Ruin Theory in Discrete Time

Jakob Graves

September 4, 2023

1 A Discrete Time Model

Here we have a discrete time model for an insurer's surplus. Denote the insurer's surplus at time n by $U(n)$ (also known as a surplus process), which is defined by

$$U(n) = u + n - \sum_{i=1}^{n} Z_i$$

for $n = 1, 2, 3, ...$, where

- $u = U(0)$ is the insurer's initial surplus, or surplus at time 0
- Z_i is the insurer's aggregate claim amount in the i^{th} time interval, $\{Z_i\}_{i=1}^{\infty}$ is an i.i.d. sequence of non-negative random variables with $E[Z_1] < 1$, PDF $\{h_k\}_{k=0}^{\infty}$, and CDF $H(x)$
- The insurer's income per unit time is 1, so that n is the total premium income up to time n

We are interested in the probability of ultimate ruin, which occurs if the surplus ever falls to 0 or below. The probability of ultimate ruin is defined as $\psi(u) = P(T_u < \infty)$ where $T_u = \min\{n \geq 1 : U_n \leq 0\}$

$$\psi(0) = \sum_{y=0}^{\infty} [1 - H(y)] = E[Z_1]$$

$$\psi(u) = \sum_{y=0}^{u-1} [1 - H(y)]\psi(u - y) + \sum_{y=u}^{\infty} [1 - H(y)]$$

1.1 Lundberg's Inequality in Discrete Time

Lundberg's Inequality gives an upper bound on the ruin probability for the discrete time risk model.

Lundberg's Inequality: For the discrete time surplus function, the probability of ultimate ruin satisfies the following inequality

$$\psi(u) \leq e^{-Ru}$$

where R is the adjustment coefficient which satisfies the following equation

$$E[\exp(R(X-1))] = 1$$

where X is the loss random variable

Proof: Page 124 *Insurance Risk and Ruin* David C. M. Dickson (2016)

1.2 Example Exercise

1. Suppose a small fire insurance company insures 100 homes. Each homeowner pays an annual premium of $500. Fires are rare, but occur with probability 1 in 1000 per house per year. In the event of a fire, the insurance company must pay out $250,000.

(a) Show that an expected value decision maker would enter this business.

(b) Show that a worst-case scenario decision maker would run screaming in fear from this business.

(c) How many years must an insurance company operate before the probability of losing money falls below 5%?

(d) How much money must the insurance company have hoarded to survive these numbers of years?

Note that the very mathematically mature subject of Ruin Theory addresses these questions in detail. See also the book in my digital collection called *Insurance Risk and Ruin* by David C. M. Dickson, and my own notes on this book.

Solution:

(a) The number of house fires that occur in a year are binomial distributed with $n = 100$ and $p = 0.001$. The expected value of a binomial distribution is given by $np = 100 \cdot 0.001 = 0.1$. Therefore the average annual payout is $0.1 \cdot \$250,000 = \$25,000$. The annual premium collected by the company is $50,000. Therefore, the business is profitable on average, with expected annual profit of $25,000.

(b) For the overly paranoid insurer, the maximum loss is $\$250,000 \cdot 100 - \$50,000 = \$24,950,000$ in the case that every house burns down (e.g. a forest fire wipes out the neighbourhood), so if we are using the worst-case scenario minimizing strategy, the optimal decision is to not enter the business.

(c) We use the discrete time risk model described above. As mentioned previously, the number of claims in a year are distributed binomailly with $n = 100$ and $p = 0.001$. Let's use it to calculate $\psi(0)$. Let $Y \sim \text{binom}(100, 0.001)$. Then since one claim is equivalent to five units of premium income, we have

$$\psi(0) = E[Z_1] = 5E[Y] = 5np = 5 \cdot 100 \cdot 0.001 = 0.5$$

So the probability of ruin starting from no initial surplus is 50%. Now lets look at the distribution of $\{Z_i\}_{i=0}^{\infty}$. For simplicity, we take P(3 claim) as the binomial probability P($x \geq 3$).

claim distribution	
P(0 claim)	0.904792
P(1 claim)	0.09057
P(2 claim)	0.004488
P(3 claim)	0.00015
Sum	1

Figure 1: The binomial CDF of X

Now the ruin probabilities are below, as calculated in Excel. See spreadsheet "discrete risk model simulation and calculator.xlsx" for more details. Recall that $h(x)$ is the claims PDF for $\{Z_i\}_{i=0}^{\infty}$, and $H(x)$ is the CDF.

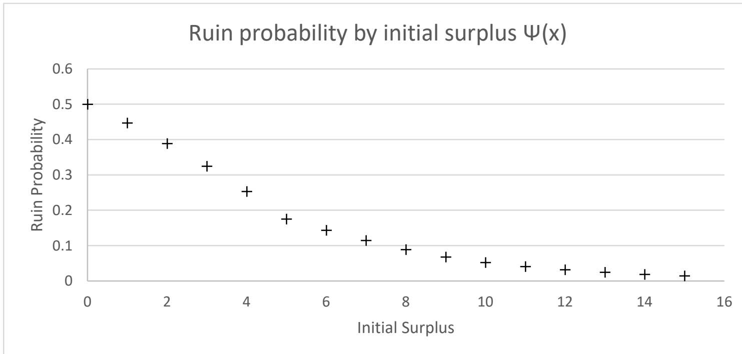
x	h(x)	H(x)	1 - H(x)	Ψ(x)
0	**0.904792**	**0.904792**	**0.095208**	**0.5**
1	0	0.90479	0.09521	0.44737
2	0	0.90479	0.09521	0.38921
3	0	0.90479	0.09521	0.32494
4	0	0.90479	0.09521	0.25391
5	**0.09057**	**0.99536**	**0.00464**	**0.17540**
6	0	0.99536	0.00464	0.14395
7	0	0.99536	0.00464	0.11501
8	0	0.99536	0.00464	0.08946
9	0	0.99536	0.00464	0.06833
10	**0.004488**	**0.99985**	**0.00015**	**0.05284**
11	0	0.99985	0.00015	0.041605
12	0	0.99985	0.00015	0.032374
13	0	0.99985	0.00015	0.025047
14	0	0.99985	0.00015	0.019417
15	**0.00015**	**1**	**0**	**0.015135**

Figure 2: The PDF, CDF, and ruin probabilities

So, the company needs to accumulate a surplus of 11 before the probability of ruin falls below 5%. This is equivalent to a \$550,000 surplus.

To answer the question, the expected yearly income is \$25,000 from part (a), and \$550,000/\$25,000=22. Therefore, we expect to have the \$550,000 surplus after 22 years of operation. The minimum number of operating years before getting the desired surplus is 11.

See below a chart that plots the ruin probability as a function of the initial surplus:



Here is some R code to compute ruin probabilities using the model defined above:

```
# this script will compute the discrete time probability of ruin for
# an insurance company, where the number of claims per period is an
# i.i.d binomial random variable

binom_ruin_prob <- function(n, p, income, claim, surplus_0, p_thresh){

  # DEFINITION OF INPUT VARIABLES
  # n is the number of insured entities
  # p is the probability an insured entity generates a claim
  # income is the premium income per period (from all insured entities)
  # claim is the payout per claim
  # surplus_0 is the initial surplus

  # OUTPUT
  # the output is the scalar probability of ruin (between 0 and 1)

  # IMPORTANT TECHNICAL CONDITIONS
  # income and claims need to be scaled such that income is 1 per period
  # in the case income is not initially 1, it must divide claim evenly
  # in other words scaled claim amounts must be on the non-negative integers
  # expected value of the first period's claim amount is less than income
  # the number of claims in a period are binomial distributed

  #----------------------------------------------------------------------#

  # optional arguments
  if (missing(p_thresh)){p_thresh <- 0.01}

  # scale premium income and claims so that income is 1 per period
```

```
# this is just how we use the model is constructed
claim <- claim/income
income <- 1

# return errors if technical conditions are not satisfied
if (n*p*claim >= 1){stop("Expected value of first period claim
                          is greater than 1. Ruin is guaranteed.
                         Review technical conditions")}
if (claim %% income != 0)
{stop("Claim is not an integral multiple of income")}

# generate the NUMBER of claims distribution...
# for simplicity when p_claim gets really small (line 33), we attach the
# inverse cumulative probability (greater than or equal to the last index1)
# at the end of dist to make it a true distribution (sum to 1)
n_clm_dist <- c()
p_claim <- 1
index1 <- 0

while (p_claim > p_thresh){ # make p_claim threshold an optional parameter
  p_claim <- dbinom(index1, n, p)
  n_clm_dist <- append(n_clm_dist, p_claim)
  index1 <- index1 + 1
}
n_clm_dist <- append(n_clm_dist, 1 - sum(n_clm_dist)) # line 17-18

# now transform this to be the VALUE of claims distribution
payout_dist <- c()
for (index2 in 0:max((claim*length(n_clm_dist)-1),surplus_0))  {
  if ((index2 %% claim == 0) & (index2 < (claim*length(n_clm_dist)-1)))
    {payout_dist <- append(payout_dist, n_clm_dist[(index2/claim)+1])}
  else {payout_dist <- append(payout_dist, 0)}
}

# then the cumulative distribution
cumulative <- cumsum(payout_dist)

# the last intermediary vector, the inverse cumulative distribution
inv_cumulative <- 1 - cumulative

# finally, the hard part: compute the ruin probability vector
# the formula is different if surplus_0 == 0, so do that first
if (surplus_0 == 0){psi <- n*p*claim
  surplus_0 <- 1
} else { # now for surplus_0 greater than 0
  # start with an intermediary vector for the second sum in the formula
```

```
    sum2 <- c()
    for (index3 in 1:surplus_0){
      if (is.na(inv_cumulative[index3 + 1])){
        sum2 <- append(sum2, 0)
      } else{ sum2 <- append(sum2, sum(inv_cumulative[(index3 + 1):
                                  length(inv_cumulative)]))
      }
    }

    # need to compute psi for every surplus level less than surplus_0
    # because the formula is recursive.
    psi <- c(sum2[1]/cumulative[1])
    for (index4 in 2:surplus_0){
      tranche <- rev(inv_cumulative[2:(index4)])
      dot <- sum(tranche*psi[1:length(psi)])
      psi <- append(psi, (dot + sum2[index4])/cumulative[1])
    }
    }

  return(psi[surplus_0])
}
```

And here is another script that will run a simulation of the surplus process. The simulations have more flexibility when it comes to the input parameters. We don't need the same technical conditions.

```
# this function simulated the discrete time surplus process

discrete_binom_sim <- function(n, p, time, income, claim, surplus_0){

  # DEFINITION OF INPUT VARIABLES
  # n is the number of insured entities
  # p is the probability an insured entity generates a claim
  # time is the number of periods to simulate
  # income is the premium income per period (from all insured entities)
  # claim is the payout per claim (constant)
  # surplus_0 is the initial surplus

  # OUTPUT
  # the output is a time series of the surplus process
  # it is a vector where entries give the surplus value at the
  # end of each period

  # IMPORTANT TECHNICAL CONDITIONS

  #-------------------------------------------------------------------#
```

```
  # scale premium income and claims so that income is 1 per period
  # this is just how we use the model is constructed
  claim <- claim/income
  income <- 1

  claim_process <- claim*rbinom(time, n, p)
  process <- c(surplus_0)

  for (index in 1:time){
    process <- append(process, process[length(process)] + 1
                      - claim_process[index])
    if (process[length(process)] <= 0){break}
  }

  process <- append(process, rep(process[length(process)],
                                 time - index))

 return(process)
}
```

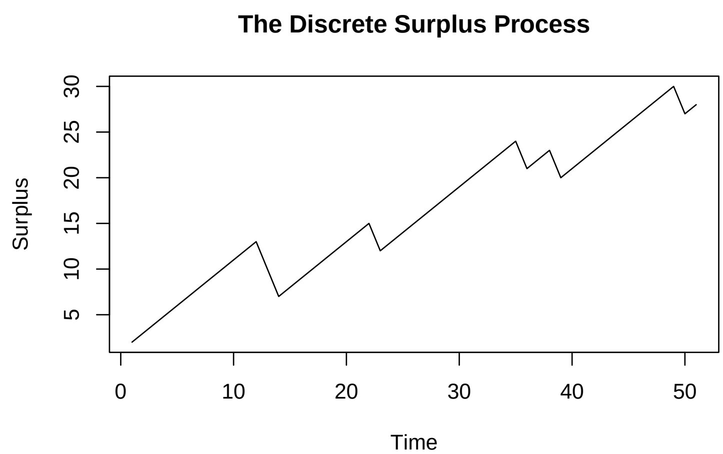


Figure 3: A sample run of the discrete surplus process simulation in R

1.2.1 Other Examples

- Page 119-120 *Insurance Risk and Ruin*, David C. M. Dickson (2016)
- Page 9-10 *Basic Ruin Theory Slides*