An. Şt. Univ. Ovidius Constanţa Vol. 12(1), 2004, 99–114

COMPARING CONVOLUTIONS WITH DHAENE-VANDEBROEK AND DE PRIL'S RECURSIONS FROM A COMPUTATIONAL POINT OF VIEW

Raluca VERNIC

Abstract

In this paper, we compare from a computational point of view three algorithms used in a specific actuarial context to evaluate the distribution function of the aggregate claims. For comparison, the number of multiplications involved is considered.

1 Introduction

One quantity of great interest in insurances is the aggregate claims distribution, which is usually estimated for a homogeneous portfolio of policies during a fixed period. Since there exists several methods to evaluate the aggregate claims distribution, an important question is which one performs better from the computational point of view and under what circumstances. Between the papers that attempt to give an answer to these questions, usually for some particular algorithms, we recall [1], [2], [5], [6] etc.

In the present paper we compare three algorithms by the number of multiplications required. The first two algorithms are exact ones, though their accuracy was disputed because of the rounding and scaling involved (see e.g. [6]), while the third algorithm is clearly an approximate one.

The paper is structured as follows: in the second section we present the algorithms, while section three is concerned with the evaluation and comparison of the number of multiplications. This section is divided into several subsections because, depending on the support of the severity distribution, the number of multiplications required by the same algorithm can be significantly different.

Key Words: Aggregate claims distribution; Convolutions; Recursive algorithms; Computational aspects.

2 The algorithms

Consider a portfolio of n policies. Let us divide this portfolio into a number of classes by clustering all policies with the same probability of a strictly positive claim amount and with the same severity distribution, as displayed in the next table. The severity distribution of a policy corresponds to the conditional distribution of the claim amount, given that a claim has occurred.

		claim probability					
		q_1	q_2	$\cdots$	q_j	$\cdots$	q_b
	$g_1(x)$						
	$g_2(x)$				$\cdots$		
claim	$\cdots$						
amount	$g_i(x)$			$\cdots$	n_{ij}		
distribution	$\cdots$						
	$g_a(x)$						

The class (i,j), $i=1,\ldots,a$, $j=1,\ldots,b$, contains all policies with severity distribution g_i and claim probability $q_j = 1-p_j$. The number of policies in class (i,j) is denoted by n_{ij}. It is assumed that for each j, $0<q_j<1$ and that the claim amounts of the individual policies are positive integer multiples of some convenient monetary unit, so that for each i, $g_i(x)$ is defined for $x=1,2,\ldots$

The probability that the aggregate claims S equal s is denoted by $p(s) := \Pr[S=s]$. We assume that the claim amounts of the policies are mutually independent.

We will now recall three methods for evaluating $\Pr[S=s]$ and we will compare them by the number of multiplications involved when calculating the distribution function (df) of S up to the value s.

2.1 Convolutions method

This is the straightforward method for evaluating $\Pr[S=s]$.

We denote by X_{ij} the r.v. claim amount for a policy belonging to the class (i,j). Then the distribution of X_{ij} is given by

$$X_{ij}\begin{pmatrix} 0 & 1 & 2 & ... & k & ... \\ p_j & q_jg_i(1) & q_jg_i(2) & ... & q_jg_i(k) & ... \end{pmatrix},\ i=1,\ldots,a,\ j=1,\ldots,b. \tag{1}$$

In order to obtain the probabilities of the aggregate claims S by simply convolutions, we rewrite it as

$$S = \sum_{i=1}^{a} \underbrace{\sum_{j=1}^{b} (X_{ij} + ... + X_{ij})}_{n_{ij}-times} = X_1 + ... + X_n.$$

Then the probabilities of S can be obtained starting from

$$p_S^{*1}(s) = p_{X_1}(s), \; s = 0, 1, ...$$

$$p_S^{*k}(s) = \sum_{y=0}^{s} p_S^{*k-1}(s-y)\, p_{X_k}(y), \; s = 0, 1, ..., \; k = 2, 3, ..., n, \qquad (2)$$

where $p_{X_k}(y) := P(X_k = y), \; k = 1, ..., n$. Then $p(s) = p_S^{*n}(s)$.

2.2 Dhaene-Vandebroek recursion

In [4], Dhaene & Vandebroek proposed the following exact recursive formula

$$p(s) = \frac{1}{s} \sum_{i=1}^{a} \sum_{j=1}^{b} n_{ij} \; v_{ij}(s), \qquad s = 1, \; 2, ... \qquad (3)$$

with the initial value given by

$$p(0) = \prod_{i=1}^{a} \prod_{j=1}^{b} (p_j)^{n_{ij}} \qquad (4)$$

and where the coefficients $v_{ij}(s)$ are determined by

$$v_{ij}(s) = \frac{q_j}{p_j} \sum_{x=1}^{s} g_i(x) \left[x \; p(s-x) - v_{ij}(s-x) \right], \qquad s = 1, \; 2, ... \qquad (5)$$

and $v_{ij}(s) = 0$ otherwise.

Notice the similarity between Panjer's recursion (see [7]) and the recursions in (3) and (5). Moreover, in the particular case when $a = b = 1$, Dhaene-Vandebroek's recursion can be reduced to Panjer's recursion as follows: all the indexes vanish and (3) becomes

$$p(s) = \frac{n}{s} v(s), \; s = 1, 2, ..., \qquad (6)$$

while (5) reduces to

$$\begin{aligned} v(s) &= \frac{q}{p}\sum_{x=1}^{s} g(x)\left[x\ p(s\text{-}x)\text{-}v(s\text{-}x)\right] = \frac{q}{p}\sum_{x=1}^{s}\left[x\ p(s\text{-}x)\text{-}\frac{s\text{-}x}{n}p\left(s\text{-}x\right)\right] g(x) = \\ &= \frac{q}{p}\sum_{x=1}^{s}\frac{(n+1)\,x-s}{n}p\left(s-x\right)g(x),\ \ s=1,2,... \end{aligned}$$

Introducing this into (6), we obtain

$$p(s)=\frac{q}{ps}\sum_{x=1}^{s}\left[(n+1)\,x-s\right]g(x)p\left(s-x\right),\ \ s=1,2,...,$$

which is exactly Panjer's recursion for this particular model (i.e. the frequency distribution here is $Binomial\,(q,n)$).

2.3 De Pril's approximation

An approximative recursive formula for the evaluation of the aggregate claims distribution in the individual model was given by De Pril in [3] as

$$f^{(r)}(s)=\frac{1}{s}\sum_{i=1}^{a}\sum_{k=1}^{min(r,s)}A(i,k)\ \sum_{x=k}^{s}x\ g_i^{*k}(x)\ f^{(r)}(s-x)\ \text{for}\ \ s=1,2,\ldots. \quad (7)$$

with

$$A(i,k)=\frac{(-1)^{k+1}}{k}\sum_{j=1}^{b}n_{ij}\ \left(\frac{q_j}{p_j}\right)^{k}, \quad (8)$$

and the initial value $p(0)$ from (4).

In this case, the convolutions g_i^{*k} are given by

$$g_i^{*k}(s)=\sum_{x=1}^{s-k+1}g_i^{*(k-1)}(s-x)g_i\left(x\right),\ \ k=2,...,s, \quad (9)$$

since $g_i\left(x\right)$ is defined for $x\geq 1$, so we must have $s-x\geq k-1$.

3 Evaluating and comparing the number of multiplications

To begin, notice that a division can be seen as a multiplication with the inverse element. Depending on g_is support, the number of multiplications required by the same algorithm can be significantly different. This is why in this section we consider three situations.

In the first case we assume that all g_i are well-defined (i.e. non-zero) for all the values in their support. The support can be 1,2,... , i.e. unbounded (infinite case); or it can be finite, upper-bounded by the value m_i for g_i, i.e. 1,...,m_i (finite case).

In the second case we assume that each g_i can be zero for some values in its definition support.

The last case will include some other particular cases that might require special attention.

3.1 A first case

3.1.1 Infinite case

We will now assume that all g_i are defined for all 1,2,..., so their support is upper unbounded, or its upper bound is very large (i.e. $g_i(x) > 0$ for $x = 1, 2, ...$). Since the upper bound is large enough, we can consider in the following that s is also very large. Let's now count the multiplications involved in the three algorithms.

Convolutions method.

We notice that in order to evaluate the df of S up to value s, all the formulas involved can be stopped at s. We start by counting the multiplications required for the evaluation of the probability functions of all X_{ij}. Since for all $i = 1, \ldots, a,\ j = 1, \ldots, b$ we need only the values $p_j, q_j g_i(1), ..., q_j g_i(s)$, this means exactly abs multiplications.

We also need to calculate $p_S^{*k}(x)$ for $x = 0, ..., s,\ k$ being fixed for the moment. From formula (2), each $p_S^{*k}(x)$ involves $x + 1$ multiplications. For $x = 0, ..., s$ this gives a total of

$$1 + 2 + ... + (s + 1) = \frac{(s+1)(s+2)}{2} \text{ multiplications.}$$

We see that this number does not depend on the convolution order k, so that evaluating all $p_S^{*k}(x)$ for $x = 0, ..., s$ and $k = 2, ..., n$ requires a number of $\frac{(n-1)(s+1)(s+2)}{2}$ multiplications, hence a total of

$$\frac{(n\text{-}1)(s\text{+}1)(s\text{+}2)}{2} + abs = \frac{n-1}{2}s^2 + \left(\frac{3(n\text{-}1)}{2} + ab\right)s + (n\text{-}1) \text{ multiplications.} \tag{10}$$

Dhaene-Vandebroek recursion. Let's now count the multiplications involved:

- one $p(s)$ needs $(ab+1)$ multiplications (see (3)), so that $p(1), ..., p(s)$ requires $(ab+1)\,s$ multiplications;
- for fixed s, formula (5) needs $(2s+1)$ multiplications; we must perform it for $i = 1, ..., a$, $j = 1, ..., b$, i.e. ab times, so we have $ab\,(2s+1)$ multiplications; we also need to evaluate $v_{ij}(1), ..., v_{ij}(s)$, which requires $\sum_{i=1}^{s} ab(2i+1) = ab\left[s\,(s+1)+s\right]$ multiplications.

Then the Dhaene-Vandebroek algorithm involves approximately

$$s\,(ab+1) + ab\left[s\,(s+1)+s\right] = abs^2 + (3ab+1)\,s \tag{11}$$

multiplications.

De Pril's approximation. We will now count the multiplications for this algorithm. We will take r small (which is the usual case), such that $r < s$.

Starting with the convolutions (9), we see from (7) that we need all the convolutions from $g_i^{*2}(2)$ till $g_i^{*r}(s), i = 1, ..., a$. We consider the following table which gives the number of multiplications required by all the convolutions till $g_i^{*r}(s)$:

$g_i^{*2}(2)$:1				
$g_i^{*2}(3)$:2	$g_i^{*3}(3)$:1			
...	...	...	...	
...	...	...	$g_i^{*(r\text{-}1)}(r)$: 2	$g_i^{*r}(r)$:1
...	...	...	...	...
$g_i^{*2}(s\text{-}1)$:$(s\text{-}2)$	$g_i^{*3}(s\text{-}1)$:(($s\text{-}3$)	...	$g_i^{*(r\text{-}1)}(s\text{-}1)$:$(s\text{-}r\text{+}1)$	$g_i^{*r}(s\text{-}1)$:$(s\text{-}r)$
$g_i^{*2}(s)$:$(s\text{-}1)$	$g_i^{*3}(s)$:$(s\text{-}2)$	...	$g_i^{*(r\text{-}1)}(s)$:$(s\text{-}r\text{+}2)$	$g_i^{*r}(s)$:$(s\text{-}r\text{+}1)$
Total: $\frac{(s\text{-}1)s}{2}$	$\frac{(s\text{-}2)(s\text{-}1)}{2}$	...	$\frac{(s\text{-}r\text{+}2)(s\text{-}r\text{+}3)}{2}$	$\frac{(s\text{-}r\text{+}1)(s\text{-}r\text{+}2)}{2}$

Therefore, for a fixed i, all the convolutions till $g_i^{*r}(s)$ need

$$\begin{aligned}
\sum_{j=s\text{-}r+2}^{s} \frac{(j\text{-}1)\,j}{2} &= \sum_{j=1}^{r-1} \frac{(s\text{-}j)\,(s\text{-}j\text{+}1)}{2} = \\
&= \frac{1}{2}\left[\left(s^2\text{+}s\right)(r\text{-}1)\text{+}\frac{(r\text{-}1)\,r\,(2r\text{-}1)}{6}\text{-}\,(2s\text{+}1)\,\frac{(r\text{-}1)\,r}{2}\right] = \\
&= \frac{r\text{-}1}{2}\left[s^2\text{-}\,(r\text{-}1)\,s\text{+}\frac{(r\text{-}2)\,r}{3}\right] \text{ multiplications.}
\end{aligned}$$

Taking now $i = 1, ..., a$, we obtain a total of $\frac{a\,(r-1)}{2}\left[s^2 - (r-1)\,s + \frac{(r-2)\,r}{3}\right]$ multiplications.

Since the evaluation of $A(i,k)$ does not involve s, we neglect it for the moment. We will now consider $f^{(r)}(x)$ from (7). We have two cases:

1. If $x < r$ then it needs: one multiplication from $\frac{1}{s}$, ax from the multiplications with $A(i,k)$, and for the last sum $2a\,(x + ... + 1)$. This gives a total of $1 + a(x^2 + 2x)$ multiplications.

2. If $x \geq r$ then we have: one multiplication from $\frac{1}{s}$, ar from the multiplications with $A(i,k)$, and for the last sum $2a\,(x + ... + (x - r + 1))$. This gives a total of $1 + a(2xr - r^2 + 2r)$ multiplications.

For $f^{(r)}(x), x = 1, ..., s$, we finally get a number of multiplications equal to

$$\sum_{x=1}^{r-1}\left[1{+}a(x^2{+}2x)\right] + \sum_{x=r}^{s}\left[1{+}a(2xr{-}r^2{+}2r)\right] = ars^2 + \left(1{-}ar^2{+}3ar\right)s + \frac{ar\,(r{-}1)\,(2r{-}7)}{6}.$$

So, for the De Pril approximation, we need a total amount of multiplications equal to

$$\frac{a\,(r{-}1)}{2}\left[s^2{-}\,(r{-}1)\,s + \frac{(r{-}2)\,r}{3}\right] + ars^2 + \left(1{-}ar^2{+}3ar\right)s + \frac{ar\,(r{-}1)\,(2r{-}7)}{6} =$$

$$= a\frac{3r-1}{2}s^2 - \frac{3ar^2 - 8ar + a - 2}{2}s + \frac{ar\,(r-1)\,(r-3)}{2}. \qquad (12)$$

Here we can add about abr multiplications for the evaluation of $A(i,k), i = 1, ..., a,\ k = 1, ..., r$.

Conclusions. From the arguments above, we can see that the evaluation of the df of S up to value s requires a number of multiplications depending on s^2 for all three algorithms. We will now compare them separately, based mainly on the coefficients of s^2.

1. **Convolutions versus Dhaene-Vandebroek**. We take into consideration (10) and (11). It is easy to see that for fixed ab, Dhaene-Vandebroek requires less multiplications when $n \geq 2ab + 1$, while convolutions are better when $n < 2ab + 1$. This is clear since (11) does not depend on n, while (10) increases with n.

2. **Convolutions versus De Pril**. From (10) and (12) we can see that De Pril is better when $n \geq a\,(3r-1)+1$. Intuitively, one could say that if $n < a\,(3r-1)+1$ convolutions seems better, but numerical evaluation shows that the difference between the number of multiplications of the two algorithms depends in this case also on b: when b increases, the maximum value of n for which convolutions are better than De Pril decreases under $a\,(3r-1)+1$. For example, taking $a = 10$ and $r = 3$, the maximum n for which convolutions give less multiplications is $\begin{cases} 79 \text{ for } b = 5 \\ 78 \text{ for } b = 10 \\ 77 \text{ for } b = 20 \end{cases}$ etc.

3. **Dhaene-Vandebroek versus De Pril**. Looking at (11) and (12) we see that De Pril performs better when $b \geq \dfrac{3r-1}{2}$ (i.e. there are more classes with different claim probabilities). Dhaene-Vandebroek will need less multiplications than De Pril when $b < \dfrac{3r-1}{2}$.

In conclusion, under reasonable assumptions (i.e. when n is large enough), when evaluating the df, the convolutions method will require a larger number of multiplications than the other two methods.

3.1.2 Finite case

We will now assume that all g_i are defined on a finite support, say on $1, 2, ..., m_i$, $i = 1, ..., a$. Then s can take only the values $0, 1, 2, ..., m_+$, where $m_+ = \sum_{i=1}^{a} n_i m_i$. Here $n_i = \sum_{j=1}^{b} n_{ij}$ is the number of policies in class i. Hence, the number of multiplications will be smaller than in the infinite case. Because the sums involved will now have some supplementary restrictions which complicate the counting, we will restrict to the case when $s = m_+$, so we count the multiplications needed to evaluate the entire distribution of S.

Convolutions method. Counting first the multiplications in (1) gives $\sum_{i=1}^{a} m_i$.

Because of the finite support of each g_i, formula (2) becomes

$$p_S^{*k}(s) = \sum_{y=\max\{0, s-m_1'-...-m_{k-1}'\}}^{\min\{s, m_k'\}} p_S^{*k-1}(s\text{-}y)\, p_{X_k}(y)\,,\ \ s{=}0, 1, ..., m_1'{+}...{+}m_k', \tag{13}$$

k=2, 3, ..., n, where $m'_k = \max\{y \,| p_{X_k}(y) > 0\}$. Therefore, it is clear that $m_+ = \sum_{i=1}^{n} m'_i$.

For simplicity, we will now assume that $m'_1 \geq ... \geq m'_n$. Counting the multiplications involved in (13), for fixed k, gives:

- for a fixed $s \in \{0, ..., m'_k - 1\}$, we need $s + 1$ multiplications;
- for a fixed $s \in \{m'_k, ..., m'_1 + ... + m'_{k-1}\}$, we need $m'_k + 1$ multiplications;
- for $s \in \{m'_1 + ... + m'_{k-1} + 1, ..., m'_1 + ... + m'_k\}$, we need $m'_k, m'_k - 1, ...$, and respectively 1 multiplications.

This gives a total of multiplications equal to

$$2\sum_{s=0}^{m'_k-1}(s+1) + (m'_k+1)\left(m'_1+...+m'_{k-1}\text{-}m'_k+1\right) = (m'_k+1)\left(m'_1+...+m'_{k-1}+1\right).$$

Now we sum for $k = 2, ..., n$ and we have

$$\sum_{k=2}^{n}(m'_k + 1)\left(m'_1 + ... + m'_{k-1} + 1\right) =$$

$$= (n\text{-}1) + \sum_{k=2}^{n} m'_k\left(m'_1+...+m'_{k-1}\right) + \sum_{k=2}^{n}\left(m'_1+...+m'_{k-1}+m'_k\right) \text{ multiplications.}$$

Hence, the total number of multiplications is

$$\sum_{i=1}^{a} m_i + (n-1) + \sum_{k=2}^{n} m'_k\left(m'_1+...+m'_{k-1}\right) + \sum_{k=2}^{n}(m'_1 + ... + m'_k). \tag{14}$$

Dhaene-Vandebroek recursion. We start the counting with formula (3), which must be performed for $s = 1, 2, ..., m_+$. This means $(1 + ab)\, m_+$ multiplications.

Because of the finite support of each g_i, the sum in formula (5) will now be $\sum_{x=1}^{\min\{s,m_i\}}$. Therefore, for fixed i, j this formula needs:

- when $s \in \{1, ..., m_i\}$, $1 + 2s$ multiplications;
- when $s \in \{m_i + 1, ..., m_+\}$, $1 + 2m_i$ multiplications for each s.

We obtain a total of

$$m_+ + 2\left[\frac{m_i\left(m_i+1\right)}{2}+m_i\left(m_+-m_i\right)\right] \text{ multiplications.}$$

When $i=1,...,a,\ j=1,...,b$, this gives

$$abm_+ + b\sum_{i=1}^{a} m_i\left(2m_+-m_i+1\right) \text{ multiplications.}$$

Then the total number of multiplications required in this finite case is

$$\left(2ab+1+2b\sum_{i=1}^{a} m_i\right)m_+ + b\sum_{i=1}^{a} m_i\left(1-m_i\right). \tag{15}$$

De Pril's approximation. The number of multiplications required for this approximation are more difficult to calculate.

We start with the convolutions (9), which will now be

$$g_i^{*k}(s)=\sum_{x=\max\{1,s-(k-1)m_i\}}^{\min\{m_i,s-k+1\}} g_i^{*(k-1)}(s-x)g_i\left(x\right),\ s=k,...,km_i,\ k=2,...,r.$$

We will use again a table that contains all the convolutions involved and the corresponding number of multiplications.

$g_i^{*2}(2)$:1	$g_i^{*3}(3)$:1	...	$g_i^{*r}(r)$:1
$g_i^{*2}(3)$:2	...	...	...
...	$g_i^{*3}(m_i+1)$:m_i-1	...	$g_i^{*r}(m_i+r\text{-}2)$:m_i-1
$g_i^{*2}(m_i)$:m_i-1	$g_i^{*3}(m_i+2)$:m_i	...	$g_i^{*r}(m_i+r\text{-}1)$:m_i
$g_i^{*2}(m_i+1)$:m_i	...	...	...
$g_i^{*2}(m_i+2)$:m_i-1	$g_i^{*3}(2m_i+1)$:m_i	...	$g_i^{*r}((r\text{-}1)\,m_i+1)$:$m_i$
...	$g_i^{*3}(2m_i+2)$:m_i-1	...	$g_i^{*r}((r\text{-}1)\,m_i+2)$:$m_i$-1
$g_i^{*2}(2m_i\text{-}1)$:2	...	...	...
$g_i^{*2}(2m_i)$:1	$g_i^{*3}(3m_i)$:1	...	$g_i^{*r}(rm_i)$:1
Total: m_i^2	$m_i\,(2m_i\text{-}1)$	...	$m_i\,[(r\text{-}1)\,m_i\text{-}\,(r\text{-}2)]$

Therefore, for a fixed i, all the convolutions $g_i^{*k},\ k=2,...,r$, need

$$\sum_{k=2}^{r} m_i\left[(k-1)\,m_i-(k-2)\right]=\frac{r-1}{2}m_i\left(rm_i-r+2\right) \text{ multiplications.}$$

For all $i \in \{1, ..., a\}$, this gives a total of

$$\frac{r-1}{2}\sum_{i=1}^{a} m_i \left(rm_i - r + 2\right) \tag{16}$$

multiplications for the convolutions involved.

We will now turn our attention to the multiplications from $f^{(r)}\left(s\right), s = 1, ..., m_+$. First, we have the term $\frac{1}{s}$, that needs a total of m_+ multiplications. Second, the terms $A\left(i,k\right)$ add a total of

$$a\sum_{s=1}^{r} s + ar\sum_{s=r+1}^{m_+} 1 = \frac{ar}{2}\left(2m_+ - r + 1\right) \quad \text{multiplications.} \tag{17}$$

Now, the most difficult part is to count the multiplications that appears in the third sum of (7). For this finite case, these sums will be

$$\sum_{i=1}^{a}\sum_{k=1}^{min(r,s)} A(i,k) \sum_{x=k}^{\min\{s,km_i\}} x\ g_i^{*k}(x)\ f^{(r)}(s-x).$$

Let's start by fixing i and varying s. We assume that r is small enough, such that $r \leq m_i$ for any i. We then have four situations:

1. $s \in \{1, ..., r\}$; this involves

$$\sum_{s=1}^{r} 2\left[s + (s-1) + ... + 1\right] = \sum_{s=1}^{r} s\left(s+1\right) = \frac{r\left(r+1\right)\left(r+2\right)}{3} \quad \text{multiplications.} \tag{18}$$

2. $s \in \{r+1, ..., m_i\}$; this involves the following number of multiplications

$$\sum_{s=r+1}^{m_i} 2\left[s + (s\text{-}1) + ... + \left(s - r + 1\right)\right] = r\sum_{s=r+1}^{m_i} \left(2s\text{-}r+1\right) = r\left(m_i+2\right)\left(m_i\text{-}r\right). \tag{19}$$

3. We fix $s \in \{km_i + 1, ..., (k+1)\, m_i\}, k \in \{1, ..., r-1\}$. Then the last two sums of $f^{(r)}\left(s\right)$ require

$$2\left[m_i + \left(2m_i - 1\right) + ... + \left(km_i - k + 1\right) + \left(s - k\right) + \left(s - k - 1\right) + ... + \left(s - r + 1\right)\right] =$$

$$= m_i k^2 + m_i k + 2\left(r - k\right)s - r^2 + r \text{ multiplications.}$$

We now have to sum this for s and k, and after some calculation we get

$$\sum_{k=1}^{r-1}\sum_{s=km_i+1}^{(k+1)m_i}\left[m_ik^2+m_ik+2\,(r\text{-}k)\,s\text{-}r^2+r\right]=$$

$$m_i\sum_{k=1}^{r-1}\left[-m_ik^2+(2rm_i\text{-}1)\,k+r\,(m_i\text{-}r+2)\right]=$$

$$=\frac{r\,(r-1)\,m_i}{2}\left[\frac{4r+7}{3}m_i-2r+3\right] \tag{20}$$

multiplications.

4. $s\in\{rm_i+1,...,m_+\}$; this involves a number of multiplications

$$\sum_{s=rm_i+1}^{m_+}2\,[m_i+(2m_i-1)+...+(rm_i-r+1)]= \tag{21}$$

$$r\,(m_+-rm_i)\,[(r+1)\,m_i-r+1]\,.$$

Now we must add all the values in (18)-(21) and let i vary, so, after some calculation, we obtain a number of multiplications equal to

$$\frac{ar\,(r+1)\,(r+2)}{3}+$$

$$+r\sum_{i=1}^{a}\left[-m_i^2\frac{2r^2+3r+1}{6}+\left(\frac{r+1}{2}+(r+1)\,m_+\right)m_i\text{-}2r\text{-}\,(r\text{-}1)\,m_+\right]=$$

$$=r\left\{\left[(r{+}1)\sum_{i=1}^{a}m_i\text{-}a\,(r\text{-}1)\right]m_+-\frac{(r{+}1)\,(2r{+}1)}{6}\sum_{i=1}^{a}m_i^2+\frac{r{+}1}{2}\sum_{i=1}^{a}m_i+\frac{a\,(r\text{-}2)\,(r\text{-}1)}{3}\right\}.$$

In conclusion, De Pril needs a total number of multiplications obtained adding the last value to (16), (17) and m_+. We get

$$\left[1{+}r\,(r{+}1)\sum_{i=1}^{a}m_i\text{-}ar\,(r\text{-}2)\right]m_+\text{-}\frac{\left(r^2{+}2\right)r}{3}\sum_{i=1}^{a}m_i^2+(2r\text{-}1)\sum_{i=1}^{a}m_i+ \tag{22}$$

$$+\frac{ar\,(r\text{-}1)\,(2r\text{-}7)}{6}.$$

Here we can add about abr multiplications for the evaluation of $A(i,k), i=1,...,a,\ k=1,...,r.$

Particular case: $m_1 = ... = m_a = m$. It is more easy to compare the number of multiplications involved in the three algorithms in this particular case. Besides, the assumption is not very restrictive.

For the convolutions method, (14) becomes in this case:

$$\frac{n\,(n-1)}{2}m^2 + \frac{n^2+n+2\,(a-1)}{2}m + (n-1)\,, \tag{23}$$

for Dhaene-Vandebroek (15) gives

$$ab(2n-1)m^2 + (ab+2abn+n)m, \tag{24}$$

while for De Pril (22) reduces to

$$\frac{ar}{3}\left[3n\,(r{+}1)\,\text{-}r^2\text{-}2\right]m^2{+}\left[a\,(2r\text{-}1)\,\text{-}anr\,(r-2)\,{+}n\right]m{+}ar\left[b{+}\frac{(r\text{-}1)\,(2r\text{-}7)}{6}\right]. \tag{25}$$

Conclusions. The following limiting values are approximative, but one can get a clear idea.

1. **Convolutions versus Dhaene-Vandebroek**. We compare the terms in m^2 from (23) and (24). Some calculations show that Dhaene-Vandebroek requires less multiplications when $n \geq 4ab+1$, while convolutions are better when $n < 4ab+1$.

2. **Convolutions versus De Pril**. From the terms in m^2 in (23) and (25) we can see that De Pril is better than convolutions if $n > 1{+}2ar\,(r+1)\,.$

3. **Dhaene-Vandebroek versus De Pril**. Looking again at the terms in m^2 in (24) and (25) we see that De Pril performs better than Dhaene-Vandebroek when $2b > r\,(r+1)$ (i.e. there are more classes with different claim probabilities, like in the infinite case).

Hence, we have the same conclusion: when evaluating the df, if n is large enough, the convolutions method will require a larger number of multiplications than the other two methods.

3.1.3 A special case

A very particular case often considered for its simplicity is when $a = 1$ and $g_1(x) = 1$ for a certain $x \in \mathbf{N}^*$, i.e. the r.v. claim amount for a policy in class $(1, j)$ is given by $X_{1j}\begin{pmatrix} 0 & x \\ p_j & q_j \end{pmatrix}$. Usually, $x = 1$, but in the following we consider a general x.

The number of multiplications for evaluating the entire distribution of S by the convolutions method can be easy obtained as

$$n^2 + n - 2. \tag{26}$$

For Dhaene-Vandebroek, we have $(b+1)\,n$ multiplications from (3) and $2nb$ from (5), hence a total of

$$(3b+1)\,n. \tag{27}$$

It is easy to see that when $n > 3b+1$, Dhaene-Vandebroek needs less multiplications than the convolutions.

Back to De Pril, (8) requires br multiplications, and (7) requires $(2r+1)\,n - r^2 + r$, while the convolutions in (9) vanish since $g_i^{*k}(kx) = 1$. Hence we have a total of

$$(2r+1)\,n - r^2 + (b+1)\,r.$$

Comparing De Pril with Dhaene-Vandebroek, we can see that the first one is better when $2r < 3b$.

In conclusion, for n and b large enough, we have the following order: De Pril is better than Dhaene-Vandebroek, which is better than convolutions.

3.2 A second case

In this case we assume that any g_i can be zero for some values in its definition support. Then some multiplications will vanish and in this case it's very difficult to directly count the number of multiplications involved. So the best way to do it is by programming the formulas involved and simultaneously counting the multiplications. In the table below we present some numerical examples. The general conclusion, that convolutions are better just for small n, seems to hold in this case also.

In the following example we consider $a = b = 3,\ n_{11} = n_{22} = n_{33},\ n_{ij} = 0$ for $i \neq j,\ q_1 = 0.4,\ q_2 = 0.3,\ q_3 = 0.2$ and

$$g_1 \begin{pmatrix} 2 & 4 \\ 0.3 & 0.1 \end{pmatrix},\ g_2 \begin{pmatrix} 1 & 3 \\ 0.2 & 0.1 \end{pmatrix},\ g_3 \begin{pmatrix} 5 \\ 0.2 \end{pmatrix}.$$

We denote by $s_{\max}$ the maximum requested value of S, i.e. we must calculate $\Pr[S = s]$ for $s = 0, ..., s_{\max}$. The following table contains the number of multiplications required by convolutions (column 3), De Pril (column 4) and Dhaene-Vandebroek (column 5). In the last column we have the algorithm giving the smallest number of multiplications for a specific choice of n_{ii} and $s_{\max}$.

n_{ii}	$s_{\max}$	Convolutions	De Pril	Dhaene-Vandebroek	Minimum
2	24	123	1088	1164	Convolutions
5	50	868	2440	2438	Convolutions
10	50	2839	2440	2438	Dhaene-Vandebroek
10	100	4179	5040	4888	Convolutions
20	100	5895	5040	4888	Dhaene-Vandebroek

3.3 Conclusion

For all the cases considered, the conclusion is the same: under reasonable assumptions (i.e. when the portfolio is large enough), the convolutions method will require a larger number of multiplications than the other two methods.

Acknowledgment. This paper was realized with the financial support of the Dutch Organization for Scientific Research (NWO no. 048.031.2003.001).

References

[1] Bruno, M.G. and Tomassetti, A., *On the computation of convolution in actuarial problems*. IME (Insurance: Mathematics & Economics) Congress, Rome 2004 (www.ime2004rome.com).

[2] De Pril, N., *Improved recursions for some compound Poisson distributions*. Insurance: Mathematics & Economics **5** (1986), 129-132.

[3] De Pril, N., *The aggregate claims distribution in the individual life model with arbitrary positive claims*. ASTIN Bulletin **19** (1989), 9-24.

[4] Dhaene, J. and Vandebroek, M., *Recursions for the individual model*. Insurance: Mathematics & Economics **16** (1995), 31-38.

[5] Hipp, C., *Speedy Panjer for phasetype claims*. IME (Insurance: Mathematics & Economics) Congress, Rome 2004 (www.ime2004rome.com).

[6] Kaas, R., *How to (and how not to) compute stop-loss premiums in practice*. Insurance: Mathematics & Economics **13** (1994), 241-254.

[7] Panjer, H.H., *Recursive evaluation of a family of compound distributions*. ASTIN Bulletin **12** (1981), 22-26.

"Ovidius" University of Constanta
Department of Mathematics and Informatics,
900527 Constanta, Bd. Mamaia 124
Romania
e-mail: rvernic@univ-ovidius.ro