ON THE EXACT CALCULATION OF THE AGGREGATE CLAIMS DISTRIBUTION IN THE INDIVIDUAL LIFE MODEL

By Karl-Heinz Waldmann

Institut für Wirtschaftstheorie und Operations Research, Universität Karlsruhe

Abstract

An iteration scheme is derived for calculating the aggregate claims distribution in the individual life model. The (exact) procedure is an efficient reformulation of De Pril's (1986) algorithm, considerably reducing both the number of arithmetic operations to be carried out and the number of data to be kept at each step of iteration. Scaling functions are used to stabilize the algorithm in case of a portfolio with a large number of policies. Some numerical results are displayed to demonstrate the efficiency of the method.

Keywords

Individual life model; aggregate claims distribution; De Pril algorithm.

1. Introduction

Consider a portfolio of m independent life insurance policies. Suppose each policy to have an amount at risk $i \in I = \{1, \ldots, a\}$ and a mortality rate q_j with $j \in J = \{1, \ldots, b\}$. Let m_{ij} denote the number of all policies with amount at risk i and mortality rate q_j.

In the individual risk model the total amount of claims, S, is the sum $S = X_1 + \ldots + X_m$ of the m individual claims $X_1, \ldots, X_m$ produced by the policies. The distribution of S, $f(s) = P(S = s)$, referred to as the aggregate claims distribution, can be obtained by successively convoluting the m two-point distributions of the individual claims. Since the numerical calculation of an m-fold convolution is usually very time-consuming, numerous approximations can be found in the literature. See, e.g., Beard, Pentikäinen and Pesonen (1984) for more details. The method derived in De Pril (1986) is a remarkable progress in computing the distribution of S exactly. Compared with Panjer's (1981) recursion formula, however, which can be thought of as the counterpart within the collective risk model, the computing time remains large (cf. Kuon, Reich and Reimers (1987), De Pril (1988), Reimers (1988)).

In the present paper we shall reformulate the iteration scheme underlying the method of De Pril (1986). A (much) more efficient organization of the data will considerably reduce both the number of arithmetic operations to be carried out and the number of data to be kept at each step of interation. Further, we shall stabilize the algorithm by introducing a suitable scaling function. This scaling function will enable us to apply the algorithm to a portfolio with an

essentially larger number of policies. Finally, some numerical results will be displayed to demonstrate the efficiency of the method.

2. THE AGGREGATE CLAIMS DISTRIBUTION

For $j \in J$, we set $p_j = 1 - q_j$, $z_j = q_j/p_j$, $m_j = \Sigma_{i \in I} m_{ij}$, and $c = \Sigma_{i \in I} \Sigma_{j \in J} i\, m_{ij}$. Further, we use $[x]$ to denote the greatest integer less than or equal to x.

It has been shown in DE PRIL (1986) that the aggregate claims distribution can be computed recursively via

$$f(0) = \prod_{j=1}^{b} (p_j)^{m_j} \tag{1}$$

and for $s = 1, \ldots, c$

$$f(s) = \frac{1}{s} \sum_{i=1}^{\min(a,\, s)} \sum_{k=1}^{[s/i]} g(i, k) f(s - ki) \tag{2}$$

where

$$g(i, k) = (-1)^{k+1} i \sum_{j=1}^{b} m_{ij} z_j^k \tag{3}$$

Theorem 1: Equation (2) can be written as

$$f(s) = \frac{1}{s} \sum_{i=1}^{\min(a,\, s)} \sum_{j=1}^{b} i\, m_{ij}\, r(s, i, j) \tag{4}$$

where, for all $i \in I$, $j \in J$, $i \leq s$

$$r(s, i, j) = z_j \{ f(s - i) - r(s - i, i, j) \} \tag{5}$$

and $r(s, i, j) = 0$ otherwise.

Proof: Let

$$r(s, i, j) := \sum_{k=1}^{[s/i]} (-1)^{k+1} z_j^k f(s - ki)$$

Then, utilizing

$$\begin{aligned} r(s, i, j) &= z_j \left\{ f(s - i) - \sum_{k=2}^{[s/i]} (-1)^{(k-1)+1} z_j^{k-1} f(s - i - (k-1) i) \right\} \\ &= z_j \left\{ f(s - i) - \sum_{k=1}^{[(s-i)/i]} (-1)^{k+1} z_j^k f(s - i - ki) \right\} \\ &= z_j \{ f(s - i) - r(s - i, i, j) \} \end{aligned}$$

the assertion immediately follows from (2). □

https://doi.org/10.2143/AST.24.1.2005083 Published online by Cambridge University Press

Equations (4) and (5) can be thought of as an efficient reformulation of equation (2). The superiority results from

(a) a lower number of arithmetic operations to be carried out at each step of iteration
(b) arrays of smaller size to keep the data needed for further iterations

To specify (a), we first study equation (2). Fix (s, i, k). Then, having already computed $g(i, k-1)$, $g(i, k)$ can be obtained as the result of

$$\{i(-1)^k\} \sum_{j=1}^{b} (-z_j) \{m_{ij} z_j^{k-1}\}$$

which can be managed by $b+1$ multiplications and b additions. Two additional multiplications and one subtraction are necessary to compute $g(i, k) f(s-ki)$. Summing over k there is a need of $(b+3)[s/i]$ multiplications and $(b+1)[s/i]$ additions/subtractions.

On the other hand, by applying equations (4) and (5), for fixed (s, i, j), one multiplication and two subtractions are necessary to compute $r(s, i, j)$. Further, one additional multiplication is needed to obtain $\{i\, m_{ij}\}\, r(s, i, j)$. Summing over j, there is a need of $2b$ multiplications and $2b$ additions/subtractions.

Now let $\xi_m(s)$ (resp. $\xi_a(s)$) denote the number of multiplications (resp. additions/subtractions) to be saved by applying equations (4) and (5) in place of equation (2) at stage s of iteration. Then it easily follows that

$$\xi_m(s) = \sum_{i=1}^{\min(a, s)} \{(b+3)[s/i] - 2b\} \approx \{(b+3) \log (a+1)\} s - 2ab$$

$$\xi_a(s) = \sum_{i=1}^{\min(a, s)} \{(b+1)[s/i] - 2b\} \approx \{(b+1) \log (a+1)\} s - 2ab$$

where use has been made of $\log (a+1) < \sum_{i=1}^{a} 1/i < 1 + \log (a)$ (cf. e.g., Ross (1983)).

Now let us specify (b). To apply iteration scheme (2), an array with ac (resp. $c+1$) elements is needed to keep $g(i, k)$ (resp. $f(s-ki)$) for further iterations. On the other hand, utilizing equations (4) and (5), an efficient implementation of $r(s, i, j)$ (resp. $f(s-i)$) needs an array with $a(a+1)b/2$ (resp. $a+1$) elements only.

To illustrate the basic idea underlying the implementation of $r(s, i, j)$, observe (see Figure 1) that the $r(s, i, j)$ within the upper triangle (solid line) have to be kept at stage s, while at stage $s+1$ the $r(s, i, j)$ of the lower triangle (dashed line) have to be retained.

To manage these data in an efficient way, we rearrange the elements of the upper triangle in an array with $a(a+1)/2$ rows and b columns, and, switching to the lower triangle, we replace the entries of $(s-i, i, j)$ (not needed any longer) by the ones of (s, i, j) (to be kept for further use) and let the other entries unchanged.

https://doi.org/10.2143/AST.24.1.2005083 Published online by Cambridge University Press

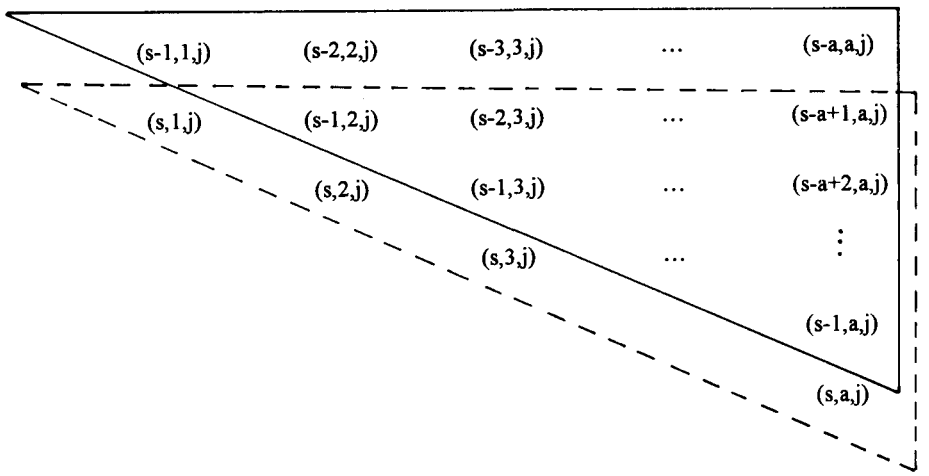

FIGURE 1. Actualization of the data.

Formally, we introduce

$$v_i = i(i-1)/2+1$$
$$w_i = 0, \qquad i \in I$$

and actualize w_i at each step $s\,(s \geq 1)$ of iteration according to

$$w_i = \begin{cases} w_i + 1, & \text{if} \quad w_i < i-1 \\ 0 & \text{otherwise} \end{cases}$$

Then w_i coincides with s modulo i and $(v_i + w_i, j)$ is the position in the array, in which the entry of (s, i, j) can be found.

3. STABILIZATION OF THE ALGORITHM WITH RESPECT TO UNDERFLOW/OVERFLOW

Applying the algorithm to a portfolio with a large number of contracts, the initial value $f(0)$ is close to zero. This fact may cause an underflow followed by an abort or irregular running of the procedure.

To discuss this aspect in more detail, let ω and Ω denote the smallest and greatest numbers that can be represented on the computer to carry out the algorithm. Suppose $f(0) < \omega$. Then the algorithm stops with an underflow. On the other hand, by formally setting $f(0)$ equal to zero, the sequence $f(s)$ of iterates degenerates to a sequence that has all its elements equal to zero, which is not consistent with the property of being a probability mass function.

There are a variety of ways to overcome this difficulty. Three methods of different efficiency and/or applicability are to be stated as methods 1 to 3. There $f^*(s)$, $0 \leq s \leq c$, is used to denote the sequence of transformed iterates.

https://doi.org/10.2143/AST.24.1.2005083 Published online by Cambridge University Press

Method 1: Suppose

$$f^*(s) = \gamma f(s), \qquad 0 \le s \le c$$

for some constant γ with $\omega < \gamma f(0) < \Omega$. Then the transformed iterates $f^*(s)$ can be obtained by formally starting (4) (resp. (2)) with $\gamma f(0)$ in place of $f(0)$. □

The use of a constant scaling function is the simplest way to stabilize the algorithm. A more refined method is to combine a constant scaling function with an exponential scaling function, which has been suggested by PANJER and WILLMOT (1986) within the collective risk theory.

Method 2: Suppose

$$f^*(s) = \gamma e^{-\alpha(s+\beta)} f(s), \qquad 0 \le s \le c$$

where α, β, γ are constants with $0 \le \alpha \le 0.5$, $\gamma > 0$, and

$$\beta = \sum_{i=1}^{a} \sum_{j=1}^{b} m_{ij} \log(p_j) \tag{6}$$

To compute $f^*(s)$, iteration scheme (4) has to be reformulated as

$$f^*(0) = \gamma e^{(1-\alpha)\beta}$$

$$f^*(s) = \frac{1}{s} \sum_{i=1}^{\min(a,\, s)} \sum_{j=1}^{b} i\, m_{ij}\, r^*(s, i, j), \qquad 1 \le s \le c$$

where, for all $i \in I$, $j \in J$

$$t(i, j) = z_j e^{-\alpha i}$$

$$r^*(s, i, j) = t(i, j)\{f^*(s-i) - r^*(s-i, i, j)\}, \qquad i \le s$$

and $r^*(s, i, j) = 0$ otherwise. □

Method 2 starts with a larger initial value as well as method 1 and additionally reduces the increase of the iterates. For large s, however, things may change and the transformation may lead to an ealier abort on account of an underflow. Our third method is one way to overcome this principal difficulty. It again starts with a larger initial value, reduces the increase of the iterates for $s \le E(S)$, and, additionally, reduces the decrease of the iterates for $s > E(S)$.

Method 3: Suppose

$$f^*(s) = \gamma e^{\alpha(s-\mu)^2} f(s), \qquad 0 \le s \le c$$

https://doi.org/10.2143/AST.24.1.2005083 Published online by Cambridge University Press

where

$$\alpha = -\beta/\mu^2$$

$$\mu = E(S) = \sum_{j=1}^{b} m_j q_j$$

and β as in (6). To compute $f^*(s)$, the modified iteration scheme reads

$$f^*(0) = \gamma$$

$$f^*(s) = \frac{1}{s} \sum_{i=1}^{\min(a,s)} \sum_{j=1}^{b} i\, m_{ij}\, r^*(s, i, j), \qquad 1 \le s \le c$$

where, for all $i \in I$, $j \in J$

$$t(s, i, j) = \begin{cases} z_j e^{\alpha i (2(s-\mu)-i)}, & i \le s \le 2a-1 \\ t(s-i, i, j)\, e^{2\alpha i^2}, & 2a \le s \le c \end{cases}$$

$$r^*(s, i, j) = t(s, i, j)\{f^*(s-i) - r^*(s-i, i, j)\}, \qquad i \le s \le c$$

and $r^*(s, i, j) = 0$ otherwise. □

It is not surprising that the last scaling function is superior to the other ones, since it is stimulated by the central limit theorem and thus best utilizes the asymptotic behavior of S as $m \to \infty$. Some numerical results to be given in the next section will illustrate the efficiency. We finally remark that $t(s, i, j)$ and $r^*(s, i, j)$ can be implemented in the same way as $r(s, i, j)$.

4. NUMERICAL RESULTS AND DISCUSSION

We consider as a starting point the portfolio discussed in GERBER (1979), p. 53.

q_j	m_{ij}				
0.03	2	3	1	2	—
0.04	—	1	2	2	1
0.05	—	2	4	2	2
0.06	—	2	2	2	1

Since the portfolio consists of 31 policies only, there is no need for a reformulation or stabilization of the algorithm. We therefore expand the portfolio by considering km_{ij} policies in place of m_{ij} (for all $i \in I$ and $j \in J$).

Let $k = 5000$ (corresponding to 155 000 policies) to illustrate the numerical progress resulting from the application of equations (4) and (5) in place of

https://doi.org/10.2143/AST.24.1.2005083 Published online by Cambridge University Press

equation (2). Then, being interested in computing the aggregate claims distribution up to the smallest c^* with $P(S > c^*) \leq 10^{-4}$, there is a saving of more than $4.4 * 10^9$ multiplications and a saving of more than $3.1 * 10^9$ additons/subtractions. Moreover, the arrays to be kept at each step of iteration can be reduced by 140 851 elements.

The maximal k implying a stable algorithm has been determined on the basis of extended numbers (i.e. $\omega = 1.9 * 10^{-4951}$, $\Omega = 1.1 * 10^{4932}$). There stable means that the algorithm does not stop with an underflow or overflow and that both $|E'(S) - E''(S)|/E''(S) \leq 10^{-5}$ and $|\mathrm{Var}'(S)^{1/2} - \mathrm{Var}''(S)^{1/2}|/\mathrm{Var}''(S)^{1/2} \leq 10^{-5}$ hold, where $E'(S)$, $\mathrm{Var}'(S)$ are determined with help of the probability mass function of S and $E''(S)$, $\mathrm{Var}''(S)$ result from the moments of the individual claims and the properties of expectation and variance. The maximal k and the associated number of policies to be obtained in this way for $\gamma = 10^{4500}$ are displayed in Table 1.

TABLE 1

STABILITY OF THE ALGORITHMS UNDER CONSIDERATION ($\gamma = 10^{4500}$)

Method	maximal k	number of policies
Equations (4) and (5)	7 900	244 900
Method 1	15 100	468 100
Method 2 ($\alpha = 0.31$)	22 100	685 100
Method 3	80 100	2 483 100

Stability of our numerical results thus means stability with respect to the first two moments. For a more theoretial treatment of the numerical stability of recursive formulae the reader is referred to PANJER and WANG (1993).

ACKNOWLEDGEMENT

I would like to thank the referees for their detailed and helpful comments.

REFERENCES

BEARD, R. E., PENTIKÄINEN, T. and PESONEN, E. (1984) *Risk Theory*. 3rd edition. Chapman and Hall, London.

DE PRIL, N. (1986) On the exact computation of the aggregate claims distribution in the individual life model. *ASTIN Bulletin* **16**, 109–112.

DE PRIL, N. (1988) Improved Approximations for the Aggregate Claims Distribution of a Life Insurance Portfolio. *Scan. Actuarial J.* **1988**, 61–68.

GERBER, H. U. (1979) *An Introduction to Mathematical Risk Theory*. Huebner Foundation Monograph 8, Philadelphia.

KUON, S., REICH, A. and REIMERS, L. (1987) Panjer vs. Kornya vs. De Pril: A comparison from a practical point of view. *ASTIN Bulletin* **17**, 183–191.

PANJER, H. H. (1981) Recursive evaluation of a family of compound distributions. *ASTIN Bulletin* **12**, 22–26.

PANJER H.H. and WILLMOT, G.E. (1986) Computational aspects of recursive evaluation of compound distributions. *Insurance: Mathematics and Economics* **5**, 113–116.
PANJER, H.H. and WANG, S. (1993) On the Stability of Recursive Algorithms. *ASTIN Bulletin*, to appear.
REIMERS, L. (1988) Letter to the Editor. *ASTIN Bulletin* **18**, 113–114.
ROSS, S.M. (1983) *Stochastic Processes*. John Wiley, New York.

Prof. Dr. KARL-HEINZ WALDMANN
Institut für Wirtschaftstheorie und Operations Research, Universität Karlsruhe, Postf. 6980, D-76128 Karlsruhe.