Актуальные проблемы *actuar* на примере одной задачи страхования.

Алексей Осипов, к.ф.-м.н.

Сиденис

June 20, 2018

Постановка задачи.

Неформально.

Рассмотрим ситуацию, когда происходят схожие события (например, пожары в городе), которые наносят ущерб разной степени. Размер ущерба случаен, и количество этих событий тоже случайно. Моделируем суммарный ущерб.

Формально.

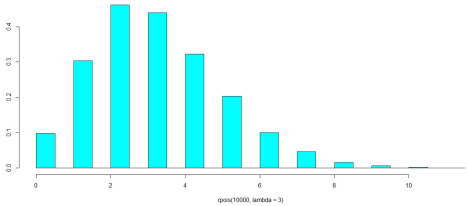
Пусть N и X — 2 распределения, N — дискретное, frequency, X — непрерывное, severity.

Задача. Моделировать $X * N$, т.е. $x_1 + \ldots + x_n$, где n распределено по N, а x_i распределены по X.

Примеры N: распределение Пуассона, биномиальное, отрицательное биномиальное (выбор из-за Panjer recursion).

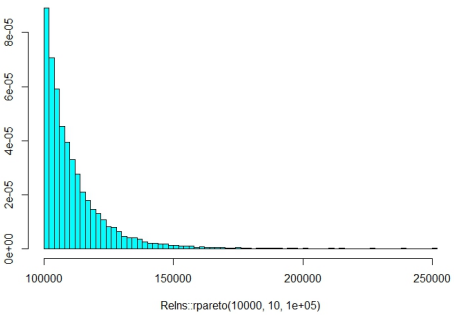
Примеры X: Парето, кусочное Парето, логнормальное (около 20 альтернатив).

Частный случай.



Мы будем брать в качестве N распределение Пуассона:

```
rpois(1000, lambda = 3)
```



В качестве X распределение Парето:

```
RelnsExtra::rpareto(10000,
    ↪ 10, 100000)
```

Почему rpareto из RelnsExtra?

Ну, можно было бы и из EnvStats. К сожалению, actuar::rpareto отличается сдвигом, а rmutil::rpareto формулой.

Функция aggregateDist из actuar.

Методы.

1. Монте Карло
2. свертка
3. Panjer recursion
4. 2 метода, основанных на нормальном распределении

Выход.

```
Fs <- aggregateDist(...); class(Fs)
[1] "aggregateDist" "ecdf"          "stepfun"
    ↪          "function"
```

Это функция, можно её вызывать, рисовать график, считать характеристики модели:

```
c(mean(Fs), quantile(Fs, 0.95)).
```

Монте Карло.

Краткое описание.

1000000 раз генерим n из N, соответствующее число раз генерим x из X, суммируем, получаем выборку из распределения, считаем характеристики.

Вход.

```
model.freq <- expression(data = rpois(lambda))
model.sev <- expression(data = RelNs::rpareto(
    ↪ alpha, xm))
Fs <- aggregateDist("simulation", nb.simul =
    ↪ numiter, model.freq, model.sev)
```

Монте Карло, детали.

```
aggregateDist("simulation", nb.simul = numiter,
    ↪  model.freq, model.sev)
```

1. **Как определить numiter?**
   Например, с помощью ЦПТ. Конечно, случаи с бесконечными средним или дисперсией, нужно обрабатывать по особому (определенные параметры у Парето).
2. **Ошибки при больших lambda в model.freq.**
   Хорошая идея привязывать numiter к lambda.
3. **Медленный метод, но очень гибкий.**
4. Есть много вариаций Монте Карло: stratified sampling, Sobol sequence, Iman Conover и т.д.

Свертка.

Краткое описание.

По дискретизованной frequency для каждого n мы можем найти вероятность получить ровно n событий, распределение суммы $X_1 + \ldots + X_n$ — это свёртка, её мы честно считаем, желательно применяя FFT.

Дискретизация.

```
freq <- discretize(ppois(x, lambda = lambda),
    ↪ from = 0, to = lambda*10, by = max(round(
    ↪ lambda/2), 1))
```

Важно добавить вероятность 0:

```
freq <- c(1-sum(freq), freq)
```

Свертка, описание.

```
sev <- discretize(RelRns::ppareto(x, shape =
    ↪ alpha, scale = xm), from = 0, to = 10*(xm
    ↪ *5), by = xm/100)
sev <- c(1-sum(sev), sev)
Fs <- aggregateDist("convolution", model.freq =
    ↪  freq, model.sev = sev, x.scale = xm/100)
```

Как задавать x.scale?

value of an amount of 1 in the severity model (monetary unit).

По сути это шаг дискретизации severity, но он и есть этот мультипликатор.

Свертка, детали, комментарии.

```
Fs <- aggregateDist("convolution", model.freq =
    ↪  freq, model.sev = sev, x.scale = xm/100)
```

1. **Как задавать дефолты?** Очень многое зависит от них: правильно указанный диапазон, шаги дискретизации.
2. **Почему равномерная дискретизация?** Это особенность метода. Естественнее неравномерная (логарифмическая), но это уже nonuniform FFT (есть nfft на python).
3. К сожалению, в actuar реализована обычная свертка, а не FFT, поэтому это **медленный метод.**

Panjer recursion.

Краткое описание.

Так как frequency принадлежит классу Panjer:

$$P(N = k) = \left(a + \frac{b}{k}\right) P(N = k - 1).$$

Эта формула и дискретизация severity и используются для создания рекурсивного алгоритма.

Вход.

распределение frequency, дискретизация severity.

```
sev <- discretize(RelNs::ppareto(x, shape =
    ↪ alpha, scale = xm), from = 0, to = 10*(xm
    ↪ *20), by = xm/100)
sev <- c(1-sum(sev), sev)
```

Panjer recursion, детали.

```
Fs <- aggregateDist("recursive", model.freq = "
    ↪ poisson", model.sev = sev, lambda =
    ↪ lambda, x.scale = xm/100, maxit = 100000)
```

frequency reduction trick

$$X * Pois(lambda) = (X * Pois(lambda/m))^{(m)}.$$

Количество сверток, которые нужно посчитать: $ln(m)/ln(2)$.

```
freqred <- (max(ceiling(log(lambda/16)/log(2)),
    ↪  0))
Fs <- aggregateDist("recursive", model.freq = "
    ↪ poisson", model.sev = sev, lambda =
    ↪ lambda/2^{freqred}, x.scale = xm/100,
    ↪ maxit = 100000, convolve = freqred)
```

Не забываем правильно указывать convolve и x.scale.

Panjer recursion, комментарии.

```
Fs <- aggregateDist("recursive", model.freq = "
    ↪ poisson", model.sev = sev, lambda =
    ↪ lambda/2^{freqred}, x.scale = xm/100,
    ↪ maxit = 100000, convolve = freqred)
```

1. **Важны дефолты.**
   Шаг дискретизации, диапазон дискретизации, при каких lambda проводить трюк сведения.
2. **Почему дискретизация равномерная?**
   Особенность метода. Естественнее, конечно, неравномерная дискретизация.
3. Трюк сведения позволяет работать даже с большими lambda.

Приближение нормальным распределением.

Мотивация.

1. Оно самое простое.
2. Что-то вроде ЦПТ.

Описание.

Приближаем нормальным с теми же средним и дисперсией.

```
Fs <- aggregateDist("normal", moments =
    ↪ aggmoments)
```

1. **А что делать, если один из моментов бесконечен?**
   Метод не работает.
2. У нас 3 входных параметра, мы приближаем моделью с 2.
   Естественно мы недооцениваем правый хвост.

Приближение степенями нормального.

Описание метода.

Стандартизуем и приближаем

$$Y + (skewness/6)(Y^2 - 1),$$

где Y из $N(0, 1)$.

Замечания.

1. **А что делать, если коэффициент асимметрии бесконечен?**
   Метод не работает, если он больше 1.
2. Считается, что мы недооцениваем правый хвост.

Описание теста.

Frequency = Poisson, Severity = Pareto.

Диапазон параметров.

lambda из Poisson от 0.1 до 100, alpha из Pareto от 2.5 до 10, xm из Pareto от 100000 до 10000000.
Всего 90 случаев.

Критерии оценки.

Сравниваем средние (мы знаем какими они должны быть) и 95%-е квантили (берем из Монте Карло с очень большим числом итераций), меряем время работы.

Результаты.

	Монте карло	Convolution	Panjer recursion	Normal approximation
Время работы	0.23/0.46 сек.	250/413 раз медленнее симуляции	1.3/3.6 раз медленнее симуляции	В 109 раз быстрее симуляции
Доля случаев с правильно оцененным средним.	97%	20%	100%	100%
Доля случаев с правильно оцененным хвостом.	92%	23%	100%	60%

Комментарии.

Замечания.

1. Свертка сработала плохо, потому что это не FFT.
2. Нормальное приближение сработало плохо, потому что не из чего не следовало, что оно должно работать.
3. Normal power approximation не применялось, потому что у некоторых тестовых случаев был неправильный skewness.
4. Монте Карло сработало быстрее, чем Panjer recursion, и на этих случаях так и должно было бы быть. Но есть случаи, которые с ним бы не сосчитались.

Чего не хватает.

1. Реализация FFT.
2. Методы, работающие с неравномерными дискретизациями.
3. Алгоритмы определения дефолтных параметров.

Заключение.

Коротко о главном.

1. С помощью actuar можно **в принципе** решить рассматриваемую задачу страхования, но основная сложность в указании дефолтных параметров.
2. Методы из actuar нельзя назвать совершенными. При их регулярном использовании будет естественно возникать идея написать что-то свое.

Большое спасибо за внимание!