

**FACULTY
OF MATHEMATICS
AND PHYSICS**
Charles University

MASTER THESIS

Bc. Matěj Drahokoupil

**Analysis of several acceleration
techniques for life insurance liability
value determination**

Department of probability and mathematical statistics

Supervisor of the master thesis: doc. RNDr. Michal Pešta, Ph.D.

Study programme: Mathematics

Study branch: Financial and insurance
mathematics

Prague 2020

I declare that I carried out this master thesis independently, and only with the cited sources, literature and other professional sources. It has not been used to obtain another or the same degree.

I understand that my work relates to the rights and obligations under the Act No. 121/2000 Sb., the Copyright Act, as amended, in particular the fact that the Charles University has the right to conclude a license agreement on the use of this work as a school work pursuant to Section 60 subsection 1 of the Copyright Act.

In Prague 5.1.2021

.....

Author's signature

I would like to thank my supervisor Doc. RNDr. Michal Pešta, Ph.D. and my consultant RNDr. Martin Janeček, Ph.D. for their valuable advice during the creation of the master thesis. Their contribution was not only in the consultation of mathematical questions but also in valuable advice concerning formal arrangement of the thesis. Especially introductory theme ideas were brought by RNDr. Martin Janeček Ph.D. which was essential to finish the thesis. I would also like to thank my brother who also helped in several stages of writing the thesis. Last but not least I would like to thank my girlfriend who was my support during studies.

Title: Analysis of several acceleration techniques for life insurance liability value determination

Author: Matěj Drahokoupil

Department: Department of probability and mathematical statistics

Supervisor: Doc. RNDr. Michal Pešta, Ph.D., Department of probability and mathematical statistics

Consultant: RNDr. Martin Janeček, Ph.D.

Abstract: The aim of the diploma thesis is to apprise the reader with a basic life insurance projection method which is used for the valuation of insurance company's liabilities. The basic projection method can be extremely time consuming in practise so another two variance reduction methods and their combination are presented to obtain either more precise liabilities estimation, or to reduce the time required for the projection. The presented methods are antithetic variate method, control-variate method and their combination later called integrated control-variate method. The final outcome of the thesis is simulation experiment which evaluates the liabilities of the group of policies and comparison of the presented variance reduction methods.

Keywords: Cash flow model, acceleration, liability value, best estimate value of liabilities, options and guarantees in life insurance.

Contents

| | |
|--|-----------|
| Introduction | 3 |
| 1 Liabilities valuation | 4 |
| 1.1 Traditional life insurance contract | 4 |
| 1.2 Guaranteed investment and profit share | 4 |
| 1.3 Best estimate liability | 5 |
| 1.4 Liability calculation assumptions | 5 |
| 1.4.1 Non-financial assumptions | 5 |
| 1.4.2 Financial assumptions | 7 |
| 1.5 Cash flows projection | 7 |
| 1.6 Cash flows calculation | 8 |
| 2 Variance reduction methods in Monte Carlo simulations | 14 |
| 2.1 Review of confidence intervals for estimating a mean | 14 |
| 2.1.1 Basic definitions | 14 |
| 2.1.2 Monte Carlo simulation application | 15 |
| 2.2 Antithetic variate method | 15 |
| 2.2.1 Underlying principle | 16 |
| 2.3 Control-variate method | 16 |
| 2.3.1 Underlying principle | 16 |
| 2.3.2 Multiple Control Variates | 18 |
| 2.4 Integrated control-variate estimator | 19 |
| 2.4.1 Underlying principle | 20 |
| 2.4.2 Integrated Control-Variate alternative I | 23 |
| 2.4.3 Integrated Control-Variates alternative II | 25 |
| 2.4.4 Estimate of the Control variate estimator variance | 25 |

| | | |
|----------|--|-----------|
| 3 | Market scenarios | 27 |
| 3.1 | Short term interest rate models | 27 |
| 4 | Implementation | 29 |
| 4.1 | Interest rate scenarios | 29 |
| 4.2 | Cashflow projections assumptions | 32 |
| 4.2.1 | Modelpoints file | 32 |
| 4.2.2 | Other assumptions | 32 |
| 4.3 | Run results | 35 |
| 4.3.1 | Run 1 - Base run | 35 |
| 4.3.2 | Run 2 - Antithetic variates run | 36 |
| 4.3.3 | Run 3 - Conventional control-variate estimator | 37 |
| 4.3.4 | Run 4 - Integrated control variate estimator | 38 |
| 4.3.5 | BEL comparison | 39 |
| 4.3.6 | Standard deviation comparison | 40 |
| 4.3.7 | Time comparison | 41 |
| | Conclusion | 45 |
| | Bibliography | 47 |
| | List of Figures | 49 |
| | List of Tables | 50 |
| A | Attachments | 51 |
| A.1 | First Attachment | 51 |

Introduction

The best estimate liability (BEL) estimation is one of the most important operations that actuaries in the insurance company do. Present value of future liabilities is basis for the Solvency II and the IFRS17 calculations.

The BEL value is dependent on economic assumptions and to get the expected value of future liabilities usually thousands of economic scenarios are used to discount the future liabilities. These economic scenarios are usually generated by Monte Carlo simulation with chosen model. Generally discounting of the future liabilities with numerous economic scenarios is very time consuming. This thesis aims on presenting mathematical methods that accelerates the projection. Two mathematical methods and their combination will be presented further in the text. The impact of each method on the time consumption will be compared.

We will compare the results of standard BEL calculation where no method is used with three acceleration methods - Antithetic variate method, Control-variate method and their combination - Integrated control-variate method. The economic scenarios will be generated using the Hull-White model. The test strain of insurance contracts is purely fictitious.

The thesis consists of four chapters. The first chapter introduces the basic principles of cash flows valuation. Standard projection of future cash flows in a life insurance company is described. The process of BEL calculation and its assumptions are presented.

The second chapter presents the Control-variate method, Antithetic variate method and Integrated control-variate method which are in the simulation part implemented and compared .

Third chapter briefly introduces the theory for simulation of interest rates and describes the Hull-White interest rate model.

The forth chapter shows the implementation of the acceleration method applied on the best estimate liabilities. Only traditional life products are used for liability calculation. Both methods are compared with the basis solution (without any method) from the perspective of time consumption.

Both Hull-White model and the Cash-flow model of liabilities are coded in Python language.

1. Liabilities valuation

1.1 Traditional life insurance contract

Traditional life insurance contract is the contract where the insured person pays the premium (single paid or regularly paid) and obtains:

- Sum assured death in case of death during the insured period (term or endowment products)
- Sum assured in case of surviving (endowment or pure endowment products)
- an annuity

The basic assumption in traditional life product is that it provides guaranteed financial coverage i.e. the value of cashflows is discounted with technical interest rate (TIR) when computing the present value of the future benefits or premiums. This means that the investment risk is borne by the insurer.

In the traditional life insurance contract the insured person knows the minimum value of a benefit obtained in case of death or in case of surviving. Both benefits can be increased by the profit shared by the insurance company.

1.2 Guaranteed investment and profit share

Traditional life insurance contracts bear a guarantee of the investment rate. The insurance company needs to interest the part of paid premium which is not deducted to cover the insured risks. This is usually hidden in the premium calculation for selected sum assured. Still it means that the company needs to invest the surplus to get at least the agreed TIR (part of the insurance contract) not to be in the red numbers for the single contract. If the investment return is higher than the guaranteed interest rate, the difference changes to a profit share. Part of the profit share is used to cover the future undesirable investment performance and the other part is paid to policyholders as a profit.

The technical interest rate might be different for each type of product. In this work, we will assume that the technical interest rate is equal to 2,5 % for all contracts. The total payoff assigned to the policyholder's contract is:

$$i_{tot} = TIR + \max(0, i_{real} - TIR) \quad (1.1)$$

where i_{tot} is the total investment return assigned to a policyholder, TIR is technical interest rate and i_{real} is the interest rate that was realized by the insurance company.

1.3 Best estimate liability

The best estimate liability is part of the Solvency II technical provision. It is defined as a "probability - weighted average of future cash-flows, taking into account the time value of money (expected present value of the future cash-flows), using the relevant risk-free interest rate term structure" by Solvency Sol. Formula 1.2 shows the present value of future cash flow projection at the valuation (time 0)

$$PVCF_0 = \sum_{t=1}^{\infty} \frac{ECF_t}{(1 + r_t)^t}, \quad (1.2)$$

where:

ECF_t is the expected cash flow at the end of period t ,

r_t is the risk-free rate.

t is the projection period (month, year, etc.)

There are only in force business calculated in the liability model - no future new business is considered. Projection is made for 50 years but not all policies end 50 years after valuation. All calculations are based on the risk-neutral principle which means that all insurance policyholders are indifferent to the risk and the expected investment return is equal to the risk-free rate.

1.4 Liability calculation assumptions

Liability calculation is based on multiple assumptions. All assumptions are considered to be on the best estimate level i.e. they reflect the real life behaviour of clients etc.. We further divide the assumptions on non-financial assumptions and financial assumptions.

1.4.1 Non-financial assumptions

Mortality

Mortality rates are generally divided into valuation mortality rates and experience mortality rates. The first mentioned are used when calculating the premiums or mathematical reserves associated with the insurance contract. These mortality rates should not distinguish between men and women and should be based on region statistical measures i.e. mortality rates from ČSU for given year.

On the other hand experience mortality rates should express the real behaviour of the insurance strain for the given insurance company. Furthermore experience mortality rates can distinguish between men and women. These mortality rates are used for decrement models i.e. number of active policies etc.. For the decrement model purposes Lee-Carter model can be used and further more partitioning for experience mortality rates can be done. Further partitioning can be for example health status (smoker, standard, healthy).

Lapses

Traditional life contracts carries an option to cancel the contract at any time and get some part of the paid premium back (surrender fees are deducted from the mathematical reserve value). The fact is that a large part of policies are canceled in the first few years of contract so the correct lapse tables are an important part of the liabilities calculations.

Lapse rates should be primarily based on insurance company data and should reflect given product or product type. This means that lapse rates can differ for term products, endowments etc. or they can differ between inception years due to some external circumstances or changes of policy.

Paid-ups

Another very important part of the decrement model are Paid-ups. Similarly to the lapses policyholder has an option to move its contract to the paid-up state. It means that policyholder does not have to pay the premiums but the sum assured is recalculated, so the present value of future benefits is equal to the value of accumulated mathematical reserve.

Paid-up rates are calculated on a similar basis as lapse rates.

Expenses

Expenses are connected to the insurance contracts. It should represent the value of expenses that need to be spend at the policy inception (initial expenses) or during the policy life time (renewal expenses).

Initial expenses usually represent the advertisement or medical treatment. Renewal expenses are usually connected with payouts, premium collections etc.

Commissions

Commissions represent the amount of money paid by the insurance company to an agent that sold the policy contract. They are usually defined as a part of

the annual premium or a part of the sum assured. They are divided into initial commissions and renewal commissions which is the same as expenses. Initial commissions are paid in the first few years and renewal commissions are paid during the policy lifetime. If the contract lapses, or the insured person dies in the first few years, insurance company returns a part of the initial commissions back to the policyholder. This state is called commissions clawback.

1.4.2 Financial assumptions

In order to calculate the liabilities of cash flows in risk-neutral world, the risk-free rate is used. Risk-free rate is a theoretical rate of return of an investment with zero risk. Government bonds yields are often for such a rate. Usually, there are two rates used in cash flow projection: one for cash flows discount and another one for an investment return on assets (based on which the profit share is distributed). Within this work we will assume that the insurance company invests in risk-free assets, and for further calculation, we will use one rate to evaluate the profit share value and also for cash flow discounting.

1.5 Cash flows projection

We need to project all cash flows for every projection period to get liabilities that occur during the insurance contracts lifetime. All cash flows are random. Projection is made for all contracts that the insurance company poses. We will define all principles on a contract. Part of the cash flows are generally paid at the beginning of the projection period and the rest of them are paid at the end of the projection period. Usually, the insurance company works with following random cash flows:

- P_t^{in} is premium paid by the policy holder the beginning of the period t
- C_t^{out} are the commissions paid by the insurance company to the agent paid at the beginning of the period t
- E_t^{out} are the expenses allocated to the contract paid at the beginning of the period t
- D_t^{out} is the death outgo paid at the end of the period t
- S_t^{out} is the surrender outgo (in case of lapse) paid at the end of the period
- M_t^{out} is the maturity outgo paid at the beginning of the next period after the maturity i.e. $t + 1$

Cash flow projection for one period with discount is showed in formula 1.3:

$$DCF_t = (CF_t^{end} + DCF_{t+1})/(1 + r_{t+1}) + CF_t^{start}, \quad (1.3)$$

where formula 1.4 shows expected cash flows at the beginning of the month and formula 1.5 shows expected cash flows at the end of the period.

$$CF_t^{start} = E(-P_t^{in} + C_t^{out} + E_t^{out} + M_{t-1}^{out}) \quad (1.4)$$

$$CF_t^{end} = E(D_t^{out} + S_t^{out}) \quad (1.5)$$

Discount rate r_t is again risk-free rate.

1.6 Cash flows calculation

In this part we will show basic formulas for cash flow calculation used in the model and in the projection. Every cash flow is probability weighted as was mentioned earlier. We will distinguish between number of living (i.e. policies in force) used for reserving purposes and number of policies in force used for decrement model with upper index *decr*. The same notation is used for number of deaths and number of maturities. Following notation will be used in the cash flows formulas as in Janeček (2006).

| | |
|--------------------|--|
| x | age of a policy holder |
| \mathbf{T}_x | the remaining number of life years at age x |
| q_x | probability that a person who is alive at the age of x will die before the age $x + 1$, $q_x = P(\mathbf{T}_x \leq 1)$ |
| p_x | probability that a person who is alive at the age of x will be alive at the age of $x + 1$, $p_x = P(\mathbf{T}_x > 1)$ |
| ${}_t p_x$ | probability that a person who is alive at the age of x will be alive at the age of $x + t$, ${}_t p_x = P(\mathbf{T}_x > t)$ |
| $q_{x,\tau}^{exp}$ | expected mortality, adjusted probability that a person who is alive at the age of x will die before the age $x + 1$, $q_x = coef_\tau \times q_x$, where $coef_\tau$ is adjustment depending on a policy year τ |
| q_t^{surr} | probability of lapse at the policy year t |
| l_t | expected number of policies in-force used for reserving at the end of projection year t , $l_t = l_{t-1} - d_t - m_t$ |
| l_t^{decr} | number of policies in-force used in decrement model at the end of the projection year t , $l_t = l_{t-1}^{decr} - d_t^{decr} - m_t - w_t$ |
| d_t | expected number of deaths at the end of the projection year t , $d_t = l_{t-1} \times q_x$ |
| d_t^{decr} | expected number of deaths in decrement model at the end of the projection year t , $d_t = l_{t-1}^{decr} \times q_{x,\tau}^{exp}$ |
| w_t | expected number of lapses at the end of the projection year t , $w_t = (l_{t-1}^{decr} - d_t^{decr}) \times wthd_t$ |
| m_t^{decr} | expected number of maturities used in decrement model at the end of the projection year t , |
| | $m_t = \begin{cases} 0 & \text{if } \tau < n \\ l_{t-1}^{decr} - d_t^{decr} - w_t & \text{if } \tau = n \end{cases}$ |
| m_t | expected number of maturities at the end of the projection year t , |
| | $m_t = \begin{cases} 0 & \text{if } \tau < n \\ l_{t-1} - d_t & \text{if } \tau = n \end{cases}$ |
| v | discount factor |
| D_t | discounted expected number of living $D_t = l_t v^t$ |
| C_t | discounted expected number of deaths $D_t = d_t v^{t+1}$ |
| M_t | Commutation number of the first order $M_t = \sum C_t$ |
| N_t | Commutation number of the first order $N_t = \sum D_t$ |
| α | acquisition costs that are paid at the inception of the contract and are expressed in absolute terms |
| β | administration costs that are expressed as a percent of sum assured |
| γ | collection costs that are expressed as percent of Gross premium |

We will consider only two types of products in our model - endowments with regular premium and endowments with single premium.

Premium

Premium for traditional life products is calculated separately for reserving purposes - Net premium with possible zillmerization and Gross premium for premium income purposes. Following formulas are only for endowment contract. Formulas for premium calculations can be found in Cipra (2006). Net premium formula for single premium contract is in 1.6

$${}_nJP_x = A_{x;n} = \frac{M_x - M_{x+n} + D_{x+n}}{D_x} \quad (1.6)$$

and for the regular premium contract is in 1.7

$${}_nP_x = \frac{A_{x;n}}{\ddot{a}_{x;n}} = \frac{M_x - M_{x+n} + D_{x+n}}{N_x - N_{x+n}} \quad (1.7)$$

Formula for the single gross premium contract including the loadings is in 1.8

$${}_nJB_x = A_{x;n} + \alpha + \beta\ddot{a}_{x;n} \quad (1.8)$$

In case of the regular gross premium contract there are collection costs in addition to single premium contract. The final formula follows from 1.9

$${}_nB_x\ddot{a}_{x;n} = A_{x;n} + \alpha + \beta\ddot{a}_{x;n} + \gamma{}_nB_x\ddot{a}_{x;n}. \quad (1.9)$$

The final expression for regular gross premium is in 1.10

$${}_nB_x = \frac{A_{x;n} + \alpha + \beta\ddot{a}_{x;n}}{(1 - \gamma)\ddot{a}_{x;n}}. \quad (1.10)$$

For simplicity we will assume that there are no premium adjustments for non-annual premium payments.

Commissions

Initial commissions are paid only in the first year of policy period. On the other hand renewal commissions are paid during the whole policy period. Initial commissions are in 1.11

$$Comm_t^{Init} = SA \times Comm_{SA}^{Init}\% + P_t \times Comm_{Prem}^{Init}\%. \quad (1.11)$$

Renewal commissions are in the 1.12

$$Comm_t^{Ren} = SA \times Comm_{SA}^{Ren}\% + P_t \times Comm_{Prem}^{Ren}\%, \quad (1.12)$$

where

| | |
|----------------------|---|
| $Comm_t^{Init}$ | initial commission at period t |
| $Comm_t^{Ren}$ | renewal commission at period t |
| $Comm_{SA}^{Init}$ | initial commission as a percentage of sum assured |
| $Comm_{Prem}^{Init}$ | initial commission as a percentage of premium |
| $Comm_{SA}^{Ren}$ | renewal commission as a percentage of sum assured |
| $Comm_{Prem}^{Ren}$ | renewal commission as a percentage of premium |
| SA | sum assured |
| P_t | premium at period t |

Expenses

Expenses are similarly to commissions divided into initial and renewal. Usually, it is assumed that expenses rise during the lifetime of policy. We will assume that they rise with an inflation. Initial expenses are again assumed to occur only in the first year of policy lifetime. Also we will assume that part of the expenses is fixed regardless the premium. Initial expenses are in 1.13

$$Exp_t^{Init} = FixExp^{Init} + VarExp^{Init}\% \times P_t. \quad (1.13)$$

Renewal expenses are in 1.14

$$Exp_t^{Ren} = FixExp^{Ren} \times (1 + i_r) + VarExp^{Ren}\% \times P_t, \quad (1.14)$$

where

| | |
|-----------------|--|
| Exp_t^{Init} | initial expenses at period t |
| Exp_t^{Ren} | renewal expenses at period t |
| $FixExp^{Init}$ | initial fixed expenses |
| $FixExp^{Ren}$ | renewal fixed expenses |
| $VarExp^{Init}$ | initial variable expenses as a percentage of premium |
| $VarExp^{Ren}$ | renewal variable expenses as a percentage of premium |
| i_t^r | inflation rate at time t |
| P_t | premium at period t |

The fixed expenses are multiplied by inflation rate at each period. Inflation rate can be either constant during the whole projection period or it can vary in time.

Mathematical reserve

The value that expresses the amount of policyholders resources that were not consumed to cover the risks is expressed by the mathematical reserve. It is also the amount that insurance company should hold to cover the future benefit payments.

The mathematical reserve for single premium policy in time t is calculated by formula 1.15

$${}_tV_x = A_{x+t;n-t} + \beta \ddot{a}_{x+t;n-t} \quad (1.15)$$

and formula for regular premium policy is in 1.16

$${}_tV_x = A_{x+t;n-t} - {}_n P_x \ddot{a}_{x+t;n-t} - \alpha \frac{\ddot{a}_{x+t;n-t}}{\ddot{a}_{x;n}} \quad (1.16)$$

Actuarial factors and commutation functions counts with TIR discounting of future premiums and benefits.

Surrender value

Surrender value is part of the mathematical reserve that is given back to policyholder in case of lapse. Usually there is a surrender during which the surrender value is zero. To get a surrender value insurance company lowers the value of mathematical reserve by surrender fee which helps the insurance company settle this cashflow. Surrender value is calculated in 1.17 if the time is greater than surrender period.

$$SurrVal_t = {}_tV_x * (1 - SurrFee_t) \quad (1.17)$$

Benefits

Benefit payments are equal to Sum assured plus accrued profit share. We do not distinguish between sum assured in case of death and sum assured in case of maturity.

Expected cash flows

To get the expected cash flows we have to take per policy values of the given cash flow and multiply it with relevant factor (expected number of in force policies, expected number of lapses, expected number of deaths, ...). The formulas are in table 1.6 Kuzminskaya (2018).

| Proj. year | | |
|---------------|-------------------|--|
| 1 | Premium Income | $P_0^{in} = l_0^{decr} \times {}_nB_x$ |
| | Total Commissions | $C_0^{out} = l_0^{decr} \times (Comm_0^{Init} + Comm_0^{Ren})$ |
| | Total Expenses | $E_0^{out} = l_0^{decr} \times (Exp_0^{Init} + Exp_0^{Ren})$ |
| | Death Outgo | $D_0^{out} = d_1^{decr} \times SA_{death}$ |
| | Maturity Outgo | $M_0^{out} = m_1^{decr} \times SA_{mat}$ |
| | Surrender Outgo | $S_0^{out} = w_1 \times SurrVal_1$ |
| ⋮ | | ⋮ |
| ⋮ | | ⋮ |
| t | Premium Income | $P_{t-1}^{in} = l_{t-1}^{decr} \times {}_nB_x$ |
| | Total Commissions | $C_{t-1}^{out} = l_{t-1}^{decr} \times (Comm_{t-1}^{Init} + Comm_{t-1}^{Ren})$ |
| | Total Expenses | $E_{t-1}^{out} = l_{t-1}^{decr} \times (Exp_{t-1}^{Init} + Exp_{t-1}^{Ren})$ |
| | Death Outgo | $D_{t-1}^{out} = d_t^{decr} \times SA_{death}$ |
| | Maturity Outgo | $M_{t-1}^{out} = m_t^{decr} \times SA_{mat}$ |
| | Surrender Outgo | $S_{t-1}^{out} = w_t \times SurrVal_t$ |

Table 1.1: Cash flows of endowment policy

2. Variance reduction methods in Monte Carlo simulations

In the first section of this thesis basic formulas for cash flow projection were presented. All formulas were considered for a single policy and the standard insurance projection is for all policies in portfolio. But the projection of the whole portfolio is just a sum of single projections. The most time consuming part of the BEL calculation is discounting future cash flows of single policy in portfolio by numerous risk-free curves generated by Monte Carlo simulation.

In the second section we will present an introduction to variance reduction methods. We will use them to handle with number of risk free curves to decrease the required time needed for stable mean estimate. These methods are Antithetic variate, Control-variate method and Integrated control-variate.

2.1 Review of confidence intervals for estimating a mean

2.1.1 Basic definitions

When estimating an unknown mean $\mu = E(X)$, we collect n *iid* samples from distribution X_1, \dots, X_n and use the sample mean Sigman (2007).

$$\bar{X}(n) = \frac{1}{n} \sum_{j=1}^n X_j. \quad (2.1)$$

Let $\sigma^2 = \text{Var}(X)$ be the variance of the distribution, then

$$\text{Var}(\bar{X}(n)) = \frac{\sigma^2}{n}. \quad (2.2)$$

CLT asserts that if $n \rightarrow \infty$ the distribution of $\frac{\sqrt{n}}{\sigma}(\bar{X}(n) - \mu)$ tends to $N(0,1)$. Letting Z denote a $N(0,1)$ r.v., for n sufficiently large we conclude, $Z_n \approx Z$ in distribution. Then for any $z \geq 0$,

$$P(|\bar{X}(n) - \mu| > z \frac{\sigma}{\sqrt{n}}) \approx P(|Z| > z) = 2P(Z > z) \quad (2.3)$$

For any α , letting $z_{\frac{\alpha}{2}}$ be such that $P(Z > z) = \alpha/2$, we thus have:

$$P(|\bar{X}(n) - \mu| > z_{\frac{\alpha}{2}} \frac{\sigma}{\sqrt{n}}) \approx \alpha, \quad (2.4)$$

which implies that μ lies inside the interval $\bar{X}(n) \pm z_{\frac{\alpha}{2}}$ with approximate probability $1 - \alpha$ as in Omelka (2018).

In the real simulation the value of σ^2 is not known just as μ is. We instead of σ^2 use its estimate - the sample variance $s^2(n)$ defined by:

$$S_{XX} = S^2(n) = \frac{1}{n-1} \sum_{j=1}^n (X_j - \bar{X})^2. \quad (2.5)$$

Similarly the sample covariance is defined by:

$$S_{XY} = \frac{1}{n-1} \sum_{j=1}^n (X_j - \bar{X}_n)(Y_j - \bar{Y}). \quad (2.6)$$

It can be shown that $s^2(n) \rightarrow \sigma^2$, with probability 1, as $n \rightarrow \infty$ and that $E(s^2(n)) = \sigma^2$, $n \geq 2$ as in Omelka (2018).

Following recursion can be derived and will be useful when implementing a simulation which requires a confidence interval:

$$\begin{aligned} \bar{X}_{n+1} &= \bar{X}_n + \frac{X_{n+1} - \bar{X}_n}{n+1}, \\ S_{n+1}^2 &= \left(1 - \frac{1}{n}\right) S_n^2 + (n+1)(\bar{X}_{n+1} - \bar{X}_n)^2. \end{aligned} \quad (2.7)$$

2.1.2 Monte Carlo simulation application

When simulating Monte Carlo simulation we do not collect the data X_1, \dots, X_n and instead we simulate them. If the number of simulations is high enough for example $n = 100\,000$ CLT can be used for constructing confidence regions.

When simulating the paths we can use the negative correlation among the variables X_1, \dots, X_n or use copies with the same mean but lower variance.

2.2 Antithetic variate method

Antithetic variate method is a method that generally reduces the variance in Monte Carlo simulations. The error correction in the simulated paths has a square root convergence so to obtain the BEL, which is the expected value of the future cash flows, we need a very large number of sample paths.

2.2.1 Underlying principle

Further in the text, we will denote our copies of $X \sim (\mu, \sigma^2)$ as X_i . We will not assume that they are independent. If $n = 2m$ for some $m \geq 1$ then

$$\bar{X}(n) = \frac{1}{2m} \sum_{j=1}^{2m} X_j = \frac{1}{m} \sum_{j=1}^m Y_j = \bar{Y}(m), \quad (2.8)$$

where

$$\begin{aligned} Y_1 &= \frac{X_1 + X_2}{2} \\ Y_2 &= \frac{X_3 + X_4}{2} \\ &\vdots \\ Y_m &= \frac{X_{n-1} + X_n}{2} \\ &\cdot \end{aligned}$$

Both estimators $\bar{X}(n)$ and $\bar{Y}(m)$ for $E(X)$ are identical. Then $Y_i = \frac{X_{2i-1} + X_{2i}}{2}$ denotes a generic Y_i . When computing variance of Y we have

$$\text{Var}(Y) = (1/4)(2\sigma^2 + 2\text{Cov}(X_1, X_2)).$$

In the case when X_1, X_2 are iid, $\text{Cov}(X_1, X_2) = 0$ and $\text{Var}(Y) = \sigma^2/2$. But if $\text{Cov}(X_1, X_2) < 0$ then $\text{Var}(Y) < \sigma^2/2$ and $\text{Var}(\bar{Y}(m)) < \frac{\sigma^2}{n}$, so the variance is reduced. We have to create negative correlation between each pair (X_{2m-1}, X_{2m}) but keep the pair iid so the CLT can be applied.

2.3 Control-variate method

Whereas the antithetic variate method relies upon the existence of a process negatively correlated to the original generating process, the control-variate method is based upon sampling a process positively correlated to the original generating process. The popularity of control-variate method lies in simulating the positively correlated process along with the main process.

2.3.1 Underlying principle

Let Y_i denote the response of interest and X_i denote the control variate observed from the replication i . Control variate $X - i$ follows some distribution with with known mean μ_X and possibly unknown variance σ^2 . We will then assume

following linear regression model for $i = 1, 2, \dots, n$:

$$\begin{aligned} Y_i &= \theta + \beta(X_i - \mu_X) + \varepsilon_i \\ \mathbf{E}[\varepsilon_i] &= 0 \\ \mathbf{Var}[\varepsilon_i] &= \sigma^2 \\ \varepsilon_i &\text{ is independent of } \varepsilon_j \text{ for } i \neq j \end{aligned} \quad , \quad (2.9)$$

For each trial the outcome of X_i is calculated along with the output of θ_i . Further we will suppose, that pairs (X_i, Y_i) , $i = 1, \dots, n$ are iid. Our goal will be to estimate the mean of Y_i .

$$\mathbf{E}[Y_i] = \theta$$

Optimizing the parameter β which will minimize the variance of θ gives its optimal value:

$$\beta^* = \frac{\sigma_Y}{\sigma_X} \rho_{XY} = \frac{\mathbf{Cov}(X, Y)}{\mathbf{Var}(X)} \quad (2.10)$$

and θ is estimated by the control-variate estimator Wei-Ning a Wei-Wen (1996):

$$\hat{\theta}_C(\beta^*) = \bar{Y} - \beta^*(\bar{X} - \mu_X) \quad (2.11)$$

Lemma 1. (*Lidebrandt (2007)*) *Control variate estimator 2.11, is unbiased and consistent.*

Proof. The expected value of 2.11 yields the unbiasedness

$$\mathbf{E}(\hat{\theta}_C(\beta^*)) = \mathbf{E}(\bar{Y}) - \mathbf{E}(\beta^*(\bar{X} - \mu_X)) = \mathbf{E}\left(\frac{1}{n} \sum_{i=1}^n Y_i\right) = \mathbf{E}(\theta) = \theta$$

Above equation holds both conditionally on $\mathbf{X} = (X_1, X_2, \dots, X_n)$ and unconditionally. The following limit guarantees consistency

$$\lim_{n \rightarrow \infty} \frac{1}{n} \sum_{i=1}^n Y(i) = \lim_{n \rightarrow \infty} \frac{1}{n} \sum_{i=1}^n (Y_i - \beta^*(X_i - \mu_X)) = \mathbf{E}(Y - \beta^*(X - \mu_X)) = \mathbf{E}(Y),$$

as in Glasserman (2004). The resulting variance for the control variate estimator is Lidebrandt (2007)

$$\mathbf{Var}(\hat{\theta}_C(\beta^*)) = \mathbf{Var}(\bar{\varepsilon}) = \frac{1}{n}(\sigma_Y^2 + \beta^2 \sigma_X^2 - 2\beta \rho_{XY} \sigma_X \sigma_Y) = \frac{(1 - \rho_{XY}^2) \sigma_Y^2}{n}. \quad (2.12)$$

The last equality follows from 2.10. In most practical applications σ_{XY} is unknown and β^* has to be estimated by its sample counterpart

$$\hat{\beta}^* = \frac{S_{XY}}{S_{XX}}.$$

The control variate estimator then becomes

$$\hat{\theta}_C(\hat{\beta}^*) = \bar{Y} - \hat{\beta}^*(\bar{X} - \mu_X). \quad (2.13)$$

Unbiasedness holds even for $\hat{\theta}_C(\hat{\beta}^*)$ from 2.13 (see proof of the Theorem 3.2(i) in Nelson (1988)). Conditional variance of $\hat{\theta}_C(\hat{\beta}^*)$ is again as in Nelson (1988):

$$\text{Var}(\hat{\theta}_C(\hat{\beta}^*)|\mathbf{X}) = \left\{ 1 + \frac{n}{n-1} \left(\frac{(\bar{X} - \mu_X)^2}{S_{XX}} \right) \right\} \cdot \frac{(1 - \rho_{XY}^2)\sigma_Y^2}{n}, \quad (2.14)$$

which leads to the unconditional

$$\text{Var}(\hat{\theta}_C(\hat{\beta}^*)) = \left\{ 1 + \frac{n}{n-1} \mathbb{E} \left(\frac{(\bar{X} - \mu_X)^2}{S_{XX}} \right) \right\} \cdot \frac{(1 - \rho_{XY}^2)\sigma_Y^2}{n}. \quad (2.15)$$

The difference between equation 2.12 and 2.15 is called the loss factor:

$$\left\{ 1 + \frac{n}{n-1} \mathbb{E} \left[\frac{(\bar{X} - \mu_X)^2}{S_{XX}} \right] \right\}.$$

The loss factor represents the loss of efficiency when estimating β^* . In the equation 2.15, both ρ_{XY} and σ_Y^2 are determined by the joint distribution of X_i and Y_i and should not be altered. Nevertheless, we can alter the joint distribution of \mathbf{X} to improve the performance of the control-variate estimator. From the Law (2014), we have:

$$\mathbb{E}[(\bar{X} - \mu_X)^2] = \text{Var}[\bar{X}] = \frac{\sigma_X^2}{n} \cdot \left[1 + 2 \sum_{j=1}^{n-1} \left(1 - \frac{j}{n} \right) \rho_j \right] \quad (2.16)$$

and

$$\mathbb{E}[S_{XX}] = \sigma_X^2 \cdot \left[1 - \frac{2}{n-1} \sum_{j=1}^{n-1} \left(1 - \frac{j}{n} \right) \rho_j \right]. \quad (2.17)$$

The correlation between X_i and X_{i+j} is denoted as ρ_j , so possible negative correlations among X_1, X_2, \dots, X_n decreases the term 2.16 and increases the term 2.17, which lowers the loss factor. Conditional on the control-variate the error terms in model 2.9 remain uncorrelated and the variance expression 2.15 stays valid.

2.3.2 Multiple Control Variates

It is possible to put together several control-variates into one control-variate, however, it is not always possible. Previous formulas can be easily extended to the multiple control-variates. We will denote $\mathbf{X} = (X_1, \dots, X_q)$ a vector of control variates with known expected values corresponding to $\mu_{\mathbf{X}} = (\mu_1, \dots, \mu_q)$ then model 2.9 becomes:

$$\begin{aligned} Y_i &= \theta + \beta'(\mathbf{X}_i - \mu_{\mathbf{X}}) + \varepsilon_i \\ \mathbb{E}[\varepsilon_i] &= 0 \\ \text{Var}[\varepsilon_i] &= \sigma^2 \\ \varepsilon_i &\text{ is independent of } \varepsilon_j \text{ for } i \neq j \end{aligned}, \quad (2.18)$$

where $\boldsymbol{\beta} = (\beta_1, \beta_2, \dots, \beta_q)'$ and $\mathbf{X}_i = (X_{i,1}, X_{i,2}, \dots, X_{i,q})'$ where $X_{i,j}$ represents control variate j observed in replication i , following a distribution with again known mean $\boldsymbol{\mu}_X = (\mu_1, \mu_2, \dots, \mu_q)'$ and possibly unknown covariance $\boldsymbol{\Sigma}_{XX}$ and is independent of the error ε_i . Estimating the control vector $\boldsymbol{\beta}$ by the LSE gives similar to the univariate version:

$$\hat{\boldsymbol{\beta}}^* = \mathbb{S}_{XX}^{-1} \mathbb{S}_{YX},$$

where \mathbb{S}_{XX} and \mathbb{S}_{XY} are the sample counterparts of the \mathcal{K}_{XX} and \mathcal{K}_{XY} .

$$\mathbb{S}_{XY} = \sum_{i=1}^n \frac{(\mathbf{X}_i - \bar{\mathbf{X}})(Y_i - \bar{Y})}{n-1}$$

,

$$\mathbb{S}_{XX} = \sum_{i=1}^n \frac{(\mathbf{X}_i - \bar{\mathbf{X}})(\mathbf{X}_i - \bar{\mathbf{X}})'}{n-1}$$

,

$$\bar{Y} = \sum_{i=1}^n \frac{Y_i}{n},$$

$$\bar{\mathbf{X}} = \sum_{i=1}^n \frac{\mathbf{X}_i}{n},$$

. This results in the control-variate estimator

$$\hat{\theta}_C(\hat{\boldsymbol{\beta}}^*) = \bar{Y} - \hat{\boldsymbol{\beta}}^{*'} (\bar{\mathbf{X}} - \boldsymbol{\mu}_X). \quad (2.19)$$

This is unbiased and consistent estimator of θ . The proof is just straight forward extension of the one dimensional case as in lemma 1. The variance of the estimator is:

$$\text{Var}[\hat{\theta}_C(\hat{\boldsymbol{\beta}}^*)] = \left\{ 1 + \frac{n}{n-1} E[(\bar{\mathbf{X}} - \boldsymbol{\mu}_X)' \mathbb{S}_{XX}^{-1} (\bar{\mathbf{X}} - \boldsymbol{\mu}_X)] \right\} \cdot \frac{(1 - \mathbf{R}_{XY}^2) \sigma_Y^2}{n}, \quad (2.20)$$

where \mathbf{R}_{XY}^2 is the multiple correlation coefficient between the response Y_i and the q -variate control variate \mathbf{X}_i .

Analogously to the univariate case, when inducing negative correlations among $\mathbf{X}_1, \mathbf{X}_2, \dots, \mathbf{X}_n$, one can expect reduction in the loss factor and reduction of the control-variate estimator variance, if the error terms in model 2.18 are conditionally uncorrelated.

2.4 Integrated control-variate estimator

Both presented variance reduction methods can be merged together. We will apply the antithetic variates on the control variates to induce negative correlations in the control-variate joint distribution between each pair of replication. We will keep the joint distribution intact within each replication. The result will be unbiased control-variate estimator with smaller variance thanks to the antithetic variates.

2.4.1 Underlying principle

To assess the control-variate estimator efficiency. We will assume that (Y_i, \mathbf{X}_i) has a $(q + 1)$ -variate normal distribution. This is reasonable because with high number of simulations the response and the control variates are jointly subject to a central limit effect.

$$\begin{aligned} \begin{pmatrix} Y_i \\ \mathbf{X}_i \end{pmatrix} \stackrel{i.i.d}{\sim} N_{q+1} \left[\begin{pmatrix} \theta \\ \boldsymbol{\mu}_X \end{pmatrix}, \begin{pmatrix} \sigma_Y^2 & \sigma_{YX} \\ \sigma_{YX} & \boldsymbol{\Sigma}_{XX} \end{pmatrix} \right] \\ , i = 1, 2, \dots, n, \end{aligned}$$

where $\sigma_{YX} = (\sigma_{YX_1}, \sigma_{YX_2}, \dots, \sigma_{YX_n})$. This condition is also sufficient condition for our model 2.18 to hold. In the standard control variate estimator control variates across the replication form a random sample. As mention earlier when inducing negative correlations between \mathbf{X}_{2i} and \mathbf{X}_{2i-2} , $i = 1, 2, \dots, n/2$. When model 2.18 holds the integrated control variate estimator remains unbiased when applying the antithetic variates on control variates. The unbiasedness follows from the proof of Theorem 3.2 in Nelson (1988). This means that despite the dependence of $\mathbf{X}_1, \mathbf{X}_2, \dots, \mathbf{X}_n$ that:

$$E[\hat{\theta}_C(\hat{\beta}) | \mathbf{X}_1, \mathbf{X}_2, \dots, \mathbf{X}_n] = \theta$$

and this leads to

$$E[\hat{\theta}_C(\hat{\beta})] = E[\theta] = \theta.$$

Furthermore, dependence induced from applying the antithetic variate on the control-variate may affect the error term in the assumed linear regression model. To assess the variance of the integrated control-variance estimator we need to assume, that conditional on the control variates, error terms remain uncorrelated across replication. With this assumption the integrated control-variate estimator preserves the variance as in 2.20 as shown in Nelson (1988). We will further assume following adjusted model:

$$\begin{aligned} Y_i &= \theta + \boldsymbol{\beta}'(\mathbf{X}_i - \boldsymbol{\mu}_X) + \varepsilon_i \\ E[\varepsilon_i] &= 0 \\ \text{Var}[\varepsilon_i] &= \sigma^2 \\ \varepsilon_i &\text{ is independent of } \mathbf{X}_i \\ \mathbf{X}_i &\sim N_q(\boldsymbol{\mu}_X, \boldsymbol{\Sigma}_{XX}) \\ \text{Cov}[\varepsilon_i, \varepsilon_j | \mathbf{X}_1, \mathbf{X}_2, \dots, \mathbf{X}_n] &= 0 \text{ for } i \neq j \end{aligned} \tag{2.21}$$

Following theorem proves that if we integrate the antithetic variate on the control-variate this way we get a lower variance of the control-variate estimator Wei-Ning a Wei-Wen (1996).

Theorem 1. *If model 2.21 holds and the individual control variates paired across the replications follow a bivariate normal distribution with common correlation, ρ , for all control variates, then $\text{Var}[\hat{\theta}_c(\hat{\beta})]$ is a nondecreasing function in ρ .*

Proof. When our model 2.21 holds, applying antithetic variates across paired replications does not change σ_Y^2 and $\mathbf{R}_{X,Y}^2$, and the integrated control variates preserves the variance expression 2.20. As mentioned earlier it is sufficient to show that the loss factor is a nondecreasing function in ρ . Assuming that model 2.21 holds and the individual paired control variates across paired replications follow a bivariate normal distribution with common correlation, ρ , for all control variates, we have:

$$\begin{aligned} \begin{pmatrix} X_{2i-1,j} \\ X_{2i,j} \end{pmatrix} &\sim N_2 \left[\begin{pmatrix} \mu_j \\ \mu_j \end{pmatrix}, \sigma_j^2 \begin{pmatrix} 1 & \rho \\ \rho & 1 \end{pmatrix} \right] \\ &, i = 1, 2, \dots, n/2, \\ &, j = 1, 2, \dots, q, \end{aligned}$$

where $\sigma_j^2 = \text{Var}(X_{i,j})$. By principal axis transformation Morrison (1976),

$$\begin{pmatrix} Z_{2i-1,j} \\ Z_{2i,j} \end{pmatrix} = \begin{pmatrix} \frac{1}{\sqrt{2}} & -\frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \end{pmatrix} \begin{pmatrix} X_{2i-1,j} - \mu_j \\ X_{2i,j} - \mu_j \end{pmatrix} = \begin{pmatrix} \frac{1}{\sqrt{2}}(X_{2i-1,j} - X_{2i,j}) \\ \frac{1}{\sqrt{2}}(X_{2i-1,j} + X_{2i,j} - 2\mu_j) \end{pmatrix} \quad (2.22)$$

we have,

$$\begin{pmatrix} Z_{2i-1,j} \\ Z_{2i,j} \end{pmatrix} \sim N_2 \left[\mathbf{0}_2, \sigma_j^2 \begin{pmatrix} (1-\rho) & 0 \\ 0 & (1+\rho) \end{pmatrix} \right] \quad (2.23)$$

$$, i = 1, 2, \dots, n/2, \quad (2.24)$$

$$, j = 1, 2, \dots, q, \quad (2.25)$$

where $\mathbf{0}_q$ denotes a q-variate zero column vector. We define following:

$$\mathbf{Z}_i = (Z_{i,1}, Z_{i,2}, \dots, Z_{i,z})',$$

and

$$\bar{\mathbf{Z}}_2 = \frac{2}{n} \sum_{i=1}^{n/2} \mathbf{Z}_{2i},$$

$$\begin{cases} \mathbf{Z}_{2i-1} = \frac{1}{\sqrt{2}}(\mathbf{X}_{2i-1} - \mathbf{X}_{2i}) \\ \mathbf{Z}_{2i} = \frac{1}{\sqrt{2}}(\mathbf{X}_{2i-1} + \mathbf{X}_{2i} - 2\mu_X) \end{cases} \quad i = 1, 2, \dots, n/2 \quad (2.26)$$

yielding

$$\begin{cases} \mathbf{X}_{2i-1} = \frac{1}{\sqrt{2}}(\mathbf{Z}_{2i-1} + \mathbf{Z}_{2i}) + \mu_X \\ \mathbf{X}_{2i} = \frac{1}{\sqrt{2}}(\mathbf{Z}_{2i-1} - \mathbf{Z}_{2i}) + \mu_X \end{cases} \quad i = 1, 2, \dots, n/2 \quad (2.27)$$

and

$$\bar{\mathbf{X}} - \mu_X = \frac{1}{\sqrt{2}} \bar{\mathbf{Z}}_2. \quad (2.28)$$

From equations 2.28 and 2.26 we get following distributions:

$$\begin{cases} \{\mathbf{X}_{2i-1}\}_{i=1}^{n/2} \stackrel{i.i.d}{\sim} N_q(\mathbf{0}_q, (1-\rho)\Sigma_{\mathbf{X}\mathbf{X}}), \\ \{\mathbf{X}_{2i}\}_{i=1}^{n/2} \stackrel{i.i.d}{\sim} N_q(\mathbf{0}_q, (1+\rho)\Sigma_{\mathbf{X}\mathbf{X}}), & i = 1, 2, \dots, n/2 \\ \bar{\mathbf{Z}}_2 \sim N_q(\mathbf{0}_q, \frac{2}{n}(1+\rho)\Sigma_{\mathbf{X}\mathbf{X}}) \end{cases}$$

implying

$$\begin{cases} \mathbf{L}_i = \sqrt{\frac{1}{(1-\rho)}} \mathbf{Z}_{2i-1} \stackrel{i.i.d}{\sim} N_q(\mathbf{0}_q, \Sigma_{\mathbf{X}\mathbf{X}}), \\ \mathbf{P}_i = \sqrt{\frac{1}{(1+\rho)}} \mathbf{Z}_{2i} \stackrel{i.i.d}{\sim} N_q(\mathbf{0}_q, \Sigma_{\mathbf{X}\mathbf{X}}), & i = 1, 2, \dots, n/2 \\ \mathbf{C} = \sqrt{\frac{n}{2(1+\rho)}} \bar{\mathbf{Z}}_2 \sim N_q(\mathbf{0}_q, \Sigma_{\mathbf{X}\mathbf{X}}) \end{cases}$$

It can be seen that the distributions of \mathbf{L}_i , \mathbf{P}_i and \mathbf{C} are not functions of ρ . From functions 2.27 and 2.28 we get:

$$\begin{aligned} \mathbb{S}_{\mathbf{X}\mathbf{X}} &= \frac{1}{n-1} \sum_{i=1}^n (\mathbf{X}_i - \bar{\mathbf{X}})(\mathbf{X}_i - \bar{\mathbf{X}})' \\ &= \frac{1}{n-1} \left[\sum_{i=1}^{n/2} (\mathbf{X}_{2i-1} - \bar{\mathbf{X}})(\mathbf{X}_{2i-1} - \bar{\mathbf{X}})' + \sum_{i=1}^{n/2} (\mathbf{X}_{2i} - \bar{\mathbf{X}})(\mathbf{X}_{2i} - \bar{\mathbf{X}})' \right] \\ &= \frac{1}{n-1} \cdot \frac{1}{2} \left[\sum_{i=1}^{n/2} (\mathbf{Z}_{2i-1} + \mathbf{Z}_{2i} - \bar{\mathbf{Z}}_2)(\mathbf{Z}_{2i-1} + \mathbf{Z}_{2i} - \bar{\mathbf{Z}}_2)' \right. \\ &\quad \left. + \sum_{i=1}^{n/2} (-\mathbf{Z}_{2i-1} + \mathbf{Z}_{2i} - \bar{\mathbf{Z}}_2)(-\mathbf{Z}_{2i-1} + \mathbf{Z}_{2i} - \bar{\mathbf{Z}}_2)' \right] \tag{2.29} \\ &= \frac{1}{n-1} \cdot \frac{1}{2} \left[2 \sum_{i=1}^{n/2} \mathbf{Z}_{2i-1} \mathbf{Z}_{2i-1}' + 2 \sum_{i=1}^{n/2} \mathbf{Z}_{2i} \mathbf{Z}_{2i}' \right. \\ &\quad \left. - 2 \bar{\mathbf{Z}}_2 \sum_{i=1}^{n/2} \mathbf{Z}_{2i}' - 2 \sum_{i=1}^{n/2} \mathbf{Z}_{2i} \bar{\mathbf{Z}}_2' + n \bar{\mathbf{Z}}_2 \bar{\mathbf{Z}}_2' \right] \\ &= \frac{1}{n-1} \left\{ \left[2 \sum_{i=1}^{n/2} \mathbf{Z}_{2i} \mathbf{Z}_{2i}' - \frac{n}{2} \bar{\mathbf{Z}}_2 \bar{\mathbf{Z}}_2' \right] + \sum_{i=1}^{n/2} \mathbf{Z}_{2i-1} \mathbf{Z}_{2i-1}' \right\} \end{aligned}$$

from the 2.28 and 2.29, we have:

$$\begin{aligned} &\frac{n}{n-1} (\bar{\mathbf{X}} - \boldsymbol{\mu}_X)' \mathbb{S}_{\mathbf{X}\mathbf{X}}^{-1} (\bar{\mathbf{X}} - \boldsymbol{\mu}_X) \\ &= \frac{n}{2} \bar{\mathbf{Z}}_2' \left\{ \left[2 \sum_{i=1}^{n/2} \mathbf{Z}_{2i} \mathbf{Z}_{2i}' - \frac{n}{2} \bar{\mathbf{Z}}_2 \bar{\mathbf{Z}}_2' \right] + \sum_{i=1}^{n/2} \mathbf{Z}_{2i-1} \mathbf{Z}_{2i-1}' \right\}^{-1} \bar{\mathbf{Z}}_2 \\ &= \frac{n}{2} \bar{\mathbf{Z}}_2' \left\{ (1+\rho) \left[2 \sum_{i=1}^{n/2} \mathbf{P}_i \mathbf{P}_i' - \frac{n}{2} \bar{\mathbf{P}} \bar{\mathbf{P}}' \right] + (1-\rho) \sum_{i=1}^{n/2} \mathbf{L}_i \mathbf{L}_i' \right\}^{-1} \bar{\mathbf{Z}}_2 \\ &= (1+\rho) \mathbf{C}' [(1+\rho) \mathbf{W}_1 + (1-\rho) \mathbf{W}_2]^{-1} \mathbf{C} \\ &= \mathbf{C}' \left[\mathbf{W}_1 + \frac{(1-\rho)}{(1+\rho)} \mathbf{W}_2 \right]^{-1} \mathbf{C} = L(p), \end{aligned}$$

where $\bar{\mathbf{P}} = \frac{2}{n} \sum_{i=1}^{n/2} \mathbf{P}_i$, and

$$\begin{aligned}\mathbf{W}_1 &= \left[\sum_{i=1}^{n/2} \mathbf{P}_i \mathbf{P}_i' - \frac{n}{2} \bar{\mathbf{P}} \bar{\mathbf{P}}' \right] \sim \text{Wishart}_q((n/2 - 1), \boldsymbol{\Sigma}_{\mathbf{X}\mathbf{X}}) \\ \mathbf{W}_2 &= \sum_{i=1}^{n/2} \mathbf{L}_i \mathbf{L}_i' \sim \text{Wishart}_q(n/2, \boldsymbol{\Sigma}_{\mathbf{X}\mathbf{X}})\end{aligned}$$

where $\text{Wishart}_p(d, \boldsymbol{\Sigma})$ represents the p -dimensional Wishart distribution with d degrees of freedom and the covariance matrix $\boldsymbol{\Sigma}$. Because the distributions of \mathbf{C} , \mathbf{W}_1 and \mathbf{W}_2 does not depend on ρ , to show that the loss factor is nondecreasing function in ρ , we only need to show that for fixed \mathbf{C} , \mathbf{W}_1 and \mathbf{W}_2 , $L(\rho_1) \leq L(\rho_2)$ if $\rho_1 < \rho_2$. Since the matrix

$$\left(\mathbf{W}_1 + \frac{(1 - \rho_1)}{(1 + \rho_1)} \mathbf{W}_2 \right) - \left(\mathbf{W}_1 + \frac{(1 - \rho_2)}{(1 + \rho_2)} \mathbf{W}_2 \right) = \frac{2(\rho_2 - \rho_1)}{(1 + \rho_1)(1 + \rho_2)} \mathbf{W}_2$$

is positive definite for $\rho_1 < \rho_2$ (see Corollary 7.7.4 in Horn and Johnson (1985)) implies that the matrix:

$$\left(\mathbf{W}_1 + \frac{(1 - \rho_1)}{(1 + \rho_1)} \mathbf{W}_2 \right)^{-1} - \left(\mathbf{W}_1 + \frac{(1 - \rho_2)}{(1 + \rho_2)} \mathbf{W}_2 \right)^{-1}$$

is positive semidefinite. So we have

$$\begin{aligned}L(\rho_1) - L(\rho_2) &= \mathbf{C}' \left(\mathbf{W}_1 + \frac{(1 - \rho_1)}{(1 + \rho_1)} \mathbf{W}_2 \right)^{-1} \mathbf{C} - \mathbf{C}' \left(\mathbf{W}_1 + \frac{(1 - \rho_2)}{(1 + \rho_2)} \mathbf{W}_2 \right)^{-1} \mathbf{C} \\ &= \mathbf{C}' \left\{ \left(\mathbf{W}_1 + \frac{(1 - \rho_1)}{(1 + \rho_1)} \mathbf{W}_2 \right)^{-1} - \left(\mathbf{W}_1 + \frac{(1 - \rho_2)}{(1 + \rho_2)} \mathbf{W}_2 \right)^{-1} \right\} \mathbf{C}\end{aligned}$$

□

Consequence of the 1 is that under the assumption mentioned earlier, applying antithetic variate on control-variate across paired replications lowers the variance of control-variate estimator compared to the conventional control-variate estimator.

2.4.2 Integrated Control-Variate alternative I

When describing the alternative integrated control-variate estimators we will work only with the univariate case of the control-variate model for $i = 1, 2, \dots, n$:

$$\begin{aligned}Y_i &= \theta + \beta'(X_i - \mu_X) + \varepsilon_i \\ \mathbb{E}[\varepsilon_i] &= 0 \\ \text{Var}[\varepsilon_i] &= \sigma^2 \\ \varepsilon_i &\text{ is independent of } X_i \\ X_i &\sim N(\mu_X, \Sigma_{XX}) \\ \text{Cov}[\varepsilon_i, \varepsilon_j | X_1, X_2, \dots, X_n] &= 0 \text{ for } i \neq j\end{aligned}\tag{2.30}$$

The integrated method developed by Kwon a Tew (1993) works with the integrated control-variate estimator, where they firstly combine the paired replications and then construct a conventional control-variate estimator based on $n/2$ independent paired replications. For the integrated control-variate method it holds

$$\begin{cases} Y_{2i-1} = \theta + \beta(X_{2i-1} - \mu_X) + \varepsilon_{2i-1} \\ Y_{2i} = \theta + \beta(X_{2i} - \mu_X) + \varepsilon_{2i} \end{cases} \quad i = 1, 2, \dots, n/2,$$

after which we get

$$Y_i^* = \theta + \beta(X_i^* - \mu_X) + \varepsilon_i^*, \quad i = 1, 2, \dots, n/2. \quad (2.31)$$

Components of the above equation are defined

$$\begin{cases} Y_i^* = \frac{Y_{2i-1} + Y_{2i}}{2} \\ X_i^* = \frac{X_{2i-1} + X_{2i}}{2} \\ \varepsilon_i^* = \frac{\varepsilon_{2i-1} + \varepsilon_{2i}}{2} \\ \varepsilon_i^* \text{ and } X_i^* \text{ are independent.} \end{cases} \quad i = 1, 2, \dots, n/2$$

Correlation coefficient between X_i^* and Y_i^* is

$$\rho_{X^*Y^*}^2 = \frac{\text{Cov}(X_i^*, Y_i^*)^2}{\text{Var}(X_i^*)\text{Var}(Y_i^*)} = \frac{\beta^2 \text{Var}(X_i^*)}{\text{Var}(Y_i^*)}.$$

Estimated control-variate estimator proposed by Kwon a Tew (1993) is

$$\begin{aligned} \hat{\theta}_{KW} &= \bar{Y}^* - \hat{\beta}_{KW}^* (\bar{X}^* - \mu_X) \\ \bar{Y}^* &= \frac{2}{n} \sum_{i=1}^{n/2} Y_i^* \\ \bar{X}^* &= \frac{2}{n} \sum_{i=1}^{n/2} X_i^* \\ \hat{\beta}_{KW}^* &= \frac{\sum_{i=1}^{n/2} (X_i^* - \bar{X}^*) (Y_i^* - \bar{Y}^*)}{\sum_{i=1}^{n/2} (X_i^* - \bar{X}^*)^2} \end{aligned}$$

The variance of control-variate estimator constructed as above is from the equation 2.20

$$\begin{aligned} \text{Var}(\bar{\theta}_{KW}) &= \left(\frac{n/2 - 2}{n/2 - 3} \right) \frac{(1 - \rho_{X^*Y^*}^2) \text{Var}(Y_i^*)}{n/2} \\ &= \left(\frac{n/2 - 2}{n/2 - 3} \right) \left[1 - \frac{\beta^2 \text{Var}(X_i^*)}{\text{Var}(Y_i^*)} \right] \frac{\text{Var}(Y_i^*)}{n/2} \\ &= \left(\frac{n/2 - 2}{n/2 - 3} \right) \frac{[\text{Var}(Y_i^*) - \beta^2 (\text{Var}(X_i^*))]}{n/2}, \\ &= \left(\frac{n/2 - 2}{n/2 - 3} \right) \frac{\text{Var}(\varepsilon_i^*)}{n/2} = \left(\frac{n/2 - 2}{n/2 - 3} \right) \frac{\text{Var}(\varepsilon_i)}{n} \\ &= \left(\frac{n/2 - 2}{n/2 - 3} \right) \frac{(1 - \rho_{XY}^2) \text{Var}(Y_i)}{n} > \left(\frac{n - 2}{n - 3} \right) \frac{(1 - \rho_{XY}^2) \text{Var}(Y_i)}{n} \end{aligned}$$

which shows that the performance of the integrated control-variate estimator proposed by Kwon a Tew (1993) is worse than the control-variate estimator without antithetic variates.

2.4.3 Integrated Control-Variates alternative II

The second intuitive integrated control-variate estimator developed as in Wei-Ning a Wei-Wen (1996) exchanges the steps of control-variate and anti-thetic variate usage compared to the previous integrated control-variate method. So there are firstly individual control-variate estimators created based on odd-numbered and even-numbered replications and then the average of these two conventional control-variate estimators is taken as the integrated control-variate estimator. The performance of this integrated control-variate estimator is the same as the previous alternative. If we denote θ_1 and θ_2 the control-variate estimators based on odd-numbered and even-numbered replications, then following holds

$$\begin{cases} \hat{\theta}_1 = \bar{Y}_1 - \hat{\beta}_1(\bar{X}_1 - \mu_x), \\ \hat{\theta}_2 = \bar{Y}_2 - \hat{\beta}_2(\bar{X}_2 - \mu_x), \\ \hat{\theta}_{12} = \frac{\hat{\beta}_1 + \hat{\beta}_2}{2}, \end{cases}$$

where

$$\begin{aligned} \bar{Y}_1 &= \frac{2}{n} \sum_{i=1}^{n/2} Y_{2i-1}, & \bar{Y}_2 &= \frac{2}{n} \sum_{i=1}^{n/2} Y_{2i}, \\ \bar{X}_1 &= \frac{2}{n} \sum_{i=1}^{n/2} X_{2i-1}, & \bar{X}_2 &= \frac{2}{n} \sum_{i=1}^{n/2} X_{2i}, \\ \hat{\beta}_1 &= \frac{\sum_{i=1}^{n/2} (Y_{2i-1} - \bar{Y}_1)(X_{2i-1} - \bar{X}_1)}{\sum_{i=1}^{n/2} (X_{2i-1} - \bar{X}_1)^2}, & & \frac{\sum_{i=1}^{n/2} (Y_{2i} - \bar{Y}_1)(X_{2i} - \bar{X}_1)}{\sum_{i=1}^{n/2} (X_{2i} - \bar{X}_1)^2}. \end{aligned}$$

Since, conditional on $\mathbf{X} = \{X_1, X_2, \dots, X_n\}$, $\varepsilon_1, \varepsilon_2, \dots, \varepsilon_n$ are uncorrelated under the assumed model 2.21, which means

$$\text{Cov}(\hat{\theta}_1, \hat{\theta}_2) = \text{E}[\text{Cov}(\hat{\theta}_1, \hat{\theta}_2 | \mathbf{X})] + \text{Cov}[\text{E}(\hat{\theta}_1 | \mathbf{X}), \text{E}(\hat{\theta}_2 | \mathbf{X})] = 0,$$

which yields again from the equations 2.20 and ??, the variance of the integrated control variate estimator Wei-Ning a Wei-Wen (1996)

$$\begin{aligned} \text{Var}(\hat{\theta}_{12}) &= \frac{1}{2} \text{Var}(\hat{\theta}_1) = \frac{1}{2} \left(\frac{n/2 - 2}{n/2 - 3} \right) \frac{\text{Var}(\varepsilon_{2i-1})}{n/2} \\ &= \left(\frac{n/2 - 2}{n/2 - 3} \right) \frac{(1 - \rho_{XY}^2) \text{Var}(Y_i)}{n} > \left(\frac{n - 2}{n - 3} \right) \frac{(1 - \rho_{XY}^2) \text{Var}(Y_i)}{n} \end{aligned}$$

2.4.4 Estimate of the Control variate estimator variance

For our purposes we will need to estimate the Control variate estimator variance. As proposed by Nelson (1988) we can used the estimate in the form

$$S_{CV}^2 = S_{YY}^2 \left(\frac{1}{n} + \frac{1}{n-1} (\bar{\mathbf{X}} - \mu_{\mathbf{X}})' S_{XX}^{-1} (\bar{\mathbf{X}} - \mu_{\mathbf{X}}) \right), \quad (2.32)$$

where

$$S_{YY}^2 = \frac{1}{(n - q - 1)} \sum_{i=1}^n (Y_i - (\mathbf{X}_i - \boldsymbol{\mu}_X)' \hat{\boldsymbol{\beta}} - \hat{\theta}_C(\hat{\boldsymbol{\beta}}^*)).$$

The equation 2.32 could cause a great increase in the time needed to calculate all scenarios due to the complexity. We will have to use a recursion formula similar to the variance recursion formula in 2.7 for the term S_{YY}^2 .

$$S_{YY,n+1}^2 = \left(1 - \frac{1}{n - q}\right) S_{YY,n}^2 + (n - q + 1) \left[\hat{\theta}_C(\hat{\boldsymbol{\beta}}^*)_{n+1} - \hat{\theta}_C(\hat{\boldsymbol{\beta}}^*)_n \right]^2,$$

where $\hat{\theta}_C(\hat{\boldsymbol{\beta}}^*)_n$ is calculated as in 2.19.

$$\hat{\theta}_C(\hat{\boldsymbol{\beta}}^*)_n = \bar{Y}_n - \hat{\boldsymbol{\beta}}^{*'} (\bar{\mathbf{X}}_n - \boldsymbol{\mu}_X). \quad (2.33)$$

Both mean terms on the right hands side of the equation 2.33 are again recursively calculated similarly to 2.7. Estimate of the $\boldsymbol{\beta}^*$ is calculated after preselected number of scenarios and holds for the rest of the projection. The term $\boldsymbol{\mu}_X$ is known.

3. Market scenarios

3.1 Short term interest rate models

For calculations purposes we will need to generate interest rate curves so it is possible to discount future cash flows. To simulate them we will use short term interest rate model - Hull-White model. Hull-White model is extension of Vasicek model. Dynamics of Hull-White model is described by the following stochastic differential equation:

$$dr_t = (\theta(t) - \alpha r_t)dt + \sigma dW_t, \quad (3.1)$$

where α and σ are constants. Function $\theta(t)$ is chosen in order to fit the input term structure of interest rates. W_t is Wiener process which is defined as in F.Mercurio (2006):

1. $W_0 = 0$;
2. W has continuous paths a.s.;
3. for any time t_i , $i = 1, \dots, m$ for which $0 = t_0 < t_1 < \dots < t_m$, the increments $W(t_1) - W(t_0), \dots, W(t_m) - W(t_{m-1})$ are independent;
4. $W(t+u) - W(t) \sim N(0, u)$.

In order to match the input term structure, $\theta(t)$ needs to be set as follows:

$$\theta(t) = \frac{\partial f^M(0, t)}{\partial T} + \alpha f^M(0, t) + \frac{\sigma^2}{2\alpha}(1 - e^{-2\alpha t}) \quad (3.2)$$

, where $f^M(0, t)$ represents market instantaneous forward rate. When integrating equation 3.2 similarly to F.Mercurio (2006) or Kuzminskaya (2018) we get:

$$\begin{aligned} r(t) &= r(s)e^{\alpha(t-s)} + \int_s^t e^{-\alpha(t-u)}\theta(u)du + \sigma \int_s^t e^{-\alpha(t-u)}dW(u) = \\ &= r(s)e^{\alpha(t-s)} + \varphi(t) - \varphi(s)e^{-\alpha(t-s)} + \sigma \int_s^t e^{-\alpha(t-u)}dW(u) \end{aligned} \quad (3.3)$$

where

$$\varphi(t) = f^M(0, t) + \frac{\sigma^2}{2\alpha^2}(1 - e^{-\alpha t})^2 \quad (3.4)$$

. The short rate $r(t)$ is given a normal distribution with the properties (Kuzminskaya (2018), F.Mercurio (2006)):

$$\begin{aligned} \mathbb{E}[r(t) | \mathcal{F}_0] &= f^M(0, t) + \frac{\sigma^2}{2\alpha^2}(1 - e^{-\alpha t})^2 \\ \text{Var}[r(t) | \mathcal{F}_0] &= \frac{\sigma^2}{2\alpha}(1 - e^{-\alpha t}) \end{aligned} \quad (3.5)$$

The input for short rate models calibration has to be continuous function of time to maturity. Very popular econometric model that interpolates yield to maturity is Nelson-Siegel-Svensson model. The yield curve is described by the function Hakala (2017):

$$R^M(0, T) = \beta_0 + \beta_1 \frac{1 - \exp\{-\frac{T}{\gamma_1}\}}{\frac{T}{\gamma_1}} + \beta_2 \left(\frac{1 - \exp\{-\frac{T}{\gamma_1}\}}{\frac{T}{\gamma_1}} - \exp\left\{-\frac{T}{\gamma_1}\right\} \right) + \beta_3 \left(\frac{1 - \exp\{-\frac{T}{\gamma_2}\}}{\frac{T}{\gamma_2}} - \exp\left\{-\frac{T}{\gamma_2}\right\} \right), \quad (3.6)$$

where $\beta_0, \beta_1, \beta_2, \beta_3, \gamma_1, \gamma_2$ are constants. Instantaneous forward rate is then a derivative of yield curve function. Derivation is shown again in Hakala (2017). Instantaneous rate formula is:

$$f^M(0, T) = \beta_0 + \beta_1 \exp\left\{-\frac{T}{\gamma_1}\right\} + \beta_2 T \frac{\exp\left\{-\frac{T}{\gamma_1}\right\}}{\gamma_1} + \beta_3 T \frac{\exp\left\{-\frac{T}{\gamma_2}\right\}}{\gamma_2}. \quad (3.7)$$

For $\theta(t)$ calculation we need also the derivative of $f^M(0, T)$:

$$\frac{\partial f^M(0, t)}{\partial T} = -\frac{\beta_1}{\gamma_1} \exp\left\{-\frac{T}{\gamma_1}\right\} + \beta_2 \left(\frac{\exp\left\{-\frac{T}{\gamma_1}\right\}}{\gamma_1} - T \frac{\exp\left\{-\frac{T}{\gamma_1}\right\}}{\gamma_1^2} \right) + \beta_3 \left(\frac{\exp\left\{-\frac{T}{\gamma_2}\right\}}{\gamma_2} - T \frac{\exp\left\{-\frac{T}{\gamma_2}\right\}}{\gamma_2^2} \right). \quad (3.8)$$

Iteration of discretized function 3.2 can be written as:

$$r(t + \Delta t) = r(t) + [\theta(t) - \alpha r(t)]\Delta t + \sigma \Delta W(t) = (1 - \alpha \Delta t)r(t) + \Delta t \theta(t) + \sqrt{\Delta t} \sigma N(0, 1), \quad (3.9)$$

where $N(0, 1)$ states for random value of standard normal distribution. Parameters $(1 - \alpha \Delta t)$ and $\Delta t \theta(t) + \sqrt{\Delta t} \sigma N(0, 1)$ do not depend on trajectory $r(t)$ and can be computed before iterative calibration computation. Calibration of the Hull-White model is further described in Hakala (2017).

4. Implementation

In this part of the thesis, we will apply earlier presented variance reduction methods. Firstly, interest rate scenarios, both for the standard calculation and also antithetic scenarios, are generated by the Hull-White model. These scenarios are used for discounting, it means that the scenarios generated by Hull-White model are risk free rates. We will use 50 000 scenarios as the scenario base and then the methods will be compared in terms of the time consumption and the resulting variance of the BEL. We will compare only following run types:

- Base run - no variance reduction used
- Antithetic run - antithetic scenarios used
- CV run - conventional control-variate vector used
- Integrated CV run - Integrated control-variate vector used

Both Hull-White model and the Life Insurance cash flow model are coded in the Python language. It was chosen due to the extensive mathematical libraries available.

4.1 Interest rate scenarios

Under the Solvency II directive, EIOPA recommends the risk-free rate usage for BEL calculations. In particular we will use the zero-coupon euro swap curve based on Deutsche Bundesbank data Bun. We have 50 observations for each year. The development of used zero-coupon swap rates is in Figure 4.1.

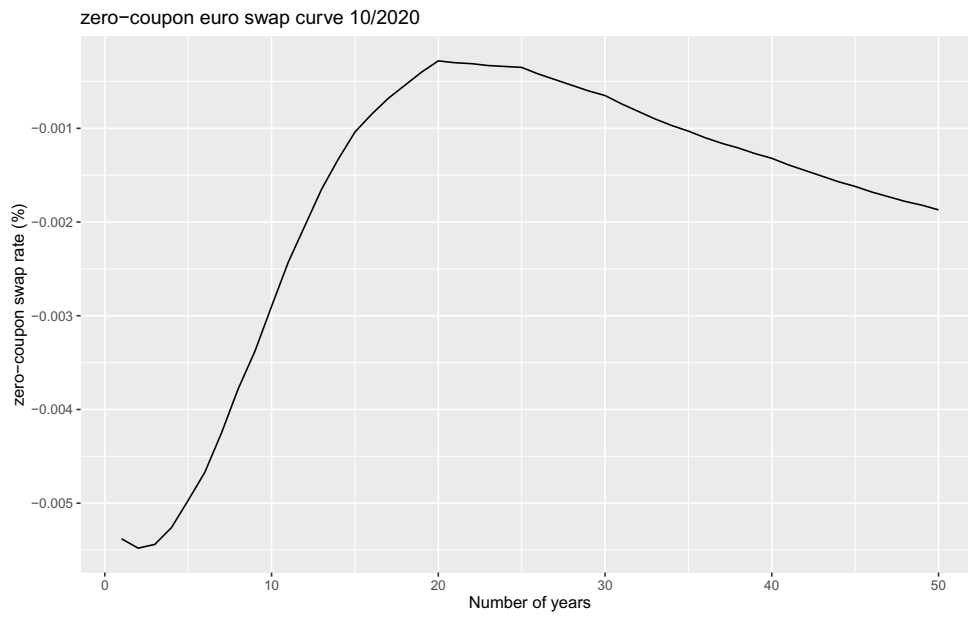


Figure 4.1: Zero-coupon euro swap curve 10/2020

Hull-White model parameters α and σ were set to $\alpha = 0.1$ and $\sigma = 0.016$. First 200 scenarios are plotted in the figure 4.2. The mean of all scenarios in comparison with the original ZCBS curve is plotted in Figure 4.3

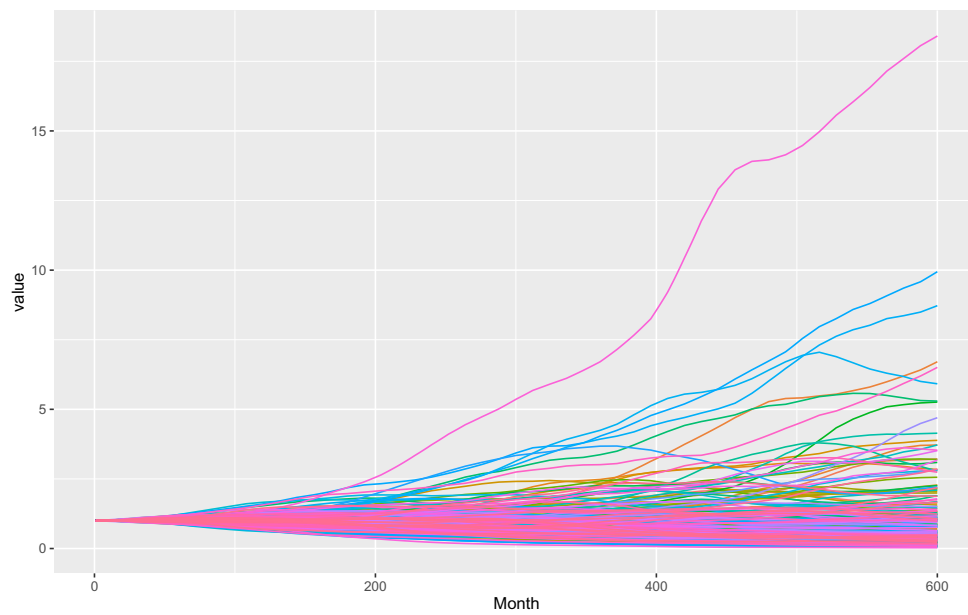


Figure 4.2: Discount factor scenarios

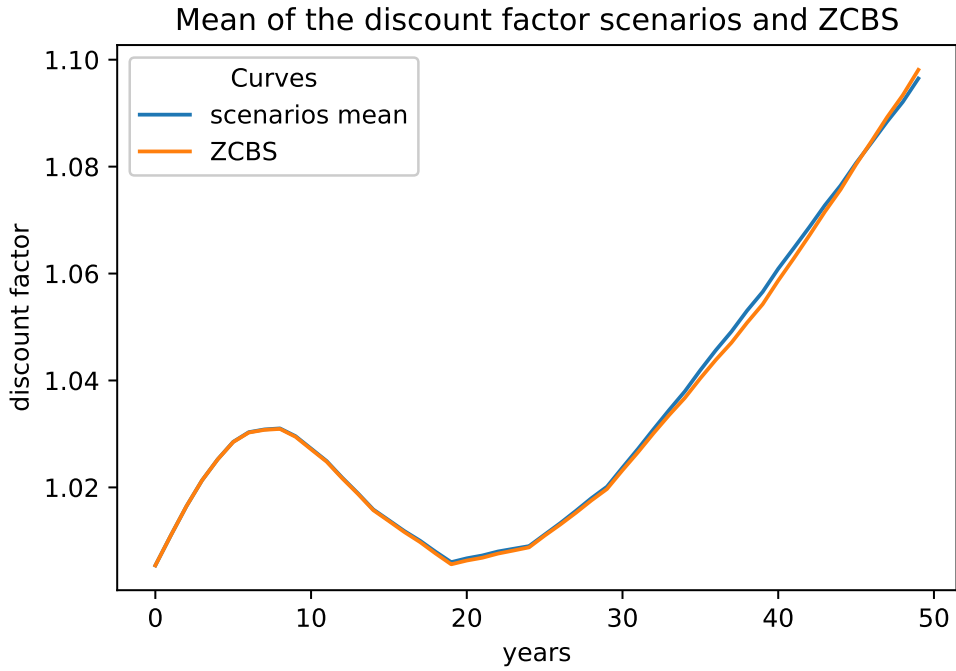


Figure 4.3: Discount factor scenarios mean and XCBZ curve

This scenarios are used in runs 1 and 3. Runs 2 and 4 use antithetic scenarios which mean in comparison with ZCBS curve is Figure 4.1

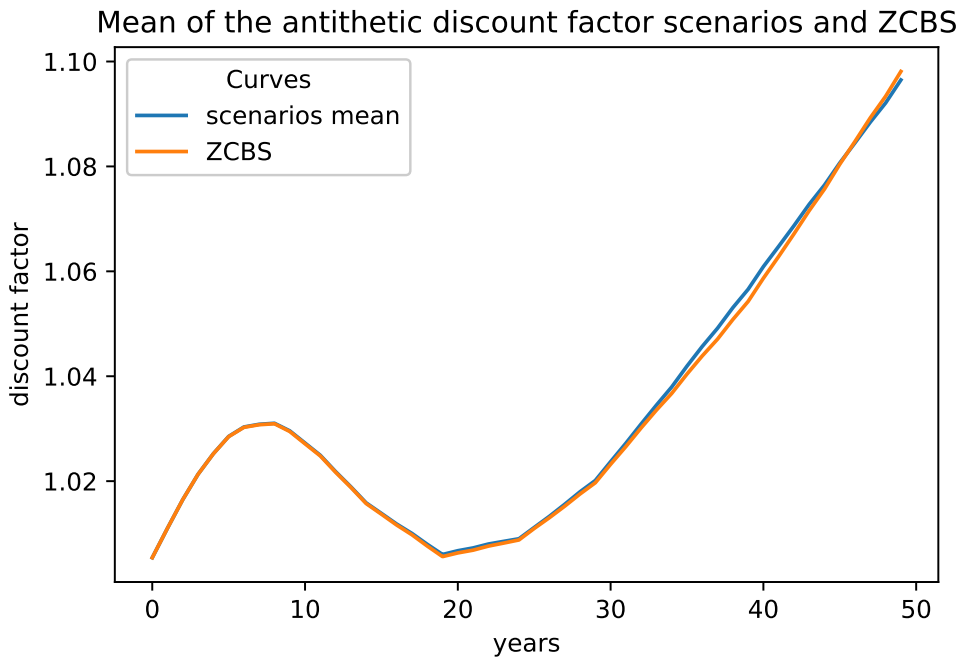


Figure 4.4: Discount factor antithetic scenarios mean and XCBZ curve

Both figures 4.3 and 4.1 are almost indistinguishable so there is no need to plot even the antithetic scenarios.

4.2 Cashflow projections assumptions

4.2.1 Modelpoints file

Our modelpoints file (file with the basic policies information - later only MP file) was provided by Martin Janeček in an exemplary cashflow model. There is no need to alter this MP file. There are 3360 policies in the MP file. The numbers of single premium and regular premium policies in the MP file are in Table 4.1

| Variable | Category | Number of observations |
|--|-----------------|------------------------|
| Policy type | Regular Premium | 2688 |
| | Single Premium | 672 |
| Sex | Female | 1680 |
| | Male | 1680 |
| Age | 20 | 960 |
| | 30 | 960 |
| | 40 | 960 |
| | 50 | 480 |
| Policy term | 10 | 960 |
| | 20 | 960 |
| | 30 | 960 |
| | 40 | 480 |
| Premium frequency (Regular premium) | Yearly | 672 |
| | Half-Yearly | 672 |
| | Quarterly | 672 |
| | Monthly | 672 |
| Sum Assured | 10 000 | 3360 |
| Inception year | 2009 | 840 |
| | 2010 | 840 |
| | 2011 | 840 |
| | 2012 | 840 |

Table 4.1: Number of observation for the MP file variables

As one can see the premium frequency is equally distributed for all possibilities (including Single premium policies). Also all inception years have the same number of observations. The only unbalanced variables are Policy term and Age. It reflects that clients older than 50 years do not write an insurance policy that often. Also the policy term longer than 40 years is not written often.

4.2.2 Other assumptions

The other assumptions used in the cash flow projection are also taken from the example cash flow model proposed by Martin Janeček.

Mortality rates

For the reserving purposes the mortality table provided by Martin Janeček will be used. It is very common that the experience mortality rates - those used for a decrement model - differ from the first order mortality rates. We will use a simplified approach where we use only the experience mortality rates coefficients. Experience mortality rates coefficient after the fifth year are equal to the fifth year value. These coefficients are in Table 4.2.

| Policy year | Coefficient |
|-------------|-------------|
| 1 | 0.5 |
| 2 | 0.6 |
| 3 | 0.7 |
| 4 | 0.8 |
| ≥ 5 | 0.9 |

Table 4.2: Experience mortality coefficients

Lapse rates

Lapse rates used in the decrement model are presented in Table 4.3. There are higher lapse rates in the first 5 years. This fact is caused by the agents who reinsure their clients mostly in the first five years which is driven by the commissions structure. It is natural to assume that the Regular premium lapse rates differ from the Single premium lapse rates.

| Policy year | Policy type | |
|-------------|-----------------|----------------|
| | Regular premium | Single premium |
| 1 | 0.1 | 0.05 |
| 2 | 0.09 | 0.04 |
| 3 | 0.08 | 0.03 |
| 4 | 0.07 | 0.02 |
| 5 | 0.06 | 0.02 |
| ≥ 6 | 0.05 | 0.02 |

Table 4.3: Lapse rates

Expenses and commissions

Expenses and commissions assumptions are in the table 4.4.

| Variable | Policy type | |
|----------------------------|-----------------|----------------|
| | Regular premium | Single premium |
| Initial commission SA | 0 | 2,5% |
| Initial commission Premium | 30% | 2,5% |
| Renewal commission SA | 0 | 0 |
| Renewal commission Premium | 5% | 0 |
| Initial expenses fix | 0 | 0 |
| Initial expenses premium | 10% | 1% |
| Renewal expenses fix | 0 | 0 |
| Renewal expenses premium | 10% | 0.5% |

Table 4.4: Expenses and commission structure

Surrender value

Surrender periods and surrender charges for both premium types are in Table 4.5.

| Variable | Policy type | |
|---------------------------|-----------------|----------------|
| | Regular premium | Single premium |
| Surrender charge | 300 | 300 |
| Surrender period (months) | 24 | 1 |

Table 4.5: Variables connected to the surrender values

Premium

Premium calculations are based on the same technical interest rate as the reserving. We will not project any administration cost reserve. It means that we will assume that the β -costs are equal to zero. The only costs which will be taken into account are α -costs. We will use the Zillmerisation to adjust the reserve with the initial costs proportion. The premium assumptions are in Table 4.6

| Variable | Policy type | |
|----------------|-----------------|----------------|
| | Regular premium | Single premium |
| TIR | 2.5% | 2.5% |
| α_{SA} | 2.5% | 2.5% |
| α_{Prm} | 25% | 2.5% |
| γ | 3% | 0 |

Table 4.6: Variables connected to the surrender values

4.3 Run results

In this section we will present the results of each run. We will then compare the resulting BEL variance and time needed to compute the results. For each run 50 000 risk-free scenarios are used to get a proper value of the BEL. For runs number 3 and 4 the BEL variance is calculated after 2 000 scenarios. This is due to the β coefficient estimate of the control variate estimator. Thanks to this fact we will look at the BEL and its standard deviation evolution after the 2 000th scenario.

4.3.1 Run 1 - Base run

The base run will serve as a benchmark for all other runs. There are no variance reduction methods used in the Base run projection. The mean BEL value evolution and its confidence band around the mean are in Figure 4.3.1.

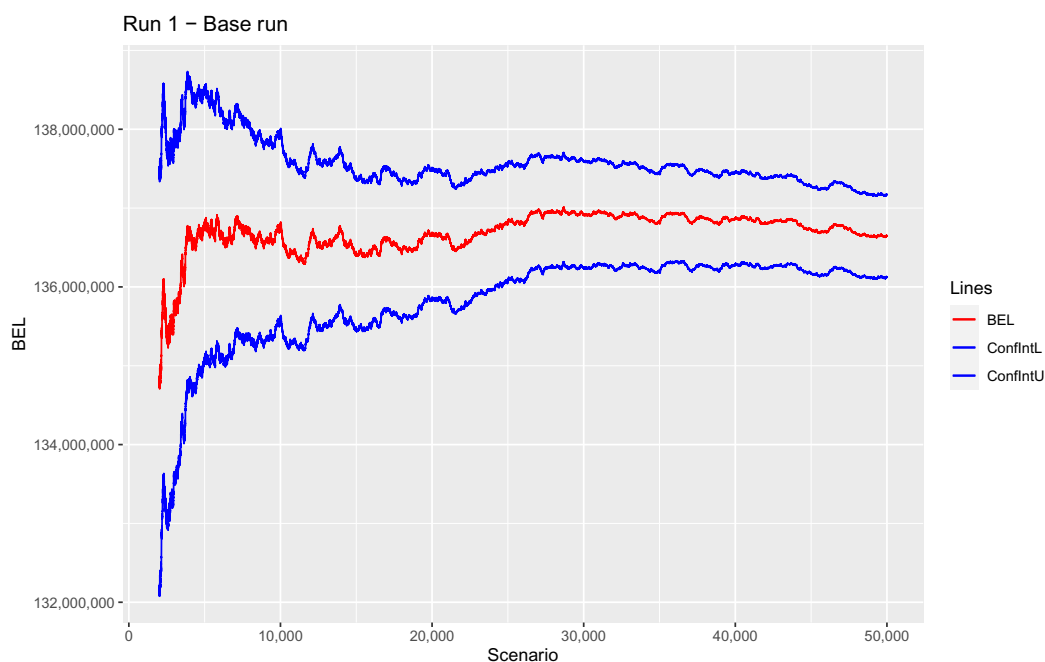


Figure 4.5: BEL evolution

As one can see the BEL value is pretty stable after 30 000 scenarios. Also the confidence band around the BEL value does not change a lot. The resulting values of the BEL, its standard deviation and time needed to calculate all scenarios are in Table 4.7

| | BEL | Standard deviation | Time (seconds) |
|-------|----------------|--------------------|----------------|
| Run 1 | 136 644 246.58 | 272 615.3963 | 7.2 |

Table 4.7: Results of the Run 1 - Base run

4.3.2 Run 2 - Antithetic variates run

To include the antithetic variates scenarios in the projection of the second run we need to alter the underlying scenario generating process. Interest rate models are based on Brownian motion so the increment of the Monte Carlo simulation is a function h of the interest rate model parameters θ and a random value generated by standard normal distribution

$$r(t + \delta t) = h(\theta, N(0, 1)),$$

where $N(0,1)$ states for random value of standard normal distribution. For the purposes of the antithetic variates method we use two paths with the same random value but different signs. The mean BEL value evolution and its confidence band around the mean are in Figure 4.3.2.

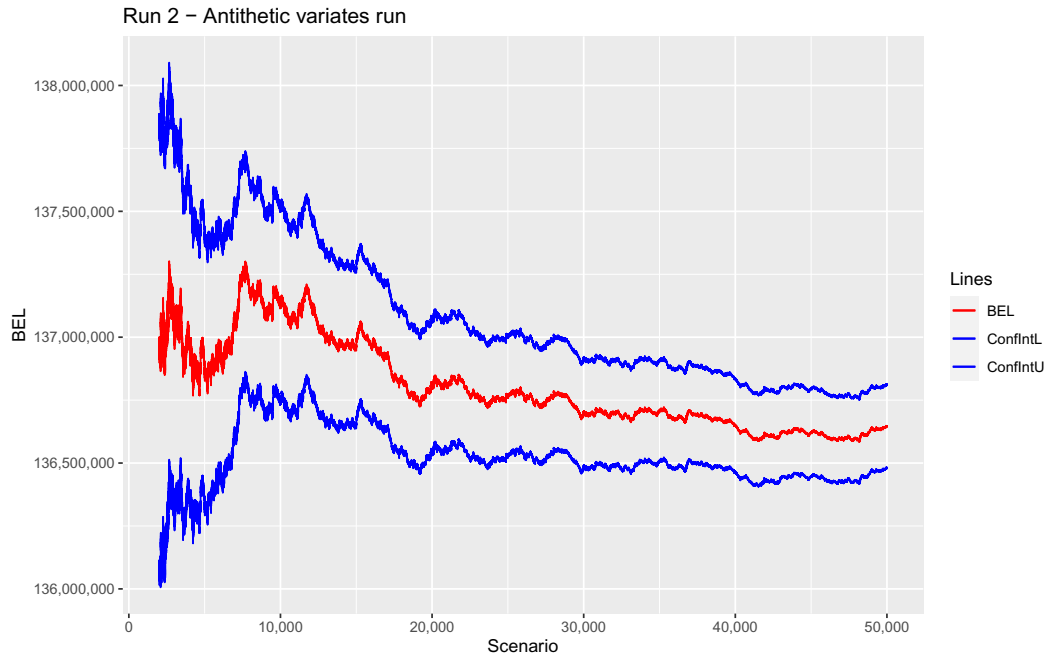


Figure 4.6: BEL evolution

When looking at this graph it is obvious that the confidence band around the BEL is narrower than in the first case. The BEL value seems to be again for stable after the 30 000 scenarios but we need to bare in mind that the scale is different in comparison with the figure with Run 1 results. Its standard deviation and time needed to calculate all scenarios is in Table 4.8.

| | BEL | Standard deviation | Time (seconds) |
|-------|----------------|--------------------|----------------|
| Run 2 | 136 646 910.67 | 82 397.2278 | 59.49 |

Table 4.8: Results of the Run 2 - Antithetic variates run

4.3.3 Run 3 - Conventional control-variate estimator

The third run uses conventional control-variate estimator to estimate the BEL and reduce its variance. The control-variate in our case will be the whole or part of the discount factors vector. This vector seems to be convenient choice because we know its real μ_X , which are the observed market values and the scenarios are generated along with the discounted cash flow values. To decide which part of the discount factors vector is the best choice, we run the control-variate estimator run multiple times with gradually increasing number of included discount factors. It means, that in every iteration, we include the discount factor for the first year plus all discount factors for next year up to the number of iteration. It means that in the first iteration we include the discount factors for the first two years and so on. We assume that discount factors in the first few years will be more important than the ones for the further years. The underlying scenario base is not the antithetic scenario base used in the second run so the results presented lower are for the conventional control-variate estimators. The resulting BEL variance estimates after all scenarios are in Figure 4.7.

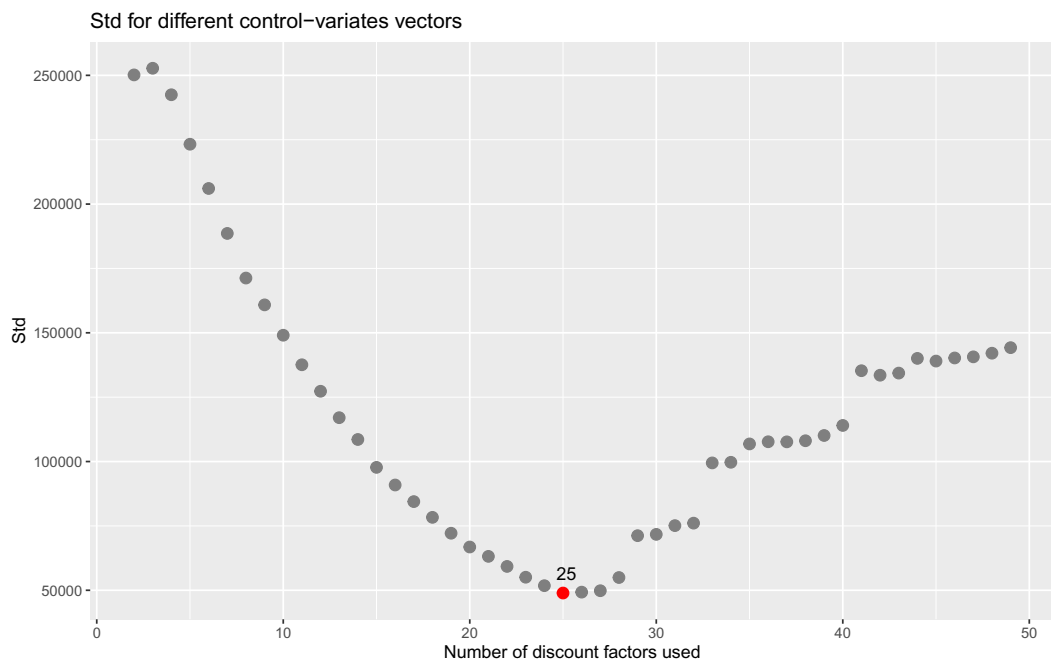


Figure 4.7: Std of control-variates run with different control-variates vector

We can see in Figure 4.7 that the minimum standard deviation is attained when using vector with discount factors up to the 25th year. We will use this control vector in all runs for control-variate method. As it was mentioned in the section 2.4.4, more complex calculations of the control-variate estimator variance estimate cause time increase of the projection. We tried to hold the underlying code as much effective as it was possible. Still there is not a negligible time difference between the base run and runs using control variate estimator. The mean BEL value evolution and its confidence band around the mean are in Figure 4.8.

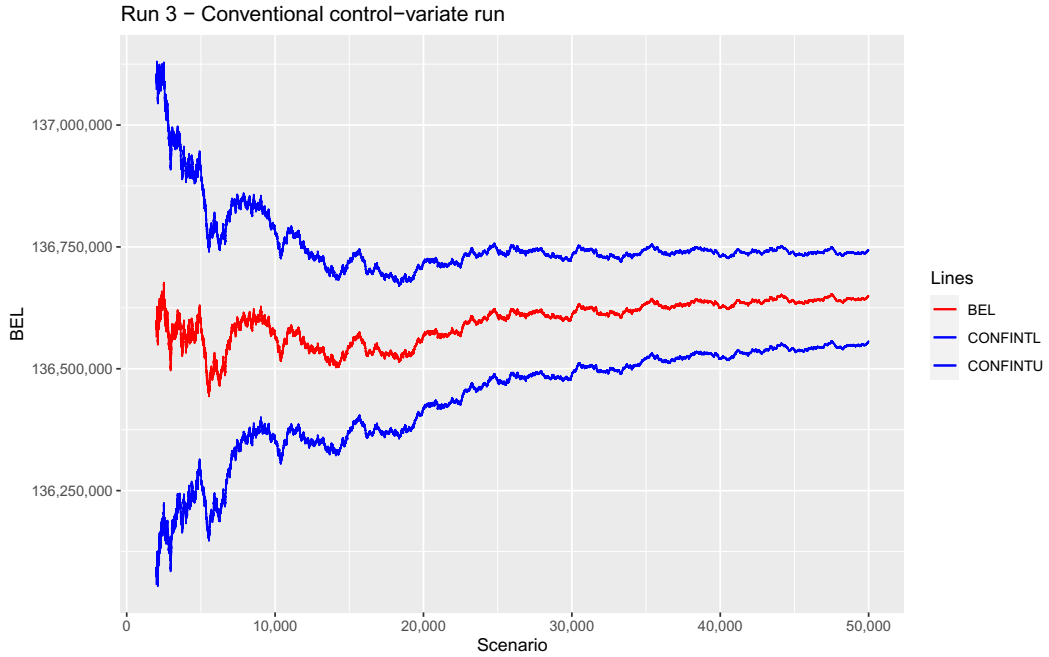


Figure 4.8: BEL evolution

When looking at this graph, it is obvious that the confidence band around the BEL is much narrower compared to the first two runs. The BEL value seems to be again stable after the 30 000 scenarios but we need to bare in mind that the scale is a bit different in comparison with the figure with Run 1 results. Its standard deviation and time needed to calculate all scenarios are in Table 4.9.

| | BEL | Standard deviation | Time (seconds) |
|-------|----------------|--------------------|----------------|
| Run 3 | 136 649 937.77 | 48 830.4616 | 62.07 |

Table 4.9: Results of the Run 3 - Conventional control variate run

4.3.4 Run 4 - Integrated control variate estimator

The forth run uses integrated control variate estimator to estimate the BEL and reduce its variance. There are antithetic scenarios applied on the control-variate in this case. As in the previous run, the discount factor vector serves as the control-variate. The results are in Figure 4.9.

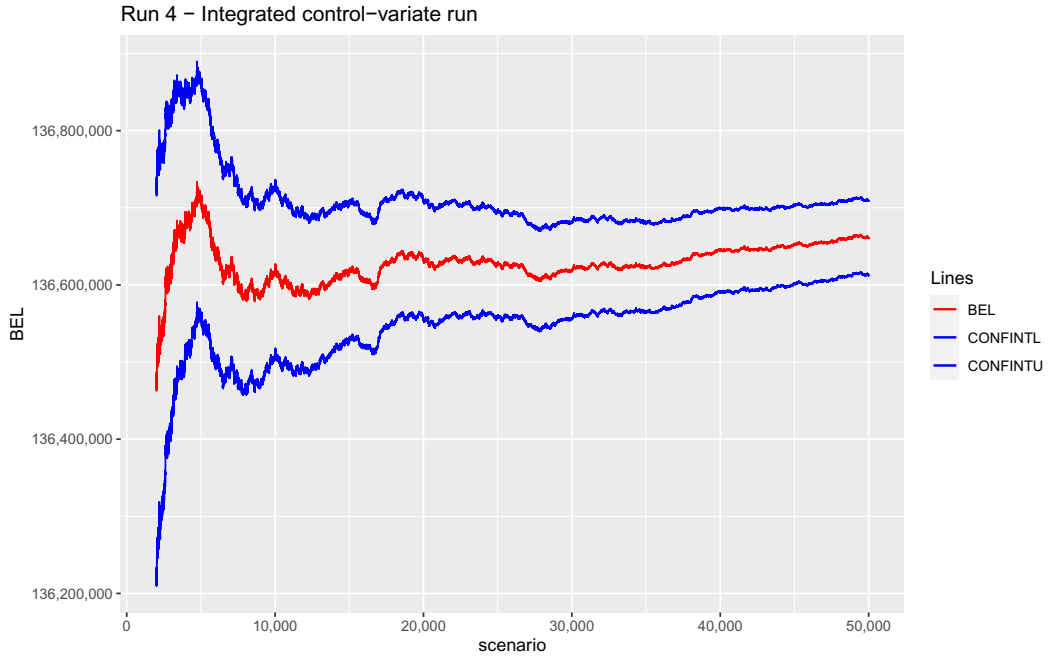


Figure 4.9: BEL evolution

When looking at this graph, it is obvious that the confidence band around the BEL is much narrower compared to the first three runs. The BEL value seems to be stable after the 20 000 scenarios but we again need to bare in mind that the scale is different from the previous figures. Its standard deviation and time needed to calculate all scenarios is in Table 4.10.

| | BEL | Standard deviation | Time (seconds) |
|-------|----------------|--------------------|----------------|
| Run 4 | 136 660 386.71 | 24 648.1113 | 79.95 |

Table 4.10: Results of the Run 4 - Integrated control variate run

4.3.5 BEL comparison

The values of BEL can differ among our runs. We want the BEL values of runs with variance reduction methods to be similar to BEL value of our base run after all scenarios. Following Figure 4.10 shows the evolution of BEL.

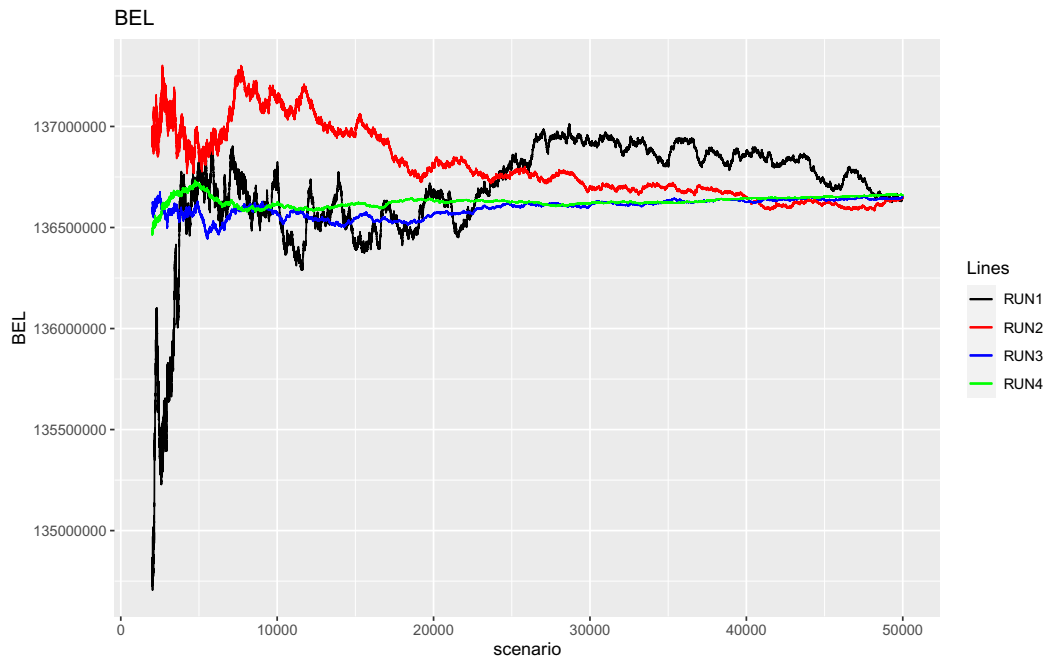


Figure 4.10: Standard deviation estimates

It is obvious that all runs converge to the same value but the first two runs are much more volatile especially from the beginning of the simulation. The runs where control-variate method is used are pretty stable. A problem can arise, when the BEL value for lower scenarios, which we would like to take as a faster alternative to the base run, significantly differ from the BEL value of the base run. The question then lies in the evaluation of the difference. It means, whether the insurance company has a materiality level for example 5% and the percentage difference between the base run and the other run is lower than this materiality level or the insurance company wants to test them with statistical test.

4.3.6 Standard deviation comparison

It is not that obvious how much the standard deviation differs between each run i.e., how much are the used variance reduction methods effective. To get a better insight we present the figure of the estimated standard deviations for all presented runs in Figure 4.11. We choose the standard deviation instead of the variance because of the scale.

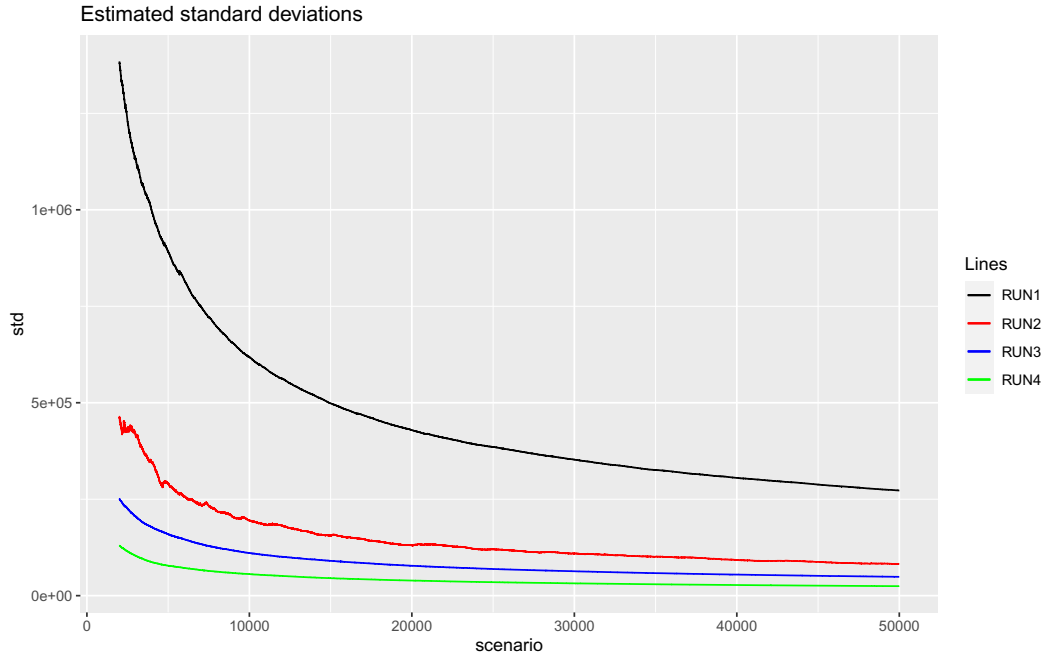


Figure 4.11: Standard deviation estimates

It is now clear that antithetic variate method, control-variate method and integrated control-variate method reduce significantly the standard deviation which implies they reduce the variance. The difference in standard deviation between the used methods themselves is significantly lower than the difference in std between the base run and the used methods. The BEL values after all scenarios and their estimated standard deviations are in Table 4.11

| Run | BEL | Standard deviation |
|-------|----------------|--------------------|
| Run 1 | 136 644 246.58 | 272 615.3963 |
| Run 2 | 136 646 910.67 | 82 397.2278 |
| Run 3 | 136 649 937.77 | 48 830.4616 |
| Run 4 | 136 660 386.71 | 24 648.1113 |

Table 4.11: Estimated standard deviations

4.3.7 Time comparison

Now we have to assess the time reduction instead of the variance reduction. The methods are primarily presented to reduce the amount of time needed to calculate the BEL. The methods would be useless if the variance is reduced but the time needed to compute the BEL with the same variance as the base run is greater than the base run time. The goal is to find the first standard deviation occurrence that is lower than the resulting base run standard deviation. As it was mentioned earlier the base run is pretty fast because there are no complicated calculations. Following Figure 4.12 can get us a better insight into the speed of runs, where variance reduction method is used. Also we can get a rough estimate

of the scenario number which is maximal possible for each run to use, because it has lower elapsed time than all scenarios from the base run.

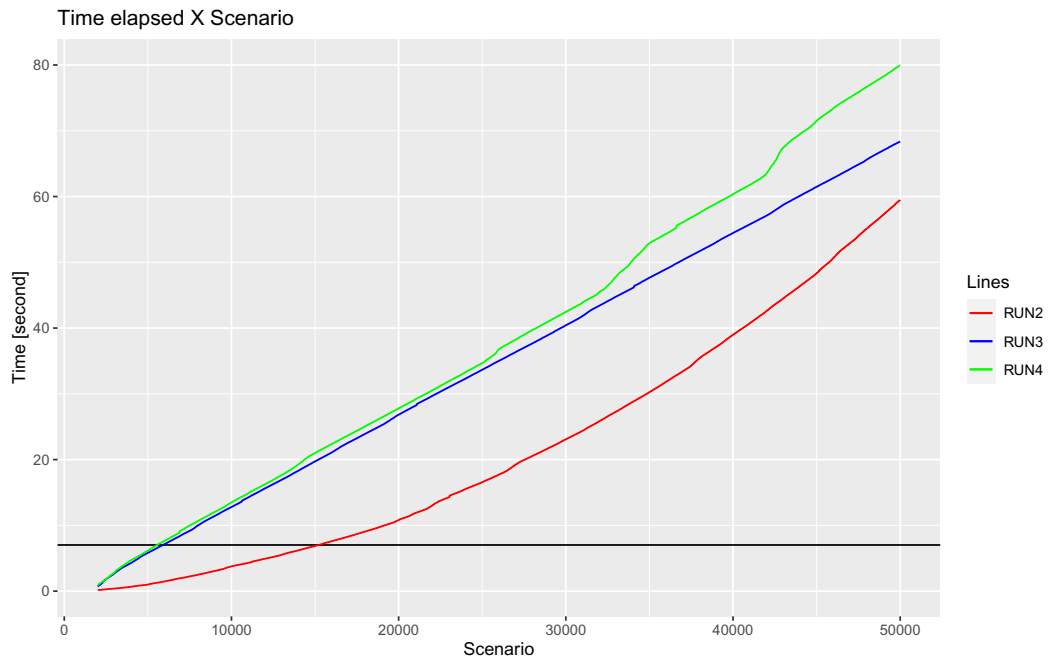


Figure 4.12: Time x Scenario

The black horizontal line represents the elapsed time of all scenarios from the base run. We can see that for the second run we should investigate the elapsed time and variance of scenarios up to approximately 15 000. For the third and fourth run it is roughly 5 500 scenarios. The next Figure 4.13 explores the standard deviation in particular elapsed times for each run.

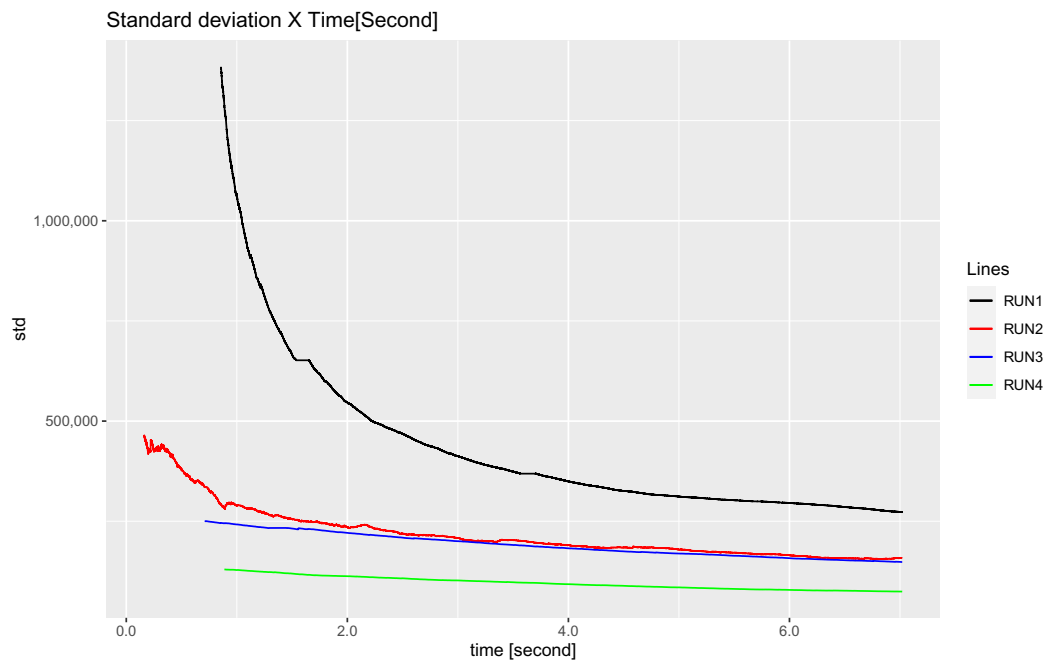


Figure 4.13: Standard deviation x Time

We can see that for our purposes i.e. for scenarios with elapsed time lower than the elapsed time of the base run after all scenarios, the antithetic variate and conventional control variate runs are similar in the terms of variance reduction. If we aim to use the minimum scenarios possible, the control-variate method is slightly better than the antithetic variate method. Integrated control variate method outperforms all other methods and significantly reduces the variance. We can now look at the scenario that has lower elapsed time and at the same time it has lower standard deviation than the base run. The results are in Table 4.12.

| Run | Scenario | BEL | Standard deviation | Elapsed time |
|-------|----------|----------------|--------------------|--------------|
| Run 2 | 5 471 | 136 905 333.02 | 272 384.6430 | 1.2442 |
| Run 3 | 2000 | 136 564 384.67 | 250 462.5799 | 0.7095 |
| Run 4 | 2000 | 136 481 345.52 | 129 695.1512 | 0.8876 |

| Run | Base time reduction (%) |
|-------|-------------------------|
| Run 2 | 82.28 |
| Run 3 | 89.89 |
| Run 4 | 87.35 |

Table 4.12: Scenarios with lower elapsed time

The absolute differences between the resulting BEL value of the base run and other runs and their relative differences are in Table 4.14.

| Run | Scenario | BEL difference | Rel. difference (%) |
|-------|----------|----------------|---------------------|
| Run 2 | 5 471 | -261 086.44 | -0.1911% |
| Run 3 | 2 000 | 79 861.91 | 0.0583% |
| Run 4 | 2 000 | 169 781.98 | 0.1243% |

Table 4.13: BEL differences

All differences are under 1% of the relative difference. The materiality threshold used in insurance companies is usually from 1% to 5%. So we can conclude that from this point of view the difference is acceptable. On the other hand we can use the paired sample t-test to evaluate the equality of the run BEL values with the base run BEL value. The Null hypothesis, alternative hypothesis and test statistic is as in Omelka (2018).

$$H_0 : \mu_X = \mu_Y,$$

$$H_1 : \mu_X \neq \mu_Y$$

$$Z_{n,m} = \frac{\bar{X}_n - \bar{Y}_m}{\sqrt{S_X^2/n + S_Y^2/m}},$$

where \bar{X}_n , \bar{Y}_m , S_X^2 and S_Y^2 are sample means and sample variances respectively. It holds

$$\frac{\bar{X}_n - \bar{Y}_m - (\mu_X - \mu_Y)}{\sqrt{S_X^2/n + S_Y^2/m}} \xrightarrow{D} N(0, 1),$$

for $m, n \rightarrow \infty$. The test values and p-values for each run are in the table ??.

| Run | Test statistic | p-value |
|-------|----------------|---------|
| Run 2 | -67.2595 | < 0.001 |
| Run 3 | 12.8853 | < 0.001 |
| Run 4 | 26.6029 | < 0.001 |

Table 4.14: Test statistic

We can conclude that the BEL values for each run using variance reduction method are not statistically equal to the BEL value of the base run. On the other hand the relative differences are low and it is probable that in the real world we could use such values.

Conclusion

The BEL calculation is one of the most important tasks in the life of actuary. When using large number of economic scenario, this task can be pretty demanding especially with numerous product types and a lot of policies. The purpose of this thesis was to present two variance reduction methods and their combination. These methods can decrease the projection time.

We assumed only the traditional type of policies, both single premium and regular premium. We introduced the main formulas and liability valuation principle. Also we presented the basics of interest rate model theory, in particular the Hull-White model which was used for the simulation of economic scenarios.

For the projection purposes we chose the Python programming language due to the already created extensive mathematical libraries. We started with the liabilities projection for our modelpoint file, which consists of 3360 in-force traditional life-insurance policies. The calculated liabilities were discounted with 50 000 simulated scenarios for our base run i.e. the benchmark for other runs. Antithetic variate, control-variate and integrated control-variate method were used to reduce the variance of the BEL value.

Every variance reduction method significantly decreased time needed to calculate BEL. Even though we rejected the null hypothesis about equality of BEL values between runs with used variance reduction method and base run, the relative differences are small and it is most likely that they would be acceptable in the insurance company (under materiality threshold difference).

We compared BEL values and its standard deviation for scenario greater than 2 000. This is due to the β^* coefficient estimate in the control-variate runs, which is estimated after 2 000 scenarios. It was shown that all observed runs were about the same in the terms of time reduction. Control-variate method slightly over performed the Integrated control-variate method from the perspective of time reduction. Control-variate method has a little simpler code than Integrated control-variate method and so the projection of 2000 scenarios, after which we estimate the β^* coefficient, is faster. On the other hand Integrated control-variate run speed is approx. 0.18 second slower but the variance is 50% lower. This implies that if we estimated the β^* coefficient after lower number of scenarios, we could presumably find Integrated control-variate method faster than control-variate method. A questions is whether the time reduction with lower scenarios used for β^* coefficient estimate is not at the expense of BEL value accuracy.

The most important part of the code were recursive formulas for sample mean and sample variace-covariance. There would be no time reduction without recursive formulas. We can conclude that the code, where antithetic variate method is used, is more time consuming than the control-variate method code - the antithetic variate code has more simple parts of the code that repeat for every scenario. This causes that the conventional control-variate method is the fastest one. Conventional control-variate method code is on the other hand more complex

and could be more difficult to implement. Both antithetic variate method and conventional control-variate methods are approximately the same in the means of variance reduction. Integrated control-variate method is absolute winner in the terms of variance reduction, but can be harder to implement. If the actuary's goal is rather time reduction with minimum coding effort, the antithetic variate method is the best choice. On the other hand for more code experienced actuary conventional control-variate method brings additional time reduction. If the actuary's goal is also the minimum variance, except from the time reduction, integrated control-variate method is the winner.

Bibliography

- Euro zero-coupon swap curve deutsche bundesbank data. URL https://www.bundesbank.de/dynamic/action/en/statistics/time-series-databases/time-series-databases/759784/759784?listId=www_skms_it05b.
- Directive 2009/138/ec of the european parliament and of the council of 25 november 2009 on the taking-up and pursuit of the business of insurance and reinsurance (solvency II). *Official Journal of the European Union*.
- CIPRA, T. (2006). *Pojistná matematika - teorie a praxe*. Second edition. Ekopress.
- F.MERCURIO, D. B. (2006). *Interest Rate Models - Theory and Practice*.
- GLASSERMAN, P. (2004). *Monte Carlo Methods in Financial Engineering*. Springer-Verlag, New York.
- HAKALA, M. (2017). *Modely úrokových měř - praktické aspekty*.
- HORN, R. and JOHNSON, C. (1985). Matrix analysis. *Cambridge university press, Cambridge*.
- JANEČEK, M. (2006). *Valuation Techniques of Life Insurance Liabilities*. PhD thesis, Charles university, Faculty of Mathematics and Physics, Praha.
- KUZMINSKAYA, K. (2018). *Acceleration of calculations in life insurance*. Diploma thesis, Charles university, Faculty of Mathematics and Physics, Praha.
- KWON, C. and TEW, J. (1993). Combined correlation induction strategies for designed simulation experiments. *IEEE Press, Los Alamitos, California*, pages 455–465.
- LAW, A. (2014). *Simulation modeling and Analysis, 2nd edition*. McGraw-Hill Education. ISBN 978-0073401324.
- LIDEBRANDT, T. (2007). *Variance Reduction Three Approaches to Control Variates*. Stockholm.
- MORRISON, D. (1976). Multivariate statistical methods. *McGraw-Hill, New York*.
- NELSON, B. (1988). Control variate remedies. *Working Paper Series No. 1988 - 004*. URL <http://users.iems.northwestern.edu/~nelsonb/Publications/ControlVariateRemedies.pdf>. (Accessed 1.12.2020).
- OMELKA, M. (2018). *NMSA331 Matematická statistika, Poznámky k přednášce*. MFF UK. URL https://www2.karlin.mff.cuni.cz/~omelka/Soubory/nmsa331/ms1_1718.pdf. (Last updated 19.2.2018, Accessed 27.12.2020).

- SIGMAN, K. (2007). Introduction to reducing variance in monte carlo simulations. *Columbia University*. URL <http://www.columbia.edu/~ks20/4703-Sigman/4703-07-Notes-ATV.pdf>. (Accessed 12.10.2020).
- WEI-NING, Y. and WEI-WEN, L. (1996). Combining antithetic variates and simulation experiments. *ACM Transaction on Modeling and Computer Simulation*, **6**.

List of Figures

| | | |
|------|--|----|
| 4.1 | Zero-coupon euro swap curve 10/2020 | 30 |
| 4.2 | Discount factor scenarios | 30 |
| 4.3 | Discount factor scenarios mean and XCBZ curve | 31 |
| 4.4 | Discount factor antithetic scenarios mean and XCBZ curve | 31 |
| 4.5 | BEL evolution | 35 |
| 4.6 | BEL evolution | 36 |
| 4.7 | Std of control-variates run with different control-variates vector | 37 |
| 4.8 | BEL evolution | 38 |
| 4.9 | BEL evolution | 39 |
| 4.10 | Standard deviation estimates | 40 |
| 4.11 | Standard deviation estimates | 41 |
| 4.12 | Time x Scenario | 42 |
| 4.13 | Standard deviation x Time | 42 |

List of Tables

| | | |
|------|---|----|
| 1.1 | Cash flows of endowment policy | 13 |
| 4.1 | Number of observation for the MP file variables | 32 |
| 4.2 | Experience mortality coefficients | 33 |
| 4.3 | Lapse rates | 33 |
| 4.4 | Expenses and commission structure | 34 |
| 4.5 | Variables connected to the surrender values | 34 |
| 4.6 | Variables connected to the surrender values | 34 |
| 4.7 | Results of the Run 1 - Base run | 35 |
| 4.8 | Results of the Run 2 - Antithetic variates run | 36 |
| 4.9 | Results of the Run 3 - Conventional control variate run | 38 |
| 4.10 | Results of the Run 4 - Integrated control variate run | 39 |
| 4.11 | Estimated standard deviations | 41 |
| 4.12 | Scenarios with lower elapsed time | 43 |
| 4.13 | BEL differences | 43 |
| 4.14 | Test statistic | 44 |

A. Attachments

A.1 First Attachment

Commented code for the cash flow model starts on the next page.

```

#libraries import
import pandas as pd
import os
import datetime
import pdb
import time
from scipy.stats import norm
import numpy as np

#basic variable definitions
start_time = time.time()
month_modelled = 0
policy_month = 0
policy_year = 0
month_in_year = 0
actual_age = 0
in_for = 0
prem_freq = 0
cal_year = 0
qx = 0
mort_exp = 0
q_exp = 0
lapse_rate = 0
disc_r = 0
SA = 0
pol_num = 0
age_at_entry = 0
Alpha_sa = 0
Alpha_prem = 0
Beta = 0
Gamma = 0
Net_res = 0
sex = 0
pol_term_y = 0
pol_term_m = 0
count = 0
CV_freq = 0
age = 0

#valuation date
val_date = datetime.datetime.strptime("1.1.2012", '%d.%m.%Y')

#technical interest rate set to 2,5%
tir=0.025

#input mortality table
mort_tab_M = pd.read_csv("mort_tab_M.csv", sep = ";", float_precision = 'high')

#Entry console setting
#One can set start and end policy for projection, run number which indicates the
#variable reduction method used and number of economic scenarios used for discounting
# RUN = 1 - basic scenario
#   = 2 - antithetic scenario
#   = 3 - control variates method

```

```

# = 4 - integrated control variates method
while True:
    try:
        start_policy = int(input("Please enter a start policy: "))
        end_policy = int(input("Please enter an end policy: "))
        run_number = int(input("Please enter a run number: "))
        n_curves = int(input("Please enter a number of scenarios"))
        break
    except ValueError:
        print("Oops! That was no valid number. Try again...")

#modelpoint assumptions, other assumptions, experience mortality factors and lapse rates
mp_file = pd.read_csv("model\\MP_FILE.csv", sep = ";")
oth_ass = pd.read_csv("model\\oth_assum.csv", sep = ";")
mort_exp = pd.read_csv("model\\mort_exp.csv", sep = ";")
lapse_rates = pd.read_csv("model\\lapse_rates.csv", sep = ";")

#interest rate file if run is 1 or 3 there is standard scenario file used
#for scenarios 2 and 4 anthitetic scenarios file is used
if run_number == 1 or run_number == 3:
    rates = pd.read_csv("model\\int_rates.csv", sep = ";")
else:
    rates = pd.read_csv("model\\int_rates_anthi.csv", sep = ";")

rates = rates.T.iloc[2:len(rates.T)]
policy = 1
invest_margin = 0.001
MATH_RES = {}

#MP file variables
age_at_entry = mp_file.iat[policy,3]
sex = mp_file.iat[policy,4]
pol_term_y = mp_file.iat[policy,5]
SA = mp_file.iat[policy,6]
prem_freq = mp_file.iat[policy,7]
CV_freq = mp_file.iat[policy,8]
count = mp_file.iat[policy,9]

#other assumptions as dictionary
other_assumptions = {"pol_type":list(oth_ass["Policy type"]),
                    "TIR":list(oth_ass["TIR"]),
                    "alpha_SA":list(oth_ass["Alpha SA"]),
                    "alpha_prem":list(oth_ass["Alpha premium"]),
                    "beta":list(oth_ass["Beta"]),
                    "gamma":list(oth_ass["Gama"]),
                    "surr_per":list(oth_ass["Surrender period"]),
                    "surr_charge":list(oth_ass["Surrender charge"]),
                    "init_comm_SA":list(oth_ass["Initial commission SA"]),
                    "init_comm_prem":list(oth_ass["Initial commission premium"]),
                    "ren_comm_SA":list(oth_ass["Renewal commission SA"]),
                    "ren_comm_prem":list(oth_ass["Renewal commission premium"]),
                    "init_exp_fix":list(oth_ass["Initial expenses fix"]),
                    "init_exp_prem":list(oth_ass["Initial expenses premium"]),
                    "ren_exp_fix":list(oth_ass["Renewal expenses fix"]),

```

```

        "ren_exp_prem":list(oth_ass["Renewal expenses premium"])
    }

```

#all classes below are calculated for each policy based on its parameters
class CPVB_RES:

```

#Class representing present value of future benefits
#the output of the class is vector for each time till the end of policy

```

```

def __init__(self, age = 0, month = 1):
    self.PVB = {}
    self.PVB_DF = pd.DataFrame()
    self.age = age
    self.month = month
    self.expired = {}
    self.unexpired = {}
    for i in range (1,13):
        self.expired[i % 12] = (i-1)/12
        self.unexpired[i % 12] = 1-(i-1)/12
        self.PVB[i]=0

```

```

def recalc(self, age):

```

```

    #recalculation using commutation factors
    self.age = age
    self.PVB[1] = ((mort_tab_M.iat[age_at_entry + age, 7] - mort_tab_M.iat[age_at_entry + pol_term_y, 7] + mort
_tab_M.iat[age_at_entry + pol_term_y, 3])
        /mort_tab_M.iat[age_at_entry + age, 3])
    for t in range (1,pol_term_y - self.age + 1):
        self.PVB[t*12 + 1] = ((mort_tab_M.iat[age_at_entry + age + t, 7] - mort_tab_M.iat[age_at_entry + pol_term
_y, 7] + mort_tab_M.iat[age_at_entry + pol_term_y, 3])
            /mort_tab_M.iat[age_at_entry + age + t, 3])

    for i in range((t-1) * 12 + 1, (t * 12) + 1):
        self.PVB[i] = self.PVB[(t-1) * 12 + 1] * self.unexpired[ i % 12] + self.PVB[(t * 12) + 1] * self.expired[i %
12]
    self.PVB_DF = pd.DataFrame.from_dict(self.PVB,orient = "index")

```

class CPVP_RES:

```

#Class representing value of future premiums
#the output of the class is vector for each time till the end of policy

```

```

def __init__(self, age = 0, month = 1):
    self.PVP = {}
    self.PVP_DF = pd.DataFrame()
    self.age = age
    self.month = month
    self.expired = {}
    self.unexpired = {}
    for i in range (1,13):
        self.expired[i % 12] = (i-1)/12
        self.unexpired[i % 12] = 1-(i-1)/12
        self.PVP[i]=0

```

```

def recalc(self, age):

```

```

#recalculation using commutation factors
self.age = age
self.PVP[1] = ((mort_tab_M.iat[age_at_entry + age, 4] - mort_tab_M.iat[age_at_entry + pol_term_y, 4])
              /mort_tab_M.iat[age_at_entry + age, 3])
for t in range (1,pol_term_y - self.age + 1):
    self.PVP[t*12 + 1] = ((mort_tab_M.iat[age_at_entry + age + t, 4] - mort_tab_M.iat[age_at_entry + pol_term
_y, 4])
                        /mort_tab_M.iat[age_at_entry + age + t, 3])

    for i in range((t-1) * 12 + 1, (t * 12) + 1):
        self.PVP[i] = self.PVP[(t-1) * 12 + 1] * self.unexpired[ i % 12] + self.PVP[(t * 12) + 1] * self.expired[i %
12]
self.PVP_DF = pd.DataFrame.from_dict(self.PVP, orient = "index")
class CZIL_RES:

#class representing a value of zillmer part of premium i.e. alfa costs redistributed till the end of policy term
def __init__(self, GROSS_PREM, PVP_RES, PVP_RES_START, age = 0, month = 1):
    self.ZIL = {}
    self.PVP_RES = PVP_RES
    self.GROSS_PREM = GROSS_PREM
    self.age = age
    self.month = month
    self.expired = {}
    self.unexpired = {}
    self.PVP_RES = PVP_RES
    self.PVP_RES_START = PVP_RES_START
    for i in range (1,13):
        self.expired[i] = (i-1)/12
        self.unexpired[i] = 1-(i-1)/12
        self.ZIL[i]=0
def recalcul(self,age):
    self.age = age
    if prem_freq == 11:
        self.ZIL[1] = 0
    else:
        self.ZIL[1] = (other_assumptions["alpha_prem"][policy_type] * self.GROSS_PREM.get_value() + SA * oth
er_assumptions["alpha_SA"][policy_type]) * self.PVP_RES.PVP[1] / self.PVP_RES_START.get_value(1)
        self.ZIL[13] = (other_assumptions["alpha_prem"][policy_type] * self.GROSS_PREM.get_value() + SA * ot
her_assumptions["alpha_SA"][policy_type]) * self.PVP_RES.get_PVP[13] / self.PVP_RES_START.get_value(1)
        for i in range(1,13):
            self.ZIL[i] = self.ZIL[1] * self.unexpired[i] + self.ZIL[13] * self.expired[i]
    return self.ZIL
def get_value(self, month):
    return self.ZIL[month]

class CNET_PREM:

#Class for Net premim calculation again using commutation factors
#output of the class is vector with net premium
def __init__(self):
    self.prem = 0
    self.prem_freq = prem_freq
def calc(self):

```

```

if self.prem_freq == 11:
    self.prem = ((mort_tab_M.iat[age_at_entry, 7] - mort_tab_M.iat[age_at_entry + pol_term_y, 7] + mort_tab_
M.iat[age_at_entry + pol_term_y, 3])
                / (mort_tab_M.iat[age_at_entry, 3]))
else:
    self.prem = ((mort_tab_M.iat[age_at_entry, 7] - mort_tab_M.iat[age_at_entry + pol_term_y, 7] + mort_tab_
M.iat[age_at_entry + pol_term_y, 3])
                / (mort_tab_M.iat[age_at_entry, 4] - mort_tab_M.iat[age_at_entry + pol_term_y, 4]))
return self.prem
def get_value(self):
    return self.prem

```

```
class CGROSS_PREM:
```

```

#Class which build Gross premium on the calculated Net premium
def __init__(self, NET_PREM, PVP_RES):
    self.GROSS_PREM = 0
    self.NET_PREM = NET_PREM
    self.PVP_RES = PVP_RES
def calc(self, age, prem_freq):
    self.age = age
    self.prem_freq = prem_freq
    self.PREM_PAID_PP = {}
    self.PVP_RES.recalc(0)
    if prem_freq == 11:
        self.GROSS_PREM = (SA * ((self.NET_PREM.calc() + other_assumptions["alpha_SA"][policy_type] + oth
er_assumptions["beta"][policy_type]*PVP_RES.PVP[1])
                              / (1 - other_assumptions["alpha_prem"][policy_type])))
    else:
        self.GROSS_PREM = (SA * ((self.NET_PREM.calc() + other_assumptions["alpha_SA"][policy_type]/PVP
_RES.PVP[1] + other_assumptions["beta"][policy_type])
                              / (1 - other_assumptions["alpha_prem"][policy_type] / PVP_RES.PVP[1] - other_assumptions["g
amma"][policy_type])))
    for i in range(1, (pol_term_y - self.age)*12 + 1):
        if (i%(12/self.prem_freq) == 1) or self.prem_freq == 12:
            self.PREM_PAID_PP[i] = self.GROSS_PREM / self.prem_freq
        else:
            self.PREM_PAID_PP[i] = 0
    self.PREM_PAID_DF = pd.DataFrame.from_dict(self.PREM_PAID_PP, orient="index")
def get_value(self):
    return self.GROSS_PREM

```

```
class CUPR:
```

```

#class that calculated UPR for premium frequency different from monthly freq
def __init__(self, prem_freq, NET_PREM, age):
    self.age = age
    self.prem_freq = prem_freq
    self.NET_PREM = NET_PREM
    self.UPR = {}
    self.pos = 0
    self.months = int(12 / self.prem_freq)
def recalc(self):
    self.UPR[1]=0

```

```

if prem_freq == 11:
    pass
else:
    for k in range(1,pol_term_y - self.age + 1):
        for j in range(self.prem_freq):
            for i in range(1,self.months + 1):
                self.UPR[12*(k-1)+j * self.months + i + 1] = SA * self.NET_PREM.prem / self.prem_freq * (self.months - i)/(self.months)
            self.UPR_DF = pd.DataFrame.from_dict(self.UPR, orient="index")
def get_value(self,pos):
    return self.UPR[pos]

```

```

class CNO_POLS:

```

```

#class implementing the decrement model

```

```

#the output of the class are the vectors of number of policies at the beginning of the month, end of the month

```

```

#number of surrenders, number of deaths and part of the policy which remains for the maturity

```

```

def __init__(self):

```

```

    self.NO_POLS_IF = {}

```

```

    self.NO_POLS_IFSM = {}

```

```

    self.NO_SURRS = {}

```

```

    self.NO_DTHS = {}

```

```

    self.NO_MATS = {}

```

```

    self.NO_DTHS = {}

```

```

    self.qx = {}

```

```

def recal(self,age):

```

```

    self.age = age

```

```

    self.age_help = age

```

```

    self.LAPSE_RATE={}

```

```

#lapse rate calculation

```

```

for i in range(1,(pol_term_y - self.age)*12 + 1):

```

```

    if self.age_help>=len(lapse_rates.index):

```

```

        if i == 1:

```

```

            self.LAPSE_RATE[i] = 1-(1-(lapse_rates["1"][len(lapse_rates.index)-1]))**(1/12)

```

```

        else:

```

```

            self.LAPSE_RATE[i] = self.LAPSE_RATE[(i-1)]

```

```

    else:

```

```

        self.LAPSE_RATE[i] = 1-(1-(lapse_rates["1"][self.age_help]))**(1/12)

```

```

    if ((i%12 == 0) and (i>0)):

```

```

        self.age_help=self.age_help+1

```

```

self.DTH_EXP = {}

```

```

self.age_help=self.age

```

```

#experiance rates calculation

```

```

for i in range(1,(pol_term_y - self.age)*12 + 1):

```

```

    if self.age_help>=len(mort_exp.index):

```

```

        if i == 1:

```

```

            self.DTH_EXP[i] = mort_exp["1"][len(mort_exp.index)-1]

```

```

        else:

```

```

            self.DTH_EXP[i] = self.DTH_EXP[i-1]

```

```

    else:

```

```

        self.DTH_EXP[i] = mort_exp["1"][self.age_help]

```

```

    if ((i%12 == 0) and (i>0)):
        self.age_help=self.age_help+1

self.MAT_RATE = {}
for i in range(1,(pol_term_y - self.age)*12 + 1):
    if i==((pol_term_y - self.age)*12):
        self.MAT_RATE[i]=1
    else:
        self.MAT_RATE[i]=0

#calculation of above mentioned vectors
for i in range(1,(pol_term_y - self.age)*12 + 1):
    self.age_help=self.age
    if i == 1:
        self.NO_POLS_IFSM[i] = mp_file.iat[policy,9]
    else:
        self.NO_POLS_IFSM[i] = self.NO_POLS_IFSM[i-1]
    self.NO_DTHS[i]=self.NO_POLS_IFSM[i] * mort_tab_M.iat[age_at_entry + self.age_help, 1] * self.DTH_
EXP[i] / 12
    self.qx[i] = mort_tab_M.iat[age_at_entry + self.age_help, 1] * self.DTH_EXP[i] / 12
    if i == ((pol_term_y - self.age)*12 - 1):
        self.NO_SURRS[i] = 0
    else:
        self.NO_SURRS[i] = (self.NO_POLS_IFSM[i] - self.NO_DTHS[i])* self.LAPSE_RATE[i]
    if i == ((pol_term_y - self.age)*12):
        self.NO_MATS[i] = self.NO_POLS_IFSM[i] - self.NO_DTHS[i]
    else:
        self.NO_MATS[i] = 0
    self.NO_POLS_IF[i] = self.NO_POLS_IFSM[i] - self.NO_DTHS[i] - self.NO_SURRS[i] - self.NO_MATS[
i]
    if i%12 == 0:
        self.age_help = self.age_help + 1
    else:
        pass
self.NO_SURRS_DF = pd.DataFrame.from_dict(self.NO_SURRS, orient="index")
self.NO_DTHS_DF = pd.DataFrame.from_dict(self.NO_DTHS, orient="index")
self.NO_MATS_DF = pd.DataFrame.from_dict(self.NO_MATS, orient="index")
self.NO_POLS_IF_DF = pd.DataFrame.from_dict(self.NO_POLS_IF, orient="index")
self.NO_POLS_IFSM_DF = pd.DataFrame.from_dict(self.NO_POLS_IFSM, orient="index")
class CRATES:

#Class calculating the dataframe which consists of the discount factors for each economic scenario
#the output of the class are investment rate vector and discount rates matrix
def __init__(self):
    self.INV_RATE = {}
    self.DISC_RATE = {}
    self.DISC_FAC = {}
    self.DISC_RATE_DF = pd.DataFrame()
    self.DISC_FAC_DF = pd.DataFrame()
    self.INV_RATE_DF = pd.DataFrame()
    for i in range(1,len(rates.index)*12+1):
        if (i-1)//12 < len(rates.index):
            self.INV_RATE[i] = (1+(rates[0][(i-1)//12]))**(1/12)-1
        else:

```

```
self.INV_RATE[i] = self.INV_RATE[i-1]
self.INV_RATE_DF = pd.DataFrame()
```

```
for j in range(1,n_curves):
    for i in range(1,len(rates.index)*12+1):
        self.DISC_RATE[i] = (1+(rates[j][(i-1)//12]))**(1/12)-1
        if i == 1:
            self.DISC_FAC[i] = 1 / (1 + self.DISC_RATE[i])
        else:
            self.DISC_FAC[i] = self.DISC_FAC[i-1] / (1 + self.DISC_RATE[i])
    if j == 1:
        self.DISC_RATE_DF = pd.DataFrame.from_dict(self.DISC_RATE, orient = "index")
        self.DISC_RATE_DF = self.DISC_RATE_DF.rename(columns = {0:1})
        self.DISC_FAC_DF = pd.DataFrame.from_dict(self.DISC_FAC, orient = "index")
        self.DISC_FAC_DF = self.DISC_FAC_DF.rename(columns = {0:1})
    else:
        self.DISC_RATE_DF[j] = pd.DataFrame.from_dict(self.DISC_RATE, orient = "index")
        self.DISC_FAC_DF[j] = pd.DataFrame.from_dict(self.DISC_FAC, orient = "index")
```

```
def recalc_age(self,age,index):
    self.age = age
    self.age_help = age
```

```
class CAGE:
```

```
#class with age output
def __init__(self,age_at_entry,start_date):
    self.age= age
    self.start_date = start_date
```

```
class CACC_BON:
```

```
#class calculating the accrued bonus
#the output is accrued bonus vector
def __init__(self,init_decb_pp,MATH_RES,ECO_RATES):
    self.init_decb_pp = init_decb_pp
    self.ECO_RATES = ECO_RATES
    self.MATH_RES = MATH_RES
    self.ACC_BON_GROSS_PP = {}
    self.ACC_BON_PP = {}
def recal(self,age):
    self.age = age
    self.monthly_tir = (1+tir) ** (1/12) -1
    for i in range(1, (pol_term_y - self.age) * 12 + 1):
        if i == 1:
            self.ACC_BON_GROSS_PP[i] = self.init_decb_pp * (1+self.ECO_RATES.INV_RATE[i]) + self.MATH_RES[i+1] / (1+self.monthly_tir) * max(0,self.ECO_RATES.INV_RATE[i] - self.monthly_tir)
            self.ACC_BON_PP[i] = self.init_decb_pp * (1 + self.ECO_RATES.INV_RATE[i] - invest_margin / 12) + self.MATH_RES[i+1] / (1+self.monthly_tir) * max(0,self.ECO_RATES.INV_RATE[i] - self.monthly_tir - invest_margin/12)
        else:
            self.ACC_BON_GROSS_PP[i] = self.ACC_BON_GROSS_PP[i - 1] * (1+self.ECO_RATES.INV_RATE[i]) + self.MATH_RES[i+1] / (1+self.monthly_tir) * max(0,self.ECO_RATES.INV_RATE[i] - tir)
```

```

        self.ACC_BON_PP[i] = (self.ACC_BON_PP[i - 1]) * (1 + self.ECO_RATES.INV_RATE[i] - invest_mar
gin / 12) + self.MATH_RES[i+1] / (1+self.monthly_tir) * max(0,self.ECO_RATES.INV_RATE[i] - self.monthly_tir
 - invest_margin/12)
class CNET_RES:

```

```

#class calculating the net reserve based on present value of future benefits and present value of future premiums
#the output is a vector of net reserve for each projection month
#there is also a method to get net reserve value for particular month

```

```

def __init__(self):
    self.NET_RES = {}
def recalc(self,PVB_RES,PVP_RES,NET_PREM,age):
    self.age = age
    self.PVB_RES = PVB_RES
    self.PVP_RES = PVP_RES
    self.PVB_RES.recalc(self.age)
    self.PVP_RES.recalc(self.age)
    self.NET_PREM = NET_PREM.calc()
    for i in range(1,(pol_term_y - self.age) * 12 + 2):
        if prem_freq == 11:
            self.NET_RES[i] = SA * self.PVB_RES.PVB[i]
        else:
            self.NET_RES[i] = SA * (self.PVB_RES.PVB[i] - self.NET_PREM * PVP_RES.PVP[i])
    self.NET_RES_DF = pd.DataFrame.from_dict(self.NET_RES, orient="index")
def get_value(self,month):
    self.month = month
    return self.NET_RES[month]

```

```

class COUTGO:

```

```

#class holding the values of per policy surrender value, death benefit and maturity benefit
#the attribute

```

```

def __init__(self):
    self.SURR_VAL_PP = {}
    self.DEATH_BEN_PP = {}
    self.MAT_BEN_PP = {}
def recalc(self,MATH_RES,ACC_BON,age):
    self.age = age
    self.MATH_RES = MATH_RES
    self.ACC_BON = ACC_BON
    for i in range(1,(pol_term_y - self.age) * 12 + 1):
        self.SURR_VAL_PP[i] = max(0,self.MATH_RES[i+1] + self.ACC_BON[i] - other_assumptions["surr_char
ge"][1])
        self.DEATH_BEN_PP[i] = max(SA,self.MATH_RES[i+1]) + self.ACC_BON[i]
        self.MAT_BEN_PP[i] = max(SA,self.MATH_RES[i+1]) + self.ACC_BON[i]
    self.DTH_BEN_DF = pd.DataFrame.from_dict(self.DEATH_BEN_PP, orient="index")
    self.SURR_VAL_DF = pd.DataFrame.from_dict(self.SURR_VAL_PP, orient="index")
    self.MAT_BEN_DF = pd.DataFrame.from_dict(self.MAT_BEN_PP, orient="index")

```

```

class CCOMM_EXP:

```

```

#class calculating the per policy commissions and expenses for each projection month

```

```

def __init__(self):
    self.COMM_PP = {}
    self.EXP_PP = {}
def recalc(self, age,GROSS_PREM):

```

```

self.age = age
self.age_help = age
self.GROSS_PREM = GROSS_PREM
try:
    other_assumptions["init_comm_SA"]
except:
    print(policy)
for i in range(1, (pol_term_y - self.age) * 12 + 1):
    if (self.age == 0 and i == 1):
        self.COMM_PP[i] = other_assumptions["init_comm_SA"][policy_type] * SA + other_assumptions["init_comm_prem"][policy_type] * self.GROSS_PREM
        self.EXP_PP[i] = other_assumptions["init_exp_fix"][policy_type] * SA + other_assumptions["init_exp_prem"][policy_type] / 12 * self.GROSS_PREM
    elif (i % 12 == 1):
        self.COMM_PP[i] = other_assumptions["ren_comm_SA"][policy_type] * SA + other_assumptions["ren_comm_prem"][policy_type] * self.GROSS_PREM
        self.EXP_PP[i] = other_assumptions["ren_exp_fix"][policy_type] * SA + other_assumptions["ren_exp_prem"][policy_type] / 12 * self.GROSS_PREM
    else:
        self.COMM_PP[i] = 0
        self.EXP_PP[i] = other_assumptions["ren_exp_fix"][policy_type] * SA + other_assumptions["ren_exp_prem"][policy_type] / 12 * self.GROSS_PREM
    if i % 12 == 0:
        self.age_help = self.age_help + 1
    else:
        pass
self.EXP_PP_DF = pd.DataFrame.from_dict(self.EXP_PP, orient="index")
self.COMM_PP_DF = pd.DataFrame.from_dict(self.COMM_PP, orient="index")

```

#the main part of the code

#-----

#initialize the cash flow dataframe

```
CASH_FLOW_ALL = pd.DataFrame(index=list(range(1, 601)), columns = [0]).fillna(0)
```

```
CASH_FLOW_ALL.to_excel("empty_frame.xlsx")
```

```
RATES = CRATES()
```

#main loop from start policy to the end policy

for policy in range(start_policy, end_policy):

#policy input

```
entry_date = datetime.datetime.strptime(mp_file["Inception date"][policy], '%d.%m.%Y')
```

```
policy_type = int(mp_file.iat[policy, 1]) - 1
```

```
age_at_entry = mp_file.iat[policy, 3]
```

```
sex = mp_file.iat[policy, 4]
```

```
pol_term_y = mp_file.iat[policy, 5]
```

```
SA = mp_file.iat[policy, 6]
```

```
prem_freq = mp_file.iat[policy, 7]
```

```
CV_freq = mp_file.iat[policy, 8]
```

```
count = mp_file.iat[policy, 9]
```

```
age_at_start = val_date.year - entry_date.year
```

```
CV_start = mp_file.iat[policy, 8]
```

#initialize instances of above classes and calculate cash flows

```
PVB_RES = CPVB_RES()
```

```

PVP_RES = CPVP_RES()
PVB_RES.recalc(age_at_start)
PVP_RES.recalc(age_at_start)
NET_PREM = CNET_PREM()
NET_PREM.calc()
NET_RES = CNET_RES()
NET_RES.recalc(PVB_RES,PVP_RES,NET_PREM,age_at_start)
UPR = CUPR(prem_freq, NET_PREM, age_at_start)
UPR.recalc()
for i in range(1,(pol_term_y - age_at_start)*12 + 2):
    if prem_freq == 11:
        MATH_RES[i] = NET_RES.NET_RES[i] + SA * other_assumptions["beta"][policy_type] * PVP_RES.PVP
[i]
    else:
        MATH_RES[i] = UPR.UPR[i] + NET_RES.NET_RES[i]
ACC_BON = CACC_BON(CV_start,MATH_RES,RATES)
ACC_BON.recalc(age_at_start)
OUTGO = COUTGO()
OUTGO.recalc(MATH_RES,ACC_BON.ACC_BON_PP,age_at_start)
GROSS_PREM = CGROSS_PREM(NET_PREM,PVP_RES)
GROSS_PREM.calc(age_at_start,prem_freq)
COMM_EXP= CCOMM_EXP()
COMM_EXP.recalc(age_at_start,GROSS_PREM.GROSS_PREM)
NO_POLS = CNO_POLS()
NO_POLS.recalc(age_at_start)

#cashflows for each policy in terms of premium, commissions and expenses
CASH_FLOW_POLS = pd.DataFrame(GROSS_PREM.PREM_PAID_DF[0] - COMM_EXP.EXP_PP_DF[0] -
COMM_EXP.COMM_PP_DF[0])

#all cashflows melted together for each projection month weighted by the number of policies, surrenders etc....
CASH_FLOW = pd.DataFrame(pd.DataFrame(NO_POLS.NO_POLS_IFSM_DF.values * CASH_FLOW_POLS.
values, columns = NO_POLS.NO_POLS_IFSM_DF.columns, index = NO_POLS.NO_POLS_IFSM_DF.index)[0]
- pd.DataFrame(NO_POLS.NO_DTHS_DF.values * OUTGO.DTH_BEN_DF.values,columns = NO_PO
LS.NO_DTHS_DF.columns, index = NO_POLS.NO_DTHS_DF.index)[0]
- pd.DataFrame(NO_POLS.NO_SURRS_DF.values * OUTGO.SURR_VAL_DF.values, columns = NO_
POLS.NO_SURRS_DF.columns, index = NO_POLS.NO_SURRS_DF.index)[0]
- pd.DataFrame(NO_POLS.NO_MATS_DF.values * OUTGO.MAT_BEN_DF.values, columns = NO_P
OLS.NO_MATS_DF.columns, index = NO_POLS.NO_MATS_DF.index)[0])

#summing up cashflows for all policies
if policy == start_policy:
    CASH_FLOW_ALL = CASH_FLOW_ALL.add(-CASH_FLOW, fill_value=0)
else:
    CASH_FLOW_ALL = CASH_FLOW_ALL.add(-CASH_FLOW, fill_value=0)

#when all cashflows for all calculated policies are calculated enomic scenarios discount for given number of curves i
s calculated
#also time needed to process every scenario is stored
DISC_CF_ALL_TEST={}
DISC_CF_ALL = []
DISC_VALUES_Y = {}
TIMES_TEST = {}

```

```
TIMES=[]
n = n_curves
z_alpha = norm.ppf(0.975)
```

```
#distinguishing between run types because each run uses different approaches
if run_number == 1:
```

```
    #base run with no variance reduction method
    #the recursive approach for calculating the mean and variance are used after
    #the scenario number 2000 to get the same approach for all runs
    row = 0
    mu_est = 0
    sigma_est = 0
    mu_m1 = 0
    RUN_RESULT_1_LIST = []
    for curve in range(1,n):
        row_m1 = row
        row = np.dot(RATES.DISC_FAC_DF[curve],CASH_FLOW_ALL)[0]
        DISC_CF_ALL.append(row)
        elapsed_time = time.time() - start_time
        if curve<2000:
            mu_est = np.mean(DISC_CF_ALL)
            sigma_est_sq = np.var(DISC_CF_ALL)
        else:
            #recursive calculation after each scenario
            mu_m1 = mu_est
            mu_est = mu_est + (row - mu_est)/(curve)
            sigma_est_m1 = sigma_est_sq
            sigma_est_sq = (1-1/(curve-1))*sigma_est_m1 + (curve)*(mu_est - mu_m1)**2
```

```
    #output after each scenario mu estimate, sigma estimate and time needed to calculate it
```

```
    row_output = {"mu":mu_est,
                  "st":np.sqrt(sigma_est_sq),
                  "time": elapsed_time}
```

```
    RUN_RESULT_1_LIST.append(row_output)
```

```
    #list with run 1 results and output to excel
```

```
    RUN_1_RESULT_DF = pd.DataFrame(RUN_RESULT_1_LIST)
```

```
    print(elapsed_time)
```

```
    RUN_1_RESULT_DF.to_excel("RUN_1_RESULT.xlsx")
```

```
elif run_number ==2:
```

```
    #second run uses the antithetic variates method so there are
```

```
    #two parallel recursive calculation for each part of the antithetic pair
```

```
    curve = 1
```

```
    RUN_RESULT_2={}

```

```
    finished = False
```

```
    mu_est1 = 0
```

```
    sigma_est = 0
```

```
    sigma_est_sq = 0
```

```
    elapsed_time = 0
```

```
    RUN_RESULT_2_LIST = []
```

```
    row = 0
```

```
    while not finished:
```

```

row_m1 = row
row = np.dot(RATES.DISC_FAC_DF[curve],CASH_FLOW_ALL)[0]
DISC_CF_ALL.append(row)
elapsed_time = time.time() - start_time
if curve<2000:
    pass
elif curve == 2000:

    #antithetic pairs mu_x and mu_y estimated after 2000 scenarios
    #sigma estimates for atithetic pairs
    mu_test=mu_est = sum(DISC_CF_ALL)/len(DISC_CF_ALL)
    mu_x = sum(DISC_CF_ALL[:,2])/len(DISC_CF_ALL)/2
    mu_y = sum(DISC_CF_ALL[1:,2])/len(DISC_CF_ALL)/2
    sigma_x_est_sq = np.var(DISC_CF_ALL[1:,2])
    sigma_y_est_sq = np.var(DISC_CF_ALL[:,2])
    sigma_est = 0
    cov_est = np.cov(DISC_CF_ALL[1:,2],DISC_CF_ALL[0:,2])[0][1]
    sigma_est_sq_x=sigma_est_sq1 = np.var(DISC_CF_ALL)
    num_x = curve/2
else:
    #odd and even scenarios recursive calculations
    if curve%2 == 0:
        x = row
        mu_x = mu_x + (row - mu_x)/(curve/2)
        cov_est = (curve/2-1)/(curve/2)*cov_est+1/(curve/2-1)*(x-mu_x)*(y-mu_y)
        mu_x = sum(DISC_CF_ALL[:,2])/len(DISC_CF_ALL)/2

    else:
        y=row
        mu_y = mu_y + (row - mu_y)/((curve-1)/2)
        mu_m1 = mu_est
        mu_est = mu_est + (row - mu_est)/(curve)
        sigma_est_m1 = sigma_est_sq_x
        sigma_est_sq_x =(1-1/(curve-1))*sigma_est_m1 + (curve)*(mu_est - mu_m1)**2

        sigma_est_sq = 1/4*(2*sigma_est_sq_x + 2*cov_est)
        sigma_est = np.sqrt(sigma_est_sq)
        l_bound = (mu_est1 - z_alpha* sigma_est / (np.sqrt(curve)))
        u_bound = (mu_est1 + z_alpha* sigma_est / (np.sqrt(curve)))

curve = curve + 1
if curve == 49998:
    finished = True

#output after each scenario mu estimate, sigma estimate and time needed to calculate it
row_output = {"mu":mu_est,
              "st":sigma_est,
              "time": elapsed_time}
RUN_RESULT_2_LIST.append(row_output)

```

#again there is list with run 2 results with mu and sigma estimated and time needed to calculate them after each scenario

```

DISC_CF_ALL_DF = pd.DataFrame(DISC_CF_ALL)
RUN_2_RESULT_DF = pd.DataFrame(RUN_RESULT_2_LIST)
run_result = [l_bound, u_bound, mu_est1, sigma_est, elapsed_time]
run_result_df = pd.DataFrame(run_result)
print(elapsed_time)

```

```

RUN_2_RESULT_DF.to_excel("RUN_2_RESULT.xlsx")
run_result_df.to_excel("run_result2.xlsx")

```

```

elif run_number == 3:

```

```

#the third run calculated the control- variate method
RUN_RESULT_3_LIST = []
curve = 1
finished = False
yearly_zcbs = RATES.DISC_FAC_DF.iloc[11::12][:25]
yearly_zcbs.index = range(25)
zcbs_input = pd.read_csv("zcbs.csv", sep = ";")
zcbs = zcbs_input[:25]
row = 0
y_est = 0
while not finished:

```

```

    #we use 2000 scenarios after which the estimates of beta and XY covariance used for control variate estimate a
    re estimated

```

```

    row_m1 = row
    row = np.dot(RATES.DISC_FAC_DF[curve],CASH_FLOW_ALL)[0]
    DISC_CF_ALL.append(row)
    elapsed_time = time.time() - start_time
    if curve<2000:
        y_est_test = np.mean(DISC_CF_ALL)
        sigma_est_sq = np.var(DISC_CF_ALL)
    elif curve == 2000:
        #control variates estimated calculation
        BEL_array_norm = DISC_CF_ALL - np.mean(DISC_CF_ALL)
        RATES_norm = yearly_zcbs.T[0:curve] - yearly_zcbs.T[0:curve].mean()

```

```

    #covariance of Y and X i.e. the multiplication of standardized BEL and standardized discount
    #factor for which we know the mean - the market values then divided by (n-1)
    cov_xy = np.dot(RATES_norm.T,BEL_array_norm)/(len(BEL_array_norm) -1)

```

```

    #covariance of discount factors
    cov_x = np.cov(yearly_zcbs[0:curve])

```

```

    #estimate of Beta
    beta_est = np.dot(np.linalg.inv(cov_x),cov_xy)
    mu_x_est = mu_x_test = mu_x_test = yearly_zcbs.T[0:curve].mean()

```

```

    #CV estimator
    y_est_test = np.mean(DISC_CF_ALL) - np.dot((yearly_zcbs.T[0:curve].mean() - zcbs["ZCBS"]),beta_est)

```

```

    #Var estimate
    sigma_array =(DISC_CF_ALL - y_est_test - np.dot((yearly_zcbs.T[0:curve] - zcbs["ZCBS"]),beta_est))**2

```

```

#loss factor and variance factors
loss_factor = 1/curve+1/(curve-1)*np.dot(np.dot((mu_x_est - zcbs["ZCBS"]).T,np.linalg.inv(cov_x)),(mu_x
_est - zcbs["ZCBS"]))
variance_factor = 1/(curve-25)*sigma_array.sum()
sigma_est_sq = loss_factor *variance_factor
mu_y_est = mu_y_test = np.mean(DISC_CF_ALL)
SS_n = np.sum((DISC_CF_ALL - y_est_test - np.dot((yearly_zcbs.T[0:curve] - zcbs["ZCBS"]),beta_est))**
2)* 1/(curve-51)
cov_x_inverse = np.linalg.inv(cov_x)
else:
#recursive calculations using beta estimated after 2000 scenarios
mu_x_testm1 = mu_x_test
mu_x_test = mu_x_test + (yearly_zcbs[curve] - mu_x_test)/curve
cov_xm1 = cov_x
cov_x = (1-1/(curve-26))*cov_xm1 + (curve-25)*np.dot((mu_x_test - mu_x_testm1),(mu_x_test - mu_x_test
m1))
cov_x_inverse = np.linalg.inv(cov_x)
mu_y_test = mu_y_test + (row - mu_y_test)/(curve)

#CV estimator
y_est_test_m1 = y_est_test
y_est_test = mu_y_test - np.dot((mu_x_test - zcbs["ZCBS"]),beta_est)

#Var estimate
SSnm1 = SS_n
SS_n = (1-1/(curve-26))*SSnm1 + (curve-25)*(y_est_test - y_est_test_m1)**2
loss_factor = 1/curve+1/(curve-1)*np.dot(np.dot((mu_x_test - zcbs["ZCBS"]).T,cov_x_inverse),(mu_x_test -
zcbs["ZCBS"]))
variance_factor = SS_n
sigma_est_sq = loss_factor *variance_factor
curve = curve + 1
if curve == 49998:
finished = True

sigma_est = np.sqrt(sigma_est_sq)

#output after each scenario mu estimate, sigma estimate and time needed to calculate it
row_output = {"mu":y_est_test,
"st":sigma_est,
"time": elapsed_time}
RUN_RESULT_3_LIST.append(row_output)

finished = False
control_variate_est = 0

#output for control variates estimator
DISC_CF_ALL_DF = pd.DataFrame(DISC_CF_ALL)
RUN_3_RESULT_DF = pd.DataFrame(RUN_RESULT_3_LIST)
print(elapsed_time)

RUN_3_RESULT_DF.to_excel("RUN_3_RESULT.xlsx")

elif run_number ==4:

```

```

#Integrated control variates method
# the calculation is very similar to the calculation of standard control variate but there are
#antithetic scenarios used and one must handle each part of the antithetic pair separately
RUN_RESULT_4_LIST = []
curve = 1
RUN_RESULT_4={}
finished = False
yearly_zcbs = RATES.DISC_FAC_DF.iloc[11::12][:25]
yearly_zcbs.index = range(25)
spot_rates_CV = 1/yearly_zcbs - 1
zcbs_input = pd.read_csv("zcbs.csv", sep = ";")
zcbs = zcbs_input[:25]
row = 0
y_est = 0
cov_y=0
while not finished:
    row_m1 = row
    row = np.dot(RATES.DISC_FAC_DF[curve],CASH_FLOW_ALL)[0]
    DISC_CF_ALL.append(row)
    elapsed_time = time.time() - start_time
    if curve<2000:
        y_est_test = np.mean(DISC_CF_ALL)
        sigma_est_sq = np.var(DISC_CF_ALL)
    elif curve == 2000:

        #again beta estimate and cov XY estimate as in the run 3
        BEL_array_norm = DISC_CF_ALL - np.mean(DISC_CF_ALL)
        RATES_norm = yearly_zcbs.T[0:curve] - yearly_zcbs.T[0:curve].mean()
        cov_xy = np.dot(RATES_norm.T,BEL_array_norm)/(len(BEL_array_norm) -1)
        cov_x = np.cov(yearly_zcbs[0:curve])
        beta_est = np.dot(np.linalg.inv(cov_x),cov_xy)
        mu_x_est = mu_x_test = yearly_zcbs.T[0:curve].mean()

    #CV estimator
    y_est_test = np.mean(DISC_CF_ALL) - np.dot((yearly_zcbs.T[0:curve].mean() - zcbs["ZCBS"]),beta_est)

    #Var estimate
    #mu estimates for each antithetic pair
    mu_y1 = np.mean(DISC_CF_ALL[:,2])
    mu_ym1 = np.mean(DISC_CF_ALL[1::2])

    #correlation between discounted bel for each antithetic pair
    corr_y = np.corrcoef(DISC_CF_ALL[:,2],DISC_CF_ALL[1::2])[0][1]
    sigma_array =(DISC_CF_ALL - y_est_test - np.dot((yearly_zcbs.T[0:curve] - zcbs["ZCBS"]),beta_est))**2

    #loss factor and variance factor calculation
    loss_factor = 1/curve+1/(curve-1)*np.dot(np.dot((mu_x_est - zcbs["ZCBS"]).T,np.linalg.inv(cov_x)),(mu_x
_est - zcbs["ZCBS"]))
    variance_factor = 1/(curve-25)*(sigma_array.sum()) *(1+corr_y)
    sigma_est_sq = loss_factor *variance_factor
    mu_y_est = mu_y_test = np.mean(DISC_CF_ALL)
    SS_n = np.sum((DISC_CF_ALL - y_est_test - np.dot((yearly_zcbs.T[0:curve] - zcbs["ZCBS"]),beta_est))**
2)* 1/(curve-51)
    cov_x_inverse = np.linalg.inv(cov_x)

```

```

y_vector = DISC_CF_ALL - np.dot((yearly_zcbs.T[0:curve] - zcbs["ZCBS"]),beta_est)

else:
#recursive calculation
mu_x_testm1 = mu_x_test
mu_x_test = mu_x_test + (yearly_zcbs[curve] - mu_x_test)/curve
cov_xm1 = cov_x
cov_x = (1-1/(curve-26))*cov_xm1 + (curve-25)*np.dot((mu_x_test - mu_x_testm1),(mu_x_test - mu_x_test
m1))
cov_x_inverse = np.linalg.inv(cov_x)
mu_y_test = mu_y_test + (row - mu_y_test)/(curve)

#CV estimator
y_est_test_m1 = y_est_test
y_est_test = mu_y_test - np.dot((mu_x_test - zcbs["ZCBS"]),beta_est)

#Var estimate recursive
SSnm1 = SS_n
SS_n = (1-1/(curve-26))*SSnm1 + (curve-25)*(y_est_test - y_est_test_m1)**2
loss_factor = 1/curve+1/(curve-1)*np.dot(np.dot((mu_x_test - zcbs["ZCBS"]).T,cov_x_inverse),(mu_x_test -
zcbs["ZCBS"]))
variance_factor = SS_n
sigma_est_sq = loss_factor *variance_factor*(1+corr_y)
curve = curve + 1
if curve == 49998:
finished = True

sigma_est = np.sqrt(sigma_est_sq)

#output after each scenario mu estimate, sigma estimate and time needed to calculate it
row_output = {"mu":y_est_test,
"st":sigma_est,
"time": elapsed_time}
RUN_RESULT_4_LIST.append(row_output)

finished = False
control_variate_est = 0

#run four output to excel
DISC_CF_ALL_DF = pd.DataFrame(DISC_CF_ALL)
RUN_4_RESULT_DF = pd.DataFrame(RUN_RESULT_4_LIST)
print(elapsed_time)

RUN_4_RESULT_DF.to_excel("RUN_4_RESULT.xlsx")
elif run_number ==5:
#run 5 is built to run the control variate run i.e. the code for third run but for different control variates
#it means this calculates run 3 for different number of discount years used as control variate
#it can be distinguished which control variate is the most suitable one
RUN_OPTIMIZATION = []
for run in range(2,50):
DISC_CF_ALL = []
RUN_RESULT_5_LIST = []
curve = 1
RUN_RESULT_5={}

```

```

finished = False
yearly_zcbs = RATES.DISC_FAC_DF.iloc[11::12][:run]
yearly_zcbs.index = range(run)
spot_rates_CV = 1/yearly_zcbs - 1
zcbs_input = pd.read_csv("zcbs.csv", sep = ";")
zcbs = zcbs_input[:run]
row = 0
y_est = 0

while not finished:
    row_m1 = row
    row = np.dot(RATES.DISC_FAC_DF[curve],CASH_FLOW_ALL)[0]
    DISC_CF_ALL.append(row)
    elapsed_time = time.time() - start_time
    if curve<2000:
        y_est = np.mean(DISC_CF_ALL)
        sigma_est_sq = np.var(DISC_CF_ALL)
    elif curve == 2000:
        BEL_array_norm = DISC_CF_ALL - np.mean(DISC_CF_ALL)
        RATES_norm = yearly_zcbs.T[0:curve] - yearly_zcbs.T[0:curve].mean()
        cov_xy = np.dot(RATES_norm.T,BEL_array_norm)/(len(BEL_array_norm) -1)
        cov_x = np.cov(yearly_zcbs[0:curve])
        beta_est = np.dot(np.linalg.inv(cov_x),cov_xy)
        mu_x_est = mu_x_test = mu_x_test = yearly_zcbs.T[0:curve].mean()
        y_est = np.mean(DISC_CF_ALL) - np.dot((yearly_zcbs.T[0:curve].mean() - zcbs["ZCBS"]),beta_est)
        sigma_array = (DISC_CF_ALL - y_est - np.dot((yearly_zcbs.T[0:curve] - zcbs["ZCBS"]),beta_est))**2
        loss_factor = 1/curve+1/(curve-1)*np.dot(np.dot((mu_x_est - zcbs["ZCBS"]).T,np.linalg.inv(cov_x)),(mu
_x_est - zcbs["ZCBS"]))
        variance_factor = 1/(curve-run)*sigma_array.sum()
        sigma_est_sq = loss_factor *variance_factor
        mu_y_est = mu_y_test = np.mean(DISC_CF_ALL)
        SS_n = np.sum((DISC_CF_ALL - y_est - np.dot((yearly_zcbs.T[0:curve] - zcbs["ZCBS"]),beta_est))**2
* 1/(curve-51)
        cov_x_inverse = np.linalg.inv(cov_x)
    else:
        mu_x_test = mu_x_test + (yearly_zcbs[curve] - mu_x_test)/curve
        mu_y_test = mu_y_test + (row - mu_y_test)/(curve)
        y_est_test_m1 = y_est_test
        y_est_test = mu_y_test - np.dot((mu_x_test - zcbs["ZCBS"]),beta_est)
        SSnm1 = SS_n
        SS_n = (1-1/(curve-run-1))*SSnm1 + (curve-run)*(y_est_test - y_est_test_m1)**2
        loss_factor = 1/curve+1/(curve-1)*np.dot(np.dot((mu_x_test - zcbs["ZCBS"]).T,cov_x_inverse),(mu_x_te
st - zcbs["ZCBS"]))
        variance_factor = SS_n
        sigma_est_sq = loss_factor *variance_factor
    curve = curve + 1

if curve == 49998:
    finished = True
    run_output = {"run": run,
        "mu":y_est_test,
        "std":np.sqrt(sigma_est_sq),
        "time": elapsed_time}

```

```
sigma_est = np.sqrt(sigma_est_sq)

finished = False
control_variate_est = 0
RUN_OPTIMIZATION.append(run_output)

DISC_CF_ALL_DF = pd.DataFrame(DISC_CF_ALL)
RUN_OPTIMIZATION_DF = pd.DataFrame(RUN_OPTIMIZATION)
RUN_5_RESULT_DF = pd.DataFrame(RUN_RESULT_5_LIST)
print(elapsed_time)
RUN_OPTIMIZATION_DF.to_excel("run_optimization.xlsx")
else:
    print("not done yet")
```