POLITECNICO DI TORINO

Master's Degree in Mathematical Engineering

Master's Degree Thesis

Reinsurance pricing model with Python application on client data

Supervisor
Prof.ssa Patrizia Semeraro

Candidate
Maria Margherita Lovera

A.Y. 2020/2021

Introduction

This thesis has been developed during a 6 months internship at Sompo International, a leading global specialty provider of property and casualty insurance and reinsurance with headquarter in Bermuda. Sompo International Reinsurance operates with experienced underwriting teams worldwide providing a broad range of reinsurance products. The main purpose of this essay is to analyze the structure and methods that characterize a reinsurance pricing process, to present a client case to have a deeper look into the company's methodologies and to introduce an alternative method to the company's standard approach. The above mentioned description and analysis are done in line with the Actuarial Team's work in the Zurich office. The data we have been working on come from a cedant that provided its submission in September and that we priced by the end of October. The central part of this thesis focuses on transposing on Python the pricing done within the company, followed by the implementation of a method well known in the literature but not available yet in the company's Excel pricing tool: the Panjer recursion. This work will most likely be useful for the Zurich Actuarial Team since it presents a method that can be easily implemented and that can be used as an alternative or comparison with the widely used and known Monte Carlo simulation. The first Chapter is a wide introduction to reinsurance where we present its origins, its forms and its types and methods. The second Chapter contains all the things we need to know when dealing with a pricing such as the basics of pricing, the ground up loss models that lie behind a pricing and the different possible approaches to it. The third Chapter deals with treaty features, both proportional and non-proportional, and their evaluation through different methods such as the Monte Carlo simulation and the Panjer recursion. Finally, the fourth and last Chapter walks us through the implementation in Python of the theory and methods above presented on real client data provided by the company.

Table of Contents

Chapter 1

Introduction to reinsurance

1.1 The origins of reinsurance and its current role

1.1.1 Historical background

Just as individuals and businesses have an interest in protecting themselves against certain risks, insurance companies need to buy cover against risks they accept under primary insurance contracts.
Nowadays reinsurance, commonly referred to as "insurance for insurers", is defined as the transfer from one insurer (the *primary insurer*) to another (the *reinsurer*) of some or all of the financial consequences of certain liabilities and premium covered by the primary insurer's policies.
It is said that the oldest known treaty of a reinsurance nature was concluded in 1370 in Genoa. However, at that time reinsurance was not the usual method of risk sharing, coinsurance was: indeed, insurers, having risk beyond their means to pay, insured these risks by sharing them with other insurers. But coinsurance had a lot of disadvantages for companies, mostly related to the fact that a company could gain an insight into another company's business and misuse this information to gain an unfair advantage both on the other company and, eventually, on the market itself.
During the last century though, thanks to the increased number of risks arising from industrialization, a greater need for reinsurance cover was needed and consequently more professional reinsurance companies were established leading to the gradual elimination of disadvantages and injustices.
The first professional reinsurance company, Cologne Re, was founded following a devastating fire in Hamburg in 1842: the loss from this event reached 18 million marks, whereas the local Hamburg Fire Fund only had 500 000 marks in reserve. This event assisted the final breakthrough of the need to share the risks of whole portfolios amongst several risk-carriers.
Thus, by establishing more and more professional reinsurance companies, the disadvantages of coinsurance were eliminated. In addition, specialization allowed the

development of new forms of reinsurance and worldwide multi-line activity allowed for a better distribution of the risks. Furthermore, by providing better reinsurance protection, direct insurers were also able to offer their clients better conditions.

1.1.2 Role of reinsurance

The primary role attributed to reinsurance is that it helps a primary insurer to achieve several practical business goals, such as insuring large exposures and financing its growth. Indeed, by purchasing reinsurance a primary insurer transfers a share of the underlying risk onto the reinsurer: therefore, the former safeguards its solvency and at the same time increases the volume or size of risk it can accept.
The primary insurer may obtain different types of reinsurance, mostly depending on its needs. In general, primary insurers have some principal functions available:

- Increase large-line capacity
  This allows a primary insurer to assume more significant risks than its financial condition and regulations would otherwise permit: reinsurers provide primary insurers with large-line capacity by accepting liability for loss exposures that the primary insurer is unwilling or unable to retain. Thus this function allows a primary insurer to fully participate in the insurance marketplace by allowing an increase in its market share while limiting the financial consequences of potential losses.

- Provide catastrophe protection
  Catastrophes (such as fire, windstorm, earthquakes) could greatly reduce the primary insurer earnings or even threaten its solvency when a large number of the insured loss exposures are concentrated in an area that experiences a catastrophe. That's why this function of reinsurance aims at protecting the primary insurer against the financial consequences of a single catastrophic event that cause multiple losses in a concentrated area.

- Stabilize loss experience
  Demographic, economic, social and natural forces cause a primary insurer's loss experience to fluctuate widely and this creates variability in its financial results. Reinsurance can smooth the resulting peaks and valleys in a primary insurer's loss experience curve and can encourage capital investment since capital investors are more likely to invest in companies with stable results.

- Provide surplus relief
  Some reinsurance agreements facilitate primary insurers premium growth by allowing them to deduct a ceding commission on loss exposures and to cede it to the reinsurer. Thus the ceding commission is an amount paid by the reinsurer to the primary insurer to cover part or all of a primary insurer's policy acquisition expenses.

- Facilitate withdrawal from a market segment
  A primary insurer may want to withdraw from a market segment that is unprofitable, undesirable or incompatible with its strategic plans. There are some options that the primary insurer has in order to withdraw from a market segment: it can either stop writing new insurance policies and continue in-force insurance until all policies expire, cancel all policies and refund the unearned premiums to insureds or withdraw from the market by purchasing portfolio reinsurance. Reinsurance can help with all of these processes, facilitating the primary insurer with the procedures while protecting him from undesirable outcomes.

- Provide underwriting guidance
  Reinsurers work with a wide variety of insurers in the domestic and global markets under many circumstances and consequently they accumulate a great deal of underwriting expertise. Thus reinsurers can assist other insurers, in particular inexperienced primary insurers entering new markets and offering new products. However it is important to state that reinsurers that provide underwriting assistance to primary insurers must respect the confidentiality of their clients' proprietary information.

1.2 Forms of reinsurance

In general, there is not a single reinsurance agreement that performs all the reinsurance functions. Instead, reinsurers have developed various forms of reinsurance (more generally reinsurance contracts) in order to be effective in helping primary insurers meet one or more of their goals. Indeed, a primary insurer often combines several agreements to meet its particular needs and each of these agreements is tailored to the specific needs of both the primary insurer and the reinsurer.
Reinsurance contracts are generally divided into two forms: *treaty reinsurance* and *facultative reinsurance*. The distinction between the two categories lies in the fact that while the former covers a whole portfolio of risks, the latter covers specific selected risks.
This difference essentially determines the design and, hence, the form of the reinsurance contract. However, many hybrid forms of reinsurance contracts exist so that it may be too simplistic to regard there to be a dichotomy between treaty and facultative reinsurance.

1.2.1 Treaty reinsurance

A reinsurance treaty is a contract *for* reinsurance rather than a contract *of* reinsurance: indeed, this contract is not used to transfer a portion of the primary insurer's risk to the reinsurer by itself but it is the parties that agree that the primary insurer cedes and the reinsurer accepts specified risks to the extent that they are underwritten

by the primary insurer.
This is the reason why treaty reinsurance is also referred to as *obligatory reinsurance*: the primary insurer is obliged to cede to the reinsurer a contractually agreed share of the risks defined in the treaty and the reinsurer is obliged to accept that share.
Of course the primary insurer is generally free to decide whether to accept the business, but where it chooses to, it is obliged to cede a certain amount or proportion of the risks to the reinsurer while the latter is bound to accept such amount or proportion of the risk if it is within the scope of the treaty.
Treaty reinsurance is efficient because the primary insurer does not have to apply for reinsurance cover in respect of each policy underwritten by it: instead, it has certainty that it will obtain appropriate reinsurance cover for a risk that it wishes to accept. Moreover, in most cases it does not have to provide the reinsurer with detailed information regarding each and every ceded risk and at the same time the reinsurer does not access each and every ceded risk. In this way the parties reduce their administrative costs of business.
Treaty reinsurance contracts can terminate on an annual basis or they can be multi year deals.

1.2.2 Facultative reinsurance

In the case of facultative reinsurance, a primary insurer decides whether it wishes to reinsure a specific risk. It is up to them to choose the right reinsurer for the deal and the reinsurer is equally free to either accept the risk or to decline it: hence the term *facultative*.
A primary insurer who elects to reinsure a risk must present the reinsurer with a precisely defined offer containing all pertinent information on the risk in question: this might result in quite high administrative costs. The reinsurer, after detailed examination, will decide whether or not to accept it: this liberty can be seen as an advantage for the latter.
Facultative reinsurance is very often used as a complement for treaty reinsurance: for example, a prospective primary insurer may seek facultative reinsurance where a risk exceeds the available treaty capacity or where it is not covered by the treaty.
Some other cases in which a primary insurer will most often turn to this form of reinsurance are the following:

- when it is left with a sum it still needs to reinsure after it has exhausted both its retention (ie. portion of a risk which a primary insurer is willing and able to carry itself) and the reinsurance capacity provided by its treaty reinsurance contract
- when it has sold a policy containing risks that are excluded from its treaty reinsurance cover

In general, there exist a known hybrid between the facultative versus treaty approach: the *facultative obligatory treaty*. This is a treaty under which the primary insurer

has the option to cede or not cede individual risks. However, the reinsurer must accept any risks that are ceded.

1.3 Types and methods of reinsurance

Reinsurance contracts can be further divided into two types, *proportional* and *non-proportional*, and both forms of reinsurance previously introduced may be either one of them.
In both cases a certain part of the risk is transferred from the primary insurer to the reinsurer. The distinction between proportional and non-proportional reinsurance lies in the definition of the part of risk to be ceded as well as the way in which premiums are shared. Furthermore, each of these two types of reinsurance is characterized by specific agreements. Thus let's introduce each type and proceed to a more detailed consideration of the characteristics of each.

1.3.1 Proportional reinsurance

In proportional reinsurance the primary insurer and the reinsurer divide premiums and losses between them at a contractually defined ratio. According to the type of treaty, this ratio may be the same for all risks covered by the contract (*quota share reinsurance*) or it may vary from risk to risk (all other proportional reinsurance types). In all cases, however, the reinsurer's share of the premiums is directly proportional to its obligation to pay any losses. For example, if the reinsurer accepts 90% of a particular risk and the primary insurer retains 10%, the premium is apportioned at a ratio of 90:10.
The terms 'risk' and 'risks' in this context refer to the risk of incurring liability resulting from reinsurance cover under an underlying policy or multiple underlying policies respectively.
The price of proportional reinsurance cover is expressed in the *reinsurance commission*: originally, this commission was intended to compensate the primary insurer for its agents' commissions, internal administration expenditures and loss adjustment costs. However, in today's highly competitive environment, the market often puts the primary insurer in a difficult position regarding the balance between costs and premiums. For this reason, there is a growing trend for reinsurers to return to the primary insurer as reinsurance commission only that part of the original premium not paid out for losses. Thus, the reinsurance commission is increasingly defined by commercial considerations rather than the primary insurer's actual operating costs.
In order to clarify the concept of reinsurance commissions under a proportional reinsurance contract, we now introduce a toy example:

Example 1.3.1. A primary insurer expects losses of 60 million, operating costs of 30 million and a profit of 10 million from a portfolio. The required original premium whould therefore be 100 million (ie. the sum of the three quantities above).
He now decides to cede 25% to a reinsurer under a *quota share reinsurance treaty* (QS): this means that the contractually defined ratio for the division of premiums and losses is the same for all risks covered by the contract (we will deal with quota share more specifically later on). Thus, the reinsurer receives 25% of the original premium (or 25 million) of which he must pay the 25% of losses (15 million). The reinsurer expects a profit of 10% too, that is 2.5 million on his premium volume of 25 million. The remainder, 7.5 million, given by 25 original premium minus 15 million loss minus 2.5 million profit, is returned to the primary insurer as his commission (see Table 1.1). This fully defrays the primary insurer's operating costs (for the primary insurer, the cost to operate 25% of the risks is 25% of 30 million: that is exactly 7.5 million).

Table 1.1: Example 1.3.1 of proportional reinsurance and reinsurance commissions

Primary insurer expectations	Premium: 100 million Losses: 60 million Operating costs: 30 million Profit: 10 million
QS: 25% ceded to reinsurer	Premium: 25 million Losses: 15 million Profit : 2.5 million Commission: 7.5 million
QS: primary insurer's 75%	Premium: 75 million Losses: 45 million Profit : 7.5 million Operating costs for 25% QS: 7.5 million

Assume, however, that due to competition the primary insurer must reduce his original premium by 2% (ie. 98 million). The quota share reinsurer's 25% would be only 24.5 million in this case but his losses would remain the same, at 15 million, and he would still like to realise his expected profit of 2.5 million. Thus now 7 million would remain as the commission and the primary insurer's operating costs would not be fully defrayed (see Table 1.2).

Table 1.2: Example 1.3.1 of proportional reinsurance and reinsurance commission with reduction of premium due to competition

Primary insurer's expectations	Premium: 98 million Losses: 60 million Operating costs: 30 million Profit: 8 million
QS: 25% ceded to reinsurer	Premium: 24.5 million Losses: 15 million Profit : 2.5 million (fixed) Commission: 7 million
QS: primary insurer's 75%	Premium: 73.5 million Losses: 45 million Profit : 5.5 million Operating costs for 25% QS: 7.5 million

After a general introduction on proportional reinsurance, we go further into details by distinguishing the main three different types of proportional reinsurance contracts: *quota share reinsurance*, *surplus reinsurance* and *proportional facultative reinsurance*.

Quota Share Treaty

Quota share reinsurance is the simplest form of proportional reinsurance: the reinsurer assumes an agreed-upon, fixed quota (percentage) of all the insurance policies written by a primary insurer within the particular branch or branches defined in the treaty. This quota determines how liability, premiums and losses are distributed between the primary insurer and the reinsurer. Let's see a toy example:

Example 1.3.2. In the following example the sum insured equals 10 000 000 and the percentage of the quota share treaty is 30%. Libaility, premium and losses are distributed between the primary insurer and the reinsurer as follows:

The parties often agree to limit the reinsurer's liability *per risk*, thus they limit the reinsurer's liability to a maximum monetary amount for losses arising under one single risk. In this scenario, the reinsurer is only bound to pay his percentage of a loss up until the per risk limit is exceeded, ie. until the limit for losses under one single policy is beat.
However, particularly in case of a natural catastrophe, multiple policies in the portfolio may be triggered: this may lead to a large-scale liability on the part of the

Table 1.3: Example 1.3.2 of quota share treaty

Primary insurer (PI) 's retention	70%
Reinsurance quota share	30%
Sum insured (SI) of the insured object	10 million
PI retains 70% of the exposure	7 million
Reinsurer assumes 30% of the exposure	3 million
Premium rate 2‰ of the SI	20 000
PI retains 70%	14 000
Reinsurer receives 30%	6 000
Loss	6 million
PI pays 70%	4.2 million
Reinsurer pays 30%	1.8 million

reinsurer under the quota share treaty. This is why the parties regularly agree on a limit *per event* to ease the situation for the reinsurer.
The quota share agreement is simple as well as cost-effective. Its disadvantage lies in the fact that it does not sufficiently address the primary insurer's various reinsurance requirements since it measures everything by the same yardstick. In particular, quota share reinsurance treaties do not help to balance a portfolio: indeed, they do not limit the exposure posed by peak risks (for example, those with very high sums insured). At the same time, such a treaty may also provide reinsurance cover where none is needed: this can unnecessarily restrict the primary insurance company's profit-making options.
By the above statements it could seem like this type of reinsurance treaty is inconvenient in most cases but it actually have its uses in different scenarios. Quota share treaties are especially suited for young, developing companies or companies which are new to a certain class of business. As their loss experience is limited, they often have difficulties in defining the correct premium: with a quota share treaty, the reinsurer takes the risk of any incorrect estimates.
Quota share reinsurance is also well suited to limiting the risk of random fluctuation and risk of change across an entire portfolio.

Surplus Treaty

Surplus reinsurance is a more sophisticated form of proportional reinsurance. With this kind of treaty, the reinsurer does not participate in all risks as for the quota share treaty: instead, the primary insurer itself retains all risks up to a certain amount of liability (its *retention*). This retention may be defined differently for each type or

class of risk. The reinsurer, for his part, it is obliged to accept the surplus, ie. the amount that exceeds the primary insurer's retention.
Of course, there must also be an upper limit to the reinsurer's obligation to accept risk. This limit is usually defined as a certain multiple of the primary insurer's retention, known as *line*. For each reinsured risk, the ratio that results between the risk retained and the risk ceded is the criterion for distributing liability, premiums and losses between the primary insurer and the reinsurer.
Also in this type of treaty, the parties regularly agree on a limit per event to ease the situation for the reinsurer in case of natural catastrophes or events that trigger multiple policies in the portfolio.
We now consider some toy examples in order to clarify the concept of surplus treaty: in all the following examples the cedent's retention is 300 000 and the reinsurer's liability (ie. the surplus) is limited to 9 lines.

Example 1.3.3. The cedent's original liability (ie. the primary insurer's liability to the policyholder) from his share in a given risk amounts to 3 million. The premium is 1.50‰ of the sum insured and the loss is 1.5 million.
The risk is shared by the cedent and the reinsurer as follows:

	Total	Cendent's retention	Reinsurer's surplus
Sum insured/liability	3 000 000	300 000 = 10%	2 700 000 = 90% (9 lines)
Premium	4 500	450 = 10%	4 050 = 90%
Loss	1 500 000	150 000 = 10%	1 350 000 = 90%

Example 1.3.4. The cedent's original liability amounts to 130 000. The premium is 1.50‰ of the sum insured and the loss is 80 000.
The risk is shared by the cedent and the reinsurer as follows:

	Total	Cendent's retention	Reinsurer's surplus
Sum insured/liability	130 000	130 000 = 100%	0 = 0%
Premium	195	195 = 100%	0 = 0%
Loss	80 000	80 000 = 100%	0 = 0%

This toy example shows that, in contrast to quota share reinsurance, the reinsurer receives no share of a risk if it does not exceed the amount defined as the primary insurer's retention.

Example 1.3.5. The cedent's original liability amounts to 3 500 000. The premium is 1.50‰ of the sum insured and the loss is 2 000 000.
The risk is shared by the cedent and the reinsurer as follows:

	Total	Cendent's retention	Reinsurer's surplus
Sum insured/liability	3 500 000	800 000[1] = 22.86%	2 700 000 = 77.14% (9 lines)
Premium	5250	1200 = 22.86%	4050 = 77.14%
Loss	2 000 000	457 200 = 22.86%	1 542 800 = 77.14%

Notice that 800 000[1] in the table comes from

$$800\,000 = 300\,000 + 500\,000$$

$$22.86\% = 8.57\% + 14.29\%$$

where 500 000 is the portion of the risk in excess of the primary insurer's maximum retention (300 000, 1 line) plus his reinsurance surplus (2 700 000, 9 lines). The same reasoning holds for premium and losses.
Thus this toy example shows that when the sum insured exceeds the surplus, the primary insurer must either carry the risk himself (as he does in this example) or else arrange suitable facultative reinsurance cover (this is the most frequent scenario).

In contrast to the quota share treaty, the surplus treaty is an excellent mean of balancing the primary insurer's portfolio and thus of limiting the heaviest exposures. As the retention can be set at various levels according to the type of risk (or class of business) and the expected loss, this type of treaty allows the primary insurer to adjust the amount of risk it accepts to fit its company's financial situation at any time. The disadvantage though is that it is complicated, and therefore expensive, to manage since it creates additional work on the accounting side.

Proportional Facultative

Under a proportional facultative reinsurance contract, the reinsurer reinsures a single risk. Whenever the primary insurer is liable under the contract, it is entitled to

be reimbursed for the relevant portion of the liability by the reinsurer under the reinsurance contract. In return, the reinsurer has a right to be paid the relevant portion of the premium as a reinsurance premium. It is presumed that both the underlying and the reinsurance policies are designed to provide for identical or closely matching cover.

1.3.2 Non-proportional reinsurance

With non-proportional reinsurance there is no set, pre-determined ratio for dividing premiums and losses between the primary insurer and the reinsurer: the share of losses that each pays will vary depending on the actual amount of loss incurred. The treaty defines an amount up to which the primary insurer will pay all losses, the *deductible*, while the reinsurer obliges himself to pay all losses above the deductible amount, up to a contractually defined cover limit. This means that, in contrast to surplus reinsurance, the reinsurer's liability is triggered only if the reinsured's liability exceedes the deductible agreed in the non-proportional reinsurance contract. As the price for this cover, the reinsurer charges a suitable portion of the original premium: thus, in contrast with proportional reinsurance, treaty wordings do not explicitly define the way premiums are to be shared by the primary insurance company and the reinsurer. Rather, from the very beginning, the reinsurer must estimate what future loss burden it can expect to pay under such a treaty. It has two methods available to do this:

- *Experience rating*
  This method is based on past loss experience: suitably adjusted, past loss statistics can give a good picture of the loss burden to be expected in the future.

- *Exposure rating*
  If no adequate loss statistics are available, the reinsurer will use the company's own risk profile and attempt to fit an exposure curve on it in order to be able to quantify the differences between the portfolio it is rating and the one it is using for comparison.

We will deal with exposure curves in a more detailed way in Chapter 2 section 3.
As for proportional reinsurance, also for non-proportional reinsurance we can identify different types of contracts or covers.

Excess of loss treaty (XoL)

Excess of loss (or XoL) reinsurance is structured quite differently from the proportional types of treaty discussed above. With proportional treaties, cessions are linked to the *sums insured*: with XoL reinsurance, in contrast, it is the *loss* that is important. Here, no matter what the sum insured, the primary insurer carries for its own account all losses incurred in the line of business named in the treaty, up to

a certain limit that we already mentioned: the *deductible*. The reinsurer pays the entire loss in excess of this amount, up to the agreed cover limit.
XoL insurance can be divided into covers per risk and covers per catastrophic event.

- Per risk XoL treaty
  For this kind of reinsurance, the primary insurer's deductible is applied on a per risk basis. This is to say that reinsurance cover is taken out for single losses which exceed the primary insurer's deductible on any one risk. Notice that a risk may refer to a single primary insurance risk or to an asset, such as a vehicle or a building. Thus for per risk XoL treaties multiple policies are aggregated only when they cover the same risk.

- Per event XoL treaty (CatXoL)
  Per event XoL reinsurance protects the primary insurer when multiple single losses on multiple different risks arise out of one single event, a catastrophe. Thus the primary insurer's deductible and the reinsurer's cover limit are both evaluated against the aggregate of any individual losses that result from any single event.

Such treaties meet the needs of those primary insurers who want reinsurance protection (at least against large losses) while retaining as much of their gross premium as possible. However, these insurers are also taking a risk that is greater than with proportional insurance, for the reinsurer provides no relief from losses below the deductible amount. Thus, in general non-proportional insurance greatly increase the odd that the primary insurer will actually have to pay in full, and for its own account, any losses near or at the agreed deductible amount.
We now present some toy examples in order to clarify the concepts of XoL reinsurance and per risk/per catastrophe treaties. For all examples, we consider a primary insurer's retention at 8 million: to further protect his retention from major losses, the primary insurer also buys a per risk XoL cover of 6 million xs 2 million. In academical notation this would be written as 6xs2 (ie. 6 million in excess of 2 million: the primary insurer pays up to 2 million while the reinsurer covers from 2 million to 8 million). As additional protection from catastrophic events, he decides to buy as well a CatXoL with the limits 9 million xs 4 million (9xs4).

Example 1.3.6. A fire leaves the primary insurer with a loss of 1 million for his own account.

Net losses

Primary insurer:	1 000 000
Per risk XoL reinsurer:	0 (2 million deductible not exceeded)
CatXoL reinsurer:	0 (4 million deductible not exceeded)

This loss amount does not trigger any of the two reinsurance contracts, since the per risk XoL is 6xs2 and the CatXoL is 9xs4. Thus the primary insurer will pay the whole amount, while the two reinsurers do not have to cover any cost.

Example 1.3.7. A major fire leaves the primary insurer with a loss of 7 million for his own account.
Since the per risk XoL is 6xs2, the primary insurer pays 2 million while the remaining 5 million are fully covered by the per risk XoL reinsurer. Notice that in this case, since the primary insurer ends up paying just 2 million, the CatXoL reinsurance contract (ie. 9xs4) is not triggered, though the CatXoL reinsurer does not have to cover any cost.

Net losses

Primary insurer:	2 000 000 (ie. per risk XoL deductible)
Per risk XoL reinsurer:	5 000 000
CatXoL reinsurer:	0

Example 1.3.8. A single earthquake leaves the primary insurer with losses for his own account as follows:

riskA	riskB	riskC	riskD	riskE
1 million	1 million	1 million	2 million	4 million

for a total loss of 9 million.
In this scenario, since the per risk XoL is 6xs2, the primary insurer has to pay in full the risks A, B, C and D and half of the riskE while the other half will be paid by the per risk XoL reinsurer (ie. 2 million). But by paying the first 4 risks in full and half of the last one, the primary insurer exceeds the CatXoL deductible (ie. 7 million), thus he just pays 4 million and the remaining amount is covered by the CatXoL reinsurer. Note tht this happens just in the case where the per risk XoLL inures to the benefit of the CatXoL.

Net losses

Primary insurer:	4 000 000 (ie. CatXoL deductible)
Per risk XoL reinsurer:	2 000 000
CatXoL reinsurer:	3 000 000

Stop loss treaty

The stop loss treaty is designed for primary insurers who are seeking comprehensive protection against fluctuations in their annual loss experience in a given class of

business. In this somewhat rare form of reinsurance, the reinsurer is obliged to cover any part of the total annual loss burden that exceeds the agreed deductible: usually, this deductible is defined as a percentage of annual premium income, but it may also be a fixed sum. It is irrelevant whether the deductible is exceeded by one single large loss or an accumulation of small and medium-sized losses.
As it is not the purpose of the stop loss treaty to relieve the primary insurer of all entrepreneurial risk, the reinsurer understandably requires the primary insurer to incur a technical loss (ie. a loss in which losses + costs > premiums) before his duty to pay is triggered.
The stop loss treaty is actually the most comprehensive form of reinsurance protection. However, reinsurers have reservations towards this type of treaty, which is the reason why it is not more widely used. There are several reasons for their restraint:

- A large amount of risk is transferred to the reinsurer while its means of influencing the exposure remain limited
- The reinsurer loses premium volume, and hence influence
- The composition of most portfolios is becoming less transparent as the insurance business becomes increasingly internationalised

Indeed, stop loss reinsurance is used to protect the primary insurer's solvency, but the reinsurer cannot increase the volume or size of primary insurance risks it is able to accept by entering into this kind of treaty.
Stop loss reinsurance comes in two different types, *excess of loss ratio* and *aggregate excess of loss*: the difference lies in the way the primary insurer's deductible and the reinsurer's cover limit are defined.
Where an *excess of loss ratio* applies, the parties to the reinsurance treaty agree on a certain percentage as the primary insurer's deductible and the reinsurer's cover limit. Then the ratio between the annual losses and the net retained premium is expressed in a percentage and tested against the percentages previously agreed upon.
Under an *aggregate excess of loss* treaty, the primary insurer is covered for the aggregate of any loss that occurs within a defined period of time. Where this kind of stop loss applies, the parties agree on the primary insurer's deductible and the reinsurer's cover limit and express them in monetary amount. The aggregate of the annual losses under the primary insurer's portfolio is then tested against these figures and with respect to those the treaty is or isn't triggered.
Stop loss treaties are most frequently used for storm and hail insurance.

Facultative excess of loss

In facultative excess of loss reinsurance, the primary insurer has the option to cede or not cede individual risks while the reinsurance company reviews individual risks and determines whether to accept or reject them. As the reinsurance is non-proportional, the parties agree that the primary insurer is liable for any loss that does not exceed

its deductible. Once the amount of a loss exceeds it, the reinsurer's liability up to a specified cover limit will be triggered.
Also in this case there's the possibility for a hybrid, known as *facultative obligatory XoL*: the primary insurer has the option to cede or not cede individual risks while the reinsurer must accept any risks that are ceded.
A facultative reinsurance contract may provide for a per event excess of loss mechanism.

Chapter 2

Ground up loss modelling

After a brief general introduction to reinsurance, we would like now to present all the steps, observations and decisions an actuary has to make in order to build a complete pricing model. In particular, the presentation will be done in line with Sompo International's work during contract renewal process and with respect to what, as an actuarial intern in the company, I am doing on a daily basis.
In order to do so, we will start by introducing some generic guidelines known throughout the industry. Then, we will go into further details with some of the methods and choices that lie behind the pricing tool used by the company.
Later on, in section 2.2 we will start to analyze in a more detailed way the models that lie behind each pricing. In particular, for non-proportional modelling the frequency/severity models will be introduced.
However, pricings do not only rely on historical experience. For this reason, in the last section the experience and exposure ratings approaches to pricing XoL reinsurance will be presented.

2.1 Basics of pricing

Like primary insurance, reinsurance is a mechanism for spreading risk. But there exists a major difference between the two: a reinsurance program is generally tailored more closely to the buyer. Indeed, each contract must be individually priced to meet the particular needs and risk level of the reinsured.
While the pricing process is the same for every client, the pricing methodology is specific to each cedent. For this reason the basic pricing tools are usually only a starting point in determining an adequate premium. It is the actuary that has to know when the assumptions in these tools are not met and how to supplement the results with additional adjustments and judgment.
In spite of that, some basic steps and modifications always need to be made in order to have reliable information. We will then introduce the methods used by most of the companies, thus also Sompo International, to correctly forecast future losses.

2.1.1 Data "*as-if*"

As we stated in the previous chapter, there are two different types of reinsurance, *proportional* and *non-proportional*, and for each of these there are standard steps that should be included in the pricing analysis and that follow standard ratemaking procedures.
One of the most important things to remember when it comes to pricing is the need to consider the historical data "*as-if*": this means that we should adjust data that are prior to the reinsurance treaty "as if" they would correspond to the treaty year. This must be done in order to forecast future information while considering the effects of inflation or market changes.
When receiving a submission from cedents, historical premiums and general losses are usually gathered in cumulative triangles with respect to the origin period and the development years. The large losses instead (usually above a specified threshold) are given separately and we often have other information, such as the date of loss and some notes concerning the loss.
What we need to do is to adjust experience to ultimate level in order to project to future periods of time. The historical losses need to be developed to an ultimate basis and if the treaty experience is insufficient to estimate loss development factors, data from other sources may need to be used. Depending on the source of these factors, adjustments for the reporting lag to the reinsurer may need to be made.
We should also adjust historical premiums to the future level. The starting point to do that is historical changes in rates and average pricing factors (ie. changes in schedule rating credits). Also the impact of rate changes anticipated during the treaty period should be taken into account. Notice that if the premium base is inflation sensitive then an exposure inflation factor should be included in the adjustment of historical premium.
Finally, the losses need to be adjusted: various sources are available for this adjustment, including the amounts used in the ceding company's own rate filings. Within the company, we dealt with this issue before renewal time. Indeed, we computed the inflation parameter over all cedents large losses in order to have it fixed over all the pricings made along the year.

2.1.2 Loss development methods

Forecasting future claims is an important part of the business of a reinsurance company. Indeed, the published profits of these companies depend not only on the actual claims paid, but on the forecasts of the claims which will have to be paid. Therefore, it is essential that a reliable estimate is available in order to ensure the financial stability of the company and its profit and loss account.
There are a number of methods which have proved useful in practice and all of these depend on finding some pattern in the way that claims have been settled in the past so that it can be applied to the future.

The actuary will select the method only after the most thorough analysis of the data at hand, which will involve:

- Checking the accuracy of all the data used
- Taking care of special features, such as large losses, cat events or other exceptional claims, since they will need careful treatment
- Selecting what type of data to use since some methods have particular data requirements
- Evaluating the result in the light of the knowledge of the business and, eventually, comparing with available external brenchmarks

There are many methods that are in use today, but we will introduce just the two methods used within the actuarial team in Zurich during the pricing process: the *Chain Ladder method* and the *Bornhuetter-Ferguson method (BF)*.
These two classic methods involve grouping the claims data in a triangle. In our case the data is shown as cumulative across the columns and classified in a row of the triangle according to when it originated and into a column according to when it emerged, but in some other companies the opposite convention is used. This is why, when dealing with data, an actuary must pay attention to be consistent with the convention in order not to mismatch information.
Another aspect that an actuary must take into account while analyzing the data is to check in which origin period the data are given. We have two options:

- Accident year (AY): claims are grouped according to the year in which they occurred
- Policy year (PY) or underwriting year (UY): claims are grouped according to the year the insurance policy incepts

Indeed, the treaty can be either on a *losses occurring basis*, for which earned premium and accident year losses should be used, or on a *risk attaching basis*, where written premium and the losses covered by policies incepting during this year are used.
Table 2.1 shows a left aligned triangle based on accident year: the data are fictitious and we assume it relates to claims incurred (ie. claims paid plus any outstanding case estimates).
It is important to know the attachment basis because calculations have different implications with the two different bases. Indeed, projecting on an AY basis gives a forecast of the cost of all claims arising from events occurring in the period covered by the data, whether they have been notified or not. If, on the other hand, the data is on a PY basis, then the results will be an estimate corresponding to all policies that have been written during the period of insurance.
Now we introduce the two methods in order to understand better how the claims reserves projection works in practice.

Table 2.1: Example of a left aligned triangle

	Development year					
AY	**1**	**2**	**3**	**4**	**5**	**6**
2015	1 900 000	2 700 000	3 600 000	4 000 000	4 200 000	4 200 000
2016	1 000 000	1 800 000	2 900 000	3 700 000	4 600 000	
2017	2 500 000	4 000 000	5 200 000	5 400 000		
2018	1 500 000	2 500 000	2 700 000			
2019	2 000 000	2 900 000				
2020	2 300 000					

Chain Ladder

The Chain Ladder method is probably one of the oldest methods of paid/incurred claims projection and still one of the most popular ones.
This method works by calculating an average factor for estimating the cumulative amount in each year starting from the cumulative amount in the previous year. This average can be formed by averaging the *loss development factors* (commonly referred to as LDFs), obtained by dividing the cumulative amount in one year over the cumulative amount in the previous year.
Let C_{ik} denote the cumulative loss amount of accident year $i = 1, \ldots, n$ at the end of development year (age) $k = 1, \ldots, n$. The amounts C_{ik} have been observed for $k \leq n + 1 - i$ whereas the other amounts have to be predicted.
The Chain Ladder algorithm consists of the stepwise prediction rule

$$\hat{C}_{i,k+1} = \hat{C}_{ik}\hat{f}_k$$

starting with $\hat{C}_{i,n+1-i} = C_{i,n+1-i}$. Here the age-to-age factor $\hat{f}_k$ is defined by

$$\hat{f}_k = \frac{\sum_{i=1}^{n-k} w_{ik} C_{ik}^{\alpha} F_{ik}}{\sum_{i=1}^{n-k} w_{ik} C_{ik}^{\alpha}}, \quad \alpha \in \{0, 1, 2\}$$

where

$$F_{ik} = \frac{C_{i,k+1}}{C_{ik}}, \quad 1 \leq i \leq n, \quad 1 \leq k \leq n - 1$$

are the individual development factors and where $w_{ik} \in [0, 1]$ are arbitrary weights which can be used by the actuary to downweight any outlying F_{ik}. Normally, $w_{ik} = 1$ for all i, k. If this is the case, then $\alpha = 1$ gives the historical chain ladder age-to-age factors, $\alpha = 0$ gives the straight average of the observed individual development factors and $\alpha = 2$ is the result of an ordinary regression of $C_{i,k+1}$ against C_{ik} with

intercept 0.
The above stepwise rule finally leads to the prediction of the trended/developed loss:

$$\hat{C}_{in} = C_{i,n+1-i}\hat{f}_{n+1-i} \cdot ... \cdot \hat{f}_{n-1} \tag{2.1}$$

where the product of the $\hat{f}_j\,, \quad j = n+1-i, \ldots, n-1$ is referred to as the age-to-ultimate factor.
Notice that we reached the prediction of C_{in} but, because of limited data, the loss development of accident year i does not need to be finished at age n. Therefore, the actuary often uses a tail factor $\hat{f}_{ult} > 1$ in order to estimate the ultimate loss amount $C_{i,ult}$ by

$$\hat{C}_{i,ult} = \hat{C}_{in}\hat{f}_{ult}.$$

A possible way to arrive at an estimate for the tail factor is a linear extrapolation of $\ln(\hat{f}_k - 1)$ by a straight line $a \cdot k + b$, $a < 0$, together with

$$\hat{f}_{ult} = \prod_{k=n}^{\infty} \hat{f}_k.$$

However, the tail factor used must be plausible and, therefore, the final tail factor is the result of the personal assessment of the future development by the actuary.

The Chain Ladder method is intuitively appealing and simple to approach, but it may present some problems:

- Since the estimate for each origin period is formed by multiplying the most recent value in each origin period by a LDF, if the most recent value is very large then the factor may overestimate the eventual losses for this period
- The LDFs must be stable across the origin periods for the method to produce sensible results and such stability is rare

For this reason, when applying the method an actuary must pay attention to the development pattern and to the LDFs: indeed, if the development pattern has changed a lot over the years, then it may be better to use only data from the most recent calendar periods in order to reflect better the current conditions.
Also, if the results affected by the LDFs appear to be highly unusual then it may be of interest to rearrange the factors. It is important to underline though that this should be done only after fully investigating the reasons: indeed, special adjustments to data using available information can help to deal with changes or adaptations.

Bornhuetter-Ferguson (BF)

The Bornhuetter-Ferguson method (or BF for short) requires some additional information with respect to the Chain Ladder method, namely the corresponding

premiums for each origin period. Remember that if we are dealing with accident years the premiums chosen should be earned premiums, whereas for policy years the premiums chosen should be written premiums.
The thrust of this method is, for each origin period, to balance the proportion of the eventual claims outgo we currently know about against a similar proportion of the premium. In order to know the proportion, we rely upon the approximation deriving from the Chain Ladder method.
The BF estimate of the reserve is achieved by the following computation:

$$\text{res} = \text{On level premium x (1 - Lag factor) x Initial loss ratio} \tag{2.2}$$

where the lag factor is the reciprocal of the age-to-ultimate factor estimated with the Chain Ladder method. The first step, therefore, is to find these lag factors.
The second step is to determine the initial loss ratio to use. Indeed, if the initial loss ratio can be estimated with sufficient accuracy, then it is likely that this method will be more accurate than the Chain Ladder method.
There are different choices that can be made. However, we will just consider the method used here in the Zurich team, where, by referring to the same notation used in the Chain Ladder method, the choice of the initial loss ratio is given by

$$r = \sum_{i=1}^{n} \frac{C_i w_i}{P_i}, \quad w_i = \frac{P_i}{P}$$

with $C_i = \sum_k C_{ik}$ being the cumulative trended/developed loss for accident year i, P_i the respective on level premium and $P = \sum_{i=1}^{n} P_i$ the total on level premium. Indeed, the loss ratio estimate used for the BF method is nothing but the average of the results obtained for each accident year from the Chain Ladder method.

2.2 Ground up loss models

The purpose of this section is to develop models of aggregate losses, ie. the total amount paid on all claims occurring in a fixed time period on a define set of contracts. Indeed, we may need a model for the aggregate amount of losses, while in other situations a model for individual losses that exceeds a specific threshold is needed.
There are two main types of models: the ***individual risk model*** and the ***collective risk model***.

Individual risk model

The individual risk model represents the aggregate loss as the sum of the amounts paid on each component of the portfolio of risks. That is,

$$S = X_1 + X_2 + ... + X_n \tag{2.3}$$

where X_i is the amount paid on the i-th contract and n is a fixed number. Furthermore, unless stated otherwise, it is assumed that $X_1, \ldots, X_n$ are independent.

Collective risk model

The collective risk model represents the aggregate losses as a sum S of a random number N of individual loss amounts $(X_1, ..., X_N)$. Hence,

$$S = X_1 + X_2 + ... + X_N, \quad N = 0, 1, 2, ... \tag{2.4}$$

where the $X_i, i = 1, ..., N$ are independent and identically distributed (iid) random variables (rvs), unless otherwise specified. More formally, the independence assumptions are:

- Conditional on $N = n$, the rvs $X_1, ..., X_n$ are iid
- Conditional on $N = n$, the common distribution of the rvs $X_1, ..., X_n$ does not depend on n
- The distribution of N does not depend in any way on the values of $X_1, X_2, ...$

Individual risk models, also referred to as *aggregate loss models*, are used for quota share and surplus treaties while collective risk models are mainly related to XoL pricings.
However, before proceeding with the introduction of the models, we briefly introduce the method used for parameters estimation by the actuaries in the Zurich team and, in general, in this field of interest: the *maximum likelihood method*.

Maximum likelihood method

The principle of maximum likelihood is relatively straightforward. It is a method that determines values for the parameters of a model. The parameter values are found such that they maximize the likelihood that the process described by the model produced the data that were actually observed.
We start with a sample of observable data $X = (X_1, \ldots, X_n)$ that has a specified model, ie. a collection of distribution functions $\{F_\theta : \theta \in \Theta\}$ indexed by the parameter space Θ. Data is observed, but we don't know which of the models F_θ it came from. We are assuming though that the model is correct, ie. that there is a θ value such that $X \sim F_\theta$. The goal then is to identify the model that explains the data the best. This amounts to identifying the true but unknown θ value. Hence, our goal is to estimate the unknown θ.
So let's suppose $X \sim F_\theta$, where the X_j are iid and the parameter θ is unknown. We further suppose that for each θ $F_\theta(x)$ admits a probability function $f(x|\theta)$. Thus, $\mathbf{f}(\mathbf{x}|\theta)$, with $\mathbf{x} = (x_1, \ldots, x_n)$, is the probability density function of the joint distribution and it measures the probability of observing the data given a model

parameter θ. Notice that since we assumed independence, the joint distribution is nothing but the product of all the individual probability distributions:

$$f(x_1, \ldots, x_n|\theta) = \prod_{i=1}^{n} f(x_i|\theta)$$

Then, the principle of maximum likelihood yields a choice of the estimator $\hat{\theta}$ as the value for the parameter that makes the observed data most probable.
The ***likelihood function*** is the density function regarded as a function of θ:

$$\mathbf{L}(\theta|\mathbf{x}) = \mathbf{f}(\mathbf{x}|\theta), \quad \theta \in \Theta$$

It is important to underline the differences between the likelihood function and the probability density function:

- The probability density function expresses the probability of observing our data given the underlying distribution parameters and it assumes that the parameters are known
- The likelihood function expresses the likelihood of parameters values occurring given the observed data and it assumes that the parameters are unknown

If $L(\theta_1) > L(\theta_2)$ then θ_1 is more likely to have been responsible for producing the observed data. In other words, F_{θ_1} is a better model than F_{θ_2} in terms of how well it fits the observed data.
Finally, the ***maximum likelihood estimator (MLE)*** of the parameter θ is the value that maximizes the likelihood function:

$$\hat{\theta}(\mathbf{x}) = \arg\max_{\theta} \mathbf{L}(\theta|\mathbf{x}).$$

Typically, we will maximize the **score function**, ie. the logarithm of the likelihood function $\ln \mathbf{L}(\theta|\mathbf{x})$, because it it simpler.
This class of estimators has an important property:
If $\hat{\theta}$ is a maximum likelihood estimate for θ, then $\hat{\eta} = g(\hat{\theta})$ is a maximum likelihood estimate for $\eta = g(\theta)$.

Proof. If g is invertible the likelihood function written as a function of η is simply given by

$$L(g^{-1}(\eta)) = L(g^{-1}(g(\theta)) = L(\theta).$$

But we know that $\hat{\theta}$ is the MLE of θ, thus the largest this function can be is $L(\hat{\theta})$. Therefore, in order to maximize, it is enough to choose $\hat{\eta}$ such that $g^{-1}(\hat{\eta}) = \hat{\theta}$, ie. take $\hat{\eta} = g(\hat{\theta})$.
If g is not invertible there is not a unique θ corresponding to each η anymore, thus in order to define $L(\eta)$ we need to make a choice for θ. Define

$$L(\eta) = \max_{\theta: g(\theta)=\eta} L(\theta)$$

then again the largest value for $L(\eta)$ occurs at $g(\hat{\theta})$ since $\hat{\theta}$ maximizes L, thus $\hat{\eta} = g(\hat{\theta})$ will be chosen.

□

This whole discussion can be extended to cases in which more than one parameter has to be estimated (as for the Normal distribution for example).

2.2.1 Individual risk models

As we already know, a proportional treaty is an agreement between a reinsurer and a ceding company in which the reinsurer assumes a given percentage of losses and premium.
The following steps should be included in the pricing analysis for proportional treaties:

1. Compile the historical experience on the treaty
   Assemble the historical premium and incurred losses on the treaty for five or more years. If this is not available, the whole environment should be adjusted "as-if" the treaty terms had been in place.

2. Exclude catastrophe and shock losses
   Cat losses are due to a single event, such as a hurricane or earthquake, which may affect a large number of risks. Shock losses are any other losses, usually affecting a single policy, which may distort the overall result. However, for different types of contracts (ie. property, casualty, ...) we can have different kinds of situations.

3. Adjust experience to ultimate level and project to future period
   The historical losses need to be developed on an ultimate basis, the historical premium has to be adjusted to the future level and the losses need to be trended to the future period.

4. Select the expected non-cat loss ratio for the treaty
   If the data used in point 3. is reliable, the expected loss ratio is simply equal to the average of the historical loss ratios adjusted to the future level. It may be worthwhile comparing this amount to the ceding company's gross experience if available.

5. Load the expected non-cat loss ratio for cat
   Even if typically there is insufficient credibility in the historical loss experience to price a loading for cat potential, this amount is critical for some treaties evaluations. Furthermore, it is important to take it into consideration.
   However, for some line of businesses cat models are available. These models

can be used by the actuaries in order to have an estimate of cats based on the client's exposure.

6. Estimate the combined ratio given ceding commission and other expenses
   After the total expected loss ratio is estimated, other features must be evaluated such as ceding commissions, reinsurer's expenses or brokerage. This will be discussed in Chapter 3.

In this section we will focus instead on the third point.
There are three basic approaches to derive the loss distribution: empirical, analytical and moment based. The empirical method can be used only when large data sets are available and in such cases a quite accurate estimate of the cumulative distribution function (cdf) is obtained. The analytical approach reduces to finding a suitable analytical expression which fits the observed data well and which is easy to handle. Finally, the moment based approach consists of estimating only the lowest characteristics of the distribution, therefore since these information do not fully define the shape of a distribution the fit to the observed data may be poor.
For proportional treaties the log-normal distribution parameterized by μ and σ is used. Indeed, the single distribution approach, in contrast with the collective risk models, assumes that the aggregate of all losses to the treaty follows a known cumulative distribution function form.

Lognormal distribution

A positive random variable Z is lognormally distributed if the logarithm of the random variable is normally distributed. Hence Z follows a lognormal(μ,σ^2) distribution if its density function is given by

$$f_Z(z;\mu,\sigma^2) = \frac{(2\pi\sigma^2)^{-\frac{1}{2}}}{z}\exp\left\{-\frac{1}{2\sigma^2}(\log z-\mu)^2\right\}$$

for $z>0$, $-\infty<\mu<+\infty$ and $\sigma>0$.
The moments of the lognormal distribution can be calculated from the moment generating function of the normal distribution. Indeed, considering Y to be a random variable normally distributed, the k-th moment m_k is defined as

$$m_k = E[Z^k] = E[e^{Yk}] = M_Y(k) = \exp\left(\mu k + \frac{\sigma^2 k^2}{2}\right)$$

where $M_X(z)$ is the moment generating function of the normal distribution.
Thus, the mean of the lognormal distribution is given by

$$E[Z] = \exp\left(\mu + \frac{1}{2}\sigma^2\right)$$

and the variance is given by

$$\mathrm{Var}[Z] = E[Z^2] - E^2[Z] = \exp(2\mu + 2\sigma^2) - \exp(2\mu + \sigma^2).$$

We now estimate the parameters with the maximum likelihood method.
If $z = (z_1, \ldots, z_n)$ are randomly selected from independent observations which follow the lognormal distribution, from (2.2.1) the function of the likelihood can be written as

$$L(\mu, \sigma^2) = \prod_{i=1}^{n} f_Z(z; \mu, \sigma^2) = (2\pi\sigma^2)^{-\frac{n}{2}} \exp\Big\{ -\sum_{i=1}^{n} \frac{1}{2\sigma^2} (\log z_i - \mu)^2 \Big\} \prod_{i=1}^{n} \frac{1}{z_i}$$

The function of loglikelihood of μ and σ^2 is the following:

$$\begin{aligned} l &= -\frac{n}{2}\log\sigma^2 - \frac{n}{2}\log 2\pi - \sum_{i=1}^{n} \frac{1}{2\sigma^2}(\log z_i - \mu)^2 - \sum_{i=1}^{n} \log z_i \\ &= -\frac{n}{2}\log\sigma^2 - \frac{n}{2}\log 2\pi - \sum_{i=1}^{n} \frac{(\log z_i)^2}{2\sigma^2} - \sum_{i=1}^{n} \frac{2\log z_i \mu}{2\sigma^2} - \frac{n\mu^2}{2\sigma^2} - \sum_{i=1}^{n} \log z_i \end{aligned}$$

We now use the first form of the loglikelihood function to compute the partial derivative with respect to σ^2 and the second one to compute the one with respect to μ:

$$\begin{aligned} \frac{\partial l}{\partial \mu} &= \sum_{i=1}^{n} \frac{\log z_i}{\sigma^2} - \frac{2n\mu}{2\sigma^2} \\ \frac{\partial l}{\partial \sigma^2} &= -\frac{n}{2\sigma^2} + \sum_{i=1}^{n} \frac{(\log z_i - \mu)^2}{2\sigma^4} \end{aligned}$$

By setting these quantities equal to zero, we obtain the the following maximum likelihood estimators:

$$\hat{\mu} = \frac{1}{n} \sum_{i=1}^{n} \log z_i$$

$$\hat{\sigma}^2 = \frac{1}{n} \sum_{i=1}^{n} (\log z_i - \hat{\mu})^2$$

Once we have the estimation of our parameters, we can retrieve the mean and the variance of Z.

2.2.2 Collective risk models

Collective risk models differ from individual risk models because of the number of losses. In fact, this time the number of losses is a random variable that needs to

be modeled too. Consequently, the distribution of S in (2.4) is obtained from the distribution of N and the distribution of the $X_j, j = 1, ..., N$: this is the reason why we refer to this model as collective model. Using this approach, the frequency and severity of claims are modelled separately and the information about these distributions are used to obtain information about S.
Modelling the distribution of N and the distribution of each X_j separately has some advantages:

- The expected number of claims changes as the business changes: growth in volume needs to be accounted for in forecasting the number of claims in future years
- The effects of general economic inflation is reflected in the losses incurred by insured parties and the claims paid by reinsurance companies
- The impact on changing individual deductibles and limits is more easily studied
- The shape of the distribution of S depends on the shapes of both distributions of N and X and this can be very useful when modifying policy details

In summary, a more accurate and flexible model can be constructed by examining frequency and severity separately.
In constructing the model (2.3) for S, N represents the actual number of losses to be insured while the X_js are the individual loss random variables. Indeed, S is the aggregate loss random variable.
In many cases of fitting frequency or severity distributions to data, several distributions may be good candidates for models. However, some distributions may be preferable for a variety of practical reasons. In general, it is useful for the severity distribution to be from a scale family (ie. if a rv X is in the scale family, then also $Y = cX$ is a member of that family) since the choice of currency should not affect the result. Also, scale families are easy to adjust for inflationary effects over time. In fact, when forecasting the costs of a future year, the anticipated rate of inflation can be factored in easily by adjusting the parameters.
A similar consideration applies to frequency distributions. Indeed, as a block of an insurance company's business grows, the number of claims can be expected to grow, all other things being equal. In the meantime, though, ideally the model selected should not depend on the length of the time period used in the study of claims frequency. This is why the expected frequency should be proportional to the length of the time period, after any adjustment for growth in business.
The derivation of distributions is not an easy task. Reinsurers normally keep data files containing detailed information about policies and claims, which are used for accounting and rate-making purposes. However, claim size distributions and other data needed for risk-theoretical analyzes can be obtained usually only after a tedious data preprocessing. Moreover, the claim statistics are often limited since files containing detailed information about some policies and claims may be missing.

There may also be situations where prior data or experience are not available at all, for example when a new type of insurance is introduced or when very large special risks are insured. Then the distribution has to be based on knowledge of similar risks or on extrapolation of some of them.
Despite the differences that lie in each scenario, there are some basic approaches that the reinsurers follow. We will not introduce all the available procedures, but just the ones that actuaries in my team use on a daily basis.

Frequency distributions

The purpose of studying counting distributions in a reinsurance context is simple. Counting distributions describe the number of losses, thus with an understanding of both the number of losses and the size of losses, one can have a deeper understanding of a variety of issues surrounding reinsurance than if one has only information about total losses.
We will focus on parametric models of loss numbers since they summarize the information about a distribution in terms of the form of the distribution and its parameter values. In particular, we will consider the *Poisson distribution* and the *Negative binomial distribution.*

Poisson distribution

As known, the Poisson distribution has probability function given by

$$p_k = \frac{e^{-\lambda}\lambda^k}{k!},\ k = 0, 1, 2, ...$$

and the mean is equal to the variance and is λ.
The Poisson distribution has two useful properties. The first is given in the following theorem:

Theorem 2.2.1. *Let $N_1, ..., N_n$ be independent Poisson variables with parameters $\lambda_1, ..., \lambda_n$. Then $N = N_1 + ... + N_n$ has a Poisson distribution with parameter $\lambda_1 + ... + \lambda_n$.*

Proof. The probability generating function of the Poisson variable N with parameter λ is given by

$$P_N(z) = E[z^N] = \sum_{k=0}^{\infty} p_k z^k = \sum_{k=0}^{\infty} \frac{(z\lambda)^k e^{-\lambda-\lambda z+\lambda z}}{k!} = e^{\lambda z-\lambda} \sum_{k=0}^{\infty} \frac{(z\lambda)^k e^{-\lambda z}}{k!} = e^{\lambda(z-1)}$$

The pgf of the sum of independent random variables is the product of the individual pgfs. Thus for the sum of Poisson random variables $N = N_1 + ... + N_n$ we have

$$P_N(z) = \prod_{j=1}^{n} P_{N_j}(z) = \prod_{j=1}^{n} \exp[\lambda_j(z-1)] = \exp\left[\sum_{j=1}^{n} \lambda_j(z-1)\right] = e^{\lambda(z-1)}$$

where $\lambda = \lambda_1 + ... + \lambda_n$. Since the pgf is unique, N must have a Poisson distribution with parameter λ. □

The second property of the Poisson distribution is particularly useful in reinsurance modelling. Suppose that the number of losses in a fixed time period follows a Poisson distribution. Further suppose that the losses can be classified into m different types (for example, losses could be classified by size as those who are below a fixed limit and those above that limit). It turns out that if one is interested in studying the number of losses above the limit, that distribution is also Poisson but with a new Poisson parameter.
It is also interesting to note that in the scenario presented above the number of losses of different types will not only be Poisson distributed, but also be independent of each other; that is, the distributions of the number of losses above the limit and the number below the limit will be independent.
Before proceeding with the theorem that formalize these ideas, we prove the following statements:

Theorem 2.2.2. *If X and Y are independent Poisson random variables with respective parameters λ_1 and λ_2, then the conditional joint distribution of X, given $X + Y = n$, is binomially distributed.*

Proof. Let $Z = X + Y$. For $k = 0, 1, \ldots, n$ we have that

$$\begin{aligned} p_X(k|Z = n) &= \frac{P(X = k, Z = n)}{P(Z = n)} \\ &= \frac{P(X = k, Y = n - k)}{P(Z = n)} \\ &= \frac{P(X = k)P(Y = n - k)}{P(Z = n)}. \end{aligned}$$

Since we know that Z is also a Poisson with mean $\lambda_1 + \lambda_2$, we get

$$p_X(k|Z = n) = \frac{e^{-\lambda_1} \cdot \frac{\lambda_1^k}{k!} \cdot e^{-\lambda_2} \cdot \frac{\lambda_2^{n-k}}{(n-k)!}}{e^{-(\lambda_1+\lambda_2)} \cdot \frac{(\lambda_1+\lambda_2)^n}{n!}} = \binom{n}{k} \cdot \left(\frac{\lambda_1}{\lambda_1 + \lambda_2}\right)^k \cdot \left(\frac{\lambda_2}{\lambda_1 + \lambda_2}\right)^{n-k}$$

hence it is a binomial distribution with parameters n and $\frac{\lambda_1}{\lambda_! + \lambda_2}$.

□

Theorem 2.2.3. *If $X_1, ..., X_n$ is a random sample from a Poisson distribution with parameter λ, then the conditional joint distribution of $X_1, ..., X_n$, given $Y = \sum_{i=1}^n X_i$, is multinomial with parameters (n, p), where $p = (p_1, \ldots, p_n)$ and $p_i = \frac{\lambda}{n\lambda} = \frac{1}{n}$.*

Proof. The joint probability mass function of the X_i is

$$p_X(x) = \prod_{i=1}^{n} e^{-\lambda} \frac{\lambda^{x_i}}{x_i!} = e^{-n\lambda} \frac{\lambda^{\sum_i x_i}}{x_1! \cdots x_n!}.$$

$Y = \sum_{i=1}^{n} X_i$ is a Poisson random variable with parameter $n\lambda$ and so

$$P\{Y = N\} = e^{-n\lambda} \frac{(n\lambda)^N}{N!}.$$

Now we have that

$$P\{(X_1 = x_1, ..., X_n = x_n) \cap (Y = N)\} = \begin{cases} e^{-n\lambda} \frac{\lambda^{\sum_i x_i}}{x1! \cdots x_n!}, \text{ if } \sum_i x_i = N \\ 0, \text{ if } \sum_i x_i \neq N \end{cases}$$

and so

$$\begin{aligned} p_X(x|Y = N) &= \frac{P\{(X_1 = x_1, \ldots, X_n = x_n) \cap (Y = N)\}}{P\{Y = N\}} \\ &= \frac{N!}{n^N x_1! \cdots x_n!} \text{ if} \sum_i x_i = N \\ &= \frac{N!}{x_1! \cdots x_n!} \left(\frac{1}{n}\right)^{x_1} \cdots \left(\frac{1}{n}\right)^{x_n} \text{ where } \sum_i x_i = N \end{aligned}$$

which is a multinomial distribution with parameters n and p, where $p = (p_1, \ldots, p_n)$ and $p_i = \frac{1}{n}$..

□

We assumed the each X_i to have parameter λ for the sake of simplicity, but nothing changes if we consider each X_i with λ_i as parameter.
We now formalize the Poisson property we nominated before:

Theorem 2.2.4. *Suppose that the number of events N is a Poisson random variable with mean λ. Further suppose that each event can be classified into one of m types with probabilities $p_1, ..., p_m$ independent of all other events. Then the number of events $N_1, ..., N_m$ corresponding to event types $1, ..., m$ respectively, are mutually independent Poisson random variables with means $\lambda p_1, ..., \lambda p_m$ respectively.*

Proof. For fixed $N = n$, the conditional joint distribution of $(N_1, ..., N_m)$ is multinomial with parameters (n, p), where $p = (p_1, ..., p_m)$. Also for fixed $N = n$, the conditional marginal distribution of N_j is binomial with parameters (n, p_j).
The joint probability function of $(N_1, \ldots, N_m)$ is given by

$$\begin{aligned} P(N_1 = n_1, \ldots, N_m = n_m) &= P(N_1 = n_1, \ldots, N_m = n_m | N = n) \cdot P(N = n) \\ &= \frac{n!}{n_1! \cdots n_m!} p_1^{n_1} \cdots p_m^{n_m} \frac{e^{-\lambda} \lambda^n}{n!} \\ &= \prod_{j=1}^{m} e^{-\lambda p_j} \frac{(\lambda p_j)^{n_j}}{n_j!} \end{aligned}$$

where $n = n_1 + \cdots + n_m$. Similarly, the marginal probability function of N_j is given by

$$\begin{aligned} P(N_j = n_j) &= \sum_{n=n_j}^{\infty} P(N_j = n_j | N = n) P(N = n) \\ &= \sum_{n=n_j}^{\infty} \binom{n}{n_j} p_j^{n_j} (1 - p_j)^{n - n_j} \frac{e^{-\lambda} \lambda^n}{n!} \\ &= e^{-\lambda} \frac{(\lambda p_j)^{n_j}}{n_j!} \sum_{n=n_j}^{\infty} \frac{[\lambda(1 - p_j)]^{n - n_j}}{(n - n_j)!} \\ &= e^{-\lambda} \frac{(\lambda p_j)^{n_j}}{n_j!} e^{\lambda(1 - p_j)} \\ &= e^{-\lambda p_j} \frac{(\lambda p_j)^{n_j}}{n_j!} \end{aligned}$$

Hence the joint probability function is the product of the marginal probability functions, establishing mutual independence.

□

After a general introduction to Poisson distribution and some of its useful properties, we will now illustrate the method of estimation by fitting a Poisson model.
Let n_k denote the number of years in which a frequency of exactly k losses occurred. If the likelihood contribution of an observation of k is p_k, then the likelihood for the entire set of observations is

$$L = \prod_{k=0}^{\infty} p_k^{n_k} \tag{2.5}$$

and the loglikelihood is

$$l = \sum_{k=0}^{\infty} n_k \log p_k.$$

The likelihood and loglikelihood functions are functions of the unknown parameters. In this case, with the Poisson distribution there is only one parameter, making the maximization easier.
For the Poisson distribution we obtain

$$p_k = \frac{e^{-\lambda}\lambda^k}{k!}$$

thus we have

$$\log p_k = -\lambda + k\log\lambda - \log k!.$$

The loglikelihood is

$$l = -\lambda n + \sum_{k=0}^{\infty} k\, n_k \log\lambda - \sum_{k=0}^{\infty} n_k \log k!.$$

Differentiating the loglikelihood with respect to λ, we obtain

$$\frac{dl}{d\lambda} = -n + \sum_{k=0}^{\infty} k\, n_k \frac{1}{\lambda}$$

and by setting the derivative of the loglikelihood to zero, the maximum likelihood estimate is obtained as the solution of the resulting equation. The estimator is then

$$\hat{\lambda} = \frac{\sum_{k=0}^{\infty} k n_k}{n}.$$

Thus the estimator has mean and variance respectively equal to

$$E[\hat{\lambda}] = \lambda \quad \text{and} \quad \text{Var}[\hat{\lambda}] = \frac{\lambda}{n}$$

Negative binomial distribution

The negative binomial distribution is often used as an alternative to the Poisson distribution. Because it has two parameters, it has more flexibility in shape than the Poisson. However, this distribution does not possess the properties that make the Poisson very versatile. In particular, Theorem 2.2.4 does not hold for the negative binomial distribution.
The probability function of the negative binomial distribution is given by

$$p_k = \binom{k+r-1}{k}\left(\frac{1}{1+\beta}\right)^r\left(\frac{\beta}{1+\beta}\right)^k,\ k = 0, 1, \ldots,\ r > 0,\ \beta > 0$$

where the binomial coefficient is to be evaluated as

$$\binom{x}{k} = \frac{x(x-1)\cdots(x-k+1)}{k!},\ k \in \mathbb{Z},\ x \in \mathbb{R}.$$

The mean and variance of the negative binomial distribution are

$$E[N] = r\beta \quad \text{and} \quad \text{Var}[N] = r\beta(1+\beta)$$

Because β is positive, it can be seen that the variance in this case exceeds the mean. This is in contrast to the Poisson distribution for which the variance is equal to the mean. This suggests that for a particular set of data, if the observed variance is larger than the observed mean, the negative binomial might be a better candidate than the Poisson distribution as a model to be fitted.
We now examine the maximum likelihood estimation for this distribution. Since the structure of the maximum likelihoood function for the entire set of observations is again (2.5), the loglikelihood for the negative binomial is

$$\begin{aligned} l &= \sum_{k=0}^{\infty} n_k \log p_k \\ &= \sum_{k=0}^{\infty} n_k \left[\log \binom{r+k-1}{k} - r\log(1+\beta) + k\log\beta - k\log(1+\beta) \right]. \end{aligned}$$

The loglikelihood is a function of the two parameters β and r. In order to find the maximum of the loglikelihood we now differentiate with respect to each of the parameters, set the derivatives equal to zero, and solve the system for the parameters. The derivatives of the loglikelihood are

$$\frac{\partial l}{\partial \beta} = \sum_{k=0}^{\infty} n_k \left(\frac{k}{\beta} - \frac{r+k}{1+\beta} \right)$$

and

$$\begin{aligned} \frac{\partial l}{\partial r} &= -\sum_{k=0}^{\infty} n_k \log(1+\beta) + \sum_{k=0}^{\infty} n_k \frac{\partial}{\partial r} \log \frac{(r+k-1)\cdots r}{k!} \\ &= -n\log(1+\beta) + \sum_{k=0}^{\infty} n_k \frac{\partial}{\partial r} \log \prod_{m=0}^{k-1} (r+m) \\ &= -n\log(1+\beta) + \sum_{k=0}^{\infty} n_k \frac{\partial}{\partial r} \sum_{m=0}^{k-1} \log(r+m) \\ &= -n\log(1+\beta) + \sum_{k=1}^{\infty} n_k \sum_{m=0}^{k-1} \frac{1}{r+m} \end{aligned}$$

Setting these equations to zero yields

$$\hat{r}\hat{\beta} = \frac{\sum_{k=0}^{\infty} k n_k}{n} = \hat{\mu}$$

and

$$n\log(1+\hat{\beta}) = \sum_{k=1}^{\infty} n_k \left(\sum_{m=0}^{k-1} \frac{1}{\hat{r}+m} \right).$$

Note that we did not solve the full system in order to underline the fact that the maximum likelihood estimator of the mean is the sample mean.

Finally, these last two equations can be solved either analytically, for example by solving the first equation for $\hat{r}$ and substituting it in the second equation, or with numerical methods (such as the Newton's method).

Severity distributions

As previously mentioned, to derive the loss distributions we can either use the empirical, analytical or moment based approach. Following the company's approach for non-proportional pricing, we will focus on the analytical methods.
It is often desirable to find an explicit analytical expression for a loss distribution. This is particularly the case if the claims statistics are too sparse to use the empirical approach. It should be stated, however, that many standard models in statistics are unsuitable for fitting the claim size distribution. The main reason for this is the strongly skewed nature of loss distributions.
For this reason, a smaller number of distributions is commonly used and, furthermore, each reinsurance company has its own approach towards modelling.
There are different possible choices made by the actuaries when it comes to distributions. This is due to the fact that non-proportional reinsurance contracts are in general more complex than the proportional ones, thus a wider variety of distributions may be needed in order to make the right choice when it comes to modeling different scenarios.
For this reason, we will introduce here just some of the basic distributions used in the company. However, before proceeding please note that one of the distributions used for modelling is once again the lognormal distribution but we will not present it again here.

Exponential distribution

The exponential distribution is the best option if we want to adopt a lighter approach. This means that, since it is a thin tailed distribution, it will tend to lead to a lower average cost per loss with respect to the data we have.
Suppose we observe the first n terms of a sample $\mathbf{x} = (x_1, \ldots, x_n)$ of random variables X, having an exponential distribution. Thus a generic term of the sequence X_j has probability density function

$$f_X(x_j) = \begin{cases} \lambda e^{-\lambda x_j} & \text{if } x_j \in [0, \infty) \\ 0 & \text{otherwise} \end{cases}$$

where the parameter λ is what needs to be estimated.
As usual we consider the likelihood and log-likelihood functions respectively:

$$L(\lambda; \mathbf{x}) = \prod_{j=1}^{n} \lambda e^{-\lambda x_j} = \lambda^n e^{-\lambda \sum_{j=1}^{n} x_j}$$

$$l(\lambda; \mathbf{x}) = n \log \lambda - \lambda \sum_{j=1}^{n} x_j$$

and by differentiating with respect to λ and setting the result equal to zero we obtain the following estimator:

$$\frac{dl}{d\lambda} = \frac{n}{\lambda} - \sum_{j=1}^{n} x_j = 0 \implies \hat{\lambda} = \frac{n}{\sum_{j=1}^{n} x_j}$$

Thus the estimator is nothing but the reciprocal of the sample mean.

Gamma distribution

Suppose we observe the first n terms of a sample $\mathbf{x} = (x_1, \ldots, x_n)$ of random variables X, this time having a Gamma distribution:

$$f(\mathbf{x}; \alpha, \beta) = \frac{\beta^\alpha \mathbf{x}^{\alpha-1} e^{-\beta \mathbf{x}}}{\Gamma(\alpha)} \quad \text{for } \mathbf{x} > 0, \quad \alpha, \beta > 0$$

Notice that if the shape parameter $\alpha = 1$, the exponential distribution results. The loglikelihood function is given by

$$l(\mathbf{x}; \alpha, \beta) = (\alpha - 1) \sum_{i=1}^{n} x_i - n \log \Gamma(\alpha) + n\alpha \log \beta - \beta \sum_{i=1}^{n} x_i \tag{2.6}$$

By computing the partial derivative of the loglikelihood function with respect to β we obtain

$$\frac{\partial l}{\partial \beta} = -\sum_{i=1}^{n} x_i + \frac{n\alpha}{\beta} = 0 \implies \hat{\beta} = \frac{\alpha}{\bar{x}}$$

where $\bar{x}$ is the sample mean.
The next step is to substitute this estimate into (2.6) in order to retrieve $\hat{\alpha}$:

$$l(\mathbf{x}; \alpha, \hat{\beta}) = (\alpha - 1) \sum_{i=1}^{n} x_i - n \log \Gamma(\alpha) + n\alpha \log \alpha - n\alpha \log \bar{x} - n\alpha$$

There are different ways in order to maximize this function since there is no closed form solution for this equation with respect to α. We would need to use an approximation algorithm or some iterative strategy in order to find a good approximation. The same holds for the initial guess α_0.
However, we will not go into further details of the above mentioned strategies since it is not the scope of this section. However, it is good to observe that the Gamma distribution is very useful in creating other distributions.

Pareto distribution

The Pareto distribution is widely used and it is usually fitted to large losses, ie. losses above a threshold. Its cumulative distribution function and probability distribution function are respectively

$$F(x;\alpha,k) = \begin{cases} 1-\left(\frac{k}{x}\right)^{\alpha} & k \le x < \infty\,; \quad \alpha, k > 0 \\ 0 \quad \text{otherwise} \end{cases}$$

$$f(x;\alpha,k) = \begin{cases} \frac{\alpha k^{\alpha}}{x^{\alpha+1}} & k \le x < \infty\,; \quad \alpha, k > 0 \\ 0 \quad \text{otherwise} \end{cases}$$

The parameter k marks a lower bound on the possible values that a Pareto distributed random variable can take on. Indeed, this is the threshold that we rely upon when dealing with large losses.
The mean and variance of a Pareto distribution are given by

$$E[X] = \frac{\alpha k}{(\alpha-1)},\ \alpha > 1\,; \quad \text{Var}[X] = \frac{\alpha k^2}{[(\alpha-1)^2(\alpha-2)]},\ \alpha > 2$$

We are interested in estimating the parameters of the Pareto distribution, so let's consider a sample $x = (x_1, \ldots, x_n)$ of random variables X having the above mentioned distribution. The likelihood function has the following form:

$$L(k,\alpha;\mathbf{x}) = \prod_{i=1}^{n} \frac{\alpha k^{\alpha}}{x_i^{\alpha+1}}\,,\ 0 < k \le min\{x_i\}\,,\ \alpha > 0$$

In general, we maximize functions with calculus. However, we need no calculus to see that L gets larger beyond bound for increases of k. But since k can be no larger than the smallest value of x in our data, the best we can do in maximizing L is to adjust k as

$$\hat{k} = min\{x_i\}$$

Now we just have to find the maximum likelihood estimate for α. Thus we consider the loglikelihood function

$$l(k,\alpha;\mathbf{x}) = \sum_{i=1}^{n} \log\left(\frac{\alpha k^{\alpha}}{x_i^{\alpha+1}}\right) = n\log\alpha + \alpha n \log k - (\alpha+1)\sum_{i=1}^{n}\log x_i$$

By setting its derivative with respect to α to 0 we get the following:

$$\frac{n}{\alpha} + l\log k - \sum_{i=1}^{n}\log x_i = 0$$

thus our final estimator is given by

$$\hat{\alpha} = \frac{n}{\sum_{i=1}^{n} \log\left(\frac{x_i}{k}\right)}$$

The Pareto distribution has a very useful property: the *memoryless* property. *Consider*

$$\bar{F}(x|X > k) = \left(\frac{k}{x}\right)^{\alpha}$$

to be the so called survival function. Then, if we consider an higher thershold $d > k$ we obtain

$$\bar{F}(x|X > d) = \frac{\left(\frac{k}{x}\right)^{\alpha}}{\left(\frac{k}{d}\right)^{\alpha}} = \left(\frac{d}{x}\right)^{\alpha}$$

ie. when we model larger losses, the model forgets the original threshold k, which is not needed anymore, and considers the new threshold d.
That implies:

- If a function has a Pareto tail and we only need to work with quite large losses, we do not need to know exactly where that tail starts. As long as we are in the tail we always have the same parameter α, whatever the threshold be
- If we have two different portfolio with different thresholds, we can still judge whether they have similar tail behavior or not, according to whether they have similar Pareto αs. Such comparison is very useful in reinsurance, where typically to get an overview per line of business one assembles data from several reinsured portfolios, all possibly having different reporting thresholds
- this comparability can lead to market values for Pareto αs, being applicable as benchmarks.

Generalized Pareto distribution

The generalized Pareto distribution (GPD) is part of the *generalized extreme value distribution (GEVD)* family and it is generally used to model excess over thresholds instead of maxima (indeed, the GEVD are usually used to model maxima).
The GPD is a two-parameter distribution with cumulative probability distribution given by

$$F(x; k, \alpha) = \begin{cases} 1 - \left(1 - \frac{kx}{\alpha}\right)^{\frac{1}{k}}, & k \neq 0 \\ 1 - e^{-\frac{x}{\alpha}}, & k = 0 \end{cases}$$

where k is the shape parameter and α is the scale parameter. Notice that both the Pareto and the exponential are special cases of the GPD (for $k < 0$ and $k = 0$

respectively).
Let $\mathbf{x} = (x_1, \ldots, x_n)$ be a sample from the generalized Pareto distribution with parameters k and α. Then, the GPD log-likelihood function is given by

$$l(b, k; \mathbf{x}) = n \log(\frac{b}{k}) - (k^{-1} + 1) \sum_{i=1}^{n} \log(1 + bx_i)$$

where $b = \frac{k}{\alpha}$. Consequently, the maximum likelihood estimators are taken to be the vales which yield a local maximum of the above equation under the constraint that $\alpha > 0$ and $1 + bx_i > 0$ for each $i = 1, \ldots, n$.
The GPD is one of the main distributional models for exceedances over thresholds. These models have been introduced in order to analyze just the useful information without considering all data even when it's not needed.

Weibull distribution

The Weibull distribution is a specific case of the *generalized extreme value (GEV) distribution*

$$H_\xi(x) = \begin{cases} \exp(-(1 + \xi x)^{-\frac{1}{\xi}})), & \xi \neq 0 \\ \exp(-e^{-x}), & \xi = 0 \end{cases}$$

where $1 + \xi x > 0$. In thec case of the Weibull distribution we have that $\xi < 0$.
In the GEV family of distributions there are the only possible non-degenerate limiting distributions for normalized bloch maxima, ie. normalized maxima $M_m = \max(X_1, \ldots, X_n)$ of iid random variables.
The Weibull distribution is a short-tailed distribution with a so called finite *right endpoint*. The right endpoint of a distribution is $x_F = \sup\{x \in \mathbb{R} : F(x) < 1\}$.

Collective model

Now that we know which models to develop for both the number of losses and the amount of a single loss, we can work on the distribution of S, where S is the aggregate loss variable defined in equation (2.4).
The random sum

$$S = X_1 + \cdots + X_N$$

has a distribution function

$$\begin{aligned} F_S(x) &= P(S \leq x) \\ &= \sum_{n=0}^{\infty} P(S \leq x|N = n)P(N = n) \\ &= \sum_{n=0}^{\infty} F_X^{*n}(x)p_n \end{aligned} \tag{2.7}$$

where $F_X(x) = P(X \leq x)$ is the common distribution function of the X_js, $p_n = P(N = n)$ and $F_X^{*n}(x)$ is the "n-fold convolution" of the cumulative distribution function of X. It can be obtained as

$$F_X^{*0}(x) = \begin{cases} 0, & x < 0 \\ 1, & x \geq 0 \end{cases}$$

and

$$F_X^{*k}(x) = \int_{-\infty}^{\infty} F_X^{*(k-1)}(x - y)dF_X(y). \tag{2.8}$$

If X is a continuous random variable with no probability on negative values, the equation (2.8) reduces to

$$F_X^{*k}(x) = \int_0^x F_X^{*(k-1)}(x - y)f_X(y)dy$$

and, by differentiating, the probability distribution function is

$$f_X^{*k}(x) = \int_0^x f_X^{*(k-1)}(x - y)f_X(y)dy.$$

Note that in the case of discrete random variables equation (2.8) is the same but with the summation instead of the integral.
The distribution (2.7) is called a *compound distribution* and the probability function for the distribution of aggregate losses is

$$f_S(x) = \sum_{n=0}^{\infty} p_n f_X^{*n}(x).$$

The probability generating function is given by

$$\begin{aligned} P_S(z) &= E[z^S] \\ &= \sum_{n=0}^{\infty} E[z^{X_1+\cdots+X_N}|N=n]P(N=n) \\ &= \sum_{n=0}^{\infty} E\left[\prod_{j=1}^{n} z^{X_j}\right] P(N=n) \\ &= \sum_{n=0}^{\infty} P(N=n)[P_X(z)]^n \\ &= E[P_X(z)^N] = P_N[P_X(z)] \end{aligned} \tag{2.9}$$

due to the independence of $X_1, \ldots, X_n$ for fixed n.
A similar relationship exists for the other generating functions. It is sometimes more convenient to use the characteristic function

$$\phi_S(z) = E(e^{izS}) = P_N[\phi_X(z)]$$

which always exists. The same holds for the moment generating function.
From (2.9) the moments of S can be obtained in terms of the moments of N and the X_js. The first two moments, ie. the mean and variance, are

$$\begin{aligned} E[S] &= E[X] \cdot E[N] \\ \mathrm{Var}[S] &= \mathrm{Var}[X] \cdot E[N] + (E[X])^2 \cdot \mathrm{Var}[N] \end{aligned}$$

These directly follow from the definitions and expressions of the probability generating and characteristic functions.

Stop-loss case

As we already know, it is common for reinsurance to be offered with a deductible applied to the aggregate losses for the period. In this scenario we would have

$$E[(S-d)_+] = \int_d^{\infty} [1 - F_S(x)]dx$$

where d is the deductible and the notation $(\cdot)_+$ means to use the value in parentheses if it is positive, but to use zero otherwise.
If the distribution is continuous, the net stop-loss premium can be computed directly from the definition as

$$E[(S-d)_+] = \int_d^{\infty} (x-d) f_S(x)dx.$$

The following theorem holds:

Theorem 2.2.5. *Suppose $P(a < S < b) = 0$. Then, for $a \leq d \leq b$,*

$$E[(S-d)_+] = \frac{b-d}{b-a}E[(S-a)_+] + \frac{d-a}{b-a}E[(S-b)_+].$$

That is, when there is an interval with no aggregate probability, the net stop-loss premium can be calculated via linear interpolation.

Proof. From the assumption, $F_S(x) = F_S(a)$, $a \leq x < b$. Then,

$$\begin{aligned} E[(S-d)_+] &= \int_d^\infty [1-F_S(x)]dx \\ &= \int_a^\infty [1-F_S(x)]dx - \int_a^d [1-F_S(x)]dx \\ &= E[(S-a)_+] - \int_a^d [1-F_S(x)]dx \\ &= E[(S-a)_+] - (d-a)[1-F_S(a)]. \end{aligned} \tag{2.10}$$

Then, by setting $d = b$ in (2.10),

$$E[(S-b)_+] = E[(S-a)_+] - (b-a)[1-F_S(a)]$$

and therefore

$$1 - F_S(a) = \frac{E[(S-a)_+] - E[(S-b)_+]}{b-a}.$$

Substituting this in (2.10) produces the desired result.

□

2.3 Experience and exposure ratings

Experience and exposure ratings are the two most prevalent and widely documented approaches to pricing XoL reinsurance contracts. Each of the two methods has its own strengths and weaknesses in any given situation and frequently these methods are used in tandem to price a contract.
Assume that for accident year $t \in \{1, \ldots, T-1\}$, the reinsurer receives the historical losses above a certain threshold A_t. Let the losses in year t be denoted by $C_{1,t}, \ldots, C_{n_t,t}$, where n_t denotes the number of losses in year t. Assume that for each accident year t we dispose of a profile with a structure as presented in Table 2.2, where LB = Lower Bound, UB = Upper Bound, TSI = Total Sum Insured and R_t is the number of rows in the profile in year t.
It is quite common to refer to the rows in the profile as "bands". The average insured value in a band is equal to the ratio between $TSI_{b_t,t}$ and $N_{r_t,t}$, where $r_t \in \{1, \ldots, R_t\}$. Quite often cedents do not only give one profile for their entire portfolio. Often, the

Table 2.2: Risk profiles

Lower Bound	Upper Bound	Number of Risks	Premium	TSI
$LB_1 = 0$	$UB_{1,t}$	$N_{1,t}$	$P_{1,t}$	$SI_{1,t}$
$LB_{2,t} = UB_{1,t}$	$UB_{2,t}$	$N_{2,t}$	$P_{2,t}$	$SI_{2,t}$
$LB_{3,t} = UB_{2,t}$	$UB_{3,t}$	$N_{3,t}$	$P_{3,t}$	$SI_{3,t}$
...	...	...	...	...
$LB_{R_t,t} = UB_{R_t-1,t}$	$UB_{R_t,t}$	$N_{R_t,t}$	$P_{R_t,t}$	$SI_{R_t,t}$

reinsurer receives profiles for different risk types, such as simple risks or commercial risks. In particular, depending from the line of business the actuary can expect to receive the individual risk bordereau instead of bands. Here we assume for simplicity that we dispose only of one profile for the entire portfolio but of course all results can be generalized.
Suppose we want to price an XoL reinsurance program covering fire on a per risk basis in year T with the structure as given in Table 2.3.

Table 2.3: XoL program

Layer	Limit	Retention
XL_1	D_2	D_1
XL_2	D_3	D_2
XL_3	D_4	D_3

What we want to do is to adapt the claim severity and frequency which was observed in the past to the current economic conditions and exposure.

2.3.1 Experience rating

The burning cost is probably the most widely known tool for pricing XoL reinsurance. It simply compares the trended/developed reinsured losses on a given portfolio with the corresponding cedent's on level premium.
The reported burning cost of layer XL_j in year t is calculated as

$$b_{jt} = \frac{\sum_{k_t=1}^{n_t} \min(D_{j+1};\, \max(0;\, C_{k_t,t} - D_j))}{P_t}$$

where in the argument of the summation we are considering the minimum value between the limit of the layer and the amount of loss that exceeds the retention for that layer.
However, past claims are under current conditions and we usually expect them to be more expensive due to the expected increase in costs during time. Therefore, if the

same portfolio is underwritten, we should expect the frequency of the losses exceeding a certain threshold to increase with time. Furthermore, when the composition of a portfolio changes, this may have an impact on the losses distribution above a given threshold. This is why we have to take into account changes in costs.
In the pricing tool of the company, this is done with the *loss development methods* introduced in Section 2.1.2. Indeed, for each layer two on level burns are computed, one with the Chain Ladder method and the other one with the BF method:

- Chain Ladder method:

$$b_{jt} = \frac{\hat{C}_{jt}}{\hat{P}_{jt}}$$

  where $\hat{C}_{jt}$ is the trended/developed loss for layer j in year t derived from equation (2.1) and P_{jt} is the on level premium. The on level premium is given by the original premium multiplied by two factors: the on level factor and the exposure trend. Concerning the former, it is derived from the rate changes. These rate changes are either given by the cedent or, if not provided, the default ones are considered. The default rates are specific for each line of business and they derive from market analyses done by the company teams over the years. Furthermore, the default rates are different if the treaty is based on accident year or policy year. Regarding the latter, it is once again derived from default rates computed over the years for each specific line of business.
  In general, it is possible for the actuary to modify these rates within the pricing tool if different directives are given, either from the cedent or by the underwriters.
  Note that with respect to equation (2.1) the indexing changed. In the Chain Ladder method we considered the accident year and the development year as indexes since we needed to make a prediction over the development years, while here we are considering the accident year and the layer since the prediction has already been made.

- BF method:

$$b_{jt} = \frac{res + C_{jt}}{P_{jt}}$$

  where res is the residual obtained from (2.2) in the BF method, C_t is the trended but not developed loss and P_{jt} is once again the on level premium.

Once both the burning costs are computed, for each year a different selection can be made. Indeed, for each year the age-to-ultimate factor is considered and if its value is greater than 2, then the burning cost from the BF method is selected. Otherwise,

the Chain Ladder burning cost is used.
Once the burning costs are selected for each year, a weighted average is made. In particular, the weights are computed with respect to the on level premiums and it is possible to choose how many years to consider for the average. The actuary can also choose to set specific weights for the burning cost average if other specific selections need to be made.
The selection depends on the "trend" that has been observed, if any. However, it is important to mention that burning costs do not give an expected loss for unused capacity, but rather for used capacity.

2.3.2 Exposure rating

The exposure rating method relies on the risk profiles with the current available portfolio information. Its objective is to estimate the proportion of the loss for the underlying policy that is expected in the excess layer.
The basic idea is, given the risks grouped as in Table 2.2, to apply a single claim distribution per risk band. But since this distribution is not known, we apply *exposure curves*. The exposure curves are constructed for loss history and they allow direct sharing of risk premium between the reinsurer and insurer, where the risk premium is a function of the deductible.
Let Y be the random variable describing the loss for a risk with insured value M, given that there is a loss. The *degree of damage* X is defined as $\frac{Y}{M}$. Let D be a deductible and define d as $\frac{D}{M}$. Let $L(d) = E[\min(d, X)]$ denote the limited expected value function for the risk. If the cedent buys non-proportional reinsurance with a deductible D, then the average retained loss for the risk with insured value M is equal to $L(d)M$. The exposure curve associated with this risk is then denoted and defined by

$$G(d) = \frac{L(d)}{L(1)} = \frac{\int_0^d (1 - F_X(x))dx}{\int_0^1 (1 - F_X(x))dx}$$

where $F_X(x)$ denotes the distribution function of X.
The exposure curve has a very simple interpretation: $G(d)$ represents the portion of the premium which is needed to cover the portion of all losses truncated to a degree of damage d. Indeed, if the exposure curve for a risk is given, its distribution function can be derived from

$$F_X(d) = \begin{cases} 1 & \text{if } d = 1 \\ 1 - \frac{G'(d)}{G'(0)} & \text{if } 0 \leq d < 1 \end{cases}$$

where $F_X(0) = 0$ and $G'(0) = \frac{1}{E[X]}$. This means that the distribution function of a risk and its exposure curve are equivalent representations.
Let's now consider the exposure rating based on a profile. Assume we want to price

a layer with deductible D_j and limit D_{j+1} for a portfolio with a profile in year T as described in Table 2.2. In all bands $b_T \in \{1, \dots, B_T\}$, calculate the ratios

$$r_{b_T,j} = \frac{D_j}{ASI_{b_T,T}}$$
$$s_{b_T,j} = \frac{D_{j+1}}{ASI_{b_T,T}}$$

where $ASI_{b_T,T} = \frac{SI_{b_T,T}}{N_{b_T,T}}$ is the average limit value in band b_T.
Denote for all $b_T \in \{1, \dots, B_T\}$ the exposure curve corresponding to the risks of band b_T as $G_{b_T}(d)$. We assume that $G_{b_T}(d) = 1$ if $d > 1$. Then $G_{b_T}(r_{b_T,j})P_{b_T,T}$ corresponds to the part of the gross premium for band b_T needed to cover all losses arising from risks in band b_T for which the degree of damage is limited to $r_{b_T,j}$.
The part of the gross premium needed to cover all losses between a degree of damage $r_{b_T,j}$ and $s_{b_T,j}$, arising from risks with an insured value of $ASI_{b_T,T}$, is equal to $(G_{b_T}(s_{b_T,j}) - G_{b_T}(r_{b_T,j}))P_{b_T,T}$. The total gross premium needed to cover all losses between D_j and D_{j+1} for the portfolio in year T is given by

$$TP_j = \sum_{b_T=1}^{B_T} (G_{b_T}(s_{b_T,j}) - G_{b_T}(r_{b_T,j}))P_{b_T,T}$$

There are different types of exposure curves that can be used for this rating process and each type of curve is mostly chosen with respect to the line of business and some factors, such as the class and size of risks. Some of the most used ones are the Swiss Re Mbbefd for property EU, the PSOLD for property US and the ILF curves for casualty.

Selection

Once both experience and exposure ratings are performed, a selection has to be made. This selection is based on the two methods and it is related to the given submission. If within the historical data provided by the cedent we have a significant number of losses, ie. good information on loss experience, we may want to select experience. The same holds if we are dealing with a portfolio that is stable over the years.
On the other hand, if we are in possession of a high number of risk profiles, either bands or single risks, and we have good knowledge of the fitting curves for the affected line of businesses, then the exposure fitting may be preferred.
In general, the selection is done with respect to the credibility. Credibility theory helps actuaries understand the risks and it allows reinsurance companies to limit its exposure to claims and losses. Thus the models are built by taking into account a number of assumptions that have been previously statistically tested in order to determine how credible they are. So once the selections for both experience and exposure are done, a final burn is computed by weighting the experience and exposure values with respect to the credibility assumptions.

A first measure of credibility is the number of claims expected during the historical period. Note that this is not the same as the actual number observed during the period. If credibility is based solely on the historical number, then more credibility will be assigned to experience rating projections that are worse than average.
As a second measure of credibility, we could look at the year-to-year variation in the projected loss cost from each of the historical periods for each line of business. Stability in this rate should add credibility even if the number of claims is relatively small.

Chapter 3

Evaluation of treaty features

As we already mentioned in the previous Chapter, after the ground up loss distribution is estimated, other features of the treaty must be evaluated. This is due to the fact that quite often some disagreements remain between the ceding company and the reinsurer about the appropriate ceding commission. For this reason, a negotiation to solve these differences usually takes place so that adjustable agreed upon features can be built into the treaty. These are different for proportional and non-proportional treaties so we are going to analyze them separately.

3.1 Proportional features

In proportional treaties the cedents receive commissions of the premiums ceded to reinsurers. This is due to the fact that they have to compensate the cost of acquiring business, the portfolio performance maintenance and monitoring and, thus, the claims handling.
There are six main features for proportional pricing: provisional commission, sliding scale commission, profit commission, loss corridor, brokerage and general taxes.
The brokerage fee is charged by the broker to execute transactions or provide intermediary specialized services such as purchases, sales, consultations, negotiations.
Regarding the other features, we will now introduce them in more details.

Sliding scale commission

A common adjustable feature is the sliding scale commission. This is a percentage of premium paid by the reinsurer to the ceding company which slides with the actual loss experience, subject to set minimum and maximum amounts.
To clarify the concept, suppose we have the commission terms given in Table 3.1, where the *provisional commission* is an interim payable commission and it is generally fixed between the minimum and maximum payable commissions.
Then the results are as in Table 3.2.

Table 3.1: Commission terms

Provisional commission:	30%
Minimum commission	25% at a 65% loss ratio
Sliding 1:1 to	35% at a 55% loss ratio
Sliding 0.5:1 to a Maximum	45% at a 35% loss ratio

Table 3.2: Sliding scale commission

Commission	Loss Ratio
45%	30% or below
45%	35%
42,5%	40%
40%	45%
37,5%	50%
35%	55%
30%	60%
25%	65% or above

In a balanced plan, it is fair to simply calculate the ultimate commission for the expected loss ratio. However, this may not be appropriate if the expected loss ratio is towards one end of the slide. For example, if the expected loss ratio is 65%, the commission from a simple calculation would be 25%, producing a 90% technical ratio including reinsurance acquisition costs (ie. the sum of the two). If the actual loss ratio is worse than 65% the reinsurer suffers the full amount, but if the actual loss ratio is better than 65% the reinsurer must pay additional commission.
It is actually more correct to see the loss ratio as a random variable and the expected loss ratio as the probability-weighted average of all possible outcomes. The expected ultimate commission ratio is then the average of all possible outcomes based on the loss ratio. This should be done by using an aggregate loss distribution model.

Profit commission

Profit commissions are a type of contingent commission whereby the commission paid from the reinsurer to the insured depends on the defined profitability of a specific book of business over a fixed period of time.
In contrast with straightforward flat commissions, which are based on the premium collected or the renewal of a single policy, the profit commission is calculated based on the financial outcomes of a group of policies. This can be useful in order to create a better alignment of interests and risk/return balance between the two parts.
Although calculations can take a number of forms, a basic formula to find the profit

commission follows this pattern:

(Reinsurance Premiums - Expenses - Incurred Claims) · Profit Percentage

where the expenses could include all expense types, such as taxes or capital charges. The insurance and reinsurance companies must find mutually acceptable terms. Note that many contracts include sliding scales for losses that lower or increase the profit commissions.

Loss corridor

A loss corridor provides that the ceding company will assume again a portion of the reinsurer's liability if the loss ratio exceeds a certain amount. For example, the corridor may be 75% of the layer from an 80% to a 90% loss ratio. If the reinsurer's loss raito is 100% before the application of the loss corridor, then it will have a net ratio of 92.5% after its application:

Table 3.3: Before and after loss corridor

	Before corridor	After corridor	
Below corridor	80%	80%	100% capped at 80%
Within corridor	10%	2.5%	10% - 75% · (90% - 80%)
Above corridor	10%	10%	100% - 90%
Total loss ratio	100%	92.5%	

As above, the proper estimate of the impact of the loss corridor should be made using an aggregate distribution. The probability and expected values for the ranges below, within and above the corridor can then be evaluated.

3.2 Non-proportional features

For non-proportional treaties there are five main features taken into account while pricing: annual aggregate deductible (AAD), aggregate annual limit (AAL), no-claim bonus (NCB), swing rate and reinstatements.

Annual Aggregate Deductible (AAD)

The AAD is a deductible-type program under which the insured agrees to pay for its own losses during the policy year up to the agreed upon annual aggregate amount. Once the reinsured has paid losses up to that amount, the reinsurer pays the remainder of losses for the annual period. This means that the reinsured is responsible for the deductible amount while the reinsurer pays the reduced amount

(from which the deductible is subtracted).
In order to clarify, let's see the following example:

Example 3.2.1. Suppose we have a XoL treaty 900 000 xs 100 000 with AAD of 1 million.

Table 3.4: AAD on a series of five cases in chronological order

	Loss	Retention	AAD	AAD to date	Reinsurance
	500 000	100 000	400 000	400 000	0
	50 000	50 000	0	400 000	0
	200 000	100 000	100 000	500 000	0
	900 000	100 000	500 000	1 000 000	300 000
	400 000	100 000	0	1 000 000	300 000
Total	2 050 000	450 000	1 000 000	1 000 000	600 000

Aggregate Annual Limit (AAL)

The aggregate annual limit is the maximum amount of coverage that a reinsurance company provides over a treaty year. Once the covered expenses reach the annual aggregate, the reinsurer stops paying out benefits even if subsequent legitimate claims are filed.
Typically rensurance companies set limits both on individual claims and on aggregate claims. The AAL is usually introduced because it would simply be too expensive not to limit coverage. Let us consider an example:

Example 3.2.2. A policy has a 2 500 000 per claim limit and an aggregate limit of 10 000 000. If the cedent makes a single claim for 5 000 000, the reinsurer company pays only 2 500 000, ie. the per claim limit, even though it is under the aggregate limit. The aggregate limit is now 7 500 000.
A second claim of 6 000 000 in the same period results in another 2 500 000 payout and a reduced aggregate limit of 5 000 000.
Three claims incur and their amounts are respectively 7 000 000, 3 000 000 and 4 000 000. At this point the reinsurer will cover 2 500 000 for the first two losses, but the third loss will be fully paid by the insurer since the aggregate annual limit of 10 000 000 has been reached.

No-Claim Bonus (NCB)

In Motor reinsurance, the NCB is a reward provided by reinsurance companies to insurance companies for making no claims during the policy term. Indeed, for every claim-free year, the reinsured receives a discount on his premium. The discount percentage increases with every passing, claim-free year.

Note that the key feature of NCB is that it is associated with the company and not the single vehicle. However, NCB cannot be claimed on the first motor reinsurance policy since there is not a claim record yet. Starting from the first renewal of the policy provided there can be a discount on the premium paid if there has been no claim during the past year and this discount will increase steadily with every claim-free year up to a maximum discount.
For the NCB feature we assume that the cedent will not claim losses if their net benefit is less than the payable NCB.

Swing rate

The swing rate offers the reinsured a target premium rate which can then be adjusted up or down depending upon the actual claim experience for the treaty for the given year. For example, if experience is good, the final rate is adjusted downward of a specific percentage. If claim experience is poor, then an additional percentage increase in premium is assumed.
A swing rate is used where the reinsured's perception of new or emerging claim experience is significantly below the reinsurer's evaluation of the experience. In essence, they are willing to bet on favorable experience. A swing rate would also be used on newer blocks of business with little experience. In this case, one sets the swing rate to give the treaty the opportunity for an "experience refund" in exchange for upside protection to the reinsurer.
This is an arrangement that allows the two parties to modify the conventional risk arrangement so that the treaty still has coverage in excess of a certain additional premium corridor as well as for very favorable experience.

Reinstatement

When the original limit of cover is all used, the reinsured will have no cover left for any further loss. In order to manage this situation, reinsurers allow the reinsured to have the original limit reinstated once it is fully or partially used up by a loss. Reinstatement can either be limited or unlimited and it can come either free or at a cost. This mostly depend on the line of business. If it is not free, then the additional premium paid is known as *reinstatement premium*.
The reinstatement premium can be calculated in two different ways:

- As to amount, where the reinstatement premium is calculated based on the size of the loss. The treaty will usually state the percentage of additional premium on which the reinstatement premium should be calculated.

$$\text{Reinstatement Premium} = \frac{\text{Loss to the reinsurer}}{\text{Cover Limit}} \cdot \text{Reinsurance Premium}$$

- As to time, where the reinstatement premium is calculated based on the size of the loss and prorated for the number of days from the occurrence of the loss to the expiry of the treaty. Thus it is computed as before, but the result is then multiplied by the ratio between the number of days passed from the date of loss and 365.

Let us consider the following example, where we suppose the reinstatement premium to be as to amount at 100% additional premium:

Example 3.2.3. An insurance company has an 80 000 000 xs 20 000 000 per risk XoL reinsurance program. It incurs some losses as shown in Table 3.5.

Table 3.5: 80 000 000 xs 20 000 000 per risk XoL reinsurance program

Loss	**Amount**	**Deductible**	**Reinsurance**	**Total reinsurance**
1	40 000 000	20 000 000	20 000 000	20 000 000
2	30 000 000	20 000 000	10 000 000	30 000 000
3	50 000 000	20 000 000	30 000 000	60 000 000
4	45 000 000	20 000 000	20 000 000	80 000 000
5	35 000 000	20 000 000	0	80 000 000

Since the reinsurance limit has been hit after four losses, the reinsured would have to bear the deductible of 20 000 000 and an additional 5 000 000 for the fourth loss and it would have to cover the last loss in full.

To avoid these scenarios reinsurers include a provision to reinstate the initial cover purchased each time it is used up by a loss. The treaty will state the number of reinstatements that the reinsurer offers with respect to the cedent's needs and specifies the percentage of additional premium related to each reinstatement.
Let us consider again Example 3.2.3, but this time with one reinstatement at additional premium. The situation is presented in Table 3.6, where in the second column (ie. "Remaining") we considered the loss amount minus the deductible.

Table 3.6: 80 000 000 xs 20 000 000 per risk XoL reinsurance program with one reinstatement

Loss	**Remaining**	**Reinstatement**	**Reinsurance**
1	20 000 000	20 000 000	0
2	10 000 000	10 000 000	0
3	30 000 000	30 000 000	0
4	25 000 000	20 000 000	5 000 000
5	15 000 000	0	15 000 000
Total		80 000 000	20 000 000

The insurance company has an additional cover of 80 000 000 that can use to bring back the original cover limit to its full amount when it is either partially or fully used by a loss. As we can see, the reinstatement covers everything for the first three losses, up to 20 000 000 for the fourth loss and nothing for the last one. This is due to the fact that the cover limit of 80 000 000 has been reached. The amount not covered by the reinstatement is covered by the XoL reinsurance program.

3.3 Evaluation

As we know from the previous chapter, the distribution function of the aggregate loss is given by

$$S(x) = \sum_{n=0}^{\infty} F_X^{*n}(x) p_n$$

where $p_n = P(N = n)$ and $F_X^{*n}(x)$ is the "n-fold convolution" of the cumulative distribution function of X.
This expression cannot be exactly evaluated for most distributions so it is necessary to rely on numerical methods. We will introduce here two of the most used methods: the *Monte Carlo simulation* and the *Panjer recursion*.

3.3.1 Monte Carlo simulation

The Monte Carlo (MC) method is one of the easiest numerical methods used to calculate the aggregate loss distribution. The logical steps are the following:

1. *For* $k = 1, \ldots, K$
   (a) *Simulate the number of losses* N *from the frequency distribution*
   (b) *Simulate independent* $X_1, \ldots, X_N$ *from the severity distribution*
   (c) *Calculate* $S_k = \sum_{i=1}^{N} X_i$
2. *Do an increment* $k = k + 1$ *and return to step 1*

All random numbers simulated in the above are independent.
Thus the obtained $S_1, \ldots, S_K$ are samples from an aggregate distribution $S(\cdot)$.
However, some problems may arise concerning the MC simulation, mostly related to the fact that the simulated portfolio may be subject to high variability, unless the number of simulations is very large.
There are different teqniques that allow for variance reduction. We will describe here one of the most used, known as *importance sampling*. This variance reduction method will be explained and introduced here since it is good knowledge to be aware of its existence, but it is not actually performed during the pricing process.

Importance sampling

Consider a random variable X and assume that it has an absolutely continuous distribution function with density f. The problem considered is the computation of the expected value

$$\theta = E[h(X)] = \int_{-\infty}^{\infty} h(x)f(x)dx \tag{3.1}$$

for some known function h. Where the analytical evaluation of this integral is difficult, mostly due to the complexity of the distribution of X, we can resort to a MC approach where we only have to be able to simulate variates from the distribution with density f.
What we do is to generate $X_1, \ldots, X_n$ independently form the density f and then compute the standard MC estimate

$$\hat{\theta}_n^{MC} = \frac{1}{n}\sum_{i=1}^{n} h(X_i).$$

The MC estimator converges to θ by the strong law of large numbers, but the speed of convergence may not be particularly fast.
Importance sampling is based on an alternative representation of the integral in (3.1). Consider a second probability density g whose support should contain that of f and define the likelihood ratio $r(x)$ as the ratio between $f(x)$ and $g(x)$ whenever $g(x) > 0$, and $r(x) = 0$ otherwise. The integral (3.1) may be written in terms of the likelihood ratio as

$$\theta = \int_{-\infty}^{\infty} h(x)r(x)g(x)dx = E_g[h(X)r(X)],$$

where E_g denotes expectation with respect to the density g. Hence we can approximate the integral in the following way:

1. *Generate $X_1, \ldots, X_n$ independently from the density g*

2. *Compute the importance sampling estimate*

$$\hat{\theta}_n^{IS} = \frac{1}{n}\sum_{i=1}^{n} h(X_i)r(X_i).$$

The main point of importance sampling lies in choosing a density g such that, for fixed n, the variance of the importance sampling estimator is considerably smaller than that of the standard MC estimator. In this way we can hope to obtain a prescribed accuracy in evaluating the integral of interest using far fewer random draws than are required in standard MC simulation.
The variances of the estimators are given by

$$\mathrm{Var}(\hat{\theta}_n^{MC}) = \frac{1}{n}(E[h(X)^2] - \theta^2)$$
$$\mathrm{Var}_g(\hat{\theta}_n^{IS}) = \frac{1}{n}(E_g[h(X)^2 r(X)^2] - \theta^2)$$

so that the aim is to make $E_g[h(X)^2 r(X)^2]$ small compared with $E[h(X)^2]$. In theory, the variance of $\hat{\theta}^{IS}$ can be reduced to zero by choosing an optimal g. To see this, suppose for the moment that h is non-negative and set

$$g^*(x) = \frac{f(x)h(x)}{E[h(X)]} \tag{3.2}$$

With this choice, the likelihood ratio becomes

$$r(x) = \frac{E[h(X)]}{h(x)}$$

Hence, $\hat{\theta}_1^{IS} = h(X_1)r(X_1) = E[h(X)]$ and the importance sampling estimator gives the correct answer in a single draw. In practice, it is of course impossible to choose a g of the form (3.2) as this requires knowledge of the quantity $E[h(X)]$ that one wants to compute. However, (3.2) can provide useful guidance in choosing a importance sampling density.

3.3.2 Panjer recursion

The Panjer method is widely used for the computation of the aggregate loss distribution since it appears that for some class of frequency distributions this calculation can be reduced to a simple recursion.

Panjer recursion

If the frequency probability mass function $\mathbb{P}(N = n)$, $n = 0, 1, \ldots$, *satisfies*

$$\mathbb{P}(N = n) = \left(a + \frac{b}{n}\right) \mathbb{P}(N = n - 1), \quad \textit{for} \quad n \geq 1 \quad \textit{and} \quad a, b \in \mathbb{R},$$

then it is said to be in Panjer class (a, b, 0). Furthermore, if the claim size distribution is discrete, then for $m \in \mathbb{N}$ *the aggregate distribution* S *satisfies the recursion*

$$\begin{aligned} \mathbb{P}(S = m) &= \frac{1}{1 - a\,\mathbb{P}(X = 0)} \sum_{j=1}^{m} \left(a + \frac{bj}{m}\right) \mathbb{P}(X = j)\, \mathbb{P}(S = m - j), \\ \mathbb{P}(S = 0) &= P_N(\mathbb{P}(X = 0)) \end{aligned} \tag{3.3}$$

where P_N is the probability generating function of N.
Note that if $\mathbb{P}(X = 0) = 0$, then $\mathbb{P}(S = 0) = \mathbb{P}(N = 0)$.

It is important to observe that in the above theorem we are making two strong assumptions: we need the frequency distribution to be in the Panjer class $(a, b, 0)$ and we require the claim size distribution to be discrete.
In the following we prove that both the Poisson and the Negative Binomial belong to the Panjer class $(a, b, 0)$ and we introduce the idea behind the discretization of the claim size distribution.

Poisson

If $N \sim Poisson(\lambda)$, then for $k \in \mathbb{N}$

$$\mathbb{P}(N = k) = \frac{\lambda^k e^{-\lambda}}{k!} = \frac{\lambda}{k} \frac{\lambda^{k-1} e^{-\lambda}}{(k-1)!} = \frac{\lambda}{k} \mathbb{P}(N = k-1)$$

thus N belongs to the Panjer class $(a, b, 0)$ with $a = 0$ and $b = \lambda$.

Negative Binomial

If $N \sim NB(r, p)$, where $r \in \mathbb{N}$ is the number of successes and $0 \leq p \leq 1$ is the probability of success, then for $k \in \mathbb{N}$ (number of failures) we have

$$\begin{aligned}\mathbb{P}(N = k) &= \binom{k+r-1}{r-1} (1-p)^k \, p^r = \frac{(k+r-1)!}{(r-1)!\, k!} (1-p)^k \, p^r \\ &= \frac{k+r-1}{k} (1-p) \frac{(k+r-2)!}{(r-1)!\, (k-1)!} (1-p)^{(k-1)} \, p^r\end{aligned}$$

thus N belongs to the Panjer class $(a, b, 0)$ with $a = 0$ and $b = (k + r - 1)(1 - p)$.

Discretization

Assume the claim size distribution is continuous with cumulative distribution function F_X. By discretizing the domain of X we define

$$h_j = F_X(j) - F_X(j-1), \forall j = 1, 2, \ldots, n$$
$$H(x) = \sum_{j \leq x} h_j$$

and therefore $F_X(0) = 0$, so $H(x) \leq F_X(x)$. We define then

$$\tilde{h}_j = F_X(j+1) - F_X(j), \forall j = 0, 1, \ldots, n-1$$
$$\tilde{H}(x) = \sum_{j \leq x} \tilde{h}_j$$

and therefore $\tilde{H}(x) \geq F_X(x)$. Thus we have

$$H(x) \leq F_X(x) \leq \tilde{H}(x) \tag{3.4}$$

where both $H(x)$ and $\tilde{H}(x)$ are discrete. Hence for the distribution of the aggregate claim we can apply the Panjer recursion and obtain lower and upper bound for $\mathbb{P}(S \leq x)$:

$$\mathbb{P}_L(S \leq x) \leq \mathbb{P}(S \leq x) \leq \mathbb{P}_U(S \leq x)$$

where $\mathbb{P}_L(S \leq x)$ and $\mathbb{P}_U(S \leq x)$ are obtained by the Panjer recursion by using $H(x)$ and $\tilde{H(x)}$ respectively.
The number of operations to calculate the aggregate loss distribution explicitly is of the order of n^3. If the maximum value for which the aggregate loss distribution should be calculated is large, the number of computations become prohibitive due to $\mathcal{O}(n^3)$ operations. The Panjer recursion instead requires $\mathcal{O}(n^2)$ operations to calculate it. The Panjer recursion formula (3.3) can be extended to a class of frequency distributions $(a, b, 1)$.

Extended Panjer recursion

If the frequency probability mass function $\mathbb{P}(N = n)$, $n = 2, 3, \ldots$, satisfies

$$\mathbb{P}(N = n) = \left(a + \frac{b}{n}\right) \mathbb{P}(N = n - 1), \quad \textit{for} \quad n \geq 2 \quad \textit{and} \quad a, b \in \mathbb{R},$$

then it is said to be in Panjer class $(a, b, 1)$. Furthermore, if the claim size distribution is discrete, then for $m \in \mathbb{N}$ the aggregate distribution S satisfies the recursion

$$\begin{aligned}
\mathbb{P}(S = m) &= \frac{1}{1 - a\,\mathbb{P}(X = 0)} \left(\mathbb{P}(N = 1) - (a + b)\,\mathbb{P}(N = 0)\right) \mathbb{P}(X = m) \\
&\quad + \sum_{j=1}^{m} \left(\frac{a + bj}{m}\right) \mathbb{P}(X = j)\,\mathbb{P}(S = m - j), \\
\mathbb{P}(S = 0) &= P_N(\mathbb{P}(X = 0))
\end{aligned}$$

where P_N is the probability generating function of N.
Note that if $\mathbb{P}(X = 0) = 0$, then $\mathbb{P}(S = 0) = \mathbb{P}(N = 0)$.

The distributions of $(a, b, 0)$ class are special cases of $(a, b, 1)$ class.
Note that this generalization of the Panjer recursion can be extended to the (a, b, l) class.

Chapter 4

Application on client data

The challenge with reinsurance treaties is that we usually have a low number of reported losses. This is the reason why we will try to estimate the frequency and severity distributions of the data submission provided by the client. Indeed, our main goal is to simulate a higher number of random values following the data distribution in order to be as close as possible to the actual distribution.
The aim of this Chapter is to present a pricing model with the main scope of implementing the aggregate loss model from scratch. Indeed, after selecting a cedant within all the ones that provided their submissions so far, we will introduce the full pricing process by starting from the client submission and concluding with the final terms. However, the goal will be to create, step by step, the compound model by translating in Python's language the procedures used in the Excel company's pricing tool. This will be a good validation for the company since it will allow for a parallelism with the pricing tool. Finally, we will implement the Panjer recursion and compare the results obtained with Monte Carlo in order to see if some improvements can be made in terms of time and efficiency.

4.1 Data submission

After choosing the cedant's submission for the analysis, we start by having a look at the folder provided.
The treaty we are looking at is a renewal contract whose package was submitted at the end of September. In the submission folder we find:

- Excel files containing the actual data, such as triangles, historical figures, risk profiles, risk listings, information on cat events..
- Slip, containing the treaty information and the final terms
- Submission email, always included in the submission folder in order to keep track of the deadlines and of the data provided

Before proceeding into the submission analysis, there are some generalities about submissions that are worth mentioning.
We are not always provided with slip, wording and/or final terms. If that is the case, ie. if we do not know which is the structure that the client wants us to price, we usually price the expiring structure unless some specifics are stated in the submission email. Indeed, the submission email is often used as an efficient and fast way for the client to underline the main changes in the structure/data with respect to the previous year's submission. However, if the treaty is new, ie. if we do not have previous submissions or pricings to refer to, we expect to have the structure provided. If that is not the case, we need to ask the underwriters for more information.
This brings us to the second main issue we may encounter in these first steps. If the submission is not complete enough for us to price the treaty or if while proceeding with the analyses we see some inconsistencies in the data, we should always ask for clarification to the underwriters. This may take a while since most of the times they have to go back to the client and ask for information. That is the reason why this is done just when strictly necessary and not for every small inconvenient or doubt.
Finally, the Excel submission can be very straightforward to analyze or quite dirty. This is why we always have to pay attention to what the client provides us and the structure we are dealing with. Indeed, most of the times we will not need all the information provided but just a part of it.
After a quick overview on the possible scenarios we might encounter when dealing with a submission, we can get started with our analysis.

4.1.1 Structure

The treaty we are dealing with is an excess of loss reinsurance contract and we know its cover and structure. We did some transformation thus the data and structure will not directly refer or correspond to any actual cedant.
The structure has four layers and we will not specify which line of business this treaty covers. The layering we will consider is the following: 2 million xs 3 million, 5 million xs 5 million, 15 million xs 10 million and 25 million xs 25 million.
From the Excel files provided in our submission we just retrieve the information we need to build the aggregate loss model in order to maintain anonymity. Indeed, we will just consider the triangles containing the development of claims along the years and the exposure amounts for each year. With "exposure amounts" we may refer to whatever exposure information is provided, such as number of insured, premiums, vehicle years and so on. The provided information depends on the line of business we are pricing. However, nothing changes if we use one or the other thus we will not go into further details regarding our submission data. Furthermore, we will also change the numbers in order to hide the true values provided in the submission.

4.1.2 Imported data

Since the exposure amounts will just be considered later on in the analysis process we will now focus on the claims information. There are 287 reported losses: these are the claims that at least once in their historical development have exceeded 3 million, which is the threshold specified by the cedant. Each row corresponds to a loss and for each loss the following information is provided:

- *Section key*, an integer number mapping the line of business. In general this is a useful information since each line of business has to be dealt with differently
- *Claim ID*, a text variable that contains the identification code for each loss
- *Date of loss*, a data type variable stating the date in which the loss occurred. The year of occurrence goes from 1989 to 2021
- *Comment*, a text variable providing information about the loss. These comments may or may not be provided and they can be useful for us to identify whether a loss is catastrophe/event related or not. This may be important because if we have cat/event losses we need to deal with them separately
- *Cat Loss*, a text variable assuming the values "Yes" or "No" if the loss is cat related or not respectively
- *Per Event Loss*, a text variable assuming the values "Yes" or "No" if the loss is event related or not respectively
- The development of losses, 20 columns of our dataset where each column contains the loss evaluated at 31/12 for each year from 2002 to 2020 and at 30/06 for 2021. Thus each row of this triangle represents the development of a single loss from 2002 to 2021

Concerning this last point, note that the client provides the single losses with their respective development thus our goal in reinsurance will be to build an aggregate development triangle to analyze the client's overall loss state.
Before proceeding with the actual data analysis, some cleansing can me made. Indeed, we can just retrieve the information we need in order to proceed with our study.
First of all, by looking at the Section key column we can notice that it just assumes one value: this means that we are dealing with just one line of business thus we do not have to make distinctions when it comes to dealing with losses and for this reason we can disregard this column.
The same holds for the catastrophe and event related columns since all the entries are equal to "No". Indeed, we can disregard these columns by keeping in mind that all the claims we are dealing with are not either cat or event related. Furthermore, we can specify once again that if these columns are not provided we can build them by checking if some information is available within the losses comments. If no comments

are available, we will just suppose that all the losses are not cat/event related and we will probably include a load for all those losses that are covered by the treaty but not present in the submission (cat load per risk, cat load per event, ...).
Regarding the claim ID, we will not really use it since it is just a loss identifier thus it is unique for each loss. However, usually it is provided because it can be useful to find a loss that may have a strange behavior in the model or that may have been updated by the client at a later point with respect to the submission. Indeed, it can happen that a loss has a relevant change occurred after the last evaluation date provided in the data and that the client wants us to know. If that is the case it is important for us to be aware of that and to update that loss according with the most recent information provided.

4.2 Data analysis

Let's now proceed with the analysis of the dataset in our possess. As we know already, we have reported losses from 1989 to 2021 thus it might be interesting to have an overall look on how many losses evaluated at 30/06/2021 are above the threshold for each year:

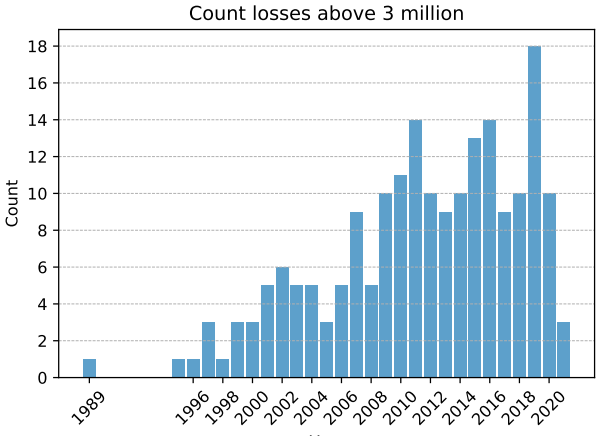


Figure 4.1: Count of losses from 1989 to 2021 at 30/06/2021 that are above the threshold.

Note that the loss count seems to increase over the years. This might be due to the increasing size of the book over time or the inflation over the years.
It can be also important to have a look at the smallest and biggest losses in order for us to have an overall idea of the situation we find ourselves in. The smallest incurred developed loss at 30/06/2021 occurred in 2014 and it is of 129 while the biggest one is of 31.9 million and it occurred in 2020. As we can notice, the gap between the smallest and biggest incurred losses is quite big. However, the cedant gave us a

threshold of 3 million. This means that in the submitted triangle we have all the losses that at least once in their historical development have exceeded 3 million.

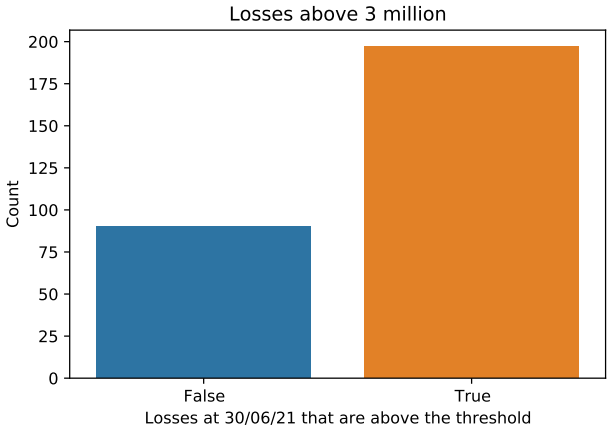


Figure 4.2: Losses that at the last evaluation date are above the client's threshold.

In Figure 4.2 we reported all the losses that at the latest evaluation date (ie. 30/06/2021) are above 3 million. The threshold is a very useful information since it allows us to build a severity model for losses above the given threshold by keeping in mind that both the threshold and the losses are subject to indexation. This is important since it can take years before a claim is settled and as we already know inflation can have a quite strong impact on claims overtime. If there is not indexation inflation could cause a loss to reach the threshold amount sooner and more frequently than expected.
Furthermore, it is relevant to know the threshold since it would not make too much sense to simulate losses that are not significant in terms of loss amount or that are not very frequent within the client submission. Also, if we consider the modeling theory introduced in Chapter 2 we know that a lot of distributions (for example the Pareto distribution) are usually used to model the severity over a specific threshold.
Another thing we should look at when it comes to the development triangle is the presence of relevant jumps in loss developments. This may be important since, whenever possible, a comparison with last year's pricing is done and it is of interest to keep track of big changes in the history of claims.
In Figure 4.3 we reported three losses, from 2007, 2008 and 2009, that have interesting changes in their historical development. For example, the 2008 loss hit the third layer's retention in 2010 but in 2013 decreased again going below 10 million. More interesting is the 2009 loss that hit the fourth layer's retention just in 2020 since it jumped over 25 million decreasing right after in 2021. This might be relevant to observe since we are dealing with a quite old loss having very recent development with a quite big increase thus it could have an impact on the pricing and, if that is the case, it is important for us to know where that impact might come from. For

example, if we are dealing with bodily injury claims we could have loss changes related to the victim's physical condition or if we are dealing with court trials both the judge and the time needed to deal with the bureaucracy might influence a lot the loss amount.

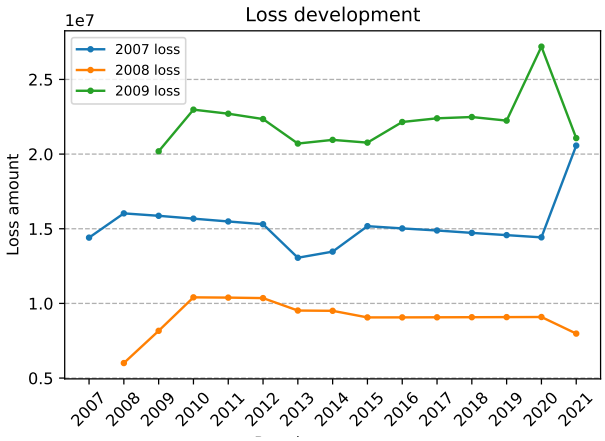


Figure 4.3: Three losses with some jumps in their development.

After a few quick observations on the dataset provided we can now have a deeper look into the claims by trying to evaluate the respective severity and frequency distributions.

4.2.1 Severity distribution

The first thing we would like to do is to have a look at the univariate distribution of claim severity in order to guess which distribution within the ones mentioned in Chapter 2 could potentially fit better our data.
For the severity modelling we will just consider the losses at their last evaluation date (ie. 30/06/2021) and we will work just with the ones above the provided threshold. Indeed, we end up having 197 losses over the 287 we had in the beginning and their distribution is reported in Figure 4.4. As we can see the distribution seems to be heavy tailed thus maybe a Pareto or Gamma distribution could fit well our claims severity. However, even if it can be helpful to have a graphical idea, that is not enough to draw conclusions. Thus we perform a Kolmogorov-Smirnov test to check which distribution could better fit our data. We chose the KS test because on Python we have the possibility to set the parameters that we want to use in order to fit our data and then compare them with a specific distribution.
In order to estimate the parameters of the distribution we use the Maximum Likelihood Method and then we perform a KS test that compares our data distribution with MLE estimated parameters and the standard distributions.
The null hypothesis states that the two distributions considered are the same, thus

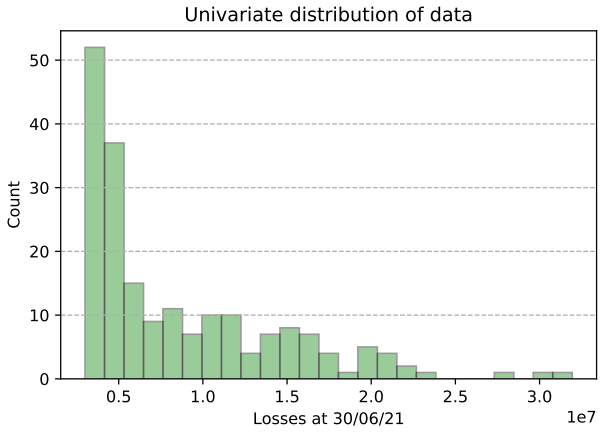


Figure 4.4: Univariate distribution of data.

if our p-value is higher than a specified alpha (in our case 0.05 since it is the value that it is usually considered) we cannot reject the null hypothesis.
The Pareto, Gamma, Exponential, Normal, LogGamma and LogNormal distributions were used for comparison. This is the output we obtained:

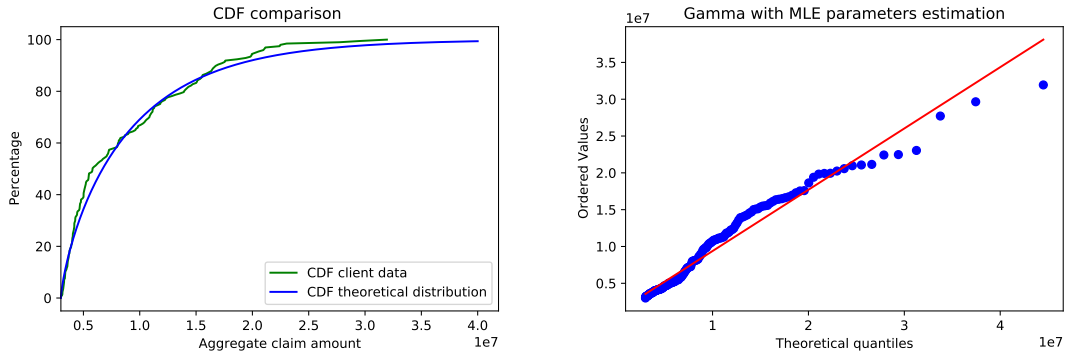
```
gamma: statistic=0.08703204260587205, pvalue=0.09527153148730717
pareto: statistic=0.10151475878943217, pvalue=0.03207368435452846
expon: statistic=0.12492337544981819, pvalue=0.0038730736970923807
loggamma: statistic=0.18081955399322003, pvalue=4.155228433178794e-06
norm: statistic=0.18735138459192763, pvalue=1.5821894236359815e-06
lognorm: statistic=0.7389885100463385, pvalue=5.8164300008446925e-111
```

The Gamma distribution appears to be by far the best one to fit our data, followed by the Pareto distribution. Thus we will simulate 10000 random numbers from a Gamma distribution with estimated parameters:

```
alpha: 0.7394863091112791
loc: 3000470.474312
scale: 8341367.9134019185
```

where α is the shape parameter. As we can see the location parameter is quite close to 3 million. This is a reasonable result since we are just considering the claims above the given threshold. The same holds for the scale parameter: we are dealing with quite big losses thus we expected a quite big number for scaling.
To have a clearer overview on the situation, we performed a two-sample KS test since it allows us to compare two distributions of two independent samples. Indeed, we compared our data distribution and the Gamma with the above estimated parameters. We obtain a p-value of 0.172 thus we do not have enough evidence to reject the null hypothesis that our data distribution and the distribution obtained by generating random numbers from a Gamma with MLE estimated parameters are the same.
By plotting our simulated data distribution against the empirical one we obtain the

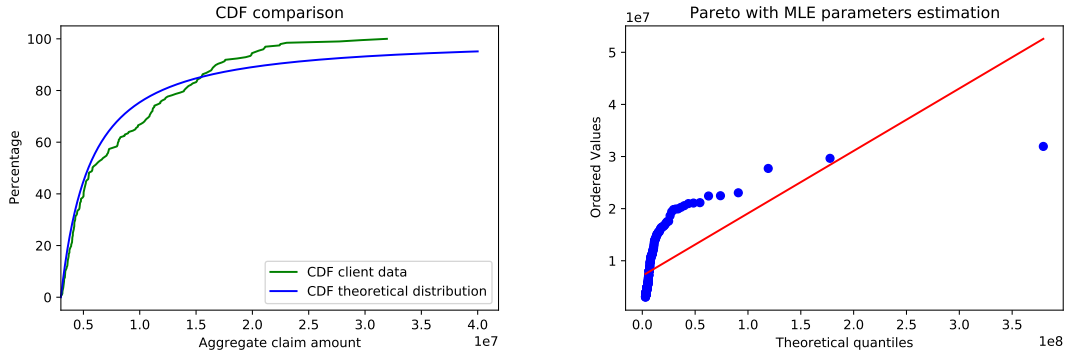
graph presented in Figure 4.5(a). We also reported in Figure 4.5(b) a QQ-plot to have another term of comparison.



((a)) Client data against theoretical distribution **((b))** QQ-plot

Figure 4.5: Gamma distribution: CDFs and QQ-plot.

Note that our biggest loss is at 31.9 million thus it is reasonable to see our data cumulative distribution function stopping earlier than the empirical one.
From the QQ-plot we can see that the graph is "right skewed", meaning that most of the data is distributed on the left side with a long tail of data extending out to the right. This seems reasonable since we are dealing with a Gamma distribution.
We did the same analysis for the Pareto distribution too in order to have a term of comparison. Here we report the cumulative distribution functions and the QQ-plot obtained:



((a)) Client data against theoretical distribution **((b))** QQ-plot

Figure 4.6: Pareto distribution: CDFs and QQ-plot.

As we can see from a graphical point of view the outcome is definitely worse with respect to the Gamma distribution. That is probably due to the fact that the Pareto

distribution is quite heavy tailed with respect to our sample thus even if we include a scaling and a locating parameter in our estimation we will not obtain a result that is close to our claims.
The p-value obtained from the two-sample KS test is in this case equal to 0.084 but even if we do not have enough evidence to reject the null hypothesis that our data distribution follows a Pareto distribution the p-value is lower than the one we obtained before and the graphs are a lot worse than the ones obtained with the Gamma distribution.
Consequently, given all the above considerations we assume that the claim severity follows a Gamma distribution with parameters estimated by the Maximum Likelihood Method applied on our data.

4.2.2 Frequency distribution

When it comes to the frequency distribution we cannot proceed in the same way as we did in the previous section because we do not have enough historical information about the frequency of losses. Thus we implement in Python the same process that lies behind the company pricing tool by keeping in mind that there are not many count distributions available for this purpose and that the standard one is the Poisson distribution. We want to find the expected number of claims occurring in 2022 in order to use it as the parameter needed to simulate the Poisson distribution.
We will consider a count triangle of claims exceeding 3 million. Note that we will consider it starting from 2007 on because we are just interested in the last 15 years of development since the older the years the more uncertainty we have (due to indexation or inflation for example). Then the loss to the layer for each year is nothing but the number of claims exceeding the threshold that occurred in that year. The reason why we do this on the claims development triangle and not just on the claim final evaluation is because we want to keep track of the history of each loss in order to be aware of when a loss is above or below the threshold during time.
In the following table we report the loss count triangle obtained by counting all losses above 3 million and then aggregating for each year, where each column corresponds to each development year:

	Development year														
Year	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**
2007	10	10	12	13	12	9	8	8	8	8	8	8	8	9	9
2008	5	5	6	5	6	6	8	7	7	6	6	6	6	5	-
2009	9	11	12	10	10	10	10	9	9	9	10	10	10	-	-
2010	17	14	14	14	12	11	12	11	11	11	11	11	-	-	-
2011	11	13	11	11	13	12	13	13	13	14	14	-	-	-	-
2012	6	8	7	7	10	11	11	11	10	10	-	-	-	-	-
2013	7	11	12	12	11	12	12	11	9	-	-	-	-	-	-
2014	10	10	9	8	9	10	9	10	-	-	-	-	-	-	-
2015	16	20	18	16	15	13	13	-	-	-	-	-	-	-	-
2016	11	14	16	16	13	14	-	-	-	-	-	-	-	-	-
2017	8	11	10	10	9	-	-	-	-	-	-	-	-	-	-
2018	7	12	11	10	-	-	-	-	-	-	-	-	-	-	-
2019	13	19	18	-	-	-	-	-	-	-	-	-	-	-	-
2020	10	10	-	-	-	-	-	-	-	-	-	-	-	-	-
2021	3	-	-	-	-	-	-	-	-	-	-	-	-	-	-

By reading the first row we know that 2007 losses over 3 million during the first development year (ie. 31/12/2007) were 10 while at the last one (ie. 30/06/2021) decreased to 9 with some ups and downs along the years in the middle. This gives a way to track 2007 losses development over time.
However, in order to properly keep track of the claims historical development we need to develop the values with the LDFs, computed from the loss count triangle as mentioned in Chapter 2.

Development year	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Age to Age LDFs	-	1,215	0,993	0,961	0,991	0,959	1,025	0,946	0,983	1,000	1,029	1,000	1,000	1,125	1,000
Age to Ultimate LDFs	1,216	1,001	1,008	1,049	1,058	1,103	1,076	1,138	1,158	1,158	1,125	1,125	1,125	1,000	1,000

In order to develop the values we consider the main diagonal of the above aggregate loss count triangle and we multiply it by the correspondent age to ultimate LDF. Note that the LDF for 2007 is the one that corresponds to the 15th development year, for 2008 the one for the 14th development year and so on thus when we develop values we should read the age to ultimate LDFs in reverse order.
Note that we will disregard 2021 from this computation since this year is incomplete (our last evaluation is at 30/06/2021). In general we could decide to make some assumptions on the last 6 months of 2021 in order to include 2021 too, for example by making assumptions on the age to ultimate LDF for the first development year, but it is also possible to just disregard it in order to maintain more consistency with the client's data submission.
Once the developed values are found we compute the burning cost for each year. As previously explained in the Experience rating section, the burning costs are obtained by computing the ratio between the developed count and the exposure amount for each year.

Year	2007	2008	2009	2010	2011	2012	2013	2014	2015	2016	2017	2018	2019	2020
Developed values	9	5	11	12	16	12	10	11	14	15	10	10	18	10
Burning costs	3,51E-06	1,82E-06	3,86E-06	3,72E-06	4,44E-06	3,09E-06	2,69E-06	2,80E-06	3,28E-06	3,42E-06	1,99E-06	2,08E-06	3,48E-06	1,86E-06

Once we have the burning cost for each year from 2007 to 2020 we compute the expected burning cost for 2022 that in our case is given by the average of all the burning costs for each year. By multiplying this value for the expected exposure amount for 2022 provided by the cedant we finally obtain the number of claims that we expect to be greater than the threshold at the ultimate position for year 2022.
The expected number of losses for 2022 is 17.24 thus we roughly expect to have 17 losses above 3 million occurring in the upcoming year. We will use this value as the parameter needed to simulate the Poisson distribution.

4.3 Aggregate loss model

Now that we have simulated both the frequency and severity distributions we can implement the aggregate loss model. Our goal is to implement both Monte Carlo simulation and Panjer recursion with aggregate terms (AAD, AAL). We will then compare the results both in terms of precision and time elapsed.

4.3.1 Monte Carlo simulation

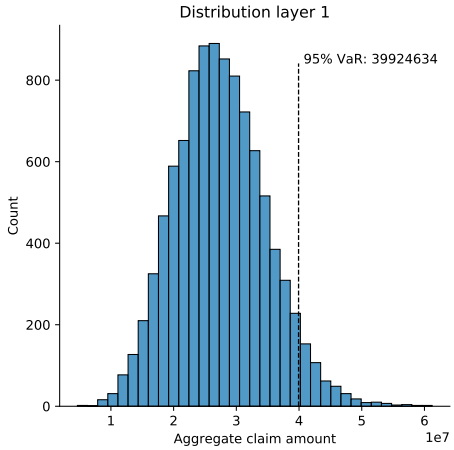
We will perform 10000 simulations and, as previously mentioned, we will consider 4 layers: 2m xs 3m, 5m xs 5m, 15m xs 10m and 25m xs 25m.
Here we present the algorithm implemented on Python for a generic layer k:

Algorithm 1 Monte Carlo simulation

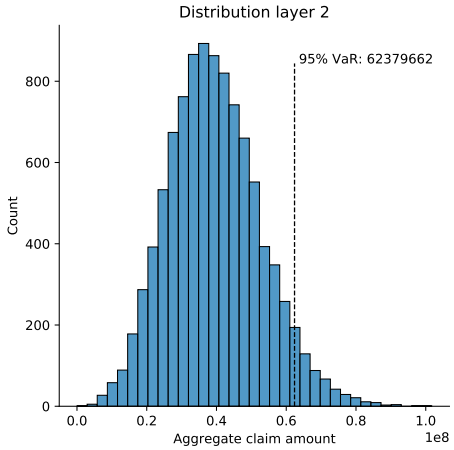
numreps $\leftarrow$ 10000
$Sk \leftarrow$ empty list ▷ Overall distribution of the expected loss to layer k
for $j \leftarrow 1$ to numreps **do**
    Simulate $N \sim Poi(\lambda)$ ▷ $\lambda \leftarrow$ Expected number of losses for 2022
    $Sk_j \leftarrow 0$ ▷ Sk_j will be equal to $\sum_{i=0}^{N} X_i$
    for $i \leftarrow 1$ to N **do**
        Simulate $X_i \sim \Gamma(shape, loc, scale)$ ▷ MLE for parameters estimation
        expLossLayer$_{ik}$ $\leftarrow$ MIN(limit$_k$, MAX(0, X_i - retention$_k$))
        $Sk_j \leftarrow Sk_j$ + expLossLayer$_{ik}$
    end for
    $Sk[j] \leftarrow Sk_j$
end for
return Sk

Note that to summarize efficiently the pseudocode we considered the algorithm for a generic layer k but in the actual Python code we did this work explicitly for each layer. For instance we initialized four empty lists named respectively S1, S2, S3 and S4 and instead of limit$_k$ and retention$_k$ we directly inserted the limit and retention of each layer (we will do the same for the Panjer recursion).
The time elapsed to run the above algorithm is 16.66 seconds. This value alone does not give much information about the model but once it will be compared with the time elapsed for the Panjer recursion it will be an interesting term of comparison for evaluation purposes.
We are now interested in visualizing for each layer the distribution Sk obtained from the MC simulation, ie. the distribution of the expected loss to the layer k. In Figure 4.7 we reported the four distributions. We will highlight in the graphs a specific value for each distribution: the Value-at-Risk (VaR).
The VaR of a random variable X at level α is the 100α percentile of the random variable. For a continuous random variable it is x such that $\mathbb{P}(X \leq x) = \alpha$. For

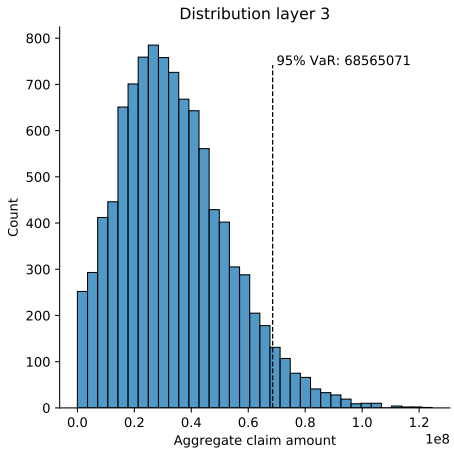
aggregate reinsurance losses, where the risk is that X is high, α is usually picked high, with values usually from 0.95 on. We will pick α euqal to 0.95 in order to stay in line with the general picks.



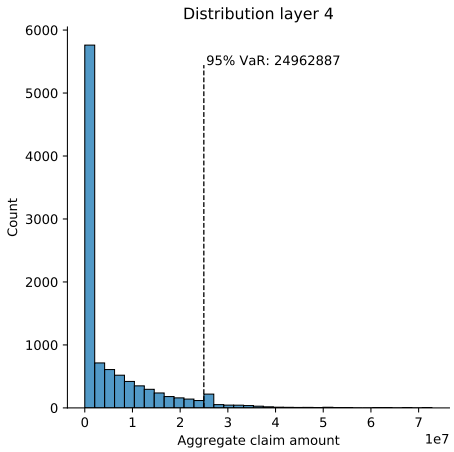
((a)) Layer 1: VaR 39.6 million



((b)) Layer 2: VaR 62.3 million



((c)) Layer 3: VaR 68.5 million

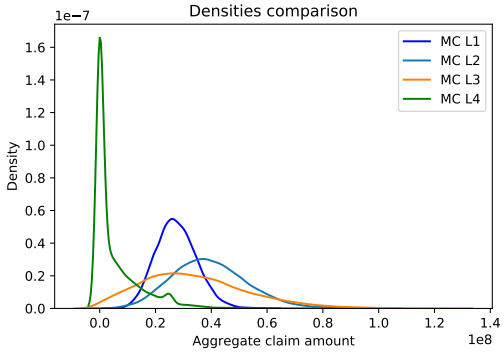


((d)) Layer 4: VaR 25 million

Figure 4.7: Monte Carlo simulation: distributions and VaR of all four layers.

The Value-at-Risk is a statistic that allows us to focus on the right tail of the distribution we are looking at. For example, by looking at the graph on the top left we know that for the first layer we have a 5% probability to have an expected loss to the layer over 39.6 million. However, note that the VaR is not a risk measure. It is a quantile mostly used in risk management and capital loading since it allows to quantify the extent of possible financial losses within a firm, portfolio or position over a specific time frame.

Let's now focus a bit more on the layers distributions. By looking at the four graphs reported above it is possible to note that the higher the layer retention is, the more the frequency drops. Indeed, in Figures 4.7(c) and 4.7(d) we can see that the mass in zero is quite higher with respect to the first two layers. This is due to the fact that while the retention increases fewer and fewer losses touch the layer thus the probability to have zero losses to the layer is higher.
By having a look at the densities and respective mean and VaR for each layer we obtain the following:



	Mean	VaR
L1	27.365.544	39.660.722
L2	39.202.731	61.844.601
L3	33.513.424	67.428.648
L4	5.291.199	25.000.000

Figure 4.8: Densities comparison with reported mean and VaR for each layer.

The table above confirms what we already stated: while the retention increases few and few losses fall into the layer thus the frequency drops while the limit conditions the loss to the layer. For instance, the fourth layer has mean distribution at 5.29 million since the retention is very high thus a lot of losses do not touch the layer and the very few ones that fall in just slightly touch the layer (note that our biggest loss is at 31.9 million). This means that we have low frequency and low severity, as expected, and this is the reason why the VaR is at the limit (25 million). Indeed, in order to reach the limit we would need either a high frequency or a high severity but both scenarios are quite unlikely because of the high retention of the layer.

4.3.2 Panjer recursion

In the above section we simulated for each layer the expected loss to the layer and we analyzed the four distributions obtained. In this section we will try to reach the same results by implementing the Panjer recursion in order to have a clear comparison of the methods used to simulate the aggregate loss model.
We will perform once again 10000 simulations and we will consider the same structure used for the Monte Carlo simulation. We kept the same dimension (ie. number of steps) for all layers and we used 100 as a value. Note also that every time a Gamma distribution will be mentioned in the pseudocode, it will be considered with the MLE parameters above estimated as it was done for the Monte Carlo simulation.

Before presenting the Panjer recursion we would like to describe the discretization process performed on the severity distribution since it is the central hypothesis needed in order to apply the above mentioned recursion. The discretization algorithm implemented on Python for a generic layer k is reported below (Algorithm 2).

Algorithm 2 Discretization process for Panjer recursion

```
dimension ← 100
eps_k ← limit_k/dimension                                   ▷ Step
lowerprob_k ← Γ_CDF(retention_k)                            ▷ Probability below layer k
upperprob_k ← Γ_CDF(retention_k+limit_k)                    ▷ Probability above layer k
x_k ← empty list                                            ▷ Step vector
x_k[1] ← retention_k                                        ▷ Starts at retention
for i ← 2 to dimension do
    x_k[i] ← retention_k+(i+1)*eps_k
end for
CDF_k toscale ← Γ_CDF(x_k)                                  ▷ CDF to scale
CDF_k below ← (CDF_k toscale-lowerprob_k)/(1-lowerprob_k)
CDF_k below.append(1)                                       ▷ Add final element
CDF_k above ← CDF_k below[1:]
CDF_k above.append(1)                                       ▷ Add final element
density_k below ← empty list                                ▷ Density below
for i ← 1 to len(CDF_k below) do
    if i ← 0 then
        d_i ← CDF_k below[i]
    else
        d_i ← CDF_k below[i] - CDF_k below[i-1]
    end if
    density_k below[i] ← d_i
end for
density_k above ← empty list                                ▷ Density above
for i ← 1 to len(CDF_k above) do
    if i ← 0 then
        d_i ← CDF_k above[i]
    else
        d_i ← CDF_k above[i] - CDF_k above[i-1]
    end if
    density_k above[i] ← d_i
end for
```

Once the severity distribution is discretized we can proceed by presenting the pseudocode describing the implementation in Python of the Panjer recursion for a generic layer k (Algorithm 3).
The starting points for the below and above recursions come from some observations

Algorithm 3 Panjer recursion

rangepanjer ← 10000
h0below$_k$ ← $e^{-(1-lowerprob_k)*\lambda}$ ▷ Starting point for below recursion
h0above$_k$ ← $e^{-(1-lowerprob_k)*\lambda*(CDF_k above[1]-1)}$ ▷ Starting point for above recursion
a$_k$ ← 0
b$_k$ ← (1-lowerprob$_k$)$*\lambda$
Sb$_k$ ← empty list ▷ Aggregate density below
Sb$_k$[1] ← h0below$_k$
for n ← 2 to rangepanjer **do**
    h ← 0
    lim ← min(n, len(x$_k$))
    for j ← 2 to lim **do**
        h ← h+ density$_k$below[j]*Sb$_k$[n-j]*(a$_k$ + b$_k$*j/n) ▷ Panjer recursion
    end for
    Sb$_k$[n] ← h
end for
Sa$_k$ ← empty list ▷ Aggregate density above
Sa$_k$[1] ← h0above$_k$
for n ← 2 to rangepanjer **do**
    h ← 0
    lim ← min(n, len(x$_k$))
    for j ← 2 to lim **do**
        h ← h+ density$_k$above[j]*Sa$_k$[n-j]*(a$_k$ + b$_k$*j/n) ▷ Panjer recursion
    end for
    Sa$_k$[n] ← h
end for
CDFpb$_k$ ← empty list ▷ Aggregate CDF below
CDFpb$_k$[1] ← Sb$_k$[1]
for i ← 2 to len(Sb$_k$) **do**
    CDFpb$_k$[i] ← Sb$_k$[i]+CDFpb$_k$[$i-1$]
end for
CDFpa$_k$ ← empty list ▷ Aggregate CDF above
CDFpa$_k$[1] ← Sa$_k$[1]
for i ← 2 to len(Sa$_k$) **do**
    CDFpa$_k$[i] ← Sa$_k$[i]+CDFpa$_k$[$i-1$]
end for

done on the expected number of losses for each layer k. By remembering that if N is a Poisson distributed random variable its probability generating function P_N is given by

$$P_N(x) = e^{\lambda(x-1)}$$

and by knowing from section 3.3.2 that $\mathbb{P}(S=0) = P_N(\mathbb{P}(X=0))$ we have that

$$\mathbb{P}(S=0) = e^{\lambda(\mathbb{P}(X=0)-1)}$$

For how we constructed the below recursion we have that $\mathbb{P}(X=0) = 0$ (because it is the first value of the below cumulative distribution function built in algorithm 2) thus the starting point is given by

$$\mathbb{P}(S=0) = \mathbb{P}(N=0) = e^{-\lambda}$$

For the above recursion instead we have that $\mathbb{P}(X=0)$ is the first value of the above cumulative distribution function built in algorithm 2 thus the starting point is

$$\mathbb{P}(S=0) = e^{\lambda(\text{CDF}_k\text{above}[1]-1)}$$

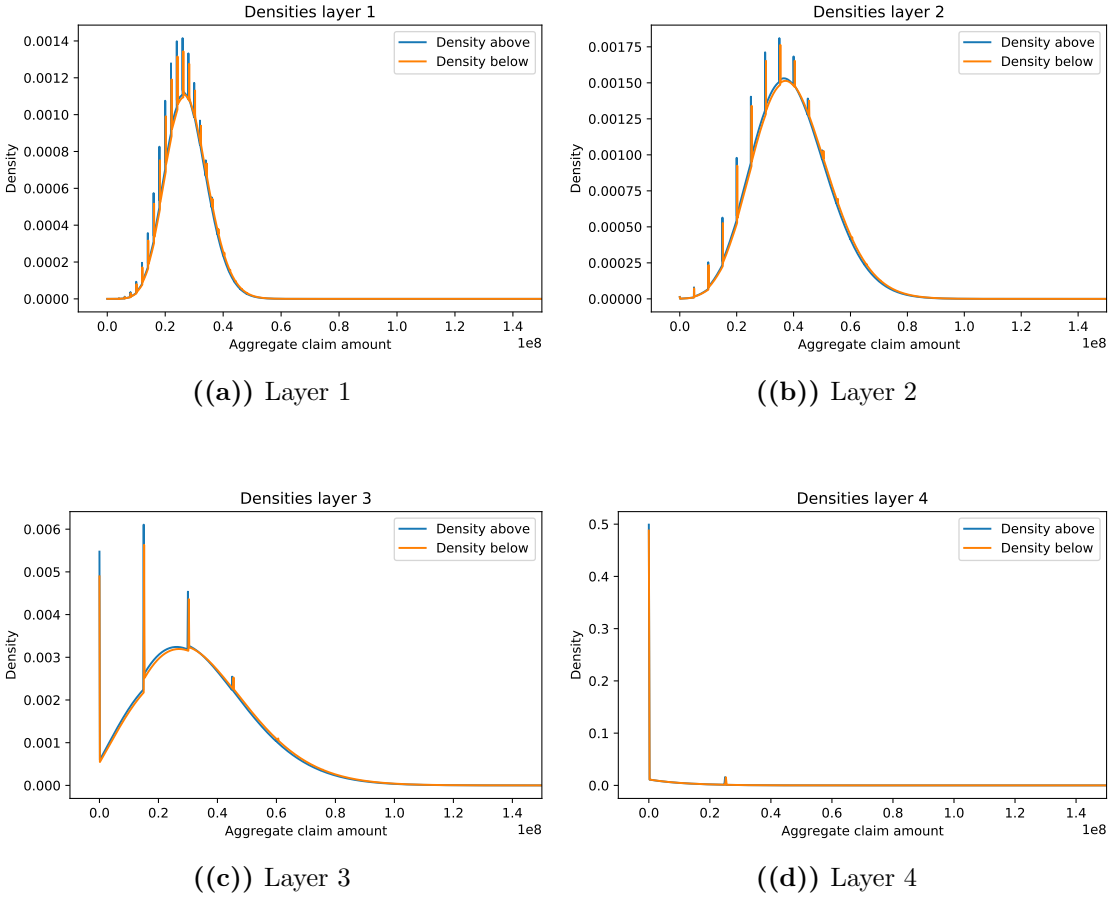
However, since we are dealing with four layers we cannot use the overall expected number of losses estimated in the frequency analysis. Indeed, λ needs to be adapted for each layer thus we will have a λ_k for each layer k. These four values are obtained in the following way:

$$\begin{aligned}
\lambda_k &= \mathbb{E}\left[\sum_{i=1}^{N} \mathbb{1}_{\{X_i > \text{retention}_k\}}\right] = \mathbb{E}_N\left[\mathbb{E}\left[\sum_{i=1}^{N} \mathbb{1}_{\{X_i > \text{retention}_k\}} | N\right]\right] \\
&= \mathbb{E}_N\left[\sum_{i=1}^{N} \mathbb{E}[\mathbb{1}_{\{X_i > \text{retention}_k\}}]\right] = \mathbb{E}_N\left[\sum_{i=1}^{N} \mathbb{P}(X_i > \text{retention}_k)\right] \\
&= \mathbb{E}_N\left[\mathbb{P}(X_i > \text{retention}_k)\sum_{i=1}^{N} 1\right] = \mathbb{P}(X_i > \text{retention}_k)\mathbb{E}[N] \\
&= (1 - F_{X_i}(\text{retention}_k))\lambda
\end{aligned}$$

Indeed, we start by computing the expected value of those losses that fall into the layer. The first row equality holds by applying the law of total expectation. It states that if X and Y are two random variables and $\mathbb{E}[X]$ is defined then:

$$\mathbb{E}[X] = \mathbb{E}_Y[\mathbb{E}[X|Y]]$$

In our case Y is N thus N is now given and we can swap expected value and summation. At this point we know that the expected value of the indicator function is nothing but the probability that X_i is bigger than the layer's retention. However, the X_i, for $i = 1, \ldots, N$ are independent and identically distributed thus this probability is a constant value and we can move it outside of the summation.
From the above described Panjer recursion algorithm we obtained the graphs reported in Figure 4.9.
Note that these plots do not represent neither continuous or discrete densities but an intermediate scenario. Indeed, we have a continuous density a part from the visible bumps in the graphs which are discrete mass points. These bumps are in



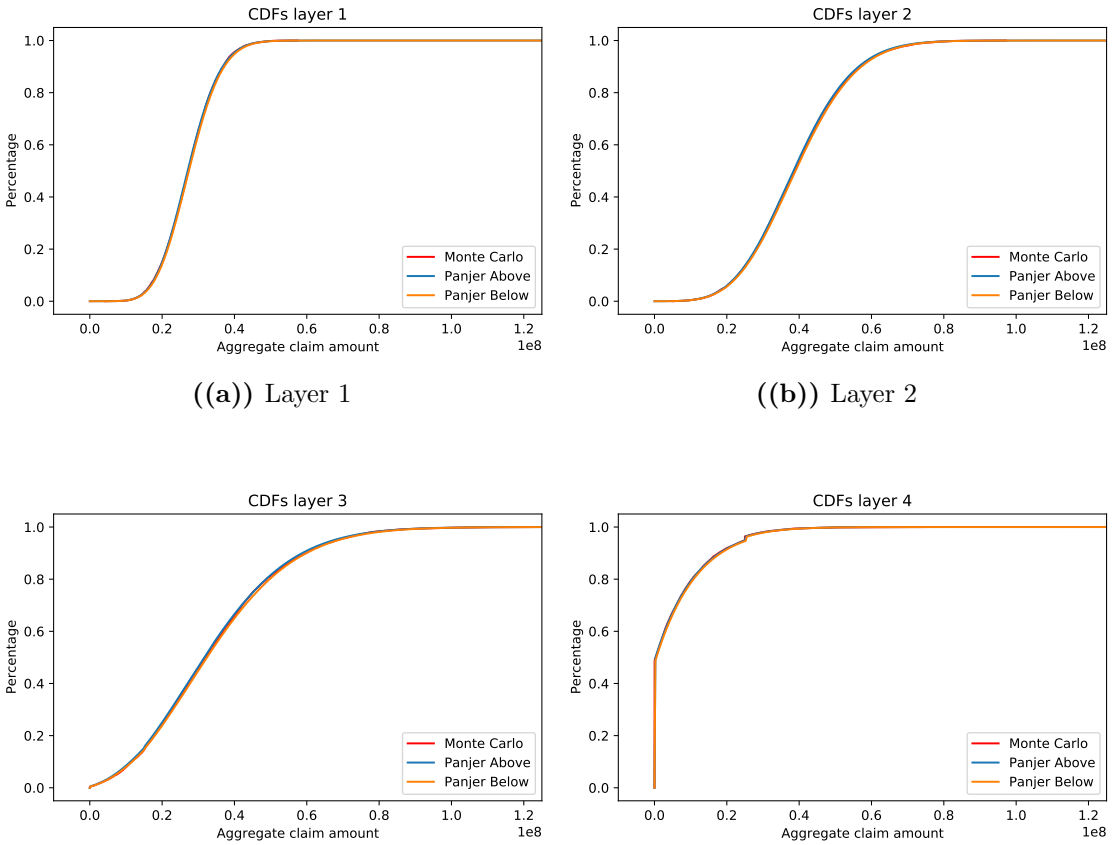
((a)) Layer 1 **((b))** Layer 2

((c)) Layer 3 **((d))** Layer 4

Figure 4.9: Panjer recursion: above and below densities of all four layers.

correspondence to the multiples of the limit characterizing the distributions and this happens because we are layering the aggregate loss distribution. Indeed, if we consider the first layer we have a 2 million limit with 3 million retention. This means that all the losses that are bigger or equal to 5 million will be a 2 million loss for the layer thus the limit and all its multiples will have a much higher mass. Note that 2 million itself does not have a high density because we are dealing with the aggregate distribution thus with our order of magnitude it is very rare to have an aggregate loss to the layer that is that small.
Another interesting thing we can observe by looking at layer 3 and layer 4 densities (Figures 4.9(c) and 4.9(d)) is that the mass in 0 is quite high, in particular in layer 4. This is due to the fact that the higher the retention is the higher is the probability to have no losses to the layer.
The time elapsed to run the above algorithm is 14.69 seconds. This is not significantly smaller that the time needed to perform the Monte Carlo simulation but it is worth mentioning that while with MC we obtain for each layer just one simulated

distribution, with the Panjer recursion we obtain two distributions that delimit the actual one. Indeed, this last method gives us a clear idea of where the true distribution should be. Furthermore, the smaller it is the gap between the below and above distribution the more precision we obtain. However, there is a trade off between precision and elapsed time. To require more precision means to increase the dimension (ie. to reduce epsilon) thus it also means to increase the time needed to run the algorithm (see next paragraph).
Finally, we present here the CDFs obtained from both the Monte Carlo simulation and the Panjer recursion in order to compare the results obtained from the two methods.



((a)) Layer 1

((b)) Layer 2

((c)) Layer 3

((d)) Layer 4

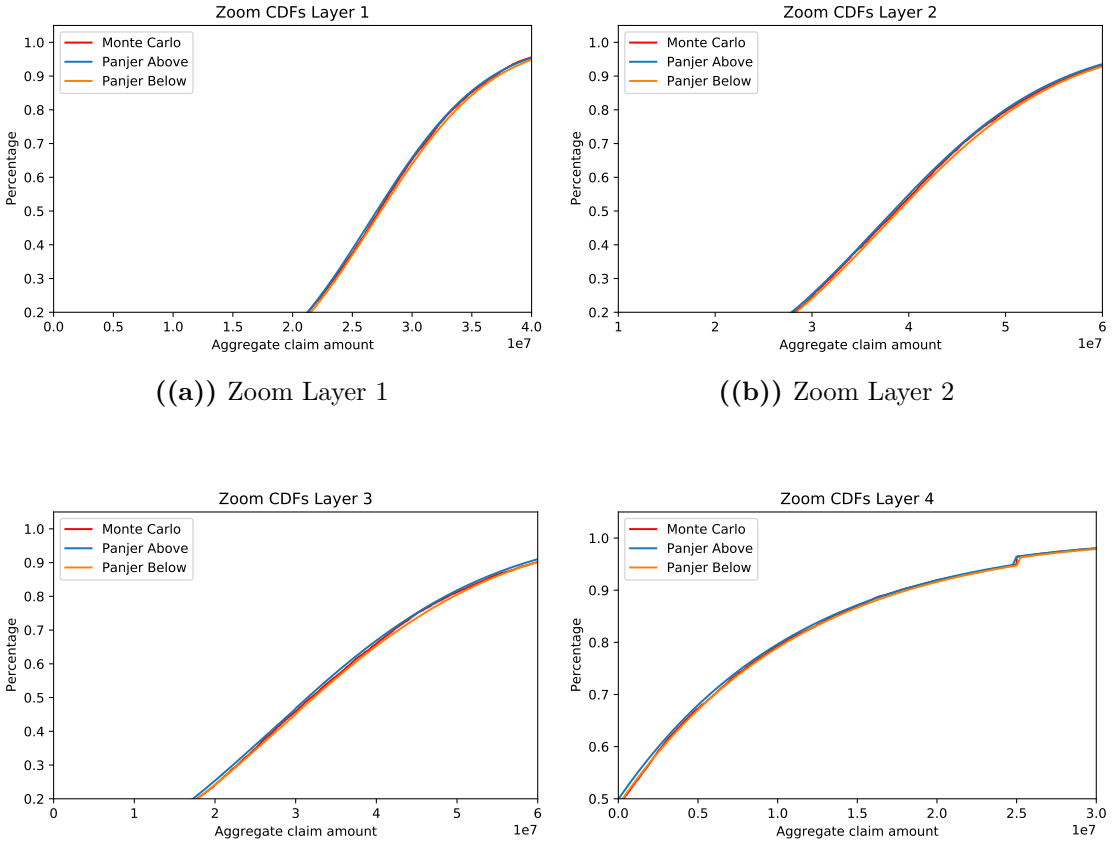
Figure 4.10: Monte Carlo simulation and Panjer recursion: CDFs comparison.

By looking at the CDFs comparison (Figure 4.10) we can see that the Monte Carlo simulation and the Panjer recursion hold quite the same results. Note that the jump in Figure 4.10(d) at 25 million is nothing but the result of the discrete mass points we observed in the Panjer densities reported in Figure 4.9. Indeed, even if these jumps are not always clearly visible they are present in all four CDFs and they are

due to the fact that in correspondence to the discrete mass points there is a big mass gathering thus they provoke a relevant increase in the cumulative distribution. Overall, our goal is to have the distribution obtained from the MC simulation falling within the Panjer interval for all layers.

4.3.3 Parameters effect on results and methods comparison

By zooming on the graphs reported in Figure 4.10 and by having a look at the first values of each distribution we can observe that with 10000 simulations the MC simulation already is in between the below and above Panjer recursions (see Figure 4.11).



((a)) Zoom Layer 1

((b)) Zoom Layer 2

((c)) Zoom Layer 3

((d)) Zoom Layer 4

Figure 4.11: Monte Carlo simulation and Panjer recursion: zoom of CDFs comparison.

Of course, by increasing the number of simulations of the MC distribution it will be more precise and it will surely fall within the interval. However, there is a trade off between precision and time elapsed. In Figure 4.12 we reported the summary of

elapsed time in relation to the number of simulations.

Monte Carlo			
Number of simulations	10.000	15.000	20.000
Elapsed time	16,66 s	25,36 s	35,21 s

Figure 4.12: Output table reporting the trade off between number of simulations and elapsed time for Monte Carlo simulation.

If we perform 15000 simulations we need 25.36 seconds to run the algorithm, almost 10 seconds more than for 10000 simulations. By increasing the number of simulations to 20000 the algorithm would require 35.21 seconds to run.
The same reasoning holds for the Panjer recursion. By increasing the dimension (ie. reducing epsilon thus having more steps) we will obtain a tighter interval but we need more time to run the algorithm (see Figure 4.13).

Panjer			
Dimension	100	200	500
Elapsed time	14,69 s	27,97 s	68,61 s

Figure 4.13: Output table reporting the trade off between dimension and elapsed time for Panjer recursion.

In order to have a significant comparison within the two methods we ran the Monte Carlo method with 100000 simulations and we assumed this distribution to be the actual distribution of data. Note that we tried different number of simulations to determine which distribution could be the best one to represent the actual distribution of data and we chose 100000 since by increasing the number thereafter we did not obtain significant differences in the obtained distributions. Afterwards, we compared the CDFs obtained from the Monte Carlo method with 10000 simulations and Panjer recursion with dimension 100 with the CDFs of the actual distribution (Figure 4.14 left). We chose 10000 and 100 respectively since with these two parameters the methods are comparable in terms of elapsed time thus we can discuss their efficiency by keeping fixed the computational effort needed to run the algorithms. Finally, for completeness we did the same comparison but doubling both the number of simulations for MC and the dimension for Panjer (Figure 4.14 right).

	Elapsed time	MSE
Monte Carlo	16,66 s	1,020E-06
Panjer	14,69 s	7,029E-07

	Elapsed time	MSE
Monte Carlo	35,21 s	1,014E-06
Panjer	27,97 s	6,320E-07

Figure 4.14: Methods comparison. *Left*: MC with 10000 simulations and Panjer with dimension 100. *Right*: MC with 20000 simulations and Panjer with dimension 200.

Note that in order to determine which model performs better we computed the Mean Squared Error (MSE), a risk function that measures the average of the errors squares, ie. the average squared difference between the estimated values and the actual ones. Since we have a structure with layering we obtained an MSE for each layer. However, we are interested into an overall comparison of the two methods thus we computed the average of the four layers MSEs. Another thing that might be worth mentioning is that for the Panjer recursion we have for each layer the below and above CDFs. However, in order to have a clear comparison with the Monte Carlo simulation we considered for each layer the average CDF where the i-th element is nothing but the average between the i-th elements of the below and above CDFs respectively.
By looking at the left table of Figure 4.14 we can see that with the same computational effort needed to run the algorithms the Panjer recursion performs better than the Monte Carlo simulation. Indeed, the MSE is lower thus the precision is higher. Furthermore, by doubling both the number of simulations and the dimension (see right table of Figure 4.14) the Panjer recursion is still better with respect to the Monte Carlo simulation and it requires less time to run. In conclusion, it might seem that the Panjer recursion is in general more accurate than the Monte Carlo simulation. However, it is still up to the actuary to decide how to proceed. We may want to use just one of the two methods with higher precision or we may decide to use both methods with lower accuracy in order to have more information. We could also choose to have faster results over more precision. The decision might depend from the situation we find ourselves in, the treaty we are dealing with or even the data submission. However, we can already conclude that the Panjer recursion can provide the same results of the Monte Carlo simulation with different methodology, same efficiency and a little more code.
In the following section we will introduce the treaty features and analyze the effects that they might have on the distributions. Note that the aggregate terms do not have an impact on the methods efficiency thus they will not affect the observations made above.

4.4 Treaty features: AAD and AAL

Now that both the Monte Carlo simulation and the Panjer recursion have been implemented we want to analyze the effects of the annual aggregate deductible (AAD) and the aggregate annual limit (AAL) over the aggregate loss distribution. Let's consider a scenario where we have an aggregate loss to the layer (we will name it X) and we are provided with both the AAL and AAD. Then we would have

$$\text{MIN(MAX(0, X - AAD), AAL)}$$

Indeed, the maximum value we could ever obtain is the the aggregate limit.

4.4.1 Implement AAD and AAL

We now present the pseudocode inserted in Algorithm 1 and Algorithm 3 respectively in order to apply the AAD followed by the AAL. Note that we will not report the full pseudocodes of MC and Panjer again but we will refer to the algorithms' line numbers in order to underline where we implemented the treaty features. We will apply these features on the MC simulation performed with 10000 simulations and the Panjer recursion with dimension 100.

Algorithm 4 Monte Carlo simulation with AAD and AAL

```
10: ...
11:  AADk ← valueAADk                                   ▷ AAD definition
12:  AALk ← valueAALk                                   ▷ AAL definition
13:  Sk_j ← min(max(0, Sk_j - AADk), AALk)              ▷ Apply AAD followed by AAL
14:  Sk[j] ← Sk_j
15: end for
16: return Sk
```

Note that instead of specifying for each layer k a value for the deductible we used "valueAADk" since we tried different numbers to really see the effect of these features and to analyze more than one scenario. The same holds for "valueAALk" for each layer k.
In the Panjer recursion in order to have a clear comparison with MC we considered the same AAD and AAL of Monte Carlo simulation with the only difference that we worked with the indexes instead (see Algorithm 5). This is due to the fact that in this case we are dealing with densities thus to obtain a correct representation we need to adapt the steps to both the AAD and AAL respectively.

Algorithm 5 Panjer recursion with AAD and AAL

```
25: ...
26: AADnk ← AADk/eps_k                                  ▷ AAD definition
27: first_k a ← 0                                       ▷ Mass in zero for above density
28: first_k b ← 0                                       ▷ Mass in zero for below density
29: for i ← 1 to AADnk do
30:     first_k a ← first_k a + Sa_k[i]
31:     first_k b ← first_k b + Sb_k[i]
32: end for
33: AADSa_k ← Sa_k[AADnk:]
34: AADSa_k[0] ← first_k a                              ▷ Overwrite first element density above
35: AADSb_k ← Sb_k[AADnk:]
36: AADSb_k[0] ← first_k b                              ▷ Overwrite first element density below
37: listk ← [0, 1*eps_k, 2*eps_k, ..., rangepanjer*eps_k]
```

38: AALnk ← (listk.index(AALk)) ▷ AAL definition
39: $last_k$a ← 0 ▷ Mass in last position for above density
40: $last_k$b ← 0 ▷ Mass in last position for below density
41: **for** i ← AALnk to (rangepanjer - AADnk) **do**
42: $last_k$a ← $last_k$a + $AADSa_k$[i]
43: $last_k$b ← $last_k$b + $AADSb_k$[i]
44: **end for**
45: $newSa_k$ ← $AADSa_k$[:AALnk]
46: $newSa_k$[AALnk] ← $last_k$a ▷ Overwrite last element density above
47: $newSb_k$ ← $AADSb_k$[:AALnk]
48: $newSb_k$[AALnk] ← $last_k$b ▷ Overwrite last element density below
49: $CDFpb_k$ ← empty list ▷ Aggregate CDF below
50: $CDFpb_k$[1] ← $newSb_k$[1]
51: **for** i ← 2 to len($newSb_k$) **do**
52: $CDFpb_k[i]$ ← $newSb_k[i]$+$CDFpb_k[i-1]$
53: **end for**
54: $CDFpa_k$ ← empty list ▷ Aggregate CDF above
55: $CDFpa_k$[1] ← $newSa_k$[1]
56: **for** i ← 2 to len($newSa_k$) **do**
57: $CDFpa_k[i]$ ← $newSa_k[i]$+$CDFpa_k[i-1]$
58: **end for**

It takes 17.94 seconds to run the Monte Carlo simulation while for the Panjer recursion we need 14.87 seconds. As we can see, for both methods the introduction of treaty features does not increase significantly the computational time needed to run the algorithms. However, these features have a big impact on the distributions and they can really change the treaty structure.

4.4.2 Results obtained

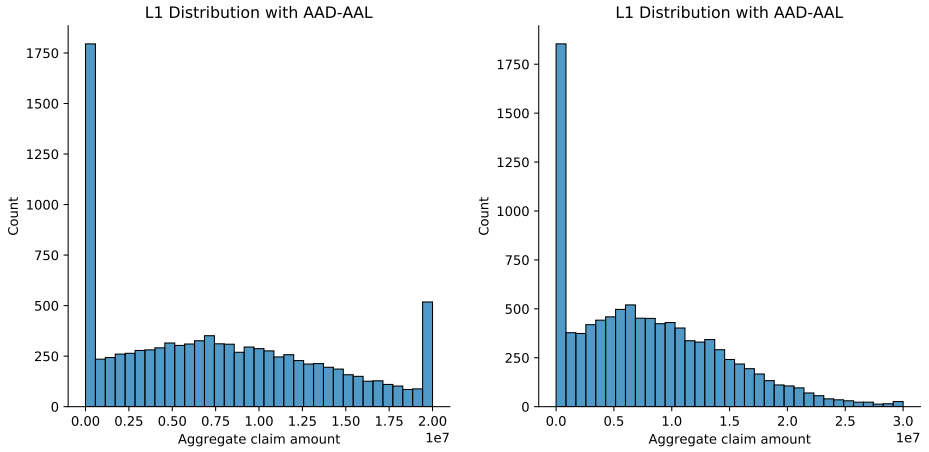
As previously mentioned, we will now apply on each layer the AAD followed by the AAL and we will try different values for both the deductibles and limits in order to analyze the effects of these treaty features on the layering. We will report just the Monte Carlo distributions plots since the features' effect on the Panjer recursion densities is the same. Finally, we will compare the two methods' CDFs and observe the similarity of the results.
In Figure 4.15 we reported the two scenarios we will consider for our analysis. We tried different values for both the AAD and AAL in order to prove how much the treaty features can influence the distributions. We kept invariant the AAD for the first and second layer and the AAL for the third and fourth one to better see the effects of each single feature on the distributions.
In order to have a first look at the effect of the treaty features on the aggregate loss

	AAD	AAL
L1	20.000.000	20.000.000
L2	15.000.000	55.000.000
L3	5.000.000	60.000.000
L4	10.000.000	25.000.000

	AAD	AAL
L1	20.000.000	30.000.000
L2	15.000.000	40.000.000
L3	10.000.000	60.000.000
L4	2.000.000	25.000.000

Figure 4.15: Left: first scenario, Right: second scenario.

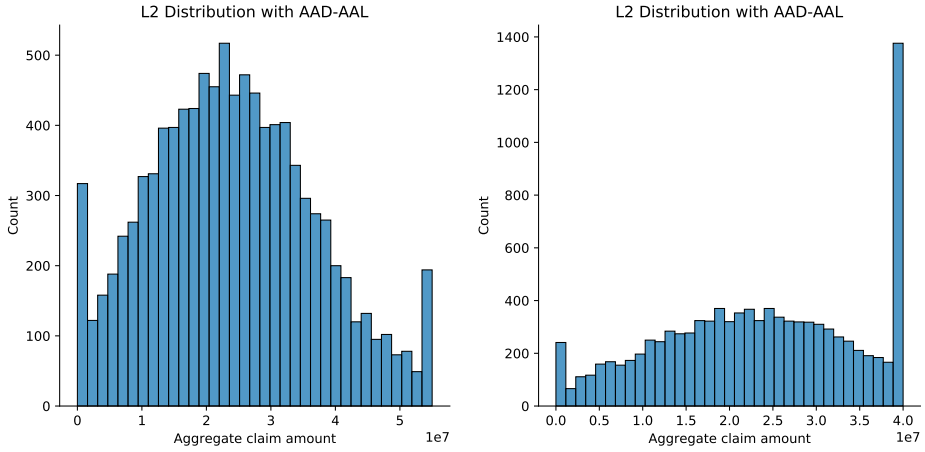
distribution we start by analyzing the first layer whose distributions are reported in Figure 4.16. In both scenarios we have an AAD of 20 million which means that all the aggregate losses are reduced by 20 million with the condition that the ones up to 20 million get capped at zero. Graphically speaking, this results into a general shift to the left of the distribution with a big mass in zero corresponding to all those losses lower than or equal to the AAD. However, the effect of this feature will be more evident for the third and fourth layers.
The AAL instead is a limit thus every aggregate loss above the aggregate limit will be capped at it. For the first scenario we have 20 million aggregate limit and, indeed, it is possible to observe in the graph that we have quite a lot of mass at 20 million with respect to the original simulated distribution and nothing after. However, in the second layer the limit increased by 10 million.



((a)) 1^{st}: AAD 20m, AAL 20m **((b))** 2^{nd}: AAD 20m, AAL 30m

Figure 4.16: First layer distributions comparison.

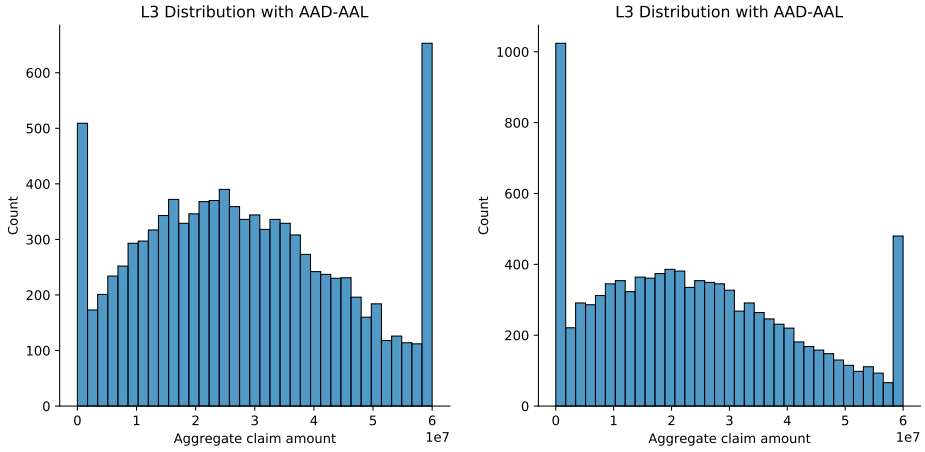
For the second layer the limit reduces and from Figure 4.17 it is clear that the effect is the opposite of what we could observe for the first layer. Indeed, the mass at the aggregate limit for the second scenario increased quite a lot since the aggregate limit is 15 million less than in the first one.
By having a look at the third layer, we can see that even if the AAL is at 60 million for both scenarios we still have a big change in the distribution (see Figure 4.18). This is due to the fact that, as previously mentioned, the AAD provokes a shift of the



((a)) 1^{st}: AAD 15m, AAL 55m **((b))** 2^{nd}: AAD 15m, AAL 40m

Figure 4.17: Second layer distributions comparison.

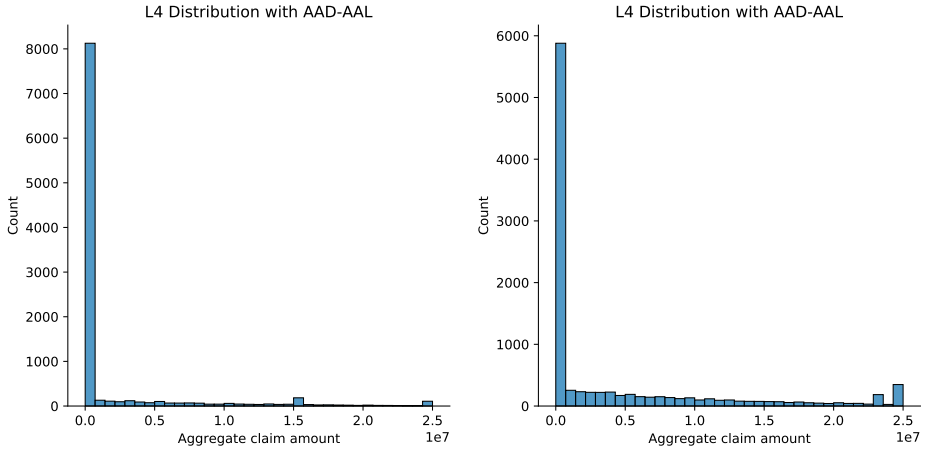
distribution to the left equal to the amount of the AAD thus we have a big mass in zero. Since in the second scenario the AAD increases by 5 million, as a consequence of the above mentioned shift the mass gathered in zero increases a lot.



((a)) 1^{st}: AAD 5m, AAL 60m **((b))** 2^{nd}: AAD 10m, AAL 60m

Figure 4.18: Third layer distributions comparison.

It is interesting to focus for a moment on the fourth layer, in particular on the first scenario. Figure 4.19(a) shows a very big mass in zero even if the AAD is not extremely big with respect to our losses scale (10 million). This is due to the fact that this layer has a quite high retention (25 million) thus just a few losses fall in and each of the ones that touch the retention is a quite small loss to the layer.



((a)) 1^{st}: AAD 10m, AAL 25m **((b))** 2^{nd}: AAD 2m, AAL 25m

Figure 4.19: Fourth layer distributions comparison.

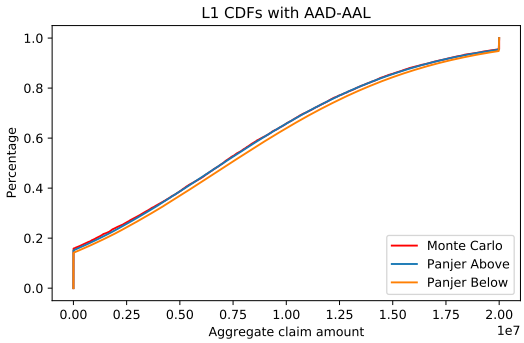
Furthermore, there is a very big mass in zero due to all those losses that do not touch the layer and most of the remaining mass falls quite close to zero since the aggregate loss to the layer will most likely tend to be small.
In the second scenario we reduced the aggregate deductible by 8 million without changing the AAL. This change afftects the distribution by pushing it to the right and, indeed, we have less mass gathered in zero.
To have a better overview of the treaty features' effects it can be interesting to look at the expectations of each distribution:

	L1	L2	L3	L4
First scenario	7.696.947	24.109.821	28.047.643	1.892.218
Second scenario	7.882.387	23.379.976	23.952.085	4.301.436

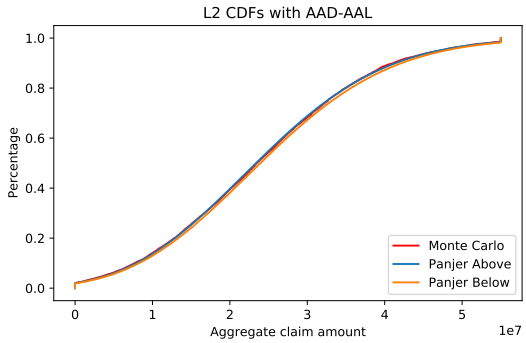
Figure 4.20: Output table reporting for each scenario the expectations of the layers' distribution.

For the first and second layers we left the AAD invariant and we changed the AAL. This causes a small change of the distributions' expectations where for the first layer it is slightly higher in the second scenario as a result of increasing the AAL while for the second layer it is smaller as a consequence of decreasing the AAL. For the third and fourth layers we modified the AAD by leaving the AAL unchanged and this provokes quite a strong change when it comes to the layers' expectations. This is a reasonable result since, as previously mentioned, the AAD causes a shift of the distribution thus we expected the mean to be affected by it in a relevant way. In particular, for the third layer it is smaller in the second scenario consequently to an increase of the AAD while for the fourth layer we see a higher expectation as a result of a decrease in the AAD.

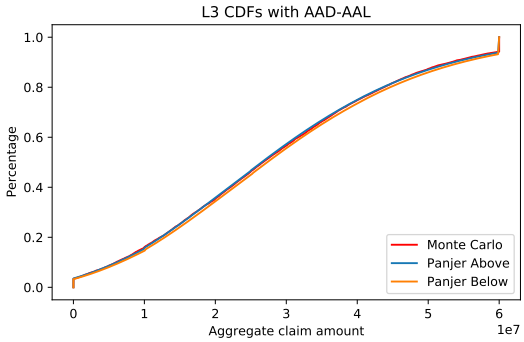
In order to check that the results obtained with the Panjer recursion are in line with the MC ones we report in Figure 4.21 the CDFs for each layer obtained by applying the treaty features corresponding to the first scenario.



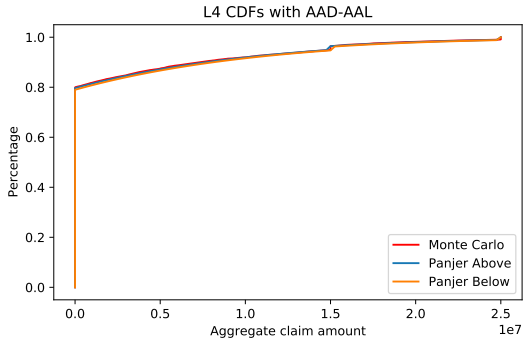
((a)) Layer 1: AAD 20m, AAL 20m



((b)) Layer 2: AAD 15m, AAL 55m



((c)) Layer 3: AAD 5m, AAL 60m



((d)) Layer 4: AAD 10m, AAL 25m

Figure 4.21: Monte Carlo simulation and Panjer recursion with AAD and AAL: CDFs comparison.

By looking at the CDFs the effect that the treaty features have on the distributions is clear. Indeed, with respect to the CDFs reported in Figure 4.10 here we observe a jump at zero provoked by the AAD, a jump to one at the aggregate limit due to the AAL effect and a general shift of the cumulative distribution to the left. The shift is quite evident for the fourth layer while for the first three layers is less apparent. The jumps at zero and at the AAL that characterize the four CDFs come from the big masses we see in all four layers' distributions as an effect of the introduction of the treaty features.
Once the results have been analyzed and an interpretation of the graphs has been given, we would like to understand what these treaty features mean for the reinsurer and the reinsured in a practical way.
As we already know, the AAD is an agreement between the two parties where the

reinsured pays by its own all the losses up to the agreed upon aggregate deductible during the policy year and once all the losses up to that amount are paid by the reinsured, the reinsurer pays the remainder of losses for the annual period. By having a look at the two scenarios analyzed above we can easily understand how much the AAD can influence a treaty. The reason why a company would buy an AAD is to have a lower premium and very often the AAD is bought just for the first layer. However, we introduced an AAD for each layer just to have a better overview on the effect that it might have on the treaty.
The aggregate annual limit instead is the maximum amount of coverage that a reinsurer provides over a year thus once the covered expenses reach the AAL the reinsurer stops paying. Differently from the AAD, the AAL does not have a big impact on the pricing itself but it can have a quite big influence on the VaR or on the aggregate loss distribution.

4.5 Conclusions

The Monte Carlo simulation is the method used by the Zurich Actuarial Team in Sompo International to simulate the aggregate loss distribution on a reinsurance structure. Despite it is a very good and well known method to implement, it is not the only option a reinsurer has. Indeed, the Panjer recursion is an alternative method that allows the actuary to obtain an interval in which the final distribution is expected to fall in. Furthermore, this recursion algorithm can be more precise than Monte Carlo without requiring a significantly higher time to run.
The Panjer recursion has both advantages and disadvantages with respect to the Monte Carlo simulation. An advantage could be that it provides an interval delimited by two distributions, one below and one above, that may allow the actuary to have some more information with respect to a single simulated distribution (for example we might have a better idea on the standard error of the estimate). Indeed, we could have a range where to expect the final distribution to be even if in the end the actuary would just retrieve the average distribution from the below and the above ones. Furthermore, by tightening up the interval we can presumably reach an higher precision than with the Monte Carlo method. Indeed, we showed that if we consider the same computational time to run each method the Panjer recursion as a lower MSE thus it is more accurate.
However, there might be some side effects coming from the Panjer recursion implementation. One may be that to perform this recursion the frequency distribution must satisfy some hypotheses and that a discretization of the severity distribution must be performed. Furthermore, the discretization requires some code in order to be carried out thus the Monte Carlo simulation is for sure straightforward and easier to implement than the Panjer recursion. Note that these observations do not change if the treaty includes some aggregate terms.

This work will enable Sompo International to eventually implement the Panjer recursion in the company pricing tool by knowing that it can be a valid alternative or a parallel method to the Monte Carlo simulation for the models simulation with all the possible advantages or disadvantages that may come with it. It will then be up to the actuary to decide which option can be the best one with respect to the situation he finds himself in. Indeed, for small clients and easy treaties a faster approach might be preferred while if we are presented with a very complicated treaty it can be ideal to be able to perform comparison between methods in order to be more precise and to have more information on the model.

Appendix A

Python code

```
### Libraries
import pandas as pd
import matplotlib.pyplot as plt
import numpy as np
import seaborn as sns
from scipy import stats
import random
import dc_stat_think as dcst
import statistics
import time
from math import e
from sklearn.metrics import mean_squared_error

### Import claims
df = pd.read_excel("claims_data.xlsm")
df.info(verbose = True)

print("Losses above 3m")
over_3m = df["30/06/2021"] >= 3e6
sns.countplot(x = over_3m)
plt.xlabel('Losses at 30/06/21 that are above the threshold')
plt.ylabel('Count')
plt.title('Losses above 3 million')
plt.show()

''' Severity '''

print("Univariate distribution of data")
df_over = df.loc[df['30/06/2021'] >= 3e6]
df_over = df_over.reset_index()
df_over = df_over.drop(columns = 'index')
sns.distplot(df_over["30/06/2021"], hist = True, color = 'green', bins
    = 25, kde = False, hist_kws={'edgecolor':'black'})
```

```
plt.xlabel("Losses at 30/06/21")
plt.ylabel("Count")
plt.title("Univariate distribution of data")
plt.grid(axis = 'y', linestyle = '--')
plt.show()

## KS test
random.seed(1000)
list_of_dists = ['expon','gamma','loggamma','lognorm','norm', 'pareto']
results = []
for i in list_of_dists:
    dist = getattr(stats, i)
    param = dist.fit(df_over['30/06/2021'])
    a = stats.kstest(df_over['30/06/2021'], i, args=param)
    results.append((i,a[0],a[1]))
results.sort(key=lambda x:float(x[2]), reverse=True)
for j in results:
    print("{}: statistic={}, pvalue={}".format(j[0], j[1], j[2]))

## Gamma distribution
random.seed(1002)
param_g = stats.gamma.fit(df_over["30/06/2021"])
num_reps = 10000
sim_gamma = stats.gamma.rvs(param_g[0], param_g[1], param_g[2], size =
    num_reps)
print('2KS test result for Gamma distribution:')
s = len(df_over['30/06/2021'])
print(stats.ks_2samp(np.asarray(df_over['30/06/2021']), sim_gamma[:s]))
x1, y1 = dcst.ecdf(df_over['30/06/2021'])
x2 = np.linspace(3000000, 40000000, 10000)
y2 = stats.gamma.cdf(x2, param_g[0], param_g[1], param_g[2])
plt.plot(x1, y1*100, 'g', label = 'CDF client data')
plt.plot(x2, y2*100, 'b', label = 'CDF theoretical distribution')
plt.xlim(left = 3e6)
plt.xlabel('Claims')
plt.ylabel('Percentage')
plt.title('CDF comparison ')
plt.legend(loc = 'lower right')
plt.show()
fig = plt.figure()
ax = fig.add_subplot(111)
res = stats.probplot(df_over['30/06/2021'], dist = stats.gamma, sparams
    = param_g, plot = ax)
ax.set_title("Gamma with MLE parameters estimation")

## Pareto distribution
random.seed(1003)
param_p = stats.pareto.fit(df_over["30/06/2021"])
num_reps = 10000
sim_pareto = stats.pareto.rvs(param_p[0], param_p[1], param_p[2], size
    = num_reps)
```

```
print('2KS test result for Pareto distribution:')
s = len(df_over['30/06/2021'])
print(stats.ks_2samp(np.asarray(df_over['30/06/2021']), sim_pareto[:s]))
x1, y1 = dcst.ecdf(df_over['30/06/2021'])
x2 = np.linspace(3000000, 40000000, 10000)
y2 = stats.pareto.cdf(x2, param_p[0], param_p[1], param_p[2])
plt.plot(x1, y1*100, 'g', label = 'CDF client data')
plt.plot(x2, y2*100, 'b', label = 'CDF theoretical distribution')
plt.xlim(left = 3e6)
plt.xlabel('Claims')
plt.ylabel('Percentage')
plt.legend(loc = 'lower right')
plt.title('CDF comparison')
plt.show()
fig = plt.figure()
ax = fig.add_subplot(111)
res = stats.probplot(df_over['30/06/2021'], dist = stats.pareto,
    sparams = param_p, plot = ax)
ax.set_title("Pareto with MLE parameters estimation")

''' Frequency '''

n_rows = df_over.shape[0]
df_over.insert(3, 'Loss count', np.ones(n_rows))
df_over.insert(3, 'Year', df_over['Date of Loss'].dt.year)
agg_loss_count = df_over.groupby('Year')['Loss count'].sum()
n = np.array([1989, 1995, 1996, 1997, 1998, 1999, 2000, 2001, 2002,
    2003, 2004, 2005, 2006, 2007, 2008, 2009, 2010, 2011, 2012, 2013,
    2014, 2015, 2016, 2017, 2018, 2019, 2020, 2021])
nxticks = [1989, 1996, 1998, 2000, 2002, 2004, 2006, 2008, 2010, 2012,
    2014, 2016, 2018, 2020]
nyticks = [0, 2, 4, 6, 8, 10, 12, 14, 16, 18]
plt.bar(n, np.asarray(agg_loss_count), width = 0.85, alpha = 0.7)
plt.grid(axis = 'y', linewidth = 0.5, linestyle = '—')
plt.xlabel('Year')
plt.ylabel('Count')
plt.title('Count losses above 3 million')
plt.xticks(nxticks, nxticks, rotation = 45)
plt.yticks(ticks = nyticks)
plt.show()

## Find the offset triangle with loss count for each year
df.insert(3, 'Year', df['Date of Loss'].dt.year)
triangle = df.iloc[:, [3, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22,
    23, 24, 25, 26]]
last_15_years = triangle['Year'] >= 2007
triangle = triangle[last_15_years]
years = np.array([2007, 2008, 2009, 2010, 2011, 2012, 2013, 2014, 2015,
     2016, 2017, 2018, 2019, 2020, 2021])
offset_triangle = pd.DataFrame()
```

```
for year in years:
  dataframe = triangle.loc[triangle['Year'] == year]
  dataframe.iloc[:, 1:] = dataframe.iloc[:, 1:].shift(periods = -(year -2007), axis = 1)
  offset_triangle = offset_triangle.append(dataframe, ignore_index = True)
offset_triangle = offset_triangle.rename(columns = {'31/12/2007': '1', '31/12/2008': '2', '31/12/2009': '3', '31/12/2010': '4', '31/12/2011': '5', '31/12/2012': '6', '31/12/2013': '7', '31/12/2014': '8', '31/12/2015': '9', '31/12/2016': '10', '31/12/2017': '11', '31/12/2018': '12', '31/12/2019': '13', '31/12/2020': '14', '30/06/2021': '15'})
offset_triangle = offset_triangle.fillna(0)
col1 = offset_triangle.loc[:, ['Year']]
cols = offset_triangle.iloc[:, 1:]
cols[cols <= 3e6] = 0
cols[cols > 3e6] = 1
offset_triangle = pd.concat([col1, cols], axis = 1, join = 'inner')
agg_count = pd.DataFrame()
for year in years:
  sum = offset_triangle.loc[offset_triangle['Year'] == year, ['1', '2', '3', '4', '5', '6', '7', '8', '9', '10', '11', '12', '13', '14', '15']].sum()
  agg_count = agg_count.append(sum, ignore_index = True)
agg_count = agg_count[['1', '2', '3', '4', '5', '6', '7', '8', '9', '10', '11', '12', '13', '14', '15']]
agg_count.insert(0, 'Year', years)
agg_count

## LDFs

# Age to Age LDFs
row = np.arange(1, 14)
row = np.flip(row)
j = 1
AtA = []
for i in row:
    ldf = agg_count.iloc[:i, j+1].sum()/agg_count.iloc[:i, j].sum()
    ldf = round(ldf, 3)
    AtA.append(ldf)
    j =j+1
AtA.append(1)
new_row1 = {'Year': 'Age to Age LDFs', '1': '-', '2': AtA[0], '3': AtA[1], '4': AtA[2], '5': AtA[3], '6': AtA[4], '7': AtA[5], '8': AtA[6], '9': AtA[7], '10': AtA[8], '11': AtA[9], '12': AtA[10], '13': AtA[11], '14': AtA[12], '15': AtA[13]}
ldfs = pd.DataFrame()
ldfs = ldfs.append(new_row1, ignore_index = True)
```

```
# Age to Ultimate LDFs
AtU = []
for i in range(14): # AtA does not contain the "-"
  l = AtA[i:]
  ldf = multiply(l)
  ldf = round(ldf, 3)
  AtU.append(ldf)
  i = i+1
AtU.append(1)
new_row2 = {'Year': 'Age to Ultimate LDFs', '1': AtU[0], '2': AtU[1], '3': AtU[2], '4': AtU[3], '5': AtU[4], '6': AtU[5], '7': AtU[6], '8': AtU[7], '9': AtU[8], '10': AtU[9],
            '11': AtU[10], '12': AtU[11], '13': AtU[12], '14': AtU[13], '15': AtU[14]}
ldfs = ldfs.append(new_row2, ignore_index = True)
ldfs = ldfs[['Year', '1', '2', '3', '4', '5', '6', '7', '8', '9', '10', '11', '12', '13', '14', '15']]
# Develop loss
loss_count = list(agg_loss_count[13:])
AtU.reverse()
trend_dev_loss = []
for i in range(14):
  trend_dev_loss.append(loss_count[i]*AtU[i])

## Burning cost and expected number of losses in 2022
v = pd.read_excel("claims_data_v.xlsm")
v_2021 = v.drop(index = 15)
h = v_2021['VY']
burning_costs = []
for i in range(14):
  burning_costs.append(trend_dev_loss[i]/h[i])
print('Burning costs:')
print(burning_costs)
avg_bc = statistics.mean(burning_costs)
print('Average of burning costs from 2007 to 2020:')
print(avg_bc)
exp_num_losses = avg_bc*v['VY'][15]
print('Expected number of losses in 2022:')
print(exp_num_losses)

"""Compound model"""

## Monte Carlo simulation
random.seed(1010)
num_simulations = 10000
S1 = []
S2 = []
S3 = []
S4 = []
start_time = time.time()
for j in range(num_simulations):
```

```
    N = stats.poisson.rvs(exp_num_losses)
    S1_j = 0
    S2_j = 0
    S3_j = 0
    S4_j = 0
    for i in range(1, N+1):
      X_i = stats.gamma.rvs(param_g[0], param_g[1], param_g[2])
      retention1 = 3000000
      limit1 = 2000000
      exp_loss_layer1 = np.minimum(limit1, np.maximum(0, X_i - retention1))
      retention2 = retention1 + limit1
      limit2 = 5000000
      exp_loss_layer2 = np.minimum(limit2, np.maximum(0, X_i - retention2))
      retention3 = retention2 + limit2
      limit3 = 15000000
      exp_loss_layer3 = np.minimum(limit3, np.maximum(0, X_i - retention3))
      retention4 = retention3 + limit3
      limit4 = 25000000
      exp_loss_layer4 = np.minimum(limit4, np.maximum(0, X_i - retention4))
      S1_j = S1_j + exp_loss_layer1
      S2_j = S2_j + exp_loss_layer2
      S3_j = S3_j + exp_loss_layer3
      S4_j = S4_j + exp_loss_layer4
      i = i+1
    S1.append(S1_j)
    S2.append(S2_j)
    S3.append(S3_j)
    S4.append(S4_j)
    j = j+1
elapsed_time = time.time() - start_time
print('Elapsed time for MC simulation:')
print(elapsed_time)

# Plots
alpha = 0.05
VaR1 = np.quantile(S1,1-alpha)
print("VaR with alpha at", alpha, ":", round(VaR1))
sns.displot(S1, kind = "hist", bins = 35)
plt.xlabel('Claim amount')
plt.ylabel('Count')
plt.title('Distribution layer 1')
plt.axvline(VaR1, color='k', linestyle='dashed', ymax = 0.9, linewidth=1)
min_ylim, max_ylim = plt.ylim()
plt.text(VaR1*1.02, max_ylim*0.9, '{}% VaR: {:.0f}'.format(95, VaR1))
plt.tight_layout()
plt.show()
```

```
256
257 VaR2 = np.quantile(S2,1-alpha)
258 print("VaR with alpha at", alpha, ":", round(VaR2))
259 sns.displot(S2, kind = "hist", bins = 35)
260 plt.xlabel('Claim amount')
261 plt.ylabel('Count')
262 plt.title('Distribution layer 2')
263 plt.axvline(VaR2, color='k', linestyle='dashed', ymax = 0.9, linewidth
    =1)
264 min_ylim, max_ylim = plt.ylim()
265 plt.text(VaR2*1.02, max_ylim*0.9, '{}% VaR: {:.0f}'.format(95, VaR2))
266 plt.tight_layout()
267 plt.show()
268
269 VaR3 = np.quantile(S3,1-alpha)
270 print("VaR with alpha at", alpha, ":", round(VaR3))
271 sns.displot(S3, kind = "hist", bins = 35)
272 plt.xlabel('Claim amount')
273 plt.ylabel('Count')
274 plt.title('Distribution layer 3')
275 plt.axvline(VaR3, color='k', linestyle='dashed', ymax = 0.9, linewidth
    =1)
276 min_ylim, max_ylim = plt.ylim()
277 plt.text(VaR3*1.02, max_ylim*0.9, '{}% VaR: {:.0f}'.format(95, VaR3))
278 plt.tight_layout()
279 plt.show()
280
281 VaR4 = np.quantile(S4,1-alpha)
282 print("VaR with alpha at", alpha, ":", round(VaR4))
283 sns.displot(S4, kind = "hist", bins = 35)
284 plt.xlabel('Claim amount')
285 plt.ylabel('Count')
286 plt.title('Distribution layer 4')
287 plt.axvline(VaR4, color='k', linestyle='dashed', ymax = 0.9, linewidth
    =1)
288 min_ylim, max_ylim = plt.ylim()
289 plt.text(VaR4*1.02, max_ylim*0.9, '{}% VaR: {:.0f}'.format(95, VaR4))
290 plt.tight_layout()
291 plt.show()
292
293 print(np.mean(S1))
294 print(np.mean(S2))
295 print(np.mean(S3))
296 print(np.mean(S4))
297
298 sns.distplot(S1, hist=False, color='blue', label = 'MC L1')
299 plt.xlabel('Claim amount')
300 plt.ylabel('Density')
301 plt.title('Densities comparison')
302 sns.distplot(S2, hist=False, label = 'MC L2')
303 sns.distplot(S3, hist=False, label = 'MC L3')
```

```
sns.distplot(S4, hist=False, color='green', label = 'MC L4')
min_ylim, max_ylim = plt.ylim()
plt.legend()
plt.show()

## Panjer recursion

random.seed(1020)
dimension = 100
retention1 = 3000000
limit1 = 2000000
eps1 = limit1/dimension
lower_prob1 = stats.gamma.cdf(retention1, param_g[0], param_g[1],
    param_g[2])
upper_prob1 = 1-stats.gamma.cdf(retention1+limit1, param_g[0], param_g
    [1], param_g[2])
x1 = []
x1.append(retention1)
for i in range(dimension):
  x1.append(int(retention1+(i+1)*eps1))
CDF1_toscale = stats.gamma.cdf(x1, param_g[0], param_g[1], param_g[2])
CDF1_below = (CDF1_toscale-lower_prob1)/(1-lower_prob1)
CDF1_below = CDF1_below.tolist()
CDF1_below.append(1)
CDF1_below = np.asarray(CDF1_below)
CDF1_above = CDF1_below[1:]
CDF1_above = CDF1_above.tolist()
CDF1_above.append(1)
CDF1_above = np.asarray(CDF1_above)
density1_below = []
for i in range(len(CDF1_below)):
  if (i == 0):
    d_i = CDF1_below[i]
  else:
    d_i = CDF1_below[i] - CDF1_below[i-1]
  density1_below.append(d_i)
density1_above = []
for i in range(len(CDF1_above)):
  if (i == 0):
    d_i = CDF1_above[i]
  else:
    d_i = CDF1_above[i] - CDF1_above[i-1]
  density1_above.append(d_i)
# Panjer recursion
h0_below1 = e**(-(1-lower_prob1)*exp_num_losses)
h0_above1 = e**((1-lower_prob1)*exp_num_losses*(CDF1_above[0]-1))
a1 = 0
b1 = (1-lower_prob1)*exp_num_losses
start_time = time.time()
range_panjer = np.arange(1, 10000)
Sb1 = []
```

```
Sb1.append(h0_below1)
for n in range_panjer:
  h = 0
  lim = min(n+1, dimension+2)
  for j in range(1, lim):
    h = h + density1_below[j]*Sb1[n-j]*(a1+b1*j/n)
  Sb1.append(h)
Sa1 = []
Sa1.append(h0_above1)
for n in range_panjer:
  h = 0
  lim = min(n+1, dimension+2)
  for j in range(1, lim):
   h = h + density1_above[j]*Sa1[n-j]*(a1+b1*j/n)
  Sa1.append(h)
elapsed_time1 = time.time() - start_time
print('Elapsed time for Panjer recursion for layer 1:')
print(elapsed_time1)
CDF_pb1 = []
CDF_pb1.append(Sb1[0])
for i in range(1, len(Sb1)):
  CDF_pb1.append(Sb1[i]+CDF_pb1[i-1])
CDF_pa1 = []
CDF_pa1.append(Sa1[0])
for i in range(1, len(Sa1)):
  CDF_pa1.append(Sa1[i]+CDF_pa1[i-1])

random.seed(2050)
retention2 = retention1 + limit1
limit2 = 5000000
eps2 = limit2/dimension
lower_prob2 = stats.gamma.cdf(retention2, param_g[0], param_g[1],
    param_g[2])
upper_prob2 = 1-stats.gamma.cdf(retention2+limit2, param_g[0], param_g
    [1], param_g[2])
x2 = []
x2.append(retention2)
for i in range(dimension):
  x2.append(int(retention2+(i+1)*eps2))
CDF2_toscale = stats.gamma.cdf(x2, param_g[0], param_g[1], param_g[2])
CDF2_below = (CDF2_toscale-lower_prob2)/(1-lower_prob2)
CDF2_below = CDF2_below.tolist()
CDF2_below.append(1)
CDF2_below = np.asarray(CDF2_below)
CDF2_above = CDF2_below[1:]
CDF2_above = CDF2_above.tolist()
CDF2_above.append(1)
CDF2_above = np.asarray(CDF2_above)
density2_below = []
for i in range(len(CDF2_below)):
  if (i == 0):
```

```
    d_i = CDF2_below[i]
  else:
    d_i = CDF2_below[i] - CDF2_below[i-1]
  density2_below.append(d_i)
density2_above = []
for i in range(len(CDF2_above)):
  if (i == 0):
    d_i = CDF2_above[i]
  else:
    d_i = CDF2_above[i] - CDF2_above[i-1]
  density2_above.append(d_i)
# Panjer recursion
h0_below2 = e**(-(1-lower_prob2)*exp_num_losses)
h0_above2 = e**((1-lower_prob2)*exp_num_losses*(CDF2_above[0]-1))
a2 = 0
b2 = (1-lower_prob2)*exp_num_losses
start_time = time.time()
range_panjer = np.arange(1, 10000)
Sb2 = []
Sb2.append(h0_below2)
for n in range_panjer:
  h = 0
  lim = min(n+1, dimension+2)
  for j in range(1, lim):
    h = h + density2_below[j]*Sb2[n-j]*(a2+b2*j/n)
  Sb2.append(h)
Sa2 = []
Sa2.append(h0_above2)
for n in range_panjer:
  h = 0
  lim = min(n+1, dimension+2)
  for j in range(1, lim):
   h = h + density2_above[j]*Sa2[n-j]*(a2+b2*j/n)
  Sa2.append(h)
elapsed_time2 = time.time() - start_time
print('Elapsed time for Panjer recursion for layer 2:')
print(elapsed_time2)
CDF_pb2 = []
CDF_pb2.append(Sb2[0])
for i in range(1, len(Sb2)):
  CDF_pb2.append(Sb2[i]+CDF_pb2[i-1])
CDF_pa2 = []
CDF_pa2.append(Sa2[0])
for i in range(1, len(Sa2)):
  CDF_pa2.append(Sa2[i]+CDF_pa2[i-1])

random.seed(2060)
retention3 = retention2 + limit2
limit3 = 15000000
eps3 = limit3/dimension
```

```
452 lower_prob3 = stats.gamma.cdf(retention3, param_g[0], param_g[1],
        param_g[2])
453 upper_prob3 = 1-stats.gamma.cdf(retention3+limit3, param_g[0], param_g
        [1], param_g[2])
454 x3 = []
455 x3.append(retention3)
456 for i in range(dimension):
457   x3.append(int(retention3+(i+1)*eps3))
458 CDF3_toscale = stats.gamma.cdf(x3, param_g[0], param_g[1], param_g[2])
459 CDF3_below = (CDF3_toscale-lower_prob3)/(1-lower_prob3)
460 CDF3_below = CDF3_below.tolist()
461 CDF3_below.append(1)
462 CDF3_below = np.asarray(CDF3_below)
463 CDF3_above = CDF3_below[1:]
464 CDF3_above = CDF3_above.tolist()
465 CDF3_above.append(1)
466 CDF3_above = np.asarray(CDF3_above)
467 density3_below = []
468 for i in range(len(CDF3_below)):
469   if (i == 0):
470     d_i = CDF3_below[i]
471   else:
472     d_i = CDF3_below[i] - CDF3_below[i-1]
473   density3_below.append(d_i)
474 density3_above = []
475 for i in range(len(CDF3_above)):
476   if (i == 0):
477     d_i = CDF3_above[i]
478   else:
479     d_i = CDF3_above[i] - CDF3_above[i-1]
480   density3_above.append(d_i)
481 # Panjer recursion
482 h0_below3 = e**(-(1-lower_prob3)*exp_num_losses)
483 h0_above3 = e**((1-lower_prob3)*exp_num_losses*(CDF3_above[0]-1))
484 a3 = 0
485 b3 = (1-lower_prob3)*exp_num_losses
486 start_time = time.time()
487 range_panjer = np.arange(1, 10000)
488 Sb3 = []
489 Sb3.append(h0_below3)
490 for n in range_panjer:
491   h = 0
492   lim = min(n+1, dimension+2)
493   for j in range(1, lim):
494     h = h + density3_below[j]*Sb3[n-j]*(a3+b3*j/n)
495   Sb3.append(h)
496 Sa3 = []
497 Sa3.append(h0_above3)
498 for n in range_panjer:
499   h = 0
500   lim = min(n+1, dimension+2)
```

```
501   for j in range(1, lim):
502    h = h + density3_above[j]*Sa3[n-j]*(a3+b3*j/n)
503   Sa3.append(h)
504 elapsed_time3 = time.time() - start_time
505 print('Elapsed time for Panjer recursion for layer 3:')
506 print(elapsed_time3)
507 CDF_pb3 = []
508 CDF_pb3.append(Sb3[0])
509 for i in range(1, len(Sb3)):
510   CDF_pb3.append(Sb3[i]+CDF_pb3[i-1])
511 CDF_pa3 = []
512 CDF_pa3.append(Sa3[0])
513 for i in range(1, len(Sa3)):
514   CDF_pa3.append(Sa3[i]+CDF_pa3[i-1])
515
516 random.seed(2070)
517 retention4 = retention3 + limit3
518 limit4 = 25000000
519 eps4 = limit4/dimension
520 lower_prob4 = stats.gamma.cdf(retention4, param_g[0], param_g[1],
      param_g[2])
521 upper_prob4 = 1-stats.gamma.cdf(retention4+limit4, param_g[0], param_g
      [1], param_g[2])
522 x4 = []
523 x4.append(retention4)
524 for i in range(dimension):
525   x4.append(int(retention4+(i+1)*eps4))
526 CDF4_toscale = stats.gamma.cdf(x4, param_g[0], param_g[1], param_g[2])
527 CDF4_below = (CDF4_toscale-lower_prob4)/(1-lower_prob4)
528 CDF4_below = CDF4_below.tolist()
529 CDF4_below.append(1)
530 CDF4_below = np.asarray(CDF4_below)
531 CDF4_above = CDF4_below[1:]
532 CDF4_above = CDF4_above.tolist()
533 CDF4_above.append(1)
534 CDF4_above = np.asarray(CDF4_above)
535 density4_below = []
536 for i in range(len(CDF4_below)):
537   if (i == 0):
538     d_i = CDF4_below[i]
539   else:
540     d_i = CDF4_below[i] - CDF4_below[i-1]
541   density4_below.append(d_i)
542 density4_above = []
543 for i in range(len(CDF4_above)):
544   if (i == 0):
545     d_i = CDF4_above[i]
546   else:
547     d_i = CDF4_above[i] - CDF4_above[i-1]
548   density4_above.append(d_i)
549 # Panjer recursion
```

```
h0_below4 = e**(-(1-lower_prob4)*exp_num_losses)
h0_above4 = e**((1-lower_prob4)*exp_num_losses*(CDF4_above[0]-1))
a4 = 0
b4 = (1-lower_prob4)*exp_num_losses
start_time = time.time()
range_panjer = np.arange(1, 10000)
Sb4 = []
Sb4.append(h0_below4)
for n in range_panjer:
  h = 0
  lim = min(n+1, dimension+2)
  for j in range(1, lim):
    h = h + density4_below[j]*Sb4[n-j]*(a4+b4*j/n)
  Sb4.append(h)
Sa4 = []
Sa4.append(h0_above4)
for n in range_panjer:
  h = 0
  lim = min(n+1, dimension+2)
  for j in range(1, lim):
   h = h + density4_above[j]*Sa4[n-j]*(a4+b4*j/n)
  Sa4.append(h)
elapsed_time4 = time.time() - start_time
print('Elapsed time for Panjer recursion for layer 4:')
print(elapsed_time4)
CDF_pb4 = []
CDF_pb4.append(Sb4[0])
for i in range(1, len(Sb4)):
  CDF_pb4.append(Sb4[i]+CDF_pb4[i-1])
CDF_pa4 = []
CDF_pa4.append(Sa4[0])
for i in range(1, len(Sa4)):
  CDF_pa4.append(Sa4[i]+CDF_pa4[i-1])
timePanjer = elapsed_time1+elapsed_time2+elapsed_time3+elapsed_time4
print(timePanjer)

# Plots
plt.title('Densities layer 1')
plt.plot(np.arange(10000)*eps1, Sa1, label = 'Density above')
plt.plot(np.arange(10000)*eps1, Sb1, label = 'Density below')
plt.xlim(-0.1e8, 1.5e8)
plt.xlabel('Aggregate claim amount')
plt.ylabel('Density')
plt.legend()
plt.tight_layout()
plt.show()

plt.title('Densities layer 2')
plt.plot(np.arange(10000)*eps2, Sa2, label = 'Density above')
plt.plot(np.arange(10000)*eps2, Sb2, label = 'Density below')
plt.xlim(-0.1e8, 1.5e8)
```

```
601 plt.xlabel('Aggregate claim amount')
602 plt.ylabel('Density')
603 plt.legend()
604 plt.tight_layout()
605 plt.show()
606
607 plt.title('Densities layer 3')
608 plt.plot(np.arange(10000)*eps3, Sa3, label = 'Density above')
609 plt.plot(np.arange(10000)*eps3, Sb3, label = 'Density below')
610 plt.xlim(-0.1e8, 1.5e8)
611 plt.xlabel('Aggregate claim amount')
612 plt.ylabel('Density')
613 plt.legend()
614 plt.tight_layout()
615 plt.show()
616
617 plt.title('Densities layer 4')
618 plt.plot(np.arange(10000)*eps4, Sa4, label = 'Density above')
619 plt.plot(np.arange(10000)*eps4, Sb4, label = 'Density below')
620 plt.xlim(-0.1e8, 1.5e8)
621 plt.xlabel('Aggregate claim amount')
622 plt.ylabel('Density')
623 plt.legend()
624 plt.tight_layout()
625 plt.show()
626
627 # CDFs
628 plt.plot(np.sort(S1), np.linspace(0, 1, len(S1), endpoint=False), label
      = 'Monte Carlo', color = 'r')
629 plt.plot(np.arange(10000)*eps1, CDF_pa1, label = 'Panjer Above')
630 plt.plot(np.arange(10000)*eps1, CDF_pb1, label = 'Panjer Below')
631 plt.xlim(-0.1e8, 1.25e8)
632 plt.legend(loc = 'lower right')
633 plt.title('CDFs layer 1')
634 plt.xlabel('Aggregate claim amount')
635 plt.ylabel('Percentage')
636 plt.tight_layout()
637 plt.show()
638
639 plt.plot(np.sort(S2), np.linspace(0, 1, len(S2), endpoint=False), label
      = 'Monte Carlo', color = 'r')
640 plt.plot(np.arange(10000)*eps2, CDF_pa2, label = 'Panjer Above')
641 plt.plot(np.arange(10000)*eps2, CDF_pb2, label = 'Panjer Below')
642 plt.legend(loc = 'lower right')
643 plt.xlim(-0.1e8, 1.25e8)
644 plt.title('CDFs layer 2')
645 plt.xlabel('Aggregate claim amount')
646 plt.ylabel('Percentage')
647 plt.tight_layout()
648 plt.show()
649
```

```
plt.plot(np.sort(S3), np.linspace(0, 1, len(S3), endpoint=False), label
    = 'Monte Carlo', color = 'r')
plt.plot(np.arange(10000)*eps3, CDF_pa3, label = 'Panjer Above')
plt.plot(np.arange(10000)*eps3, CDF_pb3, label = 'Panjer Below')
plt.legend(loc = 'lower right')
plt.xlim(-0.1e8, 1.25e8)
plt.title('CDFs layer 3')
plt.xlabel('Aggregate claim amount')
plt.ylabel('Percentage')
plt.tight_layout()
plt.show()

plt.plot(np.sort(S4), np.linspace(0, 1, len(S4), endpoint=False), label
    = 'Monte Carlo', color = 'r')
plt.plot(np.arange(10000)*eps4, CDF_pa4, label = 'Panjer Above')
plt.plot(np.arange(10000)*eps4, CDF_pb4, label = 'Panjer Below')
plt.legend(loc = 'lower right')
plt.xlim(-0.1e8, 1.25e8)
plt.title('CDFs layer 4')
plt.xlabel('Aggregate claim amount')
plt.ylabel('Percentage')
plt.tight_layout()
plt.show()

## CDFs comparison for results
# MC with 100k simulations
random.seed(9010)
num_simulations_z = 100000
S1z = []
S2z = []
S3z = []
S4z = []
start_time = time.time()
for j in range(num_simulations_z):
    N = stats.poisson.rvs(exp_num_losses)
    S1_j = 0
    S2_j = 0
    S3_j = 0
    S4_j = 0
    for i in range(1, N+1):
      X_i = stats.gamma.rvs(param_g[0], param_g[1], param_g[2])
      retention1 = 3000000
      limit1 = 2000000
      exp_loss_layer1 = np.minimum(limit1, np.maximum(0, X_i -
    retention1))
      retention2 = retention1 + limit1
      limit2 = 5000000
      exp_loss_layer2 = np.minimum(limit2, np.maximum(0, X_i -
    retention2))
      retention3 = retention2 + limit2
      limit3 = 15000000
```

```
697         exp_loss_layer3 = np.minimum(limit3, np.maximum(0, X_i -
    retention3))
698         retention4 = retention3 + limit3
699         limit4 = 25000000
700         exp_loss_layer4 = np.minimum(limit4, np.maximum(0, X_i -
    retention4))
701         S1_j = S1_j + exp_loss_layer1
702         S2_j = S2_j + exp_loss_layer2
703         S3_j = S3_j + exp_loss_layer3
704         S4_j = S4_j + exp_loss_layer4
705         i = i+1
706       S1z.append(S1_j)
707       S2z.append(S2_j)
708       S3z.append(S3_j)
709       S4z.append(S4_j)
710       j = j+1
711 elapsed_time = time.time() - start_time
712 print('Elapsed time for MC simulation:')
713 print(elapsed_time)
714 # MC 10k CDFs
715 S1 = np.sort(S1)
716 vector1 = np.arange(10000)*eps1
717 CDF_MC1 = []
718 num_simulations = 10000
719 j = 0
720 count = 0
721 while j in range(10000):
722   if count == num_simulations:
723     CDF_MC1.append(1)
724     j = j + 1
725   else:
726     if S1[count] <= vector1[j]:
727       count = count + 1
728     else:
729       CDF_MC1.append(count/num_simulations)
730       j = j + 1
731 S2 = np.sort(S2)
732 vector2 = np.arange(10000)*eps2
733 CDF_MC2 = []
734 j = 0
735 count = 0
736 while j in range(10000):
737   if count == num_simulations:
738     CDF_MC2.append(1)
739     j = j + 1
740   else:
741     if S2[count] <= vector2[j]:
742       count = count + 1
743     else:
744       CDF_MC2.append(count/num_simulations)
745       j = j + 1
```

```
S3 = np.sort(S3)
vector3 = np.arange(10000)*eps3
CDF_MC3 = []
j = 0
count = 0
while j in range(10000):
  if count == num_simulations:
    CDF_MC3.append(1)
    j = j + 1
  else:
    if S3[count] <= vector3[j]:
      count = count + 1
    else:
      CDF_MC3.append(count/num_simulations)
      j = j + 1
S4 = np.sort(S4)
vector4 = np.arange(10000)*eps4
CDF_MC4 = []
j = 0
count = 0
while j in range(10000):
  if count == num_simulations:
    CDF_MC4.append(1)
    j = j + 1
  else:
    if S4[count] <= vector4[j]:
      count = count + 1
    else:
      CDF_MC4.append(count/num_simulations)
      j = j + 1
# MC 100k CDFs
S1z = np.sort(S1z)
vector1 = np.arange(10000)*eps1
CDF_MC1z = []
j = 0
count = 0
while j in range(10000):
  if count == num_simulations_z:
    CDF_MC1z.append(1)
    j = j + 1
  else:
    if S1z[count] <= vector1[j]:
      count = count + 1
    else:
      CDF_MC1z.append(count/num_simulations_z)
      j = j + 1
S2z = np.sort(S2z)
vector2 = np.arange(10000)*eps2
CDF_MC2z = []
j = 0
count = 0
```

```
while j in range(10000):
  if count == num_simulations_z:
    CDF_MC2z.append(1)
    j = j + 1
  else:
    if S2z[count] <= vector2[j]:
      count = count + 1
    else:
      CDF_MC2z.append(count/num_simulations_z)
      j = j + 1
S3z = np.sort(S3z)
vector3 = np.arange(10000)*eps3
CDF_MC3z = []
j = 0
count = 0
while j in range(10000):
  if count == num_simulations_z:
    CDF_MC3z.append(1)
    j = j + 1
  else:
    if S3z[count] <= vector3[j]:
      count = count + 1
    else:
      CDF_MC3z.append(count/num_simulations_z)
      j = j + 1
S4z = np.sort(S4z)
vector4 = np.arange(10000)*eps4
CDF_MC4z = []
j = 0
count = 0
while j in range(10000):
  if count == num_simulations_z:
    CDF_MC4z.append(1)
    j = j + 1
  else:
    if S4z[count] <= vector4[j]:
      count = count + 1
    else:
      CDF_MC4z.append(count/num_simulations_z)
      j = j + 1
# Panjer recursion dimension 100 average CDFs
CDF1p = []
for i in range(len(CDF_pa1)):
  avg = (CDF_pa1[i]+CDF_pb1[i])/2
  CDF1p.append(avg)
CDF2p = []
for i in range(len(CDF_pa2)):
  avg = (CDF_pa2[i]+CDF_pb2[i])/2
  CDF2p.append(avg)
CDF3p = []
for i in range(len(CDF_pa3)):
```

```
848     avg = (CDF_pa3[i]+CDF_pb3[i])/2
849     CDF3p.append(avg)
850 CDF4p = []
851 for i in range(len(CDF_pa4)):
852     avg = (CDF_pa4[i]+CDF_pb4[i])/2
853     CDF4p.append(avg)
854 # MSE comparison
855 mseMC1 = mean_squared_error(CDF_MC1z, CDF_MC1)
856 mseMC2 = mean_squared_error(CDF_MC2z, CDF_MC2)
857 mseMC3 = mean_squared_error(CDF_MC3z, CDF_MC3)
858 mseMC4 = mean_squared_error(CDF_MC4z, CDF_MC4)
859 mse_avgMC = (mseMC1 + mseMC2 + mseMC3 + mseMC4)/4
860 print(mse_avgMC)
861 mseP1 = mean_squared_error(CDF_MC1z, CDF1p)
862 mseP2 = mean_squared_error(CDF_MC2z, CDF2p)
863 mseP3 = mean_squared_error(CDF_MC3z, CDF3p)
864 mseP4 = mean_squared_error(CDF_MC4z, CDF4p)
865 mse_avgP = (mseP1 + mseP2 + mseP3 + mseP4)/4
866 print(mse_avgP)
867
868 ## Monte Carlo simulation AAD-AAL First Trial
869 random.seed(1010)
870 num_simulations = 10000
871 S1 = []
872 S2 = []
873 S3 = []
874 S4 = []
875 start_time = time.time()
876 for j in range(num_simulations):
877     N = stats.poisson.rvs(exp_num_losses)
878     S1_j = 0
879     S2_j = 0
880     S3_j = 0
881     S4_j = 0
882     for i in range(1, N+1):
883       X_i = stats.gamma.rvs(param_g[0], param_g[1], param_g[2])
884       retention1 = 3000000
885       limit1 = 2000000
886       exp_loss_layer1 = np.minimum(limit1, np.maximum(0, X_i -
    retention1))
887       retention2 = retention1 + limit1
888       limit2 = 5000000
889       exp_loss_layer2 = np.minimum(limit2, np.maximum(0, X_i -
    retention2))
890       retention3 = retention2 + limit2
891       limit3 = 15000000
892       exp_loss_layer3 = np.minimum(limit3, np.maximum(0, X_i -
    retention3))
893       retention4 = retention3 + limit3
894       limit4 = 25000000
```

```
895         exp_loss_layer4 = np.minimum(limit4, np.maximum(0, X_i -
    retention4))
896         S1_j = S1_j + exp_loss_layer1
897         S2_j = S2_j + exp_loss_layer2
898         S3_j = S3_j + exp_loss_layer3
899         S4_j = S4_j + exp_loss_layer4
900         i = i+1
901     AAD1 = 20000000
902     AAL1 = 20000000
903     AAD2 = 15000000
904     AAL2 = 55000000
905     AAD3 = 5000000
906     AAL3 = 60000000
907     AAD4 = 10000000
908     AAL4 = 25000000
909     S1_j = min(max(0, (S1_j - AAD1)), AAL1)
910     S2_j = min(max(0, (S2_j - AAD2)), AAL2)
911     S3_j = min(max(0, (S3_j - AAD3)), AAL3)
912     S4_j = min(max(0, (S4_j - AAD4)), AAL4)
913     S1.append(S1_j)
914     S2.append(S2_j)
915     S3.append(S3_j)
916     S4.append(S4_j)
917     j = j+1
918 elapsed_time = time.time() - start_time
919 print('Elapsed time for MC simulation:')
920 print(elapsed_time)
921
922 print(np.mean(S1))
923 print(np.mean(S2))
924 print(np.mean(S3))
925 print(np.mean(S4))
926
927 sns.displot(S1, kind = "hist", bins = 35)
928 plt.xlabel('Aggregate claim amount')
929 plt.ylabel('Count')
930 plt.title('L1 Distribution with AAD-AAL')
931 min_ylim, max_ylim = plt.ylim()
932 plt.tight_layout()
933 plt.show()
934
935 sns.displot(S2, kind = "hist", bins = 35)
936 plt.xlabel('Aggregate claim amount')
937 plt.ylabel('Count')
938 plt.title('L2 Distribution with AAD-AAL')
939 min_ylim, max_ylim = plt.ylim()
940 plt.tight_layout()
941 plt.show()
942
943 sns.displot(S3, kind = "hist", bins = 35)
944 plt.xlabel('Aggregate claim amount')
```

```
plt.ylabel('Count')
plt.title('L3 Distribution with AAD-AAL')
min_ylim, max_ylim = plt.ylim()
plt.tight_layout()
plt.show()

sns.displot(S4, kind = "hist", bins = 35)
plt.xlabel('Aggregate claim amount')
plt.ylabel('Count')
plt.title('L4 Distribution with AAD-AAL')
min_ylim, max_ylim = plt.ylim()
plt.tight_layout()
plt.show()

## Panjer recursion AAD-AAL
dimension = 100
retention1 = 3000000
limit1 = 2000000
eps1 = limit1/dimension
lower_prob1 = stats.gamma.cdf(retention1, param_g[0], param_g[1],
    param_g[2])
upper_prob1 = 1-stats.gamma.cdf(retention1+limit1, param_g[0], param_g
    [1], param_g[2])
x1 = []
x1.append(retention1)
for i in range(dimension):
  x1.append(int(retention1+(i+1)*eps1))
CDF1_toscale = stats.gamma.cdf(x1, param_g[0], param_g[1], param_g[2])
CDF1_below = (CDF1_toscale-lower_prob1)/(1-lower_prob1)
CDF1_below = CDF1_below.tolist()
CDF1_below.append(1)
CDF1_below = np.asarray(CDF1_below)
CDF1_above = CDF1_below[1:]
CDF1_above = CDF1_above.tolist()
CDF1_above.append(1)
CDF1_above = np.asarray(CDF1_above)
density1_below = []
for i in range(len(CDF1_below)):
  if (i == 0):
    d_i = CDF1_below[i]
  else:
    d_i = CDF1_below[i] - CDF1_below[i-1]
  density1_below.append(d_i)
density1_above = []
for i in range(len(CDF1_above)):
  if (i == 0):
    d_i = CDF1_above[i]
  else:
    d_i = CDF1_above[i] - CDF1_above[i-1]
  density1_above.append(d_i)
# Panjer recursion
```

```
h0_below1 = e**(-(1-lower_prob1)*exp_num_losses)
h0_above1 = e**((1-lower_prob1)*exp_num_losses*(CDF1_above[0]-1))
a1 = 0
b1 = (1-lower_prob1)*exp_num_losses
start_time = time.time()
range_panjer = np.arange(1, 10000)
Sb1 = []
Sb1.append(h0_below1)
for n in range_panjer:
  h = 0
  lim = min(n+1, dimension+2)
  for j in range(1, lim):
    h = h + density1_below[j]*Sb1[n-j]*(a1+b1*j/n)
  Sb1.append(h)
Sa1 = []
Sa1.append(h0_above1)
for n in range_panjer:
  h = 0
  lim = min(n+1, dimension+2)
  for j in range(1, lim):
   h = h + density1_above[j]*Sa1[n-j]*(a1+b1*j/n)
  Sa1.append(h)
AADn1 = int(AAD1/eps1)
AADxp1 = ((np.arange(10000)*eps1)-AAD1)[AADn1:]
first1a = 0
first1b = 0
for i in range(AADn1+1):
  first1a = first1a + Sa1[i]
  first1b = first1b + Sb1[i]
AADSa1_new = Sa1[AADn1:]
AADSa1_new[0] = first1a
AADSb1_new = Sb1[AADn1:]
AADSb1_new[0] = first1b
xp1 = np.arange(10000)*eps1
xp1 = xp1.tolist()
AALn1_new = xp1.index(AAL1)
xp1_kept = xp1[:AALn1_new+1]
last1a = 0
last1b = 0
for i in range(AALn1_new, len(xp1)-AADn1):
  last1a = last1a + AADSa1_new[i]
  last1b = last1b + AADSb1_new[i]
Sa1_new = AADSa1_new[:AALn1_new+1]
Sa1_new[AALn1_new] = last1a
Sb1_new = AADSb1_new[:AALn1_new+1]
Sb1_new[AALn1_new] = last1b
elapsed_time1 = time.time() - start_time
print('Elapsed time for Panjer recursion for layer 1:')
print(elapsed_time1)
CDF_pb1 = []
CDF_pb1.append(Sb1_new[0])
```

```
for i in range(1, len(Sb1_new)):
  CDF_pb1.append(Sb1_new[i]+CDF_pb1[i-1])
CDF_pa1 = []
CDF_pa1.append(Sa1_new[0])
for i in range(1, len(Sa1_new)):
  CDF_pa1.append(Sa1_new[i]+CDF_pa1[i-1])

random.seed(2050)
retention2 = retention1 + limit1
limit2 = 5000000
eps2 = limit2/dimension
lower_prob2 = stats.gamma.cdf(retention2, param_g[0], param_g[1],
    param_g[2])
upper_prob2 = 1-stats.gamma.cdf(retention2+limit2, param_g[0], param_g
    [1], param_g[2])
x2 = []
x2.append(retention2)
for i in range(dimension):
  x2.append(int(retention2+(i+1)*eps2))
CDF2_toscale = stats.gamma.cdf(x2, param_g[0], param_g[1], param_g[2])
CDF2_below = (CDF2_toscale-lower_prob2)/(1-lower_prob2)
CDF2_below = CDF2_below.tolist()
CDF2_below.append(1)
CDF2_below = np.asarray(CDF2_below)
CDF2_above = CDF2_below[1:]
CDF2_above = CDF2_above.tolist()
CDF2_above.append(1)
CDF2_above = np.asarray(CDF2_above)
density2_below = []
for i in range(len(CDF2_below)):
  if (i == 0):
    d_i = CDF2_below[i]
  else:
    d_i = CDF2_below[i] - CDF2_below[i-1]
  density2_below.append(d_i)
density2_above = []
for i in range(len(CDF2_above)):
  if (i == 0):
    d_i = CDF2_above[i]
  else:
    d_i = CDF2_above[i] - CDF2_above[i-1]
  density2_above.append(d_i)
# Panjer recursion
h0_below2 = e**(-(1-lower_prob2)*exp_num_losses)
h0_above2 = e**((1-lower_prob2)*exp_num_losses*(CDF2_above[0]-1))
a2 = 0
b2 = (1-lower_prob2)*exp_num_losses
start_time = time.time()
range_panjer = np.arange(1, 10000)
Sb2 = []
Sb2.append(h0_below2)
```

```
for n in range_panjer:
  h = 0
  lim = min(n+1, dimension+2)
  for j in range(1, lim):
    h = h + density2_below[j]*Sb2[n-j]*(a2+b2*j/n)
  Sb2.append(h)
Sa2 = []
Sa2.append(h0_above2)
for n in range_panjer:
  h = 0
  lim = min(n+1, dimension+2)
  for j in range(1, lim):
   h = h + density2_above[j]*Sa2[n-j]*(a2+b2*j/n)
  Sa2.append(h)
AADn2 = int(AAD2/eps2)
AADxp2 = ((np.arange(10000)*eps2)-AAD2)[AADn2:]
first2a = 0
first2b = 0
for i in range(AADn2+1):
  first2a = first2a + Sa2[i]
  first2b = first2b + Sb2[i]
AADSa2_new = Sa2[AADn2:]
AADSa2_new[0] = first2a
AADSb2_new = Sb2[AADn2:]
AADSb2_new[0] = first2b
xp2 = np.arange(10000)*eps2
xp2 = xp2.tolist()
AALn2_new = xp2.index(AAL2)
xp2_kept = xp2[:AALn2_new+1]
last2a = 0
last2b = 0
for i in range(AALn2_new, len(xp2)-AADn2):
  last2a = last2a + AADSa2_new[i]
  last2b = last2b + AADSb2_new[i]
Sa2_new = AADSa2_new[:AALn2_new+1]
Sa2_new[AALn2_new] = last2a
Sb2_new = AADSb2_new[:AALn2_new+1]
Sb2_new[AALn2_new] = last2b
elapsed_time2 = time.time() - start_time
print('Elapsed time for Panjer recursion for layer 2:')
print(elapsed_time2)
CDF_pb2 = []
CDF_pb2.append(Sb2_new[0])
for i in range(1, len(Sb2_new)):
  CDF_pb2.append(Sb2_new[i]+CDF_pb2[i-1])
CDF_pa2 = []
CDF_pa2.append(Sa2_new[0])
for i in range(1, len(Sa2_new)):
  CDF_pa2.append(Sa2_new[i]+CDF_pa2[i-1])

random.seed(2060)
```

```
retention3 = retention2 + limit2
limit3 = 15000000
eps3 = limit3/dimension
lower_prob3 = stats.gamma.cdf(retention3, param_g[0], param_g[1],
    param_g[2])
upper_prob3 = 1-stats.gamma.cdf(retention3+limit3, param_g[0], param_g
    [1], param_g[2])
x3 = []
x3.append(retention3)
for i in range(dimension):
  x3.append(int(retention3+(i+1)*eps3))
CDF3_toscale = stats.gamma.cdf(x3, param_g[0], param_g[1], param_g[2])
CDF3_below = (CDF3_toscale-lower_prob3)/(1-lower_prob3)
CDF3_below = CDF3_below.tolist()
CDF3_below.append(1)
CDF3_below = np.asarray(CDF3_below)
CDF3_above = CDF3_below[1:]
CDF3_above = CDF3_above.tolist()
CDF3_above.append(1)
CDF3_above = np.asarray(CDF3_above)
density3_below = []
for i in range(len(CDF3_below)):
  if (i == 0):
    d_i = CDF3_below[i]
  else:
    d_i = CDF3_below[i] - CDF3_below[i-1]
  density3_below.append(d_i)
density3_above = []
for i in range(len(CDF3_above)):
  if (i == 0):
    d_i = CDF3_above[i]
  else:
    d_i = CDF3_above[i] - CDF3_above[i-1]
  density3_above.append(d_i)
# Panjer recursion
h0_below3 = e**(-(1-lower_prob3)*exp_num_losses)
h0_above3 = e**((1-lower_prob3)*exp_num_losses*(CDF3_above[0]-1))
a3 = 0
b3 = (1-lower_prob3)*exp_num_losses
start_time = time.time()
range_panjer = np.arange(1, 10000)
Sb3 = []
Sb3.append(h0_below3)
for n in range_panjer:
  h = 0
  lim = min(n+1, dimension+2)
  for j in range(1, lim):
    h = h + density3_below[j]*Sb3[n-j]*(a3+b3*j/n)
  Sb3.append(h)
Sa3 = []
Sa3.append(h0_above3)
```

```
for n in range_panjer:
  h = 0
  lim = min(n+1, dimension+2)
  for j in range(1, lim):
   h = h + density3_above[j]*Sa3[n-j]*(a3+b3*j/n)
  Sa3.append(h)
AADn3 = int(AAD3/eps3)
AADxp3 = ((np.arange(10000)*eps3)-AAD3)[AADn3:]
first3a = 0
first3b = 0
for i in range(AADn3+1):
  first3a = first3a + Sa3[i]
  first3b = first3b + Sb3[i]
AADSa3_new = Sa3[AADn3:]
AADSa3_new[0] = first3a
AADSb3_new = Sb3[AADn3:]
AADSb3_new[0] = first3b
xp3 = np.arange(10000)*eps3
xp3 = xp3.tolist()
AALn3_new = xp3.index(AAL3)
xp3_kept = xp3[:AALn3_new+1]
last3a = 0
last3b = 0
for i in range(AALn3_new, len(xp3)-AADn3):
  last3a = last3a + AADSa3_new[i]
  last3b = last3b + AADSb3_new[i]
Sa3_new = AADSa3_new[:AALn3_new+1]
Sa3_new[AALn3_new] = last3a
Sb3_new = AADSb3_new[:AALn3_new+1]
Sb3_new[AALn3_new] = last3b
elapsed_time3 = time.time() - start_time
print('Elapsed time for Panjer recursion for layer 3:')
print(elapsed_time3)
CDF_pb3 = []
CDF_pb3.append(Sb3_new[0])
for i in range(1, len(Sb3_new)):
  CDF_pb3.append(Sb3_new[i]+CDF_pb3[i-1])
CDF_pa3 = []
CDF_pa3.append(Sa3_new[0])
for i in range(1, len(Sa3_new)):
  CDF_pa3.append(Sa3_new[i]+CDF_pa3[i-1])

random.seed(2070)
retention4 = retention3 + limit3
limit4 = 25000000
eps4 = limit4/dimension
lower_prob4 = stats.gamma.cdf(retention4, param_g[0], param_g[1],
    param_g[2])
upper_prob4 = 1-stats.gamma.cdf(retention4+limit4, param_g[0], param_g
    [1], param_g[2])
x4 = []
```

```
x4.append(retention4)
for i in range(dimension):
  x4.append(int(retention4+(i+1)*eps4))
CDF4_toscale = stats.gamma.cdf(x4, param_g[0], param_g[1], param_g[2])
CDF4_below = (CDF4_toscale-lower_prob4)/(1-lower_prob4)
CDF4_below = CDF4_below.tolist()
CDF4_below.append(1)
CDF4_below = np.asarray(CDF4_below)
CDF4_above = CDF4_below[1:]
CDF4_above = CDF4_above.tolist()
CDF4_above.append(1)
CDF4_above = np.asarray(CDF4_above)
density4_below = []
for i in range(len(CDF4_below)):
  if (i == 0):
    d_i = CDF4_below[i]
  else:
    d_i = CDF4_below[i] - CDF4_below[i-1]
  density4_below.append(d_i)
density4_above = []
for i in range(len(CDF4_above)):
  if (i == 0):
    d_i = CDF4_above[i]
  else:
    d_i = CDF4_above[i] - CDF4_above[i-1]
  density4_above.append(d_i)
# Panjer recursion
h0_below4 = e**(-(1-lower_prob4)*exp_num_losses)
h0_above4 = e**((1-lower_prob4)*exp_num_losses*(CDF4_above[0]-1))
a4 = 0
b4 = (1-lower_prob4)*exp_num_losses
start_time = time.time()
range_panjer = np.arange(1, 10000)
Sb4 = []
Sb4.append(h0_below4)
for n in range_panjer:
  h = 0
  lim = min(n+1, dimension+2)
  for j in range(1, lim):
    h = h + density4_below[j]*Sb4[n-j]*(a4+b4*j/n)
  Sb4.append(h)
Sa4 = []
Sa4.append(h0_above4)
for n in range_panjer:
  h = 0
  lim = min(n+1, dimension+2)
  for j in range(1, lim):
   h = h + density4_above[j]*Sa4[n-j]*(a4+b4*j/n)
  Sa4.append(h)
AADn4 = int(AAD4/eps4)
AADxp4 = ((np.arange(10000)*eps4)-AAD4)[AADn4:]
```

```
first4a = 0
first4b = 0
for i in range(AADn4+1):
  first4a = first4a + Sa4[i]
  first4b = first4b + Sb4[i]
AADSa4_new = Sa4[AADn4:]
AADSa4_new[0] = first4a
AADSb4_new = Sb4[AADn4:]
AADSb4_new[0] = first4b
xp4 = np.arange(10000)*eps4
xp4 = xp4.tolist()
AALn4_new = xp4.index(AAL4)
xp4_kept = xp4[:AALn4_new+1]
last4a = 0
last4b = 0
for i in range(AALn4_new, len(xp4)-AADn4):
  last4a = last4a + AADSa4_new[i]
  last4b = last4b + AADSb4_new[i]
Sa4_new = AADSa4_new[:AALn4_new+1]
Sa4_new[AALn4_new] = last4a
Sb4_new = AADSb4_new[:AALn4_new+1]
Sb4_new[AALn4_new] = last4b
elapsed_time4 = time.time() - start_time
print('Elapsed time for Panjer recursion for layer 4:')
print(elapsed_time3)
CDF_pb4 = []
CDF_pb4.append(Sb4_new[0])
for i in range(1, len(Sb4_new)):
  CDF_pb4.append(Sb4_new[i]+CDF_pb4[i-1])
CDF_pa4 = []
CDF_pa4.append(Sa4_new[0])
for i in range(1, len(Sa4_new)):
  CDF_pa4.append(Sa4_new[i]+CDF_pa4[i-1])
timePanjer = elapsed_time1+elapsed_time2+elapsed_time3+elapsed_time4
timePanjer

plt.title('CDF from Panjer layer 1')
plt.plot(np.sort(S1), np.linspace(0, 1, len(S1), endpoint=False), label
    = 'Monte Carlo', color = 'r')
plt.plot(xp1_kept, CDF_pa1, label = 'Panjer Above')
plt.plot(xp1_kept, CDF_pb1, label = 'Panjer Below')
plt.title('L1 CDFs with AAD-AAL')
plt.xlabel('Aggregate claim amount')
plt.ylabel('Percentage')
plt.tight_layout()
plt.legend(loc = 'lower right')
plt.show()

plt.title('CDF from Panjer layer 2')
plt.plot(np.sort(S2), np.linspace(0, 1, len(S2), endpoint=False), label
    = 'Monte Carlo', color = 'r')
```

```
plt.plot(xp2_kept, CDF_pa2, label = 'Panjer Above')
plt.plot(xp2_kept, CDF_pb2, label = 'Panjer Below')
plt.title('L2 CDFs with AAD-AAL')
plt.xlabel('Aggregate claim amount')
plt.ylabel('Percentage')
plt.tight_layout()
plt.legend(loc = 'lower right')
plt.show()

plt.title('CDF from Panjer layer 3')
plt.plot(np.sort(S3), np.linspace(0, 1, len(S3), endpoint=False), label
    = 'Monte Carlo', color = 'r')
plt.plot(xp3_kept, CDF_pa3, label = 'Panjer Above')
plt.plot(xp3_kept, CDF_pb3, label = 'Panjer Below')
plt.title('L3 CDFs with AAD-AAL')
plt.xlabel('Aggregate claim amount')
plt.ylabel('Percentage')
plt.tight_layout()
plt.legend(loc = 'lower right')
plt.show()

plt.title('CDF from Panjer layer 4')
plt.plot(np.sort(S4), np.linspace(0, 1, len(S4), endpoint=False), label
    = 'Monte Carlo', color = 'r')
plt.plot(xp4_kept, CDF_pa4, label = 'Panjer Above')
plt.plot(xp4_kept, CDF_pb4, label = 'Panjer Below')
plt.title('L4 CDFs with AAD-AAL')
plt.xlabel('Aggregate claim amount')
plt.ylabel('Percentage')
plt.tight_layout()
plt.legend(loc = 'lower right')
plt.show()
```

Bibliography

[1] Oliver D. William, *Reinsurance and the law of aggregation*, Routledge 2021, 1st edition.

[2] Swiss Reinsurance Company, *An introduction to reinsurance*, 2022, 8th edition.

[3] David R. Clark (FCAS), *Basics of Reinsurance Pricing* first version 1996, revised 2014.

[4] Thomas Mack, *Distribution-free Calculation of the Standard Error of Chain Ladder Reserve Estimates*, Astin Bulletin 1993, 23, 213-225

[5] Thomas Mack, *The prediction error of Bornhuetter/Ferguson.*

[6] Björn Weindorfer, *A practical guide to the use of the chain-ladder method for determining technical provisions for outstanding reported claims in non-life insurance*, University of Applied Sciences bfi Vienna 2012, number 77.

[7] Pavel V. Shevchenko, *Calculation of aggregate loss distributions*, CSIRO Mathematics Informatics and Statistics 2010.

[8] R. J. Verrall, *The individual risk model: a compound distribution*, City University London 1989, 101-107.

[9] Peter Antal, *Mathematical Methods in Reinsurance*, Swiss Re 2009.

[10] S. Desmedt, M. Snoussi, X. Chenut, J.F. Walhin, *Experience and exposure rating for property per risk excess of loss reinsurance revisited*, Astin Bulletin 2012, 233-270.

[11] Thomas P. Minka, *Estimating a Gamma distribution*, 2002.

[12] M. H. Pham, C. Tsokos, B.-J. Choi, *Maximum likelihood estimation for the generalized Pareto distribution and goodness- of-fit test with censored data*,

Journal of Modern Applied Statistical Methods 2018.

[13] Ginos Brenda Faith, *Parameter Estimation for the Lognormal Distribution*, Theses and Dissertations 2009.

[14] Alexander J. McNeil, Rudiger Frey, Paul Embrechts, *Quantitative risk management*, 2015.

[15] Paul Embrechts, Claudia Kluppelberg, Thomas Mikosch, *Modelling Extremal Events for Insurance and Finance*, Springer 1997.

[16] Stuart A. Klugman, Harry H. Panjer, Gordon E. Willmot, *Loss Models. From Data to Decisions*, Wiley 1998, 1st edition.

Acknowledgements

Se mi permettete, i ringraziamenti li vorrei scrivere nella mia lingua madre (mio padre ne sarà felicissimo). Saranno più o meno sessantanove pagine, ma insomma: in triennale non li avevo scritti quindi concedetemelo. E poi chi mi conosce sa che parlo tanto.

Non posso dire che questi anni di studio siano stati facili nè tantomeno leggeri, ma certamente ho la fortuna di poter condividere con voi, voi che state leggendo e che siete arrivati fin qui con me, la bellezza di tutto ciò che c'è stato nel mezzo.

La vita mi ha insegnato che non sempre le difficoltà vengono per nuocere. Spesso e volentieri servono a metterci alla prova, tenerci in continua evoluzione e in attento ascolto di noi stessi. E quando dico "la vita" intendo dire i miei genitori, Michelangela e Giovanni Pietro (anche detti Michi e Giampi o, per la sottoscritta, mami e daddy). Loro sono le persone che voglio ringraziare per prime proprio perché è a loro che devo tutto. Mi avete insegnato la bellezza che sta in ciascuno di noi e osservandovi nella vostra quotidianità ho imparato a credere ciecamente nelle parole vita, condivisione, incontro, amore, accettazione e coraggio.

Il secondo grazie va ai miei fratelli, tutti e tre. Mi soffermerò di più sui due che mi hanno annoiata maggiormente nel corso della mia vita, ma è bene sottolineare che è per certo Mattia quello che mi ha sempre protetta e che sempre mi proteggerà. E sono sicura che se ne avesse avuto l'opportunità, sarebbe probabilmente stato il più noioso di tutti.
Grazie a Davide, che ha sempre avuto il coraggio di essere se stesso, di lottare contro un mondo che sembrava non volerlo sostenere mai e di credere nel suo valore, ogni giorno di più. Mi hai insegnato come approcciare la vita nel modo giusto, a testa alta senza vergognarsi mai. E non potrò mai essertene più grata. Spero tu possa essere la mia guida luminosa per ancora tanto, tantissimo tempo.
Grazie a Simone, che per chi ci conosce sa bene che è stato la fiamma che mi ha plasmata, fortificata e tenuta in vita. E ha sempre fatto tutte e tre queste cose contemporaneamente, assicurandosi che fossi costantemente annoiata a morte dalla sua presenza. Ma è così che noi ci vogliamo bene. Ti devo tantissimo e ti ammiro infinitamente per quello che sei, per la determinazione che ti caratterizza e per la volontà d'animo che ti porta a non stare fermo mai.

Grazie alla mia famiglia allargata, alle zie, agli zii e a tutti i cugini (acquisiti e non): siete una forza e vi ammiro tanto per tutto ciò che siete. Nella famiglia allargata ovviamente rientrano anche Ba, Diego e Lori. Grazie per essere sempre stati presenti, disponibili e di gran cuore. Grazie per la vostra energia, accoglienza e per essere dei fortissimi avversari di giochi.

Subito dopo la famiglia ci sono i Moschettieri, che mi sanno coccolare con tutti loro stessi ma se devono prendermi a schiaffi diciamo che lo fanno al 150%. Graize a Os perché la sua forza e il suo coraggio mi insegnano tanto ogni giorno. Hai tantissimo da donare e sono felice che tu scelga ogni giorno di avermi al tuo fianco in questo percorso che è la vita. Sei un'artista e andrai tanto tanto lontano, ma come direbbe Noe: "Mi raccomando ricordati chi ti ha cresiuto". Grazie a Noe per i suoi sorrisi, il suo cuore grande e il suo costante supporto nonostante ai messaggi non risponda mai. Sei preziosissima e io sono fortunata ad averti. Grazie ad Anni che è la roccia del gruppo, quella che tendenzialmente se ne esce con otto frasi che non c'entrano assolutamente niente, ma poi ne dice due che ti salvano la vita. Sei una forza della natura e di opportunità ne avrai ancora tantissime: noi ci assicureremo di farti sempre avere uno smalto rosso.

Apriamo ora il reparto università. Il grazie successivo va a quegli amici che ci sono sempre, soprattutto quando la vita ti tira porte in faccia. E di porte in faccia io e EtiEti ne abbiamo prese tantissime. Per questo sei la prima che ringrazio. Grazie per esserci sempre e ogni volta nel modo giusto. Non aggiungerò altro perché so che non serve, nonostante sia io quella che ha sempre mille parole per tutto. Ah e buon compleanno!
Grazie a Mari Poo, la sorella che non ho mai avuto ma che è sempre stata presente, in ogni tappa della mia vita. Inutile dire che quando dico "sorella" intendo "sorella maggiore" (tu capirai). Ti voglio bene, non lasciarmi mai.
Grazie a Sal, perché come mi capisci tu lo fanno in pochi. Sei forte e ti ringrazio per avermi aspettata sempre senza farmi mai sentire in difetto per nulla.
A queste tre persone se ne aggiungono altre tre così da formare lo squadrone alfa dei matematici: Gigino, Furlo e But. Ragazzi che dire: vi voglio un bene infinito! Cioè Gigino ormai è sposato e lo abbiamo perso per la tangente, Furlo è un genietto da qualche parte a Trieste e But non si sa mai se sia a Parigi o nella regione Sabauda. Siete stati la mia forza in molte occasioni e ve ne sono davvero grata.

Grazie a Lizzipiz e Giulipet, le prime coinquiline a Torino e le sorellastre che avrei avuto se fossi nata Cenerentola. In realtà non è vero, ma dovevo inserire qualcosa della Disney per ricordare al mondo che noi tre siamo le migliori in questo campo. Vi voglio bene, siete state e siete tutt'ora importantissime nel mio percorso e non posso che augurarvi un gran bene e un augurio di felicità per tutto quello che verrà.

Grazie a Bea, la mia ultima coinquilina a Torino e la mia compagna di avventure di una vita intera. Non posso pensare di affrontare un giorno senza la tua

energia, il tuo costante supporto e la tua voglia di sorridere tanto, anche quando il mondo non aiuta. Ti sono davvero grata per tutto.

Grazie a Torino, perché è lì che ho scoperto me stessa ed è lì che lascerò un gran pezzo di cuore. Sei stata la mia casa e la mia rivolta e sei tu che ospiti tutte quelle persone di cui non farò il nome, ma che sanno che rientrano in una grande sezione del mio cuore. Grazie a tutti voi, siete stati preziosi. In particolare, grazie a Vale per aver condiviso con me tanto cibo e tante belle cose, grazie alla famiglia Gili per essere sempre stata disponibile e stupenda nei miei confronti, grazie ai Gonorreici per tutta la magia e felicità che avete portato nella mia vita, grazie agli Attori Consapevoli per quei brevi ma intensi momenti accerchiati da covid e incertezze. Grazie alla mia bicicletta, Boeris, che mi ha portata dappertutto nonostante la fatica di non smontarsi metro dopo metro.

L'ultimo grazie va a Zurigo, la città che mi ha portata via dall'Italia e mi ha fatto trovare me stessa nel modo più inaspettato e bello possibile. Mi hai accompagnata nella fine di questo lungo e travagliato percorso e nel primo salto nel vuoto. Sei una città bellissima e non ti dimenticherò mai. Grazie ai miei colleghi, in particolare a Laurent e Quentin che hanno avuto la pazienza di supportarmi e sopportarmi nella stesura della tesi. Grazie ai miei coinquilini, preziosissimi nel mio percorso, persone di cuore e sempre presenti. Grazie a tutti coloro che ho incontrato e con cui ho condiviso momenti ed esperienze: avete lasciato un segno profondo.