也可以写作 simul，两个函数等价) 可以对复合层次模型模拟。关于复合层次模型及其模拟的更详细的介绍，请参阅文献 [4]。

函数语法：

```
simul(nodes, model.freq = NULL, model.sev = NULL, weights = NULL)
```

或者

```
simpf(nodes, model.freq = NULL, model.sev = NULL, weights = NULL)
```

使用说明：

1. nodes 是一个 list 对象，组成该 list 对象的每一个元素是一个数值向量，每个数值向量指定在该层次下的节点数，需要按由高层到低层的顺序指定，比如 I， J， t 的顺序。

2. model.freq 和 model.sev 指定每一层的频数和频率的随机分布。随机分布的指定采用用 R 中的表达式 (expression) 形式，且要与 R 中生成随机数的语法保持一致 (比如 rgamma)，只需要给参数赋值，而不需要指定需要生成随机数的个数。

3. weights 指定权重，按照 I， J， t 的顺序依次指定。

例子：

如果你觉得上面说的还是不太明白，那就看一下这个例子。还是考虑上面的复合层次模型，首先指定各节点处 I， J， t 的取值。

```
> nodes = list(class = 2, contract = c(4, 3), year = c(4, 4, 4,
+     4, 5, 5, 5))
```

接着指定权重：类别 1($I = 1$) 的第一份保单 ($J = 1$) 的各年权重 ($n_{11} = 4$) 是一个长度为 4 的向量，紧接着是类别 1($I = 1$) 的第二份保单 ($J = 2$) 的各年权重 ($n_{12} = 4$)，也是一个长度为 4 的向量，依此类推，每张保单每一年有一个权重，因此共需要 31 个。简便起见，权重采用产生随机数的方法进行赋值，服从 0.5 到 2.5 之间的均匀分布。

```
> set.seed(1)
> wijt = runif(31, 0.5, 2.5)
```

索赔频率分布的模型。其中 class 对应Φ_i，contract 对应$\Lambda_{ij}|\Phi_i$，year 对应$N_{ijt}|\Lambda_{ij}, \Phi_i$。我们可以对参数进行更为灵活的指定，比如对参数进行数学变换 contract = rgamma(log(class), 1)，或者跨层调用 year = rpois(weights * contract * class)。

```
> mf = expression(class = rexp(2), contract = rgamma(class, 1),
+     year = rpois(weights * contract))
```

索赔强度分布的模型。

```
> ms = expression(class = rnorm(2, sqrt(0.1)), contract = rnorm(class,
+     1), year = rlnorm(contract, 1))
```

对构建好的复合层次模型进行实际模拟。

```
> set.seed(9)
> pf = simpf(nodes = nodes, model.freq = mf, model.sev = ms, weights = wijt)
> pf

Portfolio of claim amounts

  Frequency model
    class    ~ rexp(2)
    contract ~ rgamma(class, 1)
    year     ~ rpois(weights * contract)
  Severity model
    class    ~ rnorm(2, sqrt(0.1))
```

```
  contract ~ rnorm(class, 1)
  year     ~ rlnorm(contract, 1)

Number of claims per node:
```

	class	contract	year.1	year.2	year.3	year.4	year.5
[1,]	1	1	0	0	1	1	NA
[2,]	1	2	0	2	3	0	NA
[3,]	1	3	1	1	0	1	NA
[4,]	1	4	4	3	4	1	NA
[5,]	2	1	1	2	1	0	4
[6,]	2	2	1	5	1	3	5
[7,]	2	3	0	0	2	2	0

函数 simpf 将模拟好的结果对象存储在 pf 中，默认输出索赔频率和索赔强度的模型，以及一份保单在某一年的发生的索赔案件数量。比如，上面结果中第二行第四列的数字表示在类别 1 的第二份保单，其第二年发生的案件数为 2。

5.2 模拟结果的处理

在损失数据模拟好之后，我们可以通过函数 aggregate, frequency，severity 和 weights 来调用和处理模拟结果。

函数语法：

```
aggregate(x,by = names(x$nodes), FUN = sum, classification = TRUE,
prefix = NULL, ...)
frequency(x, by = names(x$nodes),classification = TRUE, prefix = NULL, ...)
severity(x, by = head(names(x$node), -1), splitcol = NULL,
classification = TRUE, prefix = NULL, ...)
weights(object, classification = TRUE, prefix = NULL, ...)
```

使用说明：

1. x 是一个 portfolio 对象，通常是 simul 的返回值。
2. by 是汇总依据，是层次模型不同层次的名称，如 class，contract 或 year 等，可以一次指定多个。
3. FUN 是汇总方式，默认是求和 (sum)，可以改成计数 (length)，求平均 (mean)，中位数 (median)，最大值 (max)，最小值 (min) 等。
4. classification 控制是否输出分类指标，默认是输出 (TRUE)。
5. prefix 为被汇总列添加前缀。
6. splitcol 提取某一年的赔付次数，具体使用方法见例子。

例子：

默认情况下，函数 aggregate 返回某保单在某一年的总索赔数额，也就是 S_{ijt}。aggregate 的原理类似于 Excel 中的数据透视表。

```
> aggregate(pf)
```

```
     class contract  year.1  year.2  year.3  year.4 year.5
[1,]     1        1   0.000   0.000  21.221   6.741     NA
[2,]     1        2   0.000   6.882  17.247   0.000     NA
[3,]     1        3   5.251   3.303   0.000   3.463     NA
[4,]     1        4   9.908  31.799  20.537   9.576     NA
[5,]     2        1   1.281   3.684   3.542   0.000  119.3
[6,]     2        2 601.352 860.844 262.749 219.635  536.0
[7,]     2        3   0.000   0.000  86.867  11.901    0.0
```

按保单类别和保单年度分别进行汇总，返回某类别保单在某一年的平均索赔额度。比如，第一个类别总共有四张保单，其中第三张保单在第一年发生一次索赔，总的赔付额为 5.251，第四张保单在第一年发生四次索赔，总的赔付额为 9.908。因此第一类别第一年度平均赔付 =(9.908+5.251)/(1+4)=3.032。

```
> aggregate(pf, by = c("class", "year"), FUN = mean)
```

```
     class  year.1  year.2 year.3 year.4 year.5
[1,]     1   3.032   6.997  7.376  6.593     NA
[2,]     2 301.317 123.504 88.289 46.307  72.82
```

函数 frequency 返回某份保单在某一年的发生的索赔案件数量，也就是 simpf 默认输出的第二部分。该函数是 aggregate 的一个封装，当 aggregate 中的 FUN=length 时，两者是等价的。同样，我们可以通过 by 参数进行分门别类的汇总。

```
> frequency(pf, prefix = "freq.")
```

```
     class contract freq.year.1 freq.year.2 freq.year.3 freq.year.4 freq.year.5
[1,]     1        1           0           0           1           1          NA
[2,]     1        2           0           2           3           0          NA
[3,]     1        3           1           1           0           1          NA
[4,]     1        4           4           3           4           1          NA
[5,]     2        1           1           2           1           0           4
[6,]     2        2           1           5           1           3           5
[7,]     2        3           0           0           2           2           0
```

```
> frequency(pf, by = "class")
```

```
     class freq
[1,]     1   22
[2,]     2   27
```

最后，函数 severity(pf) 返回每次赔付的索赔数额，也就是式(5.1)中的 X_{ijtu}。由于某个类别的某张保单，在保障时期内可能有多次索赔，因此索赔的数量可能会多于保障时期数。而由于返回数据的组织形式是一个矩阵，矩阵的每一行代表一张保单，矩阵的每一列代表这张保单所发生的一次索赔。因此矩阵的列数等于

$$\max_{i,j} \sum_{t=1}^{n_{ij}} N_{ijt} \tag{5.4}$$

也就是说等于某张保单在所有年度可能发生的最大累计索赔次数。当某张保单的累计索赔次数小于最大次数时，空缺的数据用 NA 表示。

```
> severity(pf)

$main
     class contract claim.1 claim.2 claim.3 claim.4 claim.5 claim.6 claim.7
[1,]     1        1  21.221   6.741      NA      NA      NA      NA      NA
[2,]     1        2   4.946   1.936  8.1365   2.132   6.979      NA      NA
[3,]     1        3   5.251   3.303  3.4629      NA      NA      NA      NA
[4,]     1        4   4.848   3.050  0.6451   1.364  12.054   14.66   5.082
[5,]     2        1   1.281   2.536  1.1486   3.542   4.424   27.15  44.535
[6,]     2        2 601.352 227.828 56.5580  29.521 277.311  269.63 262.749
[7,]     2        3   8.108  78.759  8.3159   3.585      NA      NA      NA
     claim.8 claim.9 claim.10 claim.11 claim.12 claim.13 claim.14 claim.15
[1,]      NA      NA       NA       NA       NA       NA       NA       NA
[2,]      NA      NA       NA       NA       NA       NA       NA       NA
[3,]      NA      NA       NA       NA       NA       NA       NA       NA
[4,]   2.462   1.068     8.56    8.447    9.576       NA       NA       NA
[5,]  43.197      NA       NA       NA       NA       NA       NA       NA
[6,] 132.841  43.183    43.61   61.725   21.574    347.4    36.19    69.15
[7,]      NA      NA       NA       NA       NA       NA       NA       NA

$split
NULL
```

可以看出，函数将每次赔付依次列出，但却丢失了索赔发生时间的信息。可以通过设置参数 splitcol 来提取出发生在某一年的赔付案件。比如我们要把第一个事故年度发生的赔案提取出来，结果保存在 $split 处，剩余结果保存在 $main 里。从输出的结果可以看出，在第一个保单年度，第一类别的第四张保单有四次赔付，而第二类别的第三张保单没有任何赔付。

```
> severity(pf, splitcol = 1)

$main
     class contract claim.1 claim.2 claim.3 claim.4 claim.5 claim.6 claim.7
[1,]     1        1  21.221   6.741      NA      NA      NA      NA      NA
[2,]     1        2   4.946   1.936   8.136   2.132   6.979      NA      NA
[3,]     1        3   3.303   3.463      NA      NA      NA      NA      NA
[4,]     1        4  12.054  14.663   5.082   2.462   1.068    8.56   8.447
[5,]     2        1   2.536   1.149   3.542   4.424  27.153   44.54  43.197
[6,]     2        2 227.828  56.558  29.521 277.311 269.626  262.75 132.841
[7,]     2        3   8.108  78.759   8.316   3.585      NA      NA      NA
     claim.8 claim.9 claim.10 claim.11 claim.12 claim.13 claim.14
[1,]      NA      NA       NA       NA       NA       NA       NA
[2,]      NA      NA       NA       NA       NA       NA       NA
[3,]      NA      NA       NA       NA       NA       NA       NA
[4,]   9.576      NA       NA       NA       NA       NA       NA
```

```
[5,]      NA      NA       NA       NA       NA       NA       NA
[6,]  43.183   43.61    61.72    21.57    347.4    36.19    69.15
[7,]      NA      NA       NA       NA       NA       NA       NA

$split
     class contract claim.1 claim.2 claim.3 claim.4
[1,]     1        1      NA      NA      NA      NA
[2,]     1        2      NA      NA      NA      NA
[3,]     1        3   5.251      NA      NA      NA
[4,]     1        4   4.848   3.050  0.6451   1.364
[5,]     2        1   1.281      NA      NA      NA
[6,]     2        2 601.352      NA      NA      NA
[7,]     2        3      NA      NA      NA      NA
```

把前两个事故年度发生的赔案提取出来，在 $split 处，剩余结果保存 $main 里。

```
> severity(pf, splitcol = c(1, 2))

$main
     class contract claim.1 claim.2 claim.3 claim.4 claim.5 claim.6 claim.7
[1,]     1        1  21.221   6.741      NA      NA      NA      NA      NA
[2,]     1        2   8.136   2.132   6.979      NA      NA      NA      NA
[3,]     1        3   3.463      NA      NA      NA      NA      NA      NA
[4,]     1        4   2.462   1.068   8.559   8.447   9.576      NA      NA
[5,]     2        1   3.542   4.424  27.153  44.535  43.197      NA      NA
[6,]     2        2 262.749 132.841  43.183  43.612  61.725   21.57   347.4
[7,]     2        3   8.108  78.759   8.316   3.585      NA      NA      NA
     claim.8 claim.9
[1,]      NA      NA
[2,]      NA      NA
[3,]      NA      NA
[4,]      NA      NA
[5,]      NA      NA
[6,]   36.19   69.15
[7,]      NA      NA

$split
     class contract claim.1 claim.2 claim.3 claim.4 claim.5 claim.6 claim.7
[1,]     1        1      NA      NA      NA      NA      NA      NA      NA
[2,]     1        2   4.946   1.936      NA      NA      NA      NA      NA
[3,]     1        3   5.251   3.303      NA      NA      NA      NA      NA
[4,]     1        4   4.848   3.050  0.6451   1.364   12.05   14.66   5.082
[5,]     2        1   1.281   2.536  1.1486      NA      NA      NA      NA
[6,]     2        2 601.352 227.828 56.5580  29.521  277.31  269.63      NA
[7,]     2        3      NA      NA      NA      NA      NA      NA      NA
```

函数 weights 返回每份保单在每个年度的权重，通过和 wijt 进行对比可以更好的理解权重

赋值的顺序。

```
> weights(pf)
```

```
     class contract year.1 year.2 year.3 year.4 year.5
[1,]     1        1 1.0310 1.2442 1.6457 2.3164     NA
[2,]     1        2 0.9034 2.2968 2.3894 1.8216     NA
[3,]     1        3 1.7582 0.6236 0.9119 0.8531     NA
[4,]     1        4 1.8740 1.2682 2.0397 1.4954     NA
[5,]     2        1 1.9352 2.4838 1.2601 2.0549  2.369
[6,]     2        2 0.9243 1.8033 0.7511 1.0344  1.272
[7,]     2        3 0.5268 1.2648 2.2394 1.1807  1.464
```

如果令 classification=FALSE，可以很容易的计算出纯费率，纯费率被定义为保险公司对每一风险单位的平均赔款金额，在本例中我们认为实际损失额就等于实际赔款金额。

```
> aggregate(pf, classification = FALSE, )/weights(pf, classification = FALSE)
```

```
      year.1  year.2  year.3  year.4 year.5
[1,]   0.000   0.000  12.895   2.910     NA
[2,]   0.000   2.996   7.218   0.000     NA
[3,]   2.987   5.298   0.000   4.059     NA
[4,]   5.287  25.074  10.069   6.404     NA
[5,]   0.662   1.483   2.811   0.000  50.35
[6,] 650.613 477.359 349.814 212.322 421.33
[7,]   0.000   0.000  38.791  10.080   0.00
```

Chapter 6

信度理论

6.1 信度理论简述

在保险实践中，通常需要对风险类别进行划分，并按照风险的高低收取不同的费率，比如我国的交强险就将车辆类别和使用性质作为费率因子。风险分类的过程潜在假设了同一类别内的风险是同质的，但是没有任何费率厘定系统是完美的，即便两个风险属于同一个类别，它们在风险水平上也会具有异质性，这种异质性来源于我们没有考虑到的风险因素，还以交强险为例，两辆商用性质的货车可能因为司机年龄或行驶区域的不同具有相异的风险水平。

考虑在某一个风险类别中，如果一个新投保的保单没有任何先验的损失经验，那么只能对其收取手册费率 M，手册费率反映了该风险类别内全部保单组合的平均赔付水平；如果该保单有若干年的损失经验，那么根据该损失经验是否高于或低于手册费率 M 可以适当增加或降低费率，以实现对所有投保人的公平。但是，过去的损失经验具有随机性，较大的损失经验可能是由于随机波动造成的，自然而然地，我们就想到了令费率等于过去损失经验与手册费率的的某种加权。所谓*信度理论*就是一系列数量方法，使得保险人可以根据保单以往的损失经验，对其进行定价，使得保费在不同风险水平的投保人之间重新进行分配，所得到的保费称作信度保费。数学上，可以表示为：

$$\pi = Z \cdot \bar{X} + (1 - Z) \cdot M \tag{6.1}$$

其中π 是信度保费， Z 是信度因子， $\bar{X}$ 是过去损失经验的平均，M 是手册费率。信度因子 Z 反映了过去损失数据的可信程度，一方面我们要考虑数据量的大小，另一方面要考虑损失数据波动的大小。显然，数据量越大，损失波动越小，损失经验就越可信。

假设对于某份保单，我们已经具有前 n 期的损失数据 $X_1, X_2, \ldots, X_n$，那么 π 就是要预测的第 $n+1$ 年的保费。事实上，从统计学的角度讲， $\bar{X}$ 作为前 n 期数据的样本均值，是第 $n+1$ 期保费的无偏估计，而信度理论告诉我们最优的估计是对样本均值和手册费率的一个加权，是第 $n+1$ 年保费的一个*有偏估计*。当然，信度保费是一个线性估计量，是前 n 期损失数据的线性组合；而所谓的“最优”是在最小化平方误差的准则之下的，通过牺牲一定的偏差，显著地降低估计的方差，进而得到较低的平方误差。

信度通常可以分为有限波动信度和最精确信度。有限波动信度建立在传统的统计学的框架内，类似于抽样理论中在一定的相对误差和置信水平下确定样本量的思想，通过对$\bar{X}$ 正态近似，得到“完全可信”条件下所需要的样本量，也就是经验损失 $\bar{X}$ 完全可以作为下一期保费。当数据不满足完全可信条件时，通过平方根准则 $Z = \sqrt{n/n_0}$ 确定信度因子。最精确信度包括 Bühlmann 和 Bühlmann - Straub 模型，通过构造过去经验损失$X_1, X_2, \ldots, X_n$ 的线性组合，在最小化平方误差的准则下得到第 $n+1$ 期保费的线性估计。在估计模型的参数时，需要有多张

保单多期的数据，更为重要的是需要假设损失数据来自于同一个风险类别。关于这两种信度的详细论述可以参阅 [5] 和 [2]，为了更好的理解下面的内容，笔者强烈建议阅读一下这两本书。

6.2 层次信度模型

层次信度模型 (Hierarchical Credibility Model) 是由 Jewell 在 1975 年提出的。在上一小节中，我们提到在计算信度保费时需要假设保单来自同一个风险类别，而层次信度模型可以利用来自于不同风险类别的并行数据，只要这些损失数据具有某种层次结构，4.1 小节就是讲述如何模拟这种损失数据的。事实上，保险公司可以用整个业务类别 (比如火险) 的损失数据，只要这种业务的保单具按不同层次进行划分。层次信度模型可以将保费在来自于具有层次结构的保单组合的不同保单中进行分配，也就得到每张保单的信度保费。此外，拟合层次信度模型时需要测量每个节点保单的同质性，因此该模型也用于评价费率因子制定的好坏。

考虑一个两层的信度模型：在一个保单组合中，损失数据按照不同的类别 (class) 进行划分，每个类别包含不同的合同 (contract)。X_{ijt} 为每风险单位的损失 (该年度总损失 S_{ijt}/w_{ijt})，脚标 $i = 1, \ldots, I$ 表示类别， $j = 1, \ldots, J_i$ 表示第 i 个类别的不同合同， $t = 1, \ldots, n_{ij}$ 表示不同年度的观测。每个观测都对应一个权重 w_{ijt}，表示对应的风险单位数。在 4.1 节，我们介绍了如何模拟具有这种层次结构的数据，不过在那一小节，我们称之为三层 (各年 t 的索赔频率和强度作为一层)。我们分别用随机变量 Φ_i 和 Θ_{ij} 代表不同类别和合同的风险水平，模型有如下假设：

1. 随机变量 $\Phi_1, \ldots, \Phi_I$ 独立同分布。
2. 随机变量 $\Theta_{i1}, \ldots, \Theta_{iJ_i}$ 条件独立，给定 Φ_i。
3. 随机变量 $X_{ij1}, \ldots, X_{ijn_{ij}}$ 条件独立，给定 Φ_i 和 Θ_{ij}。
4. 对于所有 $t, u = 1, \ldots, n_{ij}$，

$$E[X_{ijt}|\Theta_{ij}, \Phi_i] = \mu(\Theta_{ij}, \Phi_i)$$

$$Cov(X_{ijt}, X_{iju}) = \begin{cases} \frac{\sigma^2(\Theta_{ij}, \Phi_i)}{w_{ijt}}, & t = u \\ 0, & t \neq u \end{cases}$$

定义如下保单组合的结构参数，首先是聚合保费：

$$\mu = E[E[\mu(\Theta_{ij}, \Phi_i)|\Phi_i]] \tag{6.2}$$

合同内方差的期望：

$$v = E[E[\sigma^2(\Theta_{ij}, \Phi_i)|\Phi_i]] \tag{6.3}$$

合同间方差 (类别内方差)：

$$a = E[Var[\mu(\Theta_{ij}, \Phi_i)|\Phi_i]] \tag{6.4}$$

类别间方差：

$$b = Var[E[\mu(\Theta_{ij}, \Phi_i)|\Phi_i]] \tag{6.5}$$

可以看出，Bühlmann 和 Bühlmann - Straub 模型是层次信度模型的特例 (层数为 1)。我们的目标就是根据历史的损失经验，估计每份合同的风险保费$\mu(\Theta_{ij}, \Phi_i)$，然而首先需要估计出

$\mu(\Phi_i) = E[\mu(\Theta_{ij}, \Phi_i)|\Phi_i]$。在最小化均方误差的准则下，对应的最优线性估计量为：

$$\hat{\pi}_{ij} = z_{ij}X_{ijw} + (1 - z_{ij})\hat{\pi}_i \tag{6.6}$$

$$\hat{\pi}_i = z_i X_{izw} + (1 - z_i)\hat{\mu} \tag{6.7}$$

其中的信度因子为:

$$z_{ij} = \frac{w_{ij\Sigma}}{w_{ij\Sigma} + v/a}, \qquad w_{ij\Sigma} = \sum_{t=1}^{n_{ij}} w_{ijt}$$

$$z_i = \frac{z_{i\Sigma}}{z_{i\Sigma} + a/b}, \qquad z_{i\Sigma} = \sum_{j=1}^{J_i} z_{ij}$$

数据的加权平均为：

$$X_{ijw} = \sum_{t=1}^{n_{ij}} \frac{w_{ijt}}{w_{ij\Sigma}} X_{ijt}, \qquad X_{izw} = \sum_{j=1}^{J_i} \frac{z_{ij}}{z_{i\Sigma}} X_{ijw}$$

下面分别介绍 μ, v, a, b 的估计方法。首先是聚合保费 μ 的估计量为：

$$\hat{\mu} = X_{zzw} = \sum_{i=1}^{I} \frac{z_i}{z_\Sigma} X_{izw}, \qquad z_\Sigma = \sum_{i=1}^{I} z_i \tag{6.8}$$

v 的估计量 $\hat{v}$ 为：

$$\hat{v} = \frac{1}{\sum_{i=1}^{I}\sum_{j=1}^{J_i}(n_{ij} - 1)} \sum_{i=1}^{I}\sum_{j=1}^{J_i}\sum_{t=1}^{n_{ij}} w_{ijt}(X_{ijt} - X_{ijw})^2 \tag{6.9}$$

有三种估计 a 和 b 的方法，关于这三种估计方法的更详细讨论请参见 [1]：

1) 迭代伪估计量 (iterative pseudo estimator)。

$$\tilde{a} = \frac{1}{\sum_{i=1}^{I}(J_i - 1)} \sum_{i=1}^{I}\sum_{j=1}^{J_i} z_{ij}(X_{ijw} - X_{izw})^2 \tag{6.10}$$

$$\tilde{b} = \frac{1}{I - 1} \sum_{i=1}^{I} \frac{z_i}{z_\Sigma}(X_{izw} - X_{zzw})^2 \tag{6.11}$$

这些估计量看起来简单直观。之所以 $\tilde{a}$ 和 $\tilde{b}$ 称为伪估计量是因为等式右侧的信度因子需要假设 μ, v, a, b 已知，而在估计时这些量只能使用其估计值。方程两边的相互决定的特性也决定了只能使用迭代的算法计算估计量。

2) Bühlmann - Gisler 估计量 (Bühlmann - Gisler estimator)。首先记：

$$A_i = \sum_{j=1}^{J_i} w_{ij\Sigma}(X_{ijw} - X_{iww})^2 - (J_i - 1)v \qquad c_i = w_{i\Sigma\Sigma} - \sum_{j=1}^{J_i} \frac{w_{ij\Sigma}^2}{w_{i\Sigma\Sigma}}$$

$$B = \sum_{i=1}^{I} z_{i\Sigma}(X_{izw} - \bar{X}_{zzw})^2 - (I - 1)a \qquad d = z_{\Sigma\Sigma} - \sum_{i=1}^{I} \frac{z_{i\Sigma}^2}{z_{\Sigma\Sigma}}$$

$$\bar{X}_{zzw} = \sum_{i=1}^{I} \frac{z_{i\Sigma}}{z_{\Sigma\Sigma}} X_{izw} \qquad z_{\Sigma\Sigma} = \sum_{i=1}^{I} z_{i\Sigma} = \sum_{i=1}^{I}\sum_{j=1}^{J_i} z_{ij}$$

注意到 $E(A_i) = c_i a$， $E(B) = db$，因此有 Bühlmann - Gisler 估计量：

$$\hat{a} = \frac{1}{I}\sum_{i=1}^{I} \max(\frac{A_i}{c_i}, 0) \tag{6.12}$$

$$\hat{b} = \max(\frac{B}{d}, 0) \tag{6.13}$$

Bühlmann - Gisler 估计量首先在 0 处截断，然后再取平均，因此是有偏的估计量。

3) Ohlsson 估计量 (Ohlsson estimator)。

$$\hat{a}' = \frac{\sum_{i=1}^{I} A_i}{\sum_{i=1}^{I} c_i} \tag{6.14}$$

$$\hat{b}' = \frac{B}{d} \tag{6.15}$$

与 Bühlmann - Gisler 原理类似，只不过 Ohlsson 估计量将求平均方式改为加权平均。$\hat{a}'$ 和 $\hat{b}'$ 可以为负值，但在实务中通常在 0 处截断，因此也是一个有偏的估计量。Ohlsson 估计量和 Bühlmann - Gisler 估计量原理相似，主要是为了避免迭代估计量复杂的迭代关系，简化了中间节点的计算形式。

在 actuar 包的 cm 函数支持这三种方法拟合层次信度模型，cm 是 credibility model 的缩写，“由于 cm 的工作方式和 R 中的 lm 相似，因此取名叫 cm”，该包的作者在帮助文档和发表的文章中不遗余力地重复着这个冷笑话... cm 是一个通用的信度模型函数，支持 Bühlmann，Bühlmann - Straub，层次信度模型和 Hachemeister 的信度回归模型，本小结将介绍前三种方法，下一小节介绍信度回归模型。

函数语法：

```
cm(formula, data, ratios, weights, subset,regformula = NULL, regdata,
adj.intercept = FALSE,method = c("Buhlmann-Gisler", "Ohlsson", "iterative"),
tol = sqrt(.Machine$double.eps), maxit = 100, echo = FALSE)
```

使用说明：

1. formula 指定层次信度模型，其使用方法类似于 lm 中的公式写法，但是~左边不需指定因变量。~左边按顺序指定分层的因子，比如一个保单组合分为不同的类别 (class)，在每个类别中又区分为不同的合同 (contract)，如果令: 表示两个因子的交互作用，令 + 表示不同项的分割，那么可以令 firmula=~class+class:contract。

2. data 指定保单组合的损失数据，可以是矩阵或数据框。数据列要包括保单组合的结构，各年的每风险单位的损失额 X_{ijt} 和各年风险风险单位数 w_{ijt}(可选，如果拟合 Bühlmann 模型，因为使得权重可以不提供权重信息)。

3. ratios 和 weights 指明哪些列是X_{ijt} 或w_{ijt}，weights 可选。

4. 取观测的子集进行模型拟合，需要输入逻辑表达式，要使该参数有效 data 必须是数据框 (data.frame) 形式。

5. regformula，regdata 和 adj.intercept 是信度回归模型的参数，将在下一小节讲述。

6. method 是 a， b 参数的估计方法，默认是 Buhlmann-Gisler 法。当层数为 1 时，也就是 Bühlmann 和 Bühlmann – Straub 时，三种方法估计结果相同。

7. tol 指定迭代收敛的条件，maxit 指定最大迭代次数，echo 指定是否输出迭代过程，默认是否。

例子：

首先是 Bühlmann – Straub 模型，在这里使用的是 [5] 中例 20.34 的例子。

```
> dat1 = data.frame(plyhder = c(1, 2), ratio.1 = c(NA, 18000/100),
+     ratio.2 = c(10000/50, 21000/110), ratio.3 = c(13000/60, 17000/105),
+     weight.1 = c(NA, 100), weight.2 = c(50, 110), weight.3 = c(60,
+         105))
> fit1 = cm(~plyhder, dat1, ratios = ratio.1:ratio.3, weights = weight.1:weight.3)
> fit1

Call:
cm(formula = ~plyhder, data = dat1, ratios = ratio.1:ratio.3,
    weights = weight.1:weight.3)

Structure Parameters Estimators

  Collective premium: 191.7

  Between plyhder variance: 380.9
  Within plyhder variance: 17831
```

我们可以对输出的模型对象进行提取输出结果 (summary)，预测 (predict) 等操作。输出结果与书中的信度加权重新估计 $\hat{\mu}$ 的结果一致。

```
> summary(fit1)

Call:
cm(formula = ~plyhder, data = dat1, ratios = ratio.1:ratio.3,
    weights = weight.1:weight.3)

Structure Parameters Estimators

  Collective premium: 191.7

  Between plyhder variance: 380.9
  Within plyhder variance: 17831

Detailed premiums

  Level: plyhder
    plyhder Indiv. mean Weight Cred. factor Cred. premium
    1       209.1       110    0.7015       203.9
    2       177.8       315    0.8706       179.6
```

```
> predict(fit1)

    1     2
203.9 179.6
```

如果要拟合 Bühlmann 模型，可以不指定权重，但每张保单的观测年数要相同。

```
> dat2 = data.frame(plyhder = c(1, 2), ratio.1 = c(10000/50, 18000/100),
+     ratio.2 = c(13000/60, 21000/110), ratio.3 = c(10200/52, 17000/105))
> fit2 = cm(~plyhder, dat2, ratios = ratio.1:ratio.3, method = "iterative")
> fit2

Call:
cm(formula = ~plyhder, data = dat2, ratios = ratio.1:ratio.3,
    method = "iterative")

Structure Parameters Estimators

  Collective premium: 190.9


  Between plyhder variance: 300.0
  Within plyhder variance: 166.8
```

下面拟合层次信度模型，数据我们采用 simul 函数进行模拟，索赔强度和索赔频率模型与 4.1 节相同，只是在参数取值上发生了少许变化，权重 w_{ijt} 采用 $[50, 100]$ 的均匀分布。

```
> set.seed(4)
> wijt = runif(31, 80, 100)
> nodes = list(class = 2, contract = c(4, 3), year = c(4, 4, 4,
+     4, 5, 5, 5))
> mf = expression(class = rexp(1), contract = rgamma(class + 1,
+     1), year = rpois(weights * contract))
> ms = expression(class = rnorm(2, sqrt(0.1)), contract = rnorm(class,
+     1), year = rlnorm(contract, 1))
> pf = simpf(nodes = nodes, model.freq = mf, model.sev = ms, weights = wijt)
```

计算 $X_{ijt} = S_{ijt}/w_{ijt}$，并构造数据。

```
> ratio.mat = aggregate(pf, classification = FALSE, prefix = "ratio.")/weights(pf,
+     classification = FALSE)
> dat3 = cbind(weights(pf, prefix = "weight."), ratio.mat)
```

拟合层次信度模型。

```
> fit3 = cm(~class + class:contract, data = dat3, ratio = ratio.year.1:ratio.year.5,
+     weights = weight.year.1:weight.year.5)
> summary(fit3)
```

```
Call:
cm(formula = ~class + class:contract, data = dat3, ratios = ratio.year.1:ratio.year.5,
    weights = weight.year.1:weight.year.5)

Structure Parameters Estimators

  Collective premium: 23.3

  Between class variance: 391.3
  Within class/Between contract variance: 213.2
  Within contract variance: 630.3

Detailed premiums

  Level: class
    class Indiv. mean Weight Cred. factor Cred. premium
    1     38.073      3.967  0.8792       36.29
    2      7.941      2.981  0.8455       10.31

  Level: contract
    class contract Indiv. mean Weight Cred. factor Cred. premium
    1     1        52.468      343.3  0.9915       52.330
    1     2        44.259      374.1  0.9922       44.196
    1     3         7.552      361.3  0.9919        7.785
    1     4        48.020      358.5  0.9918       47.924
    2     1         8.133      479.9  0.9939        8.146
    2     2         5.582      469.5  0.9937        5.612
    2     3        10.108      458.7  0.9936       10.109

> predict(fit3)

$class
[1] 36.29 10.31

$contract
[1] 52.330 44.196  7.785 47.924  8.146  5.612 10.109
```

还可以按照不同层次进行输出结果和预测。

```
> summary(fit3, levels = "class")

Call:
cm(formula = ~class + class:contract, data = dat3, ratios = ratio.year.1:ratio.year.5,
    weights = weight.year.1:weight.year.5)

Structure Parameters Estimators
```

```
  Collective premium: 23.3

  Between class variance: 391.3
  Within class variance: 213.2

Detailed premiums

  Level: class
    class Indiv. mean Weight Cred. factor Cred. premium
    1     38.073      3.967  0.8792       36.29
    2      7.941      2.981  0.8455       10.31
```

```
> predict(fit3, levels = "contract")

$contract
[1] 52.330 44.196  7.785 47.924  8.146  5.612 10.109
```

6.3 信度回归模型

在 Bühlmann - Straub 模型中，需要假设各年损失数据是平稳的，没有明显趋势，如果数据呈现系统性的趋势，那么 Bühlmann - Straub 模型就会低估或者高估真实保费。精算师 Hachemeister 就遇到了这种情况，他想要预测美国各州第三方责任险在未来一年的平均赔付额度，可是由于通货膨胀的影响，各年的赔付额呈递增的趋势，因此 Bühlmann - Straub 模型中的平稳性假设 ($E(X_{ij}|\Theta_i)$ 关于 j 独立) 遭到了破坏。Hachemeister 在其 1975 年发表的著名文章中将回归思想引入信度模型中，从而很好地解决了这个问题。

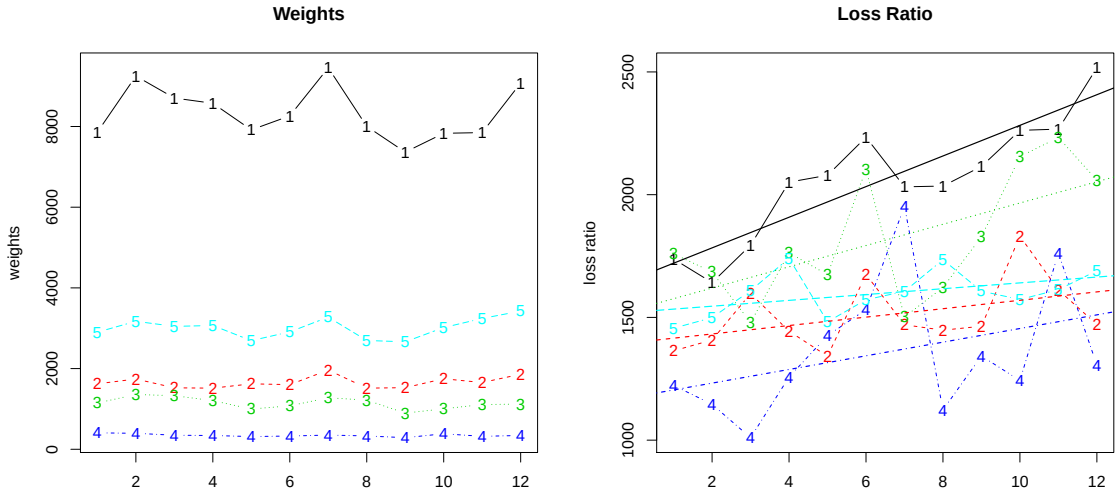
首先介绍一下在这篇文章中所使用的数据。该数据包含了美国 5 个州从 1970 年 7 月到 1973 年 6 月共 12 个季度的机动车第三方责任险的损失数据。数据总共有 5 行 25 列，每行代表一个州，第 1 列为每个州的标签，第 2 至 13 列为某个州在某季度的案均赔付额，第 14 至 25 列为对应的案件数量。R 中可以通过如下语句调用此数据。

```
> data(hachemeister)
> hachemeister
```

```
     state ratio.1 ratio.2 ratio.3 ratio.4 ratio.5 ratio.6 ratio.7 ratio.8
[1,]     1    1738    1642    1794    2051    2079    2234    2032    2035
[2,]     2    1364    1408    1597    1444    1342    1675    1470    1448
[3,]     3    1759    1685    1479    1763    1674    2103    1502    1622
[4,]     4    1223    1146    1010    1257    1426    1532    1953    1123
[5,]     5    1456    1499    1609    1741    1482    1572    1606    1735
     ratio.9 ratio.10 ratio.11 ratio.12 weight.1 weight.2 weight.3 weight.4
[1,]    2115     2262     2267     2517     7861     9251     8706     8575
[2,]    1464     1831     1612     1471     1622     1742     1523     1515
[3,]    1828     2155     2233     2059     1147     1357     1329     1204
[4,]    1343     1243     1762     1306      407      396      348      341
[5,]    1607     1573     1613     1690     2902     3172     3046     3068
     weight.5 weight.6 weight.7 weight.8 weight.9 weight.10 weight.11 weight.12
[1,]     7917     8263     9456     8003     7365      7832      7849      9077
```

[2,]	1622	1602	1964	1515	1527	1748	1654	1861
[3,]	998	1077	1277	1218	896	1003	1108	1121
[4,]	315	328	352	331	287	384	321	342
[5,]	2693	2910	3275	2697	2663	3017	3242	3425

分别作出五个州在各年的风险单位数和单位纯保费如下图。可以看出，各个州的风险单位数存在有显著的差异，第一个州的风险单位数要显著大于其他四个州，而且各州的纯保费也存在有显著的上升趋势。因此直接使用数据拟合信度模型是不合适的。一个自然的想法就是通过对数据关于时间进行回归找到这种趋势。分别对每个州的数据关于时间作回归发现，无论是斜率还是截距都有很大的差异，由于每个州的案件数量是不同的，因此回归直线的可信度也有高有低。当然我们可以将全部数据放在一起拟合一条回归直线，这样数据量虽有保证但却不能体现个体间的差异。根据信度理论的思想，我们可以将回归系数进行加权，那么问题就在于权重如何分配。



现在我们正式引入信度回归模型。给定具有 I 个风险的保单组合，令 $\mathbf{X}_i' = (X_{i1}, \ldots, X_{in})$ 为第 i 个风险的损失数据向量，令 $\mathbf{w}_i' = (w_{i1}, \ldots, w_{in})$ 为对应的权重向量。比如在 Hachemeister 数据中， $\mathbf{X}_i$ 代表第 i 个州的按均赔付额， $\mathbf{w}_i$ 代表第 i 个州的案件数量。

如果令 Θ_i 代表第 i 个州的风险水平，因此需要在给定 Θ_i 的条件下，拟合回归方程：

$$\mathbf{X} = \mathbf{Y}\boldsymbol{\beta} + \epsilon \tag{6.16}$$

其中 $\mathbf{Y}$ 是设计矩阵。与传统的线性回归模型不同的是，由于 Θ_i 的随机性，因此 $\boldsymbol{\beta}$ 具有随机效应，是关于 Θ_i 的随机变量，不妨记作 $\boldsymbol{\beta}(\Theta_i)$。模型具有如下假设：

1. 给定 Θ_i， X_{ij} 条件独立，且

$$\mathrm{E}(\mathbf{X}_i|\Theta_i) = \mathbf{Y}_i\boldsymbol{\beta}(\Theta_i) \tag{6.17}$$

$$Var(X_{ij}|\Theta_i) = \frac{\sigma^2(\Theta_i)}{w_{ij}} \tag{6.18}$$

2. 随机向量 $(\Theta_1, \mathbf{X}_1), \ldots, (\Theta_I, \mathbf{X}_I)$ 相互独立，且 $\Theta_i, \ldots, \Theta_I$ 独立同分布。

根据条件独立性，性质(6.18)可以得到数据的协方差矩阵：

$$\mathrm{Cov}(X_i, X_i'|\Theta_i) = \sigma^2(\Theta_i)\mathbf{W}_i^{-1}, \qquad W_i = \mathrm{diag}(w_{i1}, \ldots, w_{in}) \tag{6.19}$$

当不存在协变量时，亦即 $\mathbf{X}_i = (1, \ldots, 1)'$ 时，

$$E(X_i|\Theta_i) = \begin{pmatrix} 1 \\ \vdots \\ 1 \end{pmatrix} \mu(\Theta_i) \tag{6.20}$$

此时，信度回归模型就退化为 Bühlmann - Straub 信度模型

而在本案例中，当以时间为自变量，有：

$$\mathrm{E}(X_{ij}|\Theta_i) = \beta_0(\Theta_i) + \beta_1(\Theta_i) \cdot j \tag{6.21}$$

写成矩阵形式，有：

$$\mathrm{E}(\mathbf{X}_i|\Theta_i) = \begin{pmatrix} 1 & 1 \\ 1 & 2 \\ \vdots & \vdots \\ 1 & n \end{pmatrix} \begin{pmatrix} \beta_0(\Theta_i) \\ \beta_1(\Theta_i) \end{pmatrix} \tag{6.22}$$

根据如上假设，我们得到$\boldsymbol{\beta}(\Theta_i)$ 的信度估计

$$\widehat{\boldsymbol{\beta}}(\Theta_i) = \mathbf{A}_i\mathbf{B}_i + (\mathbf{I} - \mathbf{A}_i)\boldsymbol{\beta} \tag{6.23}$$

其中，

$$\mathbf{A}_i = \mathbf{T}(\mathbf{T} + (\mathbf{Y}_i'\mathbf{V}_i^{-1}\mathbf{Y}_i)^{-1})^{-1}$$

是信度矩阵。

$$\mathbf{B}_i = (\mathbf{Y}_i'\mathbf{V}_i^{-1}\mathbf{Y}_i)^{-1}\mathbf{Y}_i'\mathbf{V}_i^{-1}\mathbf{X}_i$$

是 $\mathbf{X}_i$ 的最优线性无偏估计。

$$\mathbf{V}_i = \mathrm{E}[\mathrm{Cov}(\mathbf{X}_i, \mathbf{X}_i'|\Theta_i)]$$

是组内协方差矩阵的期望。

$$\mathbf{T} = \mathrm{Cov}(\boldsymbol{\beta}(\Theta_i), \boldsymbol{\beta}(\Theta_i)')$$

是个体回归系数协方差阵 (组间协方差矩阵)。

$$\boldsymbol{\beta} = \mathrm{E}[\boldsymbol{\beta}(\Theta_i)]$$

是聚合回归系数 (聚合保费)。

在函数 cm 中，regformula，regdata 和 adj.intercept 三个参数控制信度回归模型。regformula 使用方法同参数 formula；regdata 指定回归的设计矩阵，但必须是数据框格式；adj.intercept 调整回归的截距，具体讲解见例子。

例子：

以时间为自变量，拟合 Hachemeister 数据。

```
> fit4 = cm(~state, hachemeister, regformula = ~time, regdata = data.frame(time = 1:12),
+     ratios = ratio.1:ratio.12, weights = weight.1:weight.12)
```

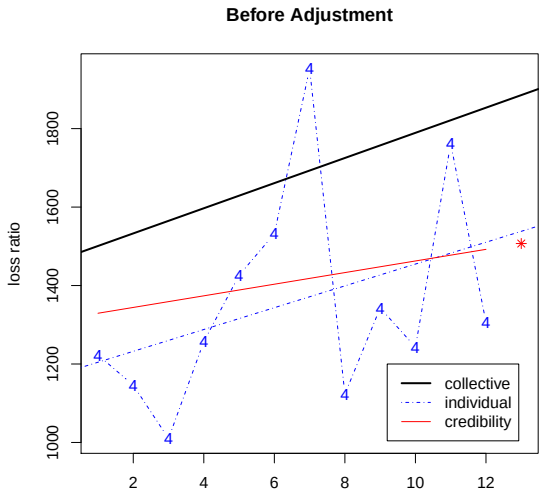
可以对模型结果进行 summary，predict 等操作，但要输入自变量数据。输出结果返回$\boldsymbol{\beta}$，$\mathbf{T}$， σ^2， $\mathbf{B}_i$， $\mathbf{A}_i$ 及 $\boldsymbol{\beta}(\Theta_i)$ 的估计值。

```
> summary(fit4, newdata = data.frame(time = 13))

Call:
cm(formula = ~state, data = hachemeister, ratios = ratio.1:ratio.12,
    weights = weight.1:weight.12, regformula = ~time, regdata = data.frame(time = 1:12))

Structure Parameters Estimators

  Collective premium: 1469 32.05

  Between state variance:  24154 2700.0
                            2700  301.8
  Within state variance: 49870187

Detailed premiums

  Level: state
    state Indiv. coef. Credibility matrix Adj. coef. Cred. premium
    1     1658.47      0.54944 3.97190    1693.52    2437
            62.39      0.06142 0.44398      57.17
    2     1398.30      0.53096 3.91217    1373.03    1651
            17.14      0.05935 0.43731      21.35
    3     1533.00      0.53173 3.76959    1545.36    2073
            43.31      0.05944 0.42137      40.61
    4     1176.70      0.47836 3.42117    1314.55    1507
            27.81      0.05347 0.38242      14.81
    5     1521.90      0.53897 3.96533    1417.41    1759
            11.87      0.06025 0.44325      26.31
```

以第 4 个州的数据为例，分别作出个体回归线，聚合数据回归线，以及信度加权后的回归线。* 号处是下一年的信度保费。

```
> pdf("pic15.pdf", height = 6, width = 6)
> matplot(dat.reg[, 9], type = "b", main = "Before Adjustment",
+     pch = "4", col = 4, lty = 4, ylab = "loss ratio", xlim = c(1,
+         13))
> abline(coef = fit4$means$state[, 4], xlim = c(1, 12), col = 4,
+     lty = 4)
> abline(coef = fit4$means$portfolio, lwd = 2)
> fit.pred = predict(fit4, newdata = data.frame(time = 1:13))
> lines(1:12, fit.pred[1:12, 4], col = 2)
```

```
> points(13, fit.pred[13, 4], col = 2, pch = 8)
> legend(9.2, 1200, legend = c("collective", "individual", "credibility"),
+     lty = c(1, 4, 1), lwd = c(2, 1, 1), col = c(1, 4, 2))
```



可以发现，信度回归线在最后几年低于个体回归线，也就是说，个体回归线的权重大于 1，为避免出现这种情况，可以将回归 x 轴的原点由 0 变为其重心：

$$j_0 = \sum_{j=1}^{n} j \frac{w_j}{w_\Sigma},$$

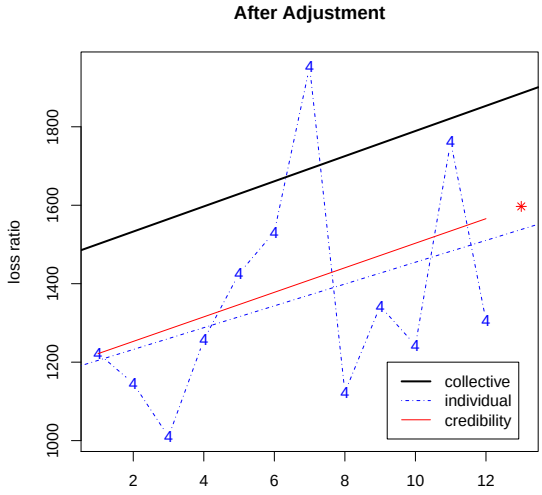
从而避免出现上面的情况，具体的证明请参阅 [2]。

调整回归截距后，虽然得到的回归直线参数意义不明确，但是其预测值却有意义。

```
> fit5 = cm(~state, hachemeister, regformula = ~time, adj.intercept = TRUE,
+     regdata = data.frame(time = 1:12), ratios = ratio.1:ratio.12,
+     weights = weight.1:weight.12)
> fit5

Call:
cm(formula = ~state, data = hachemeister, ratios = ratio.1:ratio.12,
    weights = weight.1:weight.12, regformula = ~time, regdata = data.frame(time = 1:12),
    adj.intercept = TRUE)

Structure Parameters Estimators

  Collective premium: -1675 117.1

  Between state variance:  93783    0
                               0 8046
  Within state variance: 49870187
```

重新作出回归曲线。

```
> pdf("pic16.pdf", height = 6, width = 6)
> matplot(dat.reg[, 9], type = "b", main = "After Adjustment",
+     pch = "4", col = 4, lty = 4, ylab = "loss ratio", xlim = c(1,
+         13))
> abline(coef = fit4$means$state[, 4], xlim = c(1, 12), col = 4,
+     lty = 4)
> abline(coef = fit4$means$portfolio, lwd = 2)
> fit.pred = predict(fit5, newdata = data.frame(time = 1:13))
> lines(1:12, fit.pred[1:12, 4], col = 2)
> points(13, fit.pred[13, 4], col = 2, pch = 8)
> legend(9.2, 1200, legend = c("collective", "individual", "credibility"),
+     lty = c(1, 4, 1), lwd = c(2, 1, 1), col = c(1, 4, 2))
```



Bibliography

[1] Belhadj, H., Goulet, V. and Ouellet, T. (2009), On parameter estimation in hierarchical credibility, *Astin Bulletin* 39.

[2] Bühlmann, H. and Gisler, A. (2005), *A Course in Credibility Theory and its Applications*, Springer.

[3] C. A. Hachemeister. Credibility for regression models with application to trend. In *Credibility, theory and applications, Proceedings* of the berkeley Actuarial Research Conference on Credibility, pages 129 – 163, New York, 1975. Academic Press.

[4] Goulet, V. and Pouliot, L.-P. (2008), Simulation of compound hierarchical models in R, *North American Actuarial Journal* 12, 401 – 412.

[5] Klugman, S. A., Panjer, H. H. and Willmot, G. E. (2008), *Loss Models, From Data to Decisions*, *Third Edition*, Wiley.