

# Collective risk

Ignacio Gómez, Gonzalo Pato, Gonzalo Prats

April 2023

## Contents

<b>Statement</b>	<b>1</b>
<b>Introduction</b>	<b>1</b>
<b>The project</b>	<b>2</b>
<b>Results</b>	<b>3</b>
<b>Conclusions</b>	<b>4</b>
<b>References</b>	<b>4</b>

## Statement

The clients of an insurance company generate claims according to independent (homogeneous) Poisson processes with common rate  $\lambda$ . The claims are independent and the amount of each of them follows a distribution with cdf  $F(x) = 1 - (100/x)^{2.5}$  if  $x > 100$ . New clients enroll the company according to another (homogeneous) Poisson process with rate  $\nu$  and the time they stay in the company follows an exponential distribution with rate  $\mu$ . Each policyholder pays a fixed amount  $a$  every year to the company. If the policyholder stays in the company only during a fraction of the year, she will pay the corresponding fraction of  $a$  which is assumed to be paid on a continuous manner.

If the initial capital is  $c_0$  and the initial number of clients is  $n_0$ , we want to compute the probability that the capital of the company remains positive during some given time  $t_l$ .

## Introduction

Project explanation with your own words.

Each policyholder has a fixed amount  $a$  to pay every year, so the capital of the company will increase by  $a$  times de number of clients every fraction of year.

On the other hand, they also hold the right to claim the premium on the policy, which happens according to a Poisson process with common rate  $\lambda$ . The amount of is this premium is distributed according to a Pareto distribution with parameters  $\alpha = 2.5$  and  $\beta = 100$ .

The clients of the company can leave the company at any time, and the time they stay in the company follows an exponential distribution with rate  $\mu$ .

Clients come according to an independent (homogeneous) Poisson processes with rate  $\nu$ , and they stay according to an exponentially distributed time with rate  $\mu$ .

So in general, the capital of the company at any time  $t$  will be:

$$C(t) = c_0 + at(n_0 + N_A(t) - N_D(t)) - \sum_{j=1}^{N_C(t)} X_j$$

where  $N_A(t)$  is the number of clients that arrive by time  $t$ ,  $N_D(t)$  is the number of clients that leave by time  $t$ ,  $N_C(t)$  is the number of claims that arrive by time  $t$ ,  $X_j$  is the amount of the  $j$ -th claim, and  $n(t)$  is the number of clients at time  $t$ .

## The project

Project development including the code and emphasizing the main difficulties and most important parts.

First, to simulate the amount of the claims we will define a function with the inverse of the cdf of the Pareto distribution. Mathematically, this function is:

$$F^{-1}(x) = \frac{\beta}{\sqrt[\alpha]{1-x}} = \frac{100}{\sqrt[2.5]{1-x}}$$

```
inverse.claim <- function(x) {
  return(max((100 / (1 - x)^(1 / 2.5)), 0))
}
```

The algorithm we will follow is, for each path: 1. Initialize variables  $t = 0, n = n_0, c = c_0$ . 2. Compute the time of the first event:  $dt = \text{rexp}(1, \lambda * n + \mu * n + \nu)$  3. Update the time:  $t = t + dt$  4. While  $t < t_l$ : 4.1 Update capital with the proportional part of the payments:  $c = c + n * a * dt$  4.2 Decide the type of event and update in each case: 4.2.1 If  $u < \lambda * n / (\lambda * n + \mu * n + \nu)$ , claim event so update capital  $c = c - X_j$  4.2.2 If  $u < (\lambda * n + \mu * n) / (\lambda * n + \mu * n + \nu)$ , departure event so  $n = n - 1$  4.2.3 Else, enrollment event, so  $n = n + 1$  4.3 Compute the time of the next event:  $dt = \text{rexp}(1, \lambda * n + \mu * n + \nu)$  4.4 Update the time:  $t = t + dt$

Thus for one simulation we have:

```
simulate_one_insurance <- function(c0, n0, a, tl, lambda, mu, nu) {
  # Initialize variables
  t <- 0
  n <- n0
  c <- c0
  # Update capital with annuity payments
  dt <- rexp(1, lambda * n + mu * n + nu)
  t <- t + dt
  while (t < tl) {
    # Update the capital with the proportional part of the payments
    c <- c + n * a * dt
    # Decide the type of event
    u <- runif(1)
    if (u < lambda * n / (lambda * n + mu * n + nu)) {
      # Claim event
      claim_amount <- inverse.claim(runif(1))
      c <- c - claim_amount
    } else if (u < (lambda * n + mu * n) / (lambda * n + mu * n + nu)) {
      # Departure event
      n <- max(n - 1, 0)
    } else {
      # Enrollment event
    }
  }
}
```

```

    n <- n + 1
  }
  # Stop if capital becomes negative
  if (c < 0) break
  # Time of next event
  dt <- rexp(1, lambda * n + mu * n + nu)
  t <- t + dt
}
return(c)
}

```

And for multiple:

```

simulate_insurance <- function(c0, n0, a, t1, lambda, mu, nu, MC) {
  final_capitals <- vector("numeric", MC)
  # Simulate MC paths
  final_capitals <- replicate(MC, simulate_one_insurance(c0, n0, a, t1, lambda, mu, nu))
  # Compute and return outputs
  fraction <- sum(final_capitals > 0) / MC
  pos_fc <- final_capitals[final_capitals > 0]
  mean_final_capitals <- mean(pos_fc)
  sd_final_capitals <- sd(pos_fc)
  p1 <- ggplot() +
    geom_line(mapping= aes(x = 1:length(pos_fc),
                          y = cumsum(pos_fc)/(1:length(pos_fc))))
  p2 <- ggplot() +
    geom_histogram(mapping= aes(x = pos_fc, y=after_stat(density)),
                  binwidth=5000, color = "black", fill = "white") +
    geom_density(mapping= aes(x = pos_fc, color = "red"))
  print(plot_grid(p1, p2, labels="AUTO"))
  return(list(fraction = fraction, mean_final_capital = mean_final_capitals,
             sd_final_capital = sd_final_capitals))
}

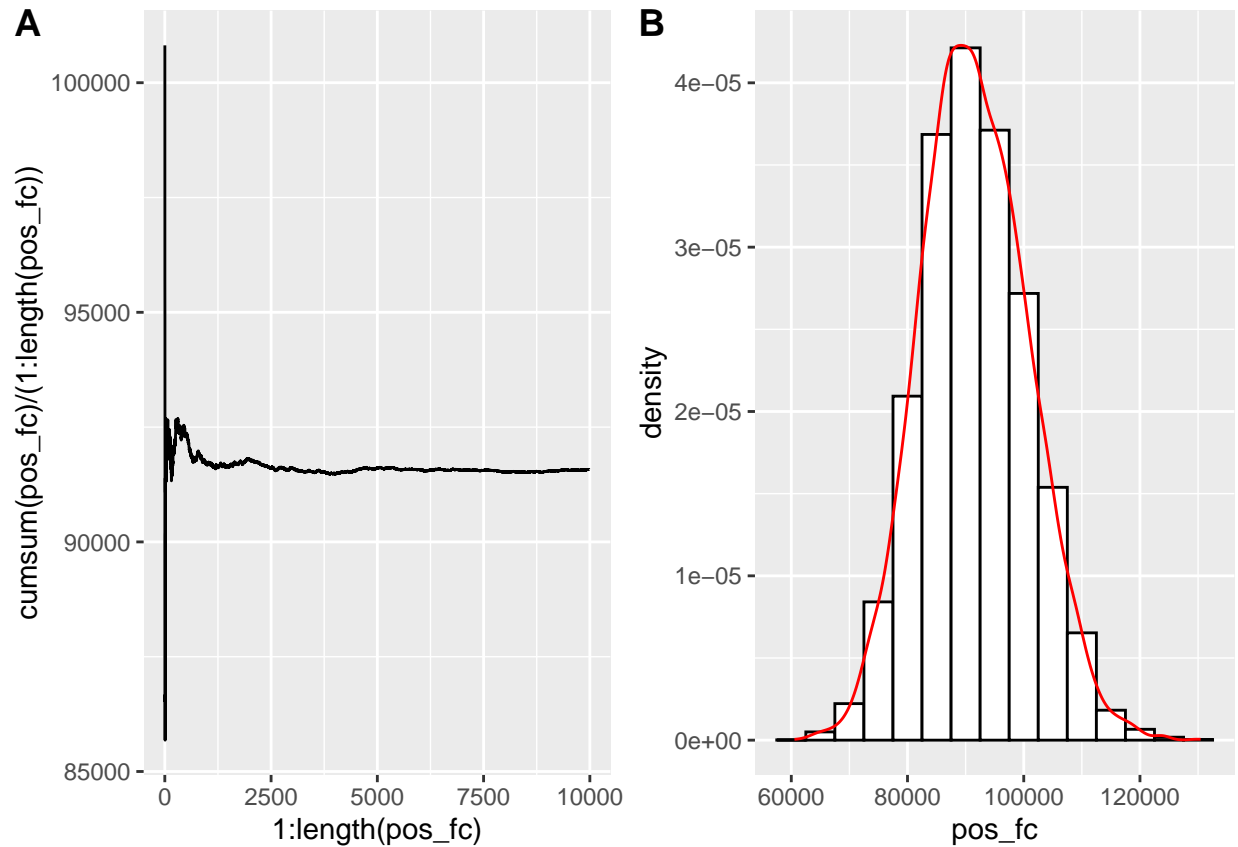
```

## Results

```

set.seed(42)
res <- simulate_insurance(c0 = 1000, n0 = 100, a = 100, t1 = 100,
                        lambda = 0.1, mu = 0.1, nu = 0.1, MC = 10000)

```



```
res
```

```
## $fraction
## [1] 0.9984
##
## $mean_final_capital
## [1] 91573.45
##
## $sd_final_capital
## [1] 9182.815
```

Numerical results, quality assessment of the approximations, and time efficiency of the algorithms.

## Conclusions

About the results, how the difficulties were solved, and possible alternative approaches. Keep the focus, the conclusions must be as brief as possible.

## References

Including textbooks, webpages, and class notes.