Algorithmic Analysis of a General Class of Discrete-based Insurance Risk Models

by

Basil Singer

A thesis
presented to the University of Waterloo
in fulfillment of the
thesis requirement for the degree of
Philosophy of Doctorate
in
Actuarial Science

Waterloo, Ontario, Canada, 2013

I hereby declare that I am the sole author of this thesis. This is a true copy of the thesis, including any required final revisions, as accepted by my examiners.

I understand that my thesis may be made electronically available to the public.

Abstract

The aim of this thesis is to develop algorithmic methods for computing particular performance measures of interest for a general class of discrete-based insurance risk models. We build upon and generalize the insurance risk models considered by Drekic and Mera (2011) and Alfa and Drekic (2007), by incorporating a threshold-based dividend system in which dividends only get paid provided some period of good financial health is sustained above a pre-specified threshold level. We employ two fundamental methods for calculating the performance measures under the more general framework.

The first method adopts the matrix-analytic approach originally used by Alfa and Drekic (2007) to calculate various ruin-related probabilities of interest such as the trivariate distribution of the time of ruin, the surplus prior to ruin, and the deficit at ruin. Specifically, we begin by introducing a particular trivariate Markov process and then expressing its transition probability matrix in a block-matrix form. From this characterization, we next identify an initial probability vector for the process, from which certain important conditional probability vectors are defined. For these vectors to be computed efficiently, we derive recursive expressions for each of them. Subsequently, using these probability vectors, we derive expressions which enable the calculation of conditional ruin probabilities and, from which, their unconditional counterparts naturally follow.

The second method used involves the first claim conditioning approach (i.e., condition on knowing the time the first claim occurs and its size) employed in many ruin theoretic articles including Drekic and Mera (2011). We derive expressions for the finite-ruin time based Gerber-Shiu function as well as the moments of the total dividends paid by a finite

time horizon or before ruin occurs, whichever happens first. It turns out that both functions can be expressed in elegant, albeit long, recursive formulas.

With the algorithmic derivations obtained from the two fundamental methods, we next focus on computational aspects of the model class by comparing six different types of models belonging to this class and providing numerical calculations for several parametric examples, highlighting the robustness and versatility of our model class. Finally, we identify several potential areas for future research and possible ways to optimize numerical calculations.

Acknowledgments

I would like to thank my supervisor, Steve Drekic, for introducing and coaching me in the field of computational probability. He has truly been supportive every step of the way, checking my derivations and numerical calculations (which are quite demanding in this field) with the thoroughest detail possible, and, most importantly, inspiring and motivating me to push my boundaries towards excellence.

I would also like to thank my committee members—Professor Yi Lu, Professor Qi-Ming He, Professor Gordon Willmot, and Professor David Landriault—for agreeing to critique my thesis and prepare valuable feedback on the quality of my work.

Dedication

This is dedicated to my parents Deena Boraie and Karim Ibrahim, my grandfather Aly Boraie, in loving memory to my grandmother, Avys Singer, and to my first actuarial science mentor, Dr Ali Hadi. Thank you for all your love and support!

Table of Contents

List of Tables xi

List of Figures xii

Nomenclature xx

1 Literature Review 1

1.1 Introduction 1

1.1.1 A Brief History of Ruin Theory 1

1.1.2 Insurance Risk Models as Proxies for Company Solvency 4

1.2 Insurance Risk Models in the Literature 5

1.2.1 Works under the Continuous Framework 5

1.2.2 Surplus Processes with Dividend Systems 7

1.2.3 Improvements in Performance Measures 9

1.2.4 Works under the Discrete Framework 10

1.2.5 Properties of Discrete-based Delayed Sparre Andersen Models . . . 13

1.3 Along Came MAMs . . . 15
1.3.1 Motivation . . . 15
1.3.2 MAMs in Ruin Theory . . . 17
1.3.3 MAMs in Discrete-based Insurance Risk Models . . . 18

2 The Model Class **20**

2.1 The Surplus Process . . . 20
2.2 Model Class Characteristics . . . 21
2.3 Performance Measures . . . 24
2.4 Essential Notation, Functions and Derivations . . . 27
2.4.1 Basic Model Notation . . . 27
2.4.2 General Functions . . . 28

3 Algorithmic Expressions for Ruin Probabilities Using MAMs **34**

3.1 Additional Notation . . . 34
3.2 Constructing the Markov Process . . . 34
3.3 The Transition Probability Matrix . . . 37
3.4 Splitting the States and Defining Probability Vectors . . . 40
3.5 Efficient Methods for Computing Probability Vectors . . . 45
3.5.1 Ceiling Values . . . 45
3.5.2 Determining the Values of the Sub-vectors of $\underline{b}_i^{(k)}$ . . . 58
3.5.3 Determining the Values of the Sub-vectors of $\underline{g}_{n,i}^{(k)}$ . . . 60

3.5.4 Determining the Values of the Sub-vectors of $\underline{h}^{(k)}_{n,-j}$ 62

3.6 Determining the Conditional Trivariate and Bivariate Ruin Probabilities . 65

3.6.1 Calculating $\psi^{(k)}_{n,i,j}(u,d)$ 66

3.6.2 Calculating $\omega^{(k)}_{n,i}(u,d)$ 76

3.6.3 Calculating $\phi^{(k)}_{n,j}(u,d)$ 78

3.7 Determining the Unconditional Trivariate and Bivariate Ruin Probabilities 79

3.7.1 Calculating $\psi_{\tau,i,j}(u,d)$ 79

3.7.2 Calculating $\omega_{\tau,i}(u,d)$ 82

3.7.3 Calculating $\phi_{\tau,j}(u,d)$ 84

4 Algorithmic Expressions for Ruin-related Quantities Using the First Claim Conditioning Method **87**

4.1 Introduction 87

4.2 Determining the Finite-ruin Time Based Gerber-Shiu Function 89

4.3 Determining the Moments of the Total Dividends before a Finite Time Horizon 94

5 Analysis and Comparison of Six DISAMs Extracted from the General Class **101**

5.1 Introduction 101

5.2 Specifying the Above and Below Modes 101

5.3 Function Derivations 107

5.4 Numerical Examples 112

6 Conclusions and Future Research Ideas **121**

References **125**

Appendix: C++ Code **132**

List of Tables

5.1 Expected surplus prior to ruin values for $T \leq 100$ 118

5.2 Expected deficit at ruin values for $T \leq 100$ 118

5.3 Cumulative trivariate ruin probabilities of the form $\Psi_{100,\lceil I_{100} \rceil,\lceil J_{100} \rceil}(10,1)$. 119

5.4 Cumulative univariate ruin probabilities of the form $\Gamma_{100}(10,1)$ 119

5.5 Expected values of total dividends by 100 time units or before ruin occurs, whichever happens first 120

5.6 Standard deviations of total dividends by 100 time units or before ruin occurs, whichever happens first 120

List of Figures

1.1 A sample path of a continuous-time surplus process with $u = 2$ and $c = 2$. 6

1.2 A sample path of a continuous-time surplus process under a threshold-based dividend system with $u = 2$ and $c = 2$. 8

1.3 A sample path of a discrete-time surplus process with $u = 2$ and $c = 2$. . . 11

2.1 A sample path of a surplus process belonging to the model class with $u = 2$ and $c = 2$. 20

2.2 Values of T, U_{T^-}, and $|U_T|$ for a sample path of a discrete-time surplus process with $u = 2$, $c = 2$, and $b = 5$. 25

3.1 A sample evolution of the trivariate stochastic process with $u = 2$, $d = 1$, $c_1 = c_2 = 1$, $c = 2$, $b = 3$, $h = 3$, $k = 5$, $D_5 = 1$, and $0 < p < 1$. 36

3.2 Analysis of $\tilde{v}_u(z_u + h + 1)$ for a surplus process with $u = 1$, $d = 1$, $c_1 = 2 \leq c = 2$, $b = 5$, and $h = 2$. 50

3.3 Analysis of $\tilde{v}_u(z_u + h + 1)$ for a surplus process with $u = 1$, $d = 1$, $c_1 > c = 2$, $b = 5$, and $h = 2$. 52

5.1 A sample path of a Consecutive Countdown surplus process with $u = 2$, $d = 3$, $c_1 = 1$, $c_2 = 2$, $c = 2$, $b = 5$, $h = 3$, and $p = 1$. 103

5.2 A sample path of a Cyclic Countdown surplus process with $u = 2$, $d = 3$, $c_1 = 1$, $c_2 = 2$, $c = 2$, $b = 5$, $h = 3$, and $p = 1$. 103

5.3 A sample path of a Consecutive Freeze surplus process with $u = 4$, $d = 2$, $c_1 = 1$, $c_2 = 2$, $c = 2$, $b = 5$, $h = 3$, and $p = 1$. 104

5.4 A sample path of a Cyclic Freeze surplus process with $u = 2$, $d = 2$, $c_1 = 1$, $c_2 = 2$, $c = 2$, $b = 5$, $h = 3$, and $p = 1$. 105

5.5 A sample path of a Consecutive Reset surplus process with $u = 2$, $d = 1$, $c_1 = 1$, $c_2 = 2$, $c = 2$, $b = 5$, $h = 3$, and $p = 1$. 106

5.6 A sample path of a Cyclic Reset surplus process with $u = 2$, $d = 1$, $c_1 = 1$, $c_2 = 2$, $c = 2$, $b = 5$, $h = 3$, and $p = 1$. 106

Nomenclature

Abbreviations:

DISAM:	Delayed independent Sparre Andersen model.
iid:	Independent and identically distributed.
MAMs:	Matrix-analytic methods.
OISAM:	Ordinary independent Sparre Andersen model.
pmf:	Probability mass function.

Conventions:

If a and b are integers with $a > b$, then $a, a+1, \ldots, b$ is considered to be an empty sequence of points.

If a and b are integers with $a > b$, then $\sum_{\ell=a}^{b} f_\ell = 0$ for any arbitrary function f_ℓ.

$0^0 = 1$.

$0 \times \infty = \infty \times 0 = 0$.

Basic Notation:

$\mathbb{Z}^-$:	The set of negative integers (i.e., {-1,-2,-3,...}).
$\mathbb{N}$:	The set of natural numbers (i.e., {0,1,2,...}).
$\mathbb{Z}^+$:	The set of positive integers (i.e., {1,2,3,...}).

$\emptyset$: The empty set.

$\lfloor x \rfloor$: The floor function of x, yielding the largest integer that is less than or equal to x.

$\lceil x \rceil$: The ceiling function of x, yielding the smallest integer that is greater than or equal to x.

$(x-y)_+$: The maximum of $x-y$ and 0.

$mod(a,b)$: a modulo b for $a \in \mathbb{N}$ and $b \in \mathbb{Z}^+$. Note that $mod(0,b) = 0$.

$\mathbf{1}_A$: The indicator function of an event A, equal to 1 if A is true and 0 if A is false.

$\overline{A}$: The complement of an event A.

$|A|$: The cardinality of (i.e., number of elements in) a set A.

A': The transpose of a vector/matrix A.

$[A]_{m,n}$: The (m,n)th block-matrix of A.

Constants:

u: The initial surplus value.

d: The initial value of the dividend counter.

c: The premium amount per time unit.

b: The dividend threshold level.

n_r: The maximum time span until the first claim occurs.

n_a: The maximum interclaim time (i.e., the maximum time span between two consecutive claims).

y_0: The minimum claim size.

y_α: The maximum claim size.

c_1: The minimum unrestricted dividend amount.

c_2: The maximum unrestricted dividend amount.

h: The maximum dividend counter value, conveying when a potential dividend can be issued.

$\underline{0}$: A $1 \times n_a$ vector of zeros.

$\mathbf{0}$: An $n_a \times n_a$ matrix of zeros.

$\underline{e}_1$: A $1 \times n_a$ vector with 1 in the first element and zeros elsewhere.

Variables:

t: A general time unit value.

y: A general (i.e., realized) claim size value.

k: The realized time unit value at which the first claim occurs.

τ: A general (i.e., realized) time of ruin value.

i: A general surplus value.

j: A general (i.e., realized) deficit at ruin value.

s: A total dividend amount value before the current time period.

s^*: A dividend amount value at the current time period. If the current time period is a non-dividend paying time period, then $s^* = 0$.

T: The time of ruin random variable.

$U(t)$: The value of the surplus process at time t under a continuous-time insurance risk model.

$U(T^-)$: The surplus prior to ruin under a continuous-time insurance risk model.

$|U(T)|$: The deficit at ruin under a continuous-time insurance risk model.

U_t: The value of the surplus process at time t under a discrete-time insurance risk model.

U_{T^-}: The surplus prior to ruin under a discrete-time insurance risk model.

$|U_T|$: The deficit at time at ruin under a discrete-time insurance risk model.

Ω_k: The sample space of possible non-ruin values that the surplus process can take at time k.

$N(t)$: The number of claims by time t under a continuous-time insurance risk model, with $N(0)$ set equal to 0.

N_t: The number of claims by time t under a discrete-time insurance risk model, with N_0 set equal to 0.

W_1: The time until the first claim occurs, where $W_1 \in \{1, 2, \ldots, n_r\}$.

W_ℓ: The time between the $(\ell - 1)$th and ℓth claims (a.k.a. ℓth interclaim time) for $\ell \in \{2, 3, 4, \ldots\}$, where $W_\ell \in \{1, 2, \ldots, n_a\}$.

X_ℓ: The size of the ℓth claim for $\ell \in \mathbb{Z}^+$, under a continuous-time insurance risk model, where $X_\ell > 0$.

W_1: The time until the first claim occurs, where $W_1 \in \{1, 2, \ldots, n_r\}$.

W_ℓ: The time between the $(\ell - 1)$th and ℓth claims (a.k.a. ℓth interclaim time) for $\ell \in \{2, 3, 4, \ldots\}$, where $W_\ell \in \{1, 2, \ldots, n_a\}$.

X_ℓ: The size of the ℓth claim for $\ell \in \mathbb{Z}^+$, under a continuous-time insurance risk model, where $X_\ell > 0$.

Y_ℓ: The size of the ℓth claim for $\ell \in \mathbb{Z}^+$, under a discrete-time insurance risk model, where $Y_\ell \in \{y_0, y_0 + 1, \ldots, y_\alpha\}$.

E_t: The value of the elapsed time counter, at time t, since the occurrence of the last claim, starting from 1 at time 0, and resetting to 1 at every claim occurring time unit.

D_t: The value of the dividend counter at time t. A dividend is potentially issued at time $t + 1$ only when D_t is equal to h.

S_t: The total dividend amounts issued by time $t \in \mathbb{N}$, with $S_0 = 0$. Its range of values depends on knowing U_{t-1}, D_{t-1}, and possibly the value of a claim at time t.

$Order$: The event variable specifying the order of claim and dividend implementation should they both need to be processed at the same time unit. If the claim is processed before the dividend, then $Order$ is set equal to $ClmFrst$; otherwise, it is set equal to $\overline{ClmFrst}$.

Probabilities:

r_ω: Equal to $Pr\{W_1 = \omega\}$.

R_ω: Equal to $Pr\{W_1 > \omega\}$.

a_ω: Equal to $Pr\{W_\ell = \omega\}$ for $\ell \in \{2, 3, 4, \ldots\}$.

A_ω: Equal to $Pr\{W_\ell > \omega\}$ for $\ell \in \{2, 3, 4, \ldots\}$.

α_y: Equal to $Pr\{Y_\ell = y\}$ for $\ell \in \mathbb{Z}^+$.

Λ_y: Equal to $Pr\{Y_\ell > y\}$ for $\ell \in \mathbb{Z}^+$.

p: Equal to $Pr\{ClmFrst\}$.

d_ℓ: Equal to the probability that an unrestricted dividend is equal to ℓ.

$d^*_\ell\{i\}$: Equal to the probability that a restricted dividend is equal to ℓ, given that the surplus process is at level i immediately before the dividend is issued.

$f^{(n)}_{u,d}(s)$: The convolution probability mass function of the first n dividends, starting from an initial surplus of u and an initial dividend counter of d. By convention, $f^{(0)}_{u,d}(s) = \mathbf{1}_{[s=0]}$.

General Functions:

z_u: The earliest time the surplus process reaches or crosses the threshold level b from an initial surplus of u.

$z^*_{u,d}$: The earliest time a dividend can be issued, starting from an initial surplus of u and an initial dividend counter of d.

$\mathcal{T}_{u,d}(t)$: The collection of dividend paying time points up to and including time t, starting from an initial surplus of u and an initial dividend counter of d.

$P_{z^*_{u,d}}(n)$: The number of periods after $z^*_{u,d}$ required for n dividends to be issued.

$Div_i(x)$: The restricted value of an unrestricted dividend of size x, given that the surplus process is at level i immediately before the dividend is issued.

$TotDiv_n(x)$: The total amount of the first n dividends, each comprised by way of a constant unrestricted dividend of size x.

$\mathcal{R}_{u,d}(t)$: The collection of all possible total dividend amounts accumulated by time t, starting from an initial surplus of u and an initial dividend counter of d.

i_{min}: The smallest possible surplus prior to ruin value.

$\tilde{v}_u(t)$: An upper bound on the level the surplus process can achieve at time t, starting from an initial surplus of u.

$V_{u_1,d_1}(t_1, t_2, y, s, s^*, Order)$: The value of the dividend counter at time t_2, starting at time t_1 ($< t_2$) from a surplus of u_1 and a dividend counter of d_1, given that the only claim occurs at time t_2 with a size equal to y, the total amount of dividends over the time period $\{t_1 + 1, t_1 + 2, \ldots, t_2 - 1\}$ is equal to s, the dividend amount at time t_2 is equal to s^*, and the value of the event variable at time t_2 is equal to $Order$. At times when any of these parameters do not play a role in determining the value of the dividend counter, we replace them with the notation ".".

Functions Assuming No Claims before Time k:

$v_{u,d}(k)$: The maximum level the surplus process can achieve at time k from an initial surplus of u and an initial dividend counter of d, immediately before the first claim and possible dividend are implemented at time k.

$\tilde{v}_{u,d}(k, n)$: An upper bound on the level the surplus process can achieve at time $k + n$ from an initial surplus of u and an initial dividend counter of d, given that the first claim occurred at time k.

Performance Measures:

$\psi^{(k)}_{n,i,j}(u,d)$: Equal to $Pr\{T = k+n, U_{T^-} = i, |U_T| = j \ | U_k \in \Omega_k, U_0 = u, D_0 = d\}$.

$\omega^{(k)}_{n,i}(u,d)$: Equal to $Pr\{T = k+n, U_{T^-} = i \ | U_k \in \Omega_k, U_0 = u, D_0 = d\}$.

$\phi^{(k)}_{n,j}(u,d)$: Equal to $Pr\{T = k+n, |U_T| = j \ | U_k \in \Omega_k, U_0 = u, D_0 = d\}$.

$\psi_{\tau,i,j}(u,d)$: Equal to $Pr\{T = \tau, U_{T^-} = i, |U_T| = j \ | U_0 = u, D_0 = d\}$.

$\omega_{\tau,i}(u,d)$: Equal to $Pr\{T = \tau, U_{T^-} = i \ | U_0 = u, D_0 = d\}$.

$\phi_{\tau,j}(u,d)$: Equal to $Pr\{T = \tau, |U_T| = j \ | U_0 = u, D_0 = d\}$.

$\Psi_{\tau,i,j}(u,d)$: Equal to $Pr\{T \leq \tau, U_{T^-} \leq i, |U_T| \leq j \ | U_0 = u, D_0 = d\}$.

$\Gamma_\tau(u,d)$: Equal to $Pr\{T \leq \tau \ | U_0 = u, D_0 = d\}$.

$\widetilde{G}_{v,m}(u,d)$: Equal to $E[v^T w(U_{T^-}, |U_T|)\mathbf{1}_{[T\leq m]} | U_0 = u, D_0 = d]$ under a DISAM, where $v \in (0,1]$ is a discount rate and $w(U_{T^-}, |U_T|)$ is a specified penalty function.

$G_{v,m}(u,d)$: Equal to $E[v^T w(U_{T^-}, |U_T|)\mathbf{1}_{[T\leq m]} | U_0 = u, D_0 = d]$ under an OISAM, where $v \in (0,1]$ is a discount rate and $w(U_{T^-}, |U_T|)$ is a specified penalty function.

$\widetilde{\mathcal{D}}^r_m(u,d)$: Equal to $E[S^r_{\min\{T,m\}} | U_0 = u, D_0 = d]$ under a DISAM.

$\mathcal{D}^r_m(u,d)$: Equal to $E[S^r_{\min\{T,m\}} | U_0 = u, D_0 = d]$ under an OISAM.

Chapter 1

Literature Review

1.1 Introduction

1.1.1 A Brief History of Ruin Theory

The concept of insurance and pooling risk is quite an old one since the early formation of human civilizations. Ever since money came into existence, societies have found it extremely beneficial to develop safety mechanisms in the form of financial compensations for their citizens, in the face of possible personal misfortunes (e.g., floods, sickness, death of a family member). Consequently, the role of insurance became more pronounced. As a result, both private and public institutions were established to protect members and organizations against particular potential losses through financial compensation and/or services.

Nowadays, insurance has become a part of everyday life for both members and institutions, at least in developed societies. In the foreword of his 1984 book *The Historian and Business of Insurance*, Oliver Westall states—in the context of the British economy—that "the involvement of insurance in so many aspects of the economy has meant that its history has reflected the evolution of business and social life in a most intimate and revealing

way." Extending this quote, we may even be so bold to say that maintaining the well-being of these risk-protection agencies is crucial for the well-being of human society. Thus, it is essential to keep insurance companies financially healthy and productive, and minimize the risk of ruin (i.e., bankruptcy).

The research conducted in this thesis lies under the umbrella of ruin theory (a.k.a. collective risk theory), which mathematically studies the insurance company's vulnerability to being ruined (i.e., bankrupt) at some point in the future. This is done by mathematically formulating an insurance risk model which involves setting up at least one surplus process. A surplus process (a.k.a. risk reserve) is typically classified as the net financial amount of an insurance company after taking away operational expenses. Once the process/processes has/have been set up, the goal is then to calculate or derive expressions for particular ruin-related quantities known as performance measures or risk measures, such as the probability of ultimate ruin (i.e., the probability that an insurance company will eventually go bankrupt).

The first known publication of an insurance risk model was a compound Poisson risk model published in 1903 by Filip Lundberg, who can be considered as the father of ruin theory. The assumptions of his model were quite simple and the main objective was to derive the probability that an insurance company's surplus will become negative (i.e., get ruined) at some date in the future. Other risk measures examined by Lundberg and others in the early twentieth century involved upper and lower bounds on probabilities of ruin, which were particularly useful when calculating the probability of ruin was done manually.

In fact, it was not until the 1980's that the focus of the ruin theory literature became

oriented towards computationally intensive and practical models. By looking at recent publications (i.e., 1990's to late 2000's) in top actuarial and statistical journals, we find that there has been an increasing number of articles which focus on practical issues such as what is shown in the following list:

(1) Analysis of surplus models under a very general set of assumptions:

   (a) Multivariate models, of which an overview is given in Anastasiadis and Chuvoka (2012).

   (b) A model incorporating a constant interest rate, with claim instants that are governed by a Markovian arrival process, and claim sizes that are matrix exponentially distributed as in Mitric et al. (2012).

   (c) Dependent interclaim time and claim size models as in Willmot and Woo (2012) and Mihálykó and Mihálykó (2011).

   (d) A Lévy insurance risk model as in Bo et al. (2012) (note: there were errors in the original manuscript, which were corrected in the two-page follow-up Bo et al. (2013)).

   (e) A discrete renewal risk model as in Cossette et al. (2006).

(2) Use of statistical techniques in estimating parameters and calculating performance measures:

   (a) Using a hybrid of crude Monte Carlo and asymptotic estimation to estimate the finite-time probability of ruin in an $AR(1)$ model as in Tang and Yuan (2012).

   (b) Using the Log phase-type class to fit heavy-tailed data as in Ahn et al. (2012).

(c) Calculating the probability of ultimate ruin via a bootstrap approach as in Baumgartner and Gatto (2010).

(d) Calculating the adjustment coefficient in an $ARMA(p, q)$ risk model as in Christ and Steinebach (1995).

(3) Development of recursive techniques for computing different types of ruin probabilities efficiently for particular insurance risk models:

(a) A discrete-based insurance risk model with random dividends as in Drekic and Mera (2011).

(b) An Erlangian approximation to insurance risk models with phase-type interclaim times and claim sizes as in Stanford et al. (2011).

(c) An insurance risk model with phase-type distributed claim sizes as in Drekic et al. (2004).

Hence, as the literature becomes abundant with both realistic and mathematically tractable models, insurance companies will have a greater variety of quantitative tools and methods at their disposal.

1.1.2 Insurance Risk Models as Proxies for Company Solvency

Notwithstanding this, we note that insurance risk models can only play a minor role in analyzing an insurance company's financial well-being. This is because the assumptions made about the nature of the surplus process throughout time are overly simplistic. Certain extrinsic factors may set a completely different fate for the company's surplus should

they occur (source: Willmot (2011)). For instance, a company may face a sudden increase in competition (e.g., new players issuing insurance products with competitive prices) or a sudden loss of reputation (e.g., being subjected to multiple bad faith lawsuits). Either case will heavily require adaptive strategic planning that will render an insurance risk model obsolete since new forces (i.e., sources of expenses) will not be captured by the stochastic model. Hence, insurance risk models cannot be over-relied upon due to the dynamic nature of the real world.

Nonetheless, stochastic models can be useful in scenario testing and even in making short-term predictions (sources: Landriault (2011); Willmot (2011)). Thus, it would be useful for practitioners to include some of the insurance risk models available in the literature in analyzing the solvency of their businesses.

1.2 Insurance Risk Models in the Literature

1.2.1 Works under the Continuous Framework

The main focus in the literature has been on the formulation/analysis of insurance risk models under a continuous-time framework. The most typical model is a surplus process, starting with an initial capital, receiving a constant premium throughout time, and processing a random number of claims, each of which is of a random size. Mathematically, using notation commonly used in the literature, this process can be written as follows:

$$U(t) = u + ct - \sum_{\ell=1}^{N(t)} X_\ell \text{ for } t \geq 0 \text{ and with } U(0) = u \text{ and } N(0) = 0.$$

In the above equation, $U(t)$ represents the company's surplus at time t, u is the initial wealth of the insurance company, c is the constant premium rate per time unit, $N(t)$ is the

number of claims that have occurred by time t, and X_ℓ is the size of the ℓth claim. Note that this model has two sources of randomness: $N(t)$ and the X_ℓ's. The main objective of using this model (and more general ones of a similar nature) is to calculate the probability that the surplus of an insurance company would be 0 or less at some specified (or unspecified) future date, given an initial capital of u. Figure 1.1 below gives an example of this typical kind of surplus process:

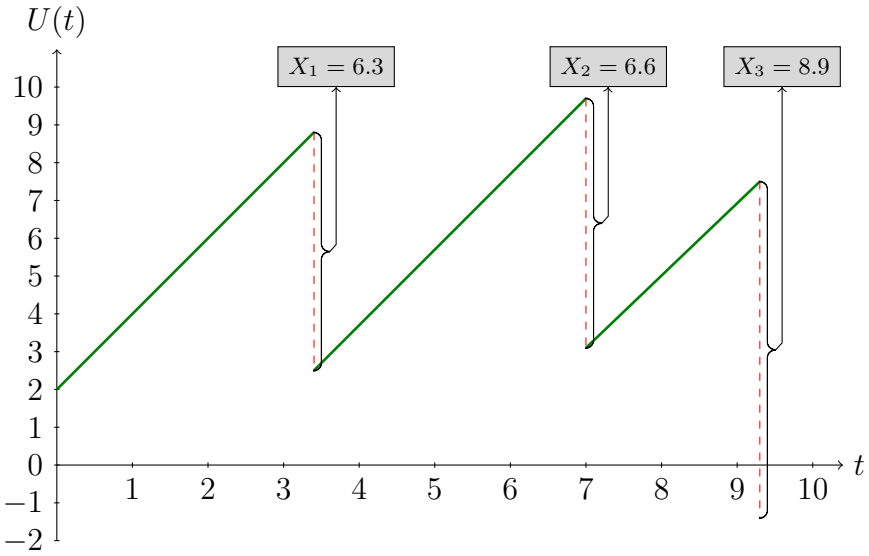

Figure 1.1: A sample path of a continuous-time surplus process with $u = 2$ and $c = 2$.

This simple model was first introduced by Lundberg in 1903: the claim number process, $\{N(t), t \geq 0\}$, was assumed to be a Poisson process with parameter λ, and the X_ℓ's were positively distributed and iid (i.e., independent and identically distributed) and independent of the number of claims $N(t)$ (e.g., see Seal (1969), pp. 12–14). Lundberg's framework was the starting point for deriving explicit expressions for risk measures such as the probability of ultimate ruin and the moments of the time of ruin.

We, of course, acknowledge the simplicity of the model above: there are many aspects we should take into account when developing models of practical value. Still, that was the starting point. Eventually, generalizations of the above-mentioned model became abundant in the literature. Along with the generalized models listed in Section 1.1.1, here are a few more examples:

(1) Copula models such as in Czado et al. (2012).

(2) Semi-Markov processes such as in Albrecher and Boxma (2005) where the independence assumption between $N(t)$ and the X_i's is dropped.

(3) $\{N(t), t \geq 0\}$ being a mixed Poisson process, with interclaim times being generally distributed (as opposed to the classical exponential distribution which is a result of Lundberg's original model) as in Grandell (1997).

(4) $\{N(t), t \geq 0\}$ being a *delayed* (or modified) renewal process, with claim sizes being generally distributed as in Willmot (2004).

Next to distributional generalizations, there are structural generalizations, which include incorporating other factors into the model. One such generalization, which we examine in this thesis, involves incorporating a threshold-based dividend system (a.k.a. dividend strategy), which will be covered in the next sub-section.

1.2.2 Surplus Processes with Dividend Systems

In practice, insurance companies issue payments to their shareholder members, as most of them are private enterprises. The amount and duration of these payments are determined by many factors such as the shareholder's position and the financial health of the insurance company. Since a shareholder's primary interest in the business is to make a high amount

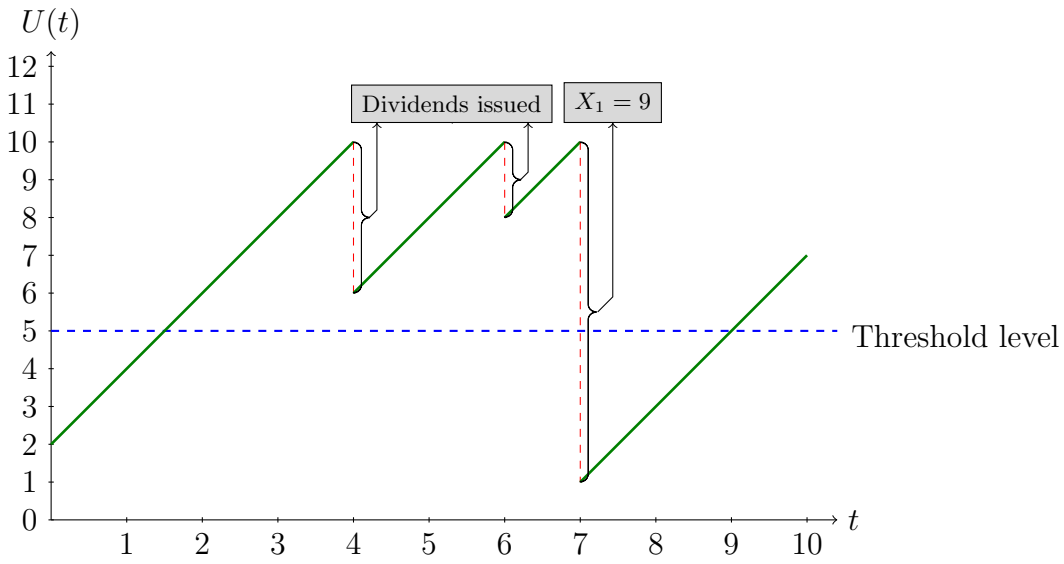

Figure 1.2: A sample path of a continuous-time surplus process under a threshold-based dividend system with $u = 2$ and $c = 2$.

of profit, a successful insurance company would therefore be one that has highly generous dividend systems for its owners. Thus, it would be of significant practical interest to incorporate dividends in insurance risk models.

Insurance risk models incorporating dividend systems were originally proposed by de Finetti (1957). In most insurance risk models including this feature, the common norm of issuing dividends is that they are applied only during periods of time when the surplus process is considered as financially healthy (see Avanzi and Wong (2012) for an example of an insurance risk model that does not follow this norm). Financial health is usually classified by comparing the value of the surplus process to a pre-specified threshold level such as the one depicted in Figure 1.2 (some models, although less common in the literature, incorporate multiple threshold levels such as in Lin and Sendova (2008)). Once the surplus process reaches or crosses this threshold level, it is classified as financially healthy and a dividend system is set in motion. In addition, dividends should never jeopardize

the financial health of an insurance company; that is, the dividend amounts issued should never be allowed to be sufficient enough to bring the surplus process below the threshold level.

One major advantage, both practically and mathematically, of incorporating threshold levels in insurance risk models is that they can keep risk reserves (i.e., surpluses) realistically bounded. Seal states on page 122 of his 1969 book that the "concept of an indefinite future for a risk business [such as an insurance company] coupled with the resulting infinite risk reserve [...] was, however, criticized as unrealistic." For a comprehensive overview of insurance risk models incorporating dividend systems, see Avanzi (2009).

1.2.3 Improvements in Performance Measures

Next to developing more flexible insurance risk models, a lot of research has been dedicated to calculating and deriving algebraic expressions for complex, albeit useful, performance measures such as expected discounted total dividends prior to ruin and the Gerber-Shiu function (proposed and formalized by Gerber and Shiu (1998)). The latter function has especially become popularly studied in the literature within the last decade because of its generality (i.e., we can extract several different performance measures under special cases). Specifically, the Gerber-Shiu function involves a general function, known as the penalty function, depending on particular ruin-related quantities associated with the surplus process, such as the surplus prior to ruin (i.e., the value of the surplus process immediately before the ruin-causing claim) and the deficit at ruin (i.e., how far the surplus process is below 0 at the time of ruin). Hence, by deriving general performance measure functions, surplus processes can be analyzed under different aspects.

Also, next to the development of more complex performance measures (thanks to the advancements in computational speed), several works in simulation and efficient algorithms have recently been developed in the literature for calculating ruin probabilities and other risk measures (for early works, see, for instance, Dufresne and Gerber (1989) and Dickson (1995)). The focus of this thesis lies in this area of the literature, specifically on deriving efficient algorithms for computing ruin-related quantities associated with a particular class of insurance risk models under a completely discrete framework (i.e., all elements determining the behaviour of surplus process, such as time and claim size, are discrete-valued). Since our model is based on this framework, and on particular computational techniques for calculating specific performance measures of interest, we dedicate the following two sections in this literature review to elaborate on the research conducted in both areas.

1.2.4 Works under the Discrete Framework

"Unlike continuous-time risk models, discrete-time risk models have not attracted much attention and the literature counts fewer contributions. Yet discrete-time risk models also have their special features and are closer to reality" (source: Li et al. (2009)).

Mathematically, using notation commonly used in the literature, a discrete-based process can be written as follows:

$$U_t = u + ct - \sum_{\ell=1}^{N_t} Y_\ell \text{ for } t \in \mathbb{N} \text{ and with } U_0 = u \text{ and } N_0 = 0.$$

Analogous to the continuous-time risk models, U_t represents the company's surplus at time t, u is the initial wealth of the insurance company, c is the constant premium rate per time unit, N_t is the number of claims that have occurred by time t, and Y_ℓ is the size of the ℓth claim. All these variables are discrete. Note again that this model has two sources of

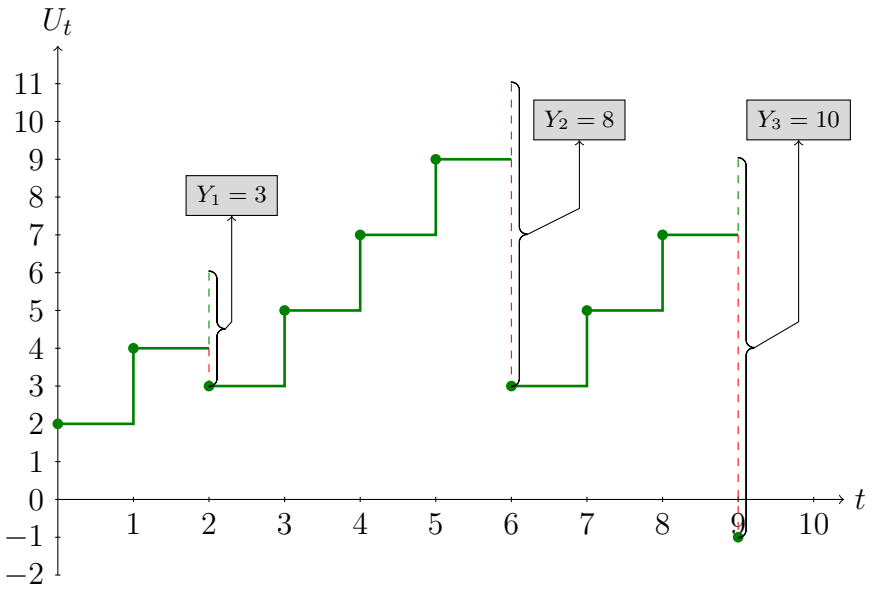

Figure 1.3: A sample path of a discrete-time surplus process with $u = 2$ and $c = 2$.

randomness: N_t and the Y_ℓ's. Figure 1.2 above provides a simple transition scheme of this discrete-based process.

It may come as a surprise that the discrete framework has received very little attention before the 1990's (this also includes discrete-time surplus processes with continuous claim amounts, which is much scarcer in the literature); rather, the focus has always seemed to have been on continuous-time insurance risk models. In fact, the first paper that, so to speak, popularized the discrete framework was Gerber's 1988 article under the amusing title *Mathematical Fun with the Compound Binomial Process*. The model Gerber used is a discrete analogue of the classical compound Poisson model: the compound binomial process. Specifically, the model assumes that at every discrete time point, a maximum of one claim can occur with a known (constant) probability value. The claim sizes are assumed to be iid discrete random variables, and the claim occurrence (or non-occurrence) at each time point is independent of subsequent occurrences (or non-occurrences) at other time

points. Regarding interclaim times, instead of having them exponentially distributed as in the continuous case, these claim times are geometrically distributed—since, in some sense, the geometric distribution is considered to be the discrete counterpart of the exponential distribution because of the memoryless property it satisfies. By employing conditional probabilistic arguments and martingale theory, in addition to some algebra, Gerber derives the probability of ultimate ruin, as well as probabilities involving the surplus prior to ruin and the deficit at ruin.

Several papers have been written since Gerber's 1988 article deriving other risk measures for more general discrete-based risk models (see Li et al. (2009) for a comprehensive overview of discrete-time models studied). Here are some notable generalization examples of interest:

(1) Distribution and performance measure generalizations:

   (a) In Cossette et al. (2010) and Cossette et al. (2004), dependency structures are assumed.

   (b) In Pavlova and Willmot (2004), the number of claims are assumed to follow a discrete-time renewal process and individual claim amounts are assumed to be discrete, iid and independent of the number of claims.

   (c) In Cheng et al. (2000), Lundberg's fundamental equation is introduced for Gerber's compound binomial surplus model, and explicit formulas for particular probabilities of ruin are derived.

(2) Structural generalizations:

   (a) In Drekic and Mera (2011), Jie-hua and Wei (2008), Landriault (2008), Bao

(2007), and Tan and Yang (2006), a threshold-based dividend system is incorporated.

(b) In Jie-hua and Wei (2010), interest rates, next to a threshold-based dividend system, are incorporated.

(c) In Yuen and Guo (2001), multiple claim types are incorporated.

(d) In dos Reis (2000), recovery times (where a recovery time is an uninterrupted length of time during which a surplus process is negatively valued) are allowed.

As can be seen, there has been a growing number of recent papers on discrete-based insurance risk models, most of which are extensions of the compound binomial framework and focusing on deriving marginal distributional results for certain risk measures, such as the time of ruin, the surplus prior to ruin, and the deficit at ruin. However, as mentioned by Alfa and Drekic (2007), "there appear to be few results in the literature for computing the joint probability distribution of these fundamental ruin-related quantities of interest in the delayed Sparre Andersen model." We address the works in this area of study next.

1.2.5 Properties of Discrete-based Delayed Sparre Andersen Models

These models (a.k.a. discrete-based renewal risk models) assume that the time until the first claim occurs follows an arbitrary discrete distribution, and is independent of the subsequent interclaim times. The main reason behind this assumption lies in the view that the first claim time is not a valid interclaim time. Rather, since the process is forcibly started from time 0, and proceeding on the premise that a claim has occurred at time 0 would simply be an artificial, convenient assumption, the time until the first claim should be considered as an incomplete interclaim time period, and, hence, its probability distribution need not necessarily be the same as its subsequent counterparts. This is a more

general class of models than the one studied by Pavlova and Willmot (2004), where the first interclaim time is assumed to follow the *stationary* or *equilibrium* distribution (e.g., see Karlin and Taylor (1975), pp. 192–193) of the iid subsequent interclaim times, and is independent of them.

Keeping in mind that, taxonomically, Sparre Andersen models only assume independence between interclaim times (including the time until the first claim), a commonly used type of Sparre Andersen model is one that further assumes that these interclaim times are independent of claim size. Thus, for specificity on the model class we use in this thesis, we denote this special class of models as discrete-based delayed independent Sparre Andersen models which will be abbreviated from this point forward to DISAMs (or DISAM for a singular model). A special case of DISAMs is when the time for the first claim to occur follows the same distribution as subsequent interclaim times. We denote models following this special case as discrete-based ordinary independent Sparre Andersen models, which will be abbreviated from this point forward to OISAMs (or OISAM for a singular model).

This thesis generalizes the DISAM originally presented in Alfa and Drekic (2007), creating a new class, so to speak, of insurance risk models. The aim then is to derive computational algorithms for calculating the following performance measures (more details will be given in the next chapter):

(1) The *trivariate* ruin probability: the joint probability of the time of ruin, the surplus prior to ruin, and the deficit at ruin.

(2) Two *bivariate* ruin probabilities:

   (a) The joint probability of the time of ruin and the surplus prior to ruin.

(b) The joint probabilities of the time of ruin and the deficit at ruin.

(3) The finite-ruin time based Gerber-Shiu function.

(4) The moments of the total dividends paid by a finite time horizon or before ruin occurs, whichever happens first.

For the last two performance measures, the first claim conditioning method is used (i.e., condition on knowing the time the first claim occurs and its size, and exploit the renewal property of the model), which is standard practice in the field of ruin theory. As for the trivariate and bivariate ruin probabilities, as employed by Alfa and Drekic (2007), we use a combination of matrix-analytic methods (MAMs for short) and conditional probabilistic arguments to derive expressions and computational procedures to calculate these quantities.

To complete the literature review on the methods used in developing the DISAM class in this thesis, the final section provides a recapitulation of the field to which MAMs pertain.

1.3 Along Came MAMs

"It is unworthy of excellent men to lose hours like slaves in the labor of calculation which could safely be relegated to anyone else if machines were used" (Gottfried Leibniz, source: Edwards (1979)).

1.3.1 Motivation

With the advancements of computational power within the second half of the previous century, the field of computational probability has become rich and multidisciplinary (e.g.,

telecommunications, queueing theory, inventory theory, computer engineering, ruin theory). The main objective of computational probability is to develop statistical models that are both general and tractable.

In his 1981 book, Marcel Neuts introduced a new genre, under a queueing context, in computational probability: MAMs. MAMs are techniques used to analyze stochastic models with matrix components under a Markovian framework. Note that these stochastic models need not be multivariate in order to include matrix components; rather, transition probability matrices and initial probability vectors are used to analyze the behaviour of stochastic processes. Examples include the analysis of discrete and continuous phase-type distributions—which are generalizations of geometric and exponential distributions, respectively—and Markovian arrival processes—which are generalizations of the Poisson process.

The main advantage of working under a Markovian setup is that we do not need to keep track of the entire history of the underlying stochastic process. Rather, in order to predict the value of a Markovian-based process at some specific future date, we only need to know its present value; past knowledge is deemed irrelevant. Notwithstanding the fact that these models are less general than those that keep track of past values as well as the present one, they are nonetheless more general than models whose processes do not depend on any path information (i.e., neither the present value nor past ones) for prediction. They also have the advantage of being mathematically tractable with many quantities explicitly expressed in terms of elegant matrix and vector components.

However, the main drawback of using MAMs is that calculating matrices can be compu-

tationally costly. Specifically, computational complexity (i.e., the time it takes to perform a desired task using a computer program) can be a constraining factor in making calculations when dealing with matrices of high dimension. Albeit that computational speed is increasing, and hence making high-dimensional matrix manipulation more convenient, it is essential to continually develop computational complexity reduction techniques in order to optimize the use of these models. This is expressed in the preface of Neuts' 1981 book, where he identifies computational probability, which incorporates MAMs, as "the study of stochastic models with a genuine added concern for algorithmic feasibility over a wide, realistic range of parameter values."

1.3.2 MAMs in Ruin Theory

In recent times, within the field of ruin theory, MAMs have been used to calculate performance measures of insurance risk models strongly generalized from basic ones. Examples under the continuous framework include the following models:

(1) In Mitric et al. (2012) (previously mentioned in Section 1.1.1), claim instants are assumed to be governed by a Markovian arrival process, and claim sizes are matrix exponentially distributed.

(2) In Badescu and Landriault (2009), a general non-renewal insurance risk model is presented, using continuous phase-type distributions, Markovian arrival processes, and results from fluid queues.

(3) In Li (2008), a continuous-time Sparre Andersen model is presented, assuming that interclaim times are continuous phase-type distributed. Different probabilities of ruin are calculated and a matrix expression for the expected discounted dividends prior to ruin in the presence of a constant dividend barrier is derived.

(4) In Stanford et al. (2000), both the interclaim times and the claim sizes are assumed to be continuous phase-type distributed and a numerical approach is used to calculate and plot probabilities of ruin with respect to changes made to particular parameters in the model (e.g., initial reserve ratio and relative security loading).

1.3.3 MAMs in Discrete-based Insurance Risk Models

To the best of our knowledge, the only publications in the literature of which we are aware that use MAMs in discrete-based insurance risk models are Drekic and Mera (2011), Wu and Li (2010), Wu and Li (2008), and Alfa and Drekic (2007). In Wu and Li (2008), interclaim times are assumed to be iid discrete positive random variables and individual claim amounts are assumed be discrete phase-type distributed. Using MAMs, the probability of ultimate ruin and a certain bivariate ruin probability are derived.

The same authors in their more recent paper (i.e., Wu and Li (2010)) introduce a model where the number of claims follows a generalized $(a, b, 0)$ class of distributions, and risk models with both discrete and continuous claim amounts are formulated and examined. These authors also present recursive formulas for these two types of models including the special case when the number of claims follows a particular class of discrete phase-type distributions (which belong to the their defined $(a, b, 0)$ class). The DISAM class used in this thesis follows along the lines of Drekic and Mera (2011) and Alfa and Drekic (2007), whose assumptions and notation are very similar to what will be presented in the subsequent chapters.

This concludes the literature review of this thesis. In the next chapter, we define our model class and its underlying assumptions. The following two chapters show the

methodologies used for deriving and calculating the performance measures of interest and their algebraic results. The subsequent chapter considers specific sub-classes of the general one, with several numerical examples provided. Finally, we conclude this thesis with a discussion of potential areas for future research.

Chapter 2

The Model Class

2.1 The Surplus Process

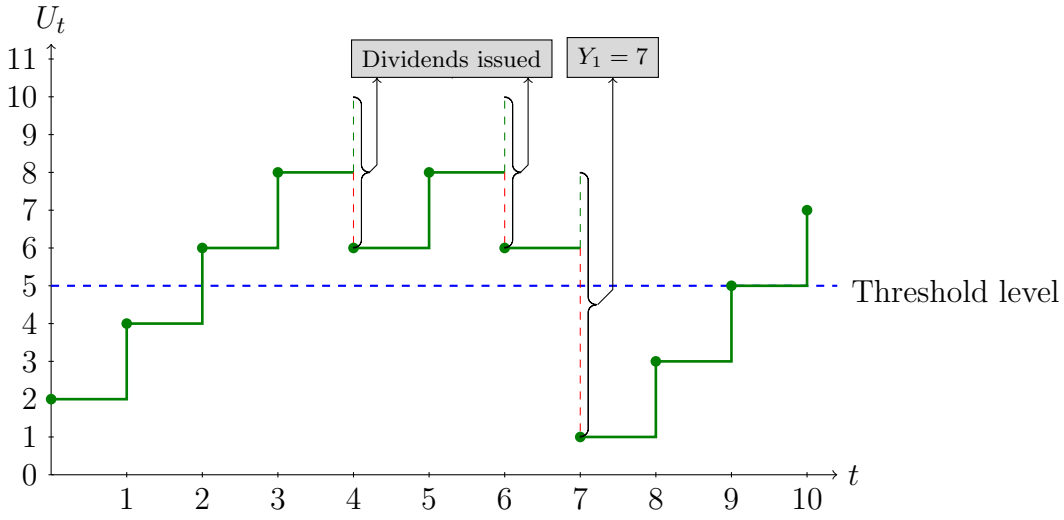

Figure 2.1: A sample path of a surplus process belonging to the model class with $u = 2$ and $c = 2$.

Members of our model class (e.g., see Figure 2.1 above) share the following surplus process equation:

$$U_t = u + ct - \sum_{\ell=1}^{N_t} Y_\ell - S_t \text{ for } t \in \mathbb{N},$$

where U_t is the value of the surplus process (a.k.a. surplus value) at time unit t, u is a constant value indicating the initial value of the surplus process at time 0, c is a constant

value representing the premium amount collected per time unit, N_t is a random variable indicating the number of claims by time t, Y_ℓ is a random variable, indicating the size of the ℓth claim, and S_t is a random variable indicating the total dividend amount by time t. In addition, regarding the variable S_t, $S_0 = 0$ and the range of values this variable can take at any time t is dictated by knowing the value of the surplus amount at time $t-1$, the value of a so-called dividend counter (to be explained shortly) at time $t-1$, and, possibly, the value of the claim size (should a claim have occurred) at time t.

Thus, there are three sources of randomness affecting the behaviour of the surplus process. We now turn to the main properties of the model class.

2.2 Model Class Characteristics

The following is the list of assumptions of our insurance risk model class:

(1) We are working under a discrete framework.

(2) At every time period, a premium of constant size $c \in \mathbb{N}$ is collected. Premiums are collected before any potential claims and dividends. We remark that while a premium value of zero is not truly realistic under an insurance risk model context, we incorporate the possibility that $c = 0$ since our derivations still hold without any loss of generality. Furthermore, as the model class we are developing can be viewed simply as a stochastic process, there may be some practical uses in other areas of study where one is working with a process that is doomed to decay (e.g., depreciation value analysis, wear-and-tear scenarios of a particular biological process).

(3) All claim sizes, $Y_1, Y_2, Y_3, \ldots$, are iid and positively valued, with probability mass

function (pmf) denoted by $\alpha_y = Pr\{Y_\ell = y\}$ for $\ell \in \mathbb{Z}^+$, tail probability function denoted by $\Lambda_y = Pr\{Y_\ell > y\}$ for $\ell \in \mathbb{Z}^+$, a minimum support value of y_0, and a maximum support value denoted by y_α, which is not necessarily finite.

(4) The time until the first claim, W_1, is positively valued with pmf denoted by $r_\omega = Pr\{W_1 = \omega\}$, tail probability function denoted by $R_\omega = Pr\{W_1 > \omega\}$, and a maximum support value denoted by n_r, which is finite. We remark that the finite support assumption can be dropped when using the first claim conditioning approach in Chapter 4.

(5) Subsequent interclaim times, $W_2, W_3, W_4, \ldots$, are iid and positively valued, with pmf denoted by $a_\omega = Pr\{W_\ell = \omega\}$ for $\ell \in \{2, 3, 4, \ldots\}$, tail probability function denoted by $A_\omega = Pr\{W_\ell > \omega\}$ for $\ell \in \{2, 3, 4, \ldots\}$, and a maximum support value denoted by n_a, which is finite. We remark that the finite support assumption can be dropped when using the first claim conditioning approach in Chapter 4. We also refer to $W_2, W_3, W_4, \ldots$ as *ordinary* interclaim times.

(6) Claim sizes, the time until the first claim, and all subsequent interclaim times are independent of one another.

(7) Dividend payment periods are determined by a dividend counter, which we denote by D_t for $t \in \mathbb{N}$, with $D_0 = d$, where d is a pre-assigned initial setting amount. The behaviour of this counter at time $t \in \mathbb{Z}^+$ is influenced by both its value and the surplus value at the previous time unit (i.e., D_{t-1} and U_{t-1}, respectively). In particular, the quantity $h \in \mathbb{Z}^+$ represents the maximum value the dividend counter can take, indicating that a dividend is scheduled to be issued at the next time unit

(i.e., for a dividend to be issued at time $t+1$, we require $D_t = h$ and $U_t \geq b$ to hold true). Whether or not a dividend is actually issued at the next time unit depends on additional factors, which are discussed in the next two assumptions.

(8) Individual dividends are non-negatively distributed and the value of a dividend depends on the surplus level, i, immediately before it is to be issued, with pmf denoted by $d_\ell^*\{i\}$. Such dividends are referred to as *restricted* in the sense that each of their amounts are never allowed to be large enough to bring the surplus process below a threshold level $b \in \mathbb{N}$. The *unrestricted* counterparts of these dividends, on the other hand, are the original amounts that would have been issued had it not been for the threshold level constraint. Thus, an unrestricted dividend whose amount is large enough to bring the surplus process below the threshold level b, will have its restricted version just equal to the amount that will bring the surplus process down to (or at) level b. In particular, if a dividend is to be issued and the surplus process is at level b, then the restricted version will be equal to 0 so that the surplus process remains at level b.

The pmf of a restricted dividend satisfies the following equation:

$$d_\ell^*\{i\} = \begin{cases} d_\ell & \text{if } \ell = \min\{(i-b)_+, c_1\}, \min\{(i-b)_+, c_1\}+1, \\ & \qquad \ldots, \min\{(i-b)_+, c_2\}-1; \\ \displaystyle\sum_{\omega=\min\{(i-b)_+, c_2\}}^{c_2} d_\omega & \text{if } \ell = \min\{(i-b)_+, c_2\}; \\ 0 & \text{otherwise,} \end{cases}$$

where d_ℓ denotes the pmf of an unrestricted dividend which we assume lies between c_1 and c_2 such that c_1 and c_2 are non-negative integers, $c_1 \leq c_2$, and $\sum_{\ell=c_1}^{c_2} d_\ell = 1$. Note that the above definition of $d_\ell^*\{i\}$ even allows for i to be less than b, which in this case yields $d_\ell^*\{i\} = \mathbf{1}_{[\ell=0]}$.

In the special case where we allow $c_1 = c_2 = \infty$, we set, by convention, $d_\ell^*\{i\} = \mathbf{1}_{[\ell=(i-b)_+]}$.

(9) Should a claim occur during a dividend payment period, the priority of which to implement first is determined by a Bernoulli random variable, with p serving as the probability of processing a claim before a dividend, and $1-p$ serving as the probability of processing a claim after a dividend. We assume that there is a zero probability of having both items processed at exactly the same time.

Next, we present the performance measures of interest, whose algorithmic expressions we aim to derive under our model class.

2.3 Performance Measures

Before presenting the performance measures, we begin by defining the following random variables:

(1) T is the time of ruin (i.e., the first time the surplus process becomes negative).

(2) U_{T^-} is the surplus prior to ruin (i.e., the value of the surplus process immediately before it becomes negative).

(3) $|U_T|$ is the deficit at ruin (i.e., how far the surplus process is below 0 at the time of ruin).

Figure 2.2 gives an idea on where these random variables lie on a sample path:

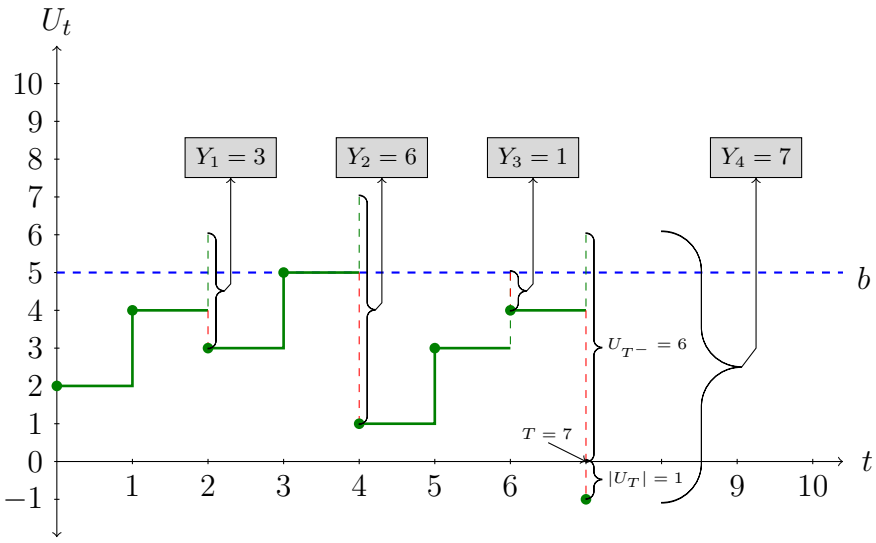

Figure 2.2: Values of T, U_{T^-}, and $|U_T|$ for a sample path of a discrete-time surplus process with $u = 2$, $c = 2$, and $b = 5$.

In this thesis, we focus on deriving algorithmic expressions for the following performance measures pertaining to the above random variables:

(1) $\psi_{\tau,i,j}(u,d) = Pr\{T = \tau, U_{T^-} = i, |U_T| = j | U_0 = u, D_0 = d\}$. We refer to this risk measure as the trivariate ruin probability since it represents the joint pmf of the time of ruin, the surplus prior to ruin, and the deficit at ruin.

(2) $\omega_{\tau,i}(u,d) = Pr\{T = \tau, U_{T^-} = i | U_0 = u, D_0 = d\}$. We refer to this risk measure as a bivariate ruin probability since it represents the joint pmf of the time of ruin and the surplus prior to ruin.

(3) $\phi_{\tau,j}(u,d) = Pr\{T = \tau, |U_T| = j | U_0 = u, D_0 = d\}$. We refer to this risk measure as a bivariate ruin probability since it represents the joint pmf of the time of ruin and the deficit at ruin.

(4) $\widetilde{G}_{v,m}(u,d) = E[v^T w(U_{T^-}, |U_T|)\mathbf{1}_{[T\leq m]}|U_0 = u, D_0 = d]$, where $v \in (0,1]$ (known as the discount factor), $w(.,.)$ is an arbitrary non-negative function of the surplus prior to ruin and the deficit at ruin (known as the penalty function), and $m \in \mathbb{Z}^+$ is a fixed value (known as the finite time horizon). We refer to this function as the finite-ruin time based Gerber-Shiu function. We note that if $w(U_{T^-}, |U_T|)$ is bounded for all values of T, then by letting m tend to infinity, we can apply the dominated convergence theorem (e.g., see Resnick (1999), Theorem 5.3.3) to obtain

$$\lim_{m\to\infty} E[v^T w(U_{T^-}, |U_T|)\mathbf{1}_{[T\leq m]}|U_0 = u, D_0 = d]$$

$$= \lim_{m\to\infty} E[v^T w(U_{T^-}, |U_T|)\mathbf{1}_{[T<m+1]}|U_0 = u, D_0 = d]$$

$$= E[v^T w(U_{T^-}, |U_T|)\mathbf{1}_{[T<\infty]}|U_0 = u, D_0 = d].$$

The final expression above is simply referred to as the Gerber-Shiu function. We also note that our definition of $\tilde{G}_{v,m}(u,d)$ follows along the lines of other finite-ruin time based Gerber-Shiu functions, such as those recently studied by Garrido and Zhou (2010) and Kuznetsov and Morales (in press) under a continuous-time setting.

(5) $\widetilde{\mathcal{D}}^r_m(u,d) = E[S^r_{\min\{T,m\}}|U_0 = u, D_0 = d]$. We refer to this function as the rth moment of the total dividends issued by a finite time horizon of $m \in \mathbb{Z}^+$ or before ruin occurs, whichever happens first. Since we assume that dividends can never bring the surplus process below level b, then $S_t < \infty$ holds true for all values of t. Note also that S_t is a non-negative, non-decreasing function of t, since it is defined as the total dividends by time t. Thus, if $T < \infty$ holds true almost surely (e.g., if U_t is

bounded by an upper ceiling value), then we can apply the monotone convergence theorem (e.g., see Resnick (1999), Theorem 5.3.1) to obtain

$$\lim_{m\to\infty} E[S^r_{\min\{T,m\}}|U_0 = u, D_0 = d] = E[S^r_T|U_0 = u, D_0 = d].$$

The right-hand side expression is simply referred to as the rth moment of the total dividends prior to ruin.

We use MAMs to derive algorithmic expressions for the above trivariate and bivariate ruin probabilities. We show the details in Chapter 3. As for the finite-ruin time based Gerber-Shiu function and the moments of the total dividends issued before a finite time horizon or before ruin occurs, whichever happens first, we apply the first claim conditioning approach. This is covered in Chapter 4.

We dedicate the remainder of this chapter to defining fundamental notation and functions, including some useful derivations associated with them, that will be employed under the matrix-analytic and first claim conditioning methods. Other notation not mentioned here is local to the method used, and hence, will be introduced in its corresponding chapter.

2.4 Essential Notation, Functions and Derivations

2.4.1 Basic Model Notation

The following notation will always be used under its respective context in this thesis:

- y: A general (i.e., realized) claim size. If we are assessing the surplus process at a non-claim paying time period, then y is set equal to 0.

- k: The realized time unit value at which the first claim occurs.

- $Order$: The event variable specifying the order of claim and dividend implementation should they both need to be processed at the same time unit. If the claim is processed before the dividend, then $Order$ is set equal to $ClmFrst$; otherwise, it is set equal to $\overline{ClmFrst}$ (i.e., if the claim is processed after the dividend).

- s: A total dividend amount value before the current time period.

- s^*: A dividend amount value at the current time period. If the current time period is a non-dividend paying time period, then $s^* = 0$. Of course, the value of s^* is influenced by the values of U_{t-1}, $Order$, and y. For instance, assuming that t is a dividend paying time period, if $U_{t-1} + c > b$, but $U_{t-1} + c - y < b$ and $Order = ClmFrst$, then $s^* = 0$. If, however, $Order = \overline{ClmFrst}$ and all other values remain unchanged, then $s^* > 0$.

2.4.2 General Functions

Define the following functions:

- z_u: The earliest time the surplus process reaches or crosses the threshold level b from an initial surplus of u. Mathematically, we have the following expression:

$$z_u = \begin{cases} \infty & \text{if } c = 0 \text{ and } u < b; \\ 0 & \text{if } c = 0 \text{ and } u \geq b; \\ \left\lceil \dfrac{(b-u)_+}{c} \right\rceil & \text{if } c \in \mathbb{Z}^+. \end{cases}$$

- $z^*_{u,d}$: The earliest time a dividend can be issued, starting from an initial surplus of u and an initial dividend counter of d. Note that it is infinite if z_u is infinite.

- $\mathcal{T}_{u,d}(t)$: The collection of dividend paying time points up to and including time t, starting from an initial surplus of u and an initial dividend counter of d. Note that $\mathcal{T}_{u,d}(t) = \emptyset$ if $t < z^*_{u,d}$, since no dividend can be paid before time $z^*_{u,d}$.

- $P_{z^*_{u,d}}(n)$: The number of periods after $z^*_{u,d}$ required for n dividends to be issued.

- $f^{(n)}_{u,d}(s)$: The convolution pmf of the first n dividends, starting from an initial surplus of u and an initial dividend counter of d. By convention, $f^{(0)}_{u,d}(s) = \mathbf{1}_{[s=0]}$.

Of particular importance is the case where no claims are assumed to have occurred during the time period over which the first n dividends are issued. If this assumption holds, then a recursive expression for $f^{(n)}_{u,d}(s)$ can be derived as follows:

Letting Z_ℓ represent the amount of the ℓth restricted dividend, and using the function $P_{z^*_{u,d}}(n)$ defined above, we first note that

$$f^{(n)}_{u,d}(s) = Pr\left\{\sum_{\ell=1}^{n} Z_\ell = s \middle| U_0 = u, D_0 = d\right\}.$$

Using the law of total probability (e.g., see Baclawski (2008), Chapter 6), we can write the following recursive expression:

$$\begin{aligned} f^{(n)}_{u,d}(s) &= \sum_{\omega=0}^{s} Pr\left\{\sum_{\ell=1}^{n} Z_\ell = s \middle| \sum_{\ell=1}^{n-1} Z_\ell = \omega, U_0 = u, D_0 = d\right\} f^{(n-1)}_{u,d}(\omega) \\ &= \sum_{\omega=0}^{s} Pr\left\{Z_n = s - \omega \middle| \sum_{\ell=1}^{n-1} Z_\ell = \omega, U_0 = u, D_0 = d\right\} f^{(n-1)}_{u,d}(w). \end{aligned}$$

Assuming no claims have occurred, the value of the surplus process at the time of the nth dividend, but immediately before its implementation, is given by $u+c[z^*_{u,d}+P_{z^*_{u,d}}(n-1)]-\omega$. Thus, it follows that

$$Pr\left\{Z_n = s-\omega \middle| \sum_{\ell=1}^{n-1} Z_\ell = \omega, U_0 = u, D_0 = d\right\} = d^*_{s-\omega}\{u+c[z^*_{u,d}+P_{z^*_{u,d}}(n-1)]-\omega\}.$$

As a result, we obtain the following overall formula:

$$f^{(n)}_{u,d}(s) = \begin{cases} \mathbf{1}_{[s=0]} & \text{if } n=0; \\ d^*_s\{u+cz^*_{u,d}\} & \text{if } n=1; \\ \displaystyle\sum_{\omega=0}^{s} d^*_{s-\omega}\{u+c[z^*_{u,d}+P_{z^*_{u,d}}(n-1)]-\omega\} f^{(n-1)}_{u,d}(\omega) & \text{if } n>1. \end{cases}$$

- $TotDiv_n(x)$: The total amount of the first n dividends, each comprised by way of a constant unrestricted dividend of size x.

Once again, we wish to focus on the situation where no claims are assumed to have occurred during the time period over which the first n dividends are issued. If this assumption holds, then a recursive expression for $TotDiv_n(x)$ can be derived with the aid of the functions $\mathcal{T}_{u,d}(t)$ and $P_{z^*_{u,d}}(n)$ defined above, which we describe below:

First of all, $TotDiv_0(x) = 0$ since the amount is clearly 0 when no dividends are paid. Secondly, it is clear that

$$TotDiv_1(x) = \min\{x, u + cz^*_{u,d} - b\}.$$

Next, in order to find $TotDiv_2(x)$, we start by noting that at time $z^*_{u,d}$, the surplus process is at $u + cz^*_{u,d} - TotDiv_1(x)$. One dividend payment period later, $P_{z^*_{u,d}}(1)$ in this case, the process is at level $u + c[z^*_{u,d} + P_{z^*_{u,d}}(1)] - TotDiv_1(x)$ just prior to the dividend being paid. Hence,

$$TotDiv_2(x) = TotDiv_1(x) + \min\{x, u + c[z^*_{u,d} + P_{z^*_{u,d}}(1)] - TotDiv_1(x) - b\}.$$

Similarly, to calculate $TotDiv_3(x)$, we start by noting that at time $z^*_{u,d} + P_{z^*_{u,d}}(1)$, the surplus process is at level $u + c[z^*_{u,d} + P_{z^*_{u,d}}(1)] - TotDiv_2(x)$. One dividend payment period later, $P_{z^*_{u,d}}(2) - P_{z^*_{u,d}}(1)$ in this case, the process is at level $u + c[z^*_{u,d} + P_{z^*_{u,d}}(2)] - TotDiv_2(x)$ just prior to the dividend being paid. Hence,

$$TotDiv_3(x) = TotDiv_2(x) + \min\{x, u + c[z^*_{u,d} + P_{z^*_{u,d}}(2)] - TotDiv_2(x) - b\}.$$

Therefore, to calculate $TotDiv_n(x)$, we apply the same inductive procedure as above to obtain the following expression:

$$TotDiv_n(x) = \begin{cases} 0 & \text{if } n = 0; \\ \min\{x, u + cz^*_{u,d} - b\} & \text{if } n = 1; \\ TotDiv_{n-1}(x) + \min\{x, u + c[z^*_{u,d} + P_{z^*_{u,d}}(n-1)] & \\ \qquad - TotDiv_{n-1}(x) - b\} & \text{if } n > 1. \end{cases}$$

- $\mathcal{R}_{u,d}(t)$: The collection of all possible total dividend amounts accumulated by time t, starting from an initial surplus of u and an initial dividend counter of d. Using the functions $TotDiv_n(x)$ and $\mathcal{T}_{u,d}(t)$ defined above, an expression for $\mathcal{R}_{u,d}(t)$ can be written as follows:

$$\mathcal{R}_{u,d}(t) = \{TotDiv_{|\mathcal{T}_{u,d}(t)|}(c_1), TotDiv_{|\mathcal{T}_{u,d}(t)|}(c_1)+1, \ldots, TotDiv_{|\mathcal{T}_{u,d}(t)|}(c_2)\}.$$

  Note that for $t < z^*_{u,d}$, $\mathcal{R}_{u,d}(t) = \{0\}$ since $\mathcal{T}_{u,d}(t) = \emptyset$.

- $V_{u_1,d_1}(t_1, t_2, y, s, s^*, Order)$: The value of the dividend counter at time t_2, starting at time t_1 ($< t_2$) from a surplus of u_1 and a dividend counter of d_1, given that the only claim occurs at time t_2 with a size equal to y, the total amount of dividends over the time period $\{t_1+1, t_1+2, \ldots, t_2-1\}$ is equal to s, the dividend amount at time t_2 is equal to s^*, and the value of the event variable at time t_2 is equal to $Order$.

  We remark that this function can capture the case when no claim occurs at time t_2, simply by setting $y = 0$. Furthermore, at times when any of these arguments do not play a role in determining the value of the dividend counter, we replace them with the notation ".". For example, if t_2 is a non-dividend paying time period, then the last two arguments play no role in determining the value of the dividend counter, and thus, for simplicity, we write the function as $V_{u_1,d_1}(t_1, t_2, y, s, ., .)$. However, we note that this function does not exist for certain combinations of s and s^*. For instance, if $Order = ClmFrst$ and y is large enough to bring the surplus process below level b, then s^* must be equal to 0. Otherwise, $V_{u_1,d_1}(t_1, t_2, y, s, s^*, Order)$ is undefined.

  In addition, due to the assumption that only one claim takes place (at time t_2) over

the time period $\{t_1 + 1, t_2 + 2, \ldots, t_2\}$, the function $V_{u_1,d_1}(t_1, t_2, y, s, s^*, Order)$ is *stationary* in the sense that

$$V_{u_1,d_1}(t_1, t_2, y, s, s^*, Order) = V_{u_1,d_1}(0, t_2 - t_1, y, s, s^*, Order) \text{ for } t_1 < t_2.$$

With these essential definitions in hand, we turn to the algorithmic section of the thesis. We begin with MAMs which will be used for deriving computational formulas of the desired trivariate and bivariate ruin probabilities.

Chapter 3

Algorithmic Expressions for Ruin Probabilities Using MAMs

3.1 Additional Notation

The following basic notation will be used throughout this chapter:

- $\underline{0}$: A $1 \times n_a$ vector of zeros.
- $\mathbf{0}$: An $n_a \times n_a$ matrix of zeros.
- $\underline{e}_1$: A $1 \times n_a$ vector with 1 in the first element, zeros elsewhere.

Further notation will be provided in later sections since specific terminologies will be introduced from particular concepts to be covered. We now proceed to constructing the fundamental Markov process in order to apply MAMs.

3.2 Constructing the Markov Process

In addition to assuming that $U_0 = u$ and $D_0 = d$, by conditioning on the occurrence of the first claim to have taken place at time $k \in \mathbb{Z}^+$, we can formulate a Markov process for

the time points $t \in \{k, k+1, k+2, \ldots\}$. Exploiting the renewal property of the surplus process, we can subsequently consider time k as our "new" time 0. Hence, for $t \geq k$, in order to determine values of the Markov process at time $t+1$, all we need to know is particular information about the surplus process at time t. Specifically, we need to know the values of the variables y, s^*, and $Order$ at time t in order to analyze the process at time $t+1$.

We define E_t to be the value of an elapsed claim time counter, at time t, since the occurrence of the last claim. The use of this counter under a DISAM context was initially introduced by Alfa and Drekic (2007), which was based on an earlier result derived by Alfa and Neuts (1995). The counter E_t behaves as follows: it starts at 1 at time 0, increments by 1 so long as a claim does not occur, and then resets to 1 at claim-occurring time instants. Thus, following the occurrence of the first claim, the range of values for E_t is $\{1, 2, \ldots, n_a\}$ as n_a is the largest possible ordinary interclaim time.

Therefore, we construct the trivariate stochastic process $\{(U_t, D_t, E_t), t = k, k+1, k+2, \ldots\}$, which happens to exhibit a Markovian nature. For instance, if, at time t, we know that the surplus process is below level b and no claim has occurred at time $t+1$, then

$$\begin{aligned}(U_{t+1}, D_{t+1}, E_{t+1}) &= (U_t + c, V_{U_t,D_t}(t, t+1, 0, 0, ., .), E_t + 1) \\ &= (U_t + c, V_{U_t,D_t}(0, 1, 0, 0, ., .), E_t + 1),\end{aligned}$$

where the last equality follows as a result of the earlier stationarity property. Considering all such cases, we obtain the following Markovian relationships for all $t = k, k+1, k+2, \ldots$:

$$(U_{t+1}, D_{t+1}, E_{t+1}) = \begin{cases} (U_t + c, V_{U_t,D_t}(0,1,0,0,.,.), E_t + 1) & \text{if } U_t < b \text{ and } y = 0; \\ (U_t + c - y, V_{U_t,D_t}(0,1,y,0,.,.), 1) & \text{if } U_t < b \text{ and } y > 0; \\ (U_t + c, V_{U_t,D_t}(0,1,0,0,.,.), E_t + 1) & \text{if } U_t \geq b,\ D_t < h \text{ and } y = 0; \\ (U_t + c - y, V_{U_t,D_t}(0,1,y,0,.,.), 1) & \text{if } U_t \geq b,\ D_t < h \text{ and } y > 0; \\ (U_t + c - s^*, V_{U_t,D_t}(0,1,0,0,s^*,.), E_t + 1) & \text{if } U_t \geq b,\ D_t = h \text{ and } y = 0; \\ (U_t + c - y - s^*, V_{U_t,D_t}(0,1,y,0,s^*,Order), 1) & \text{if } U_t \geq b,\ D_t = h \text{ and } y > 0. \end{cases}$$

Figure 3.1 below shows how this trivariate stochastic process may evolve over time.

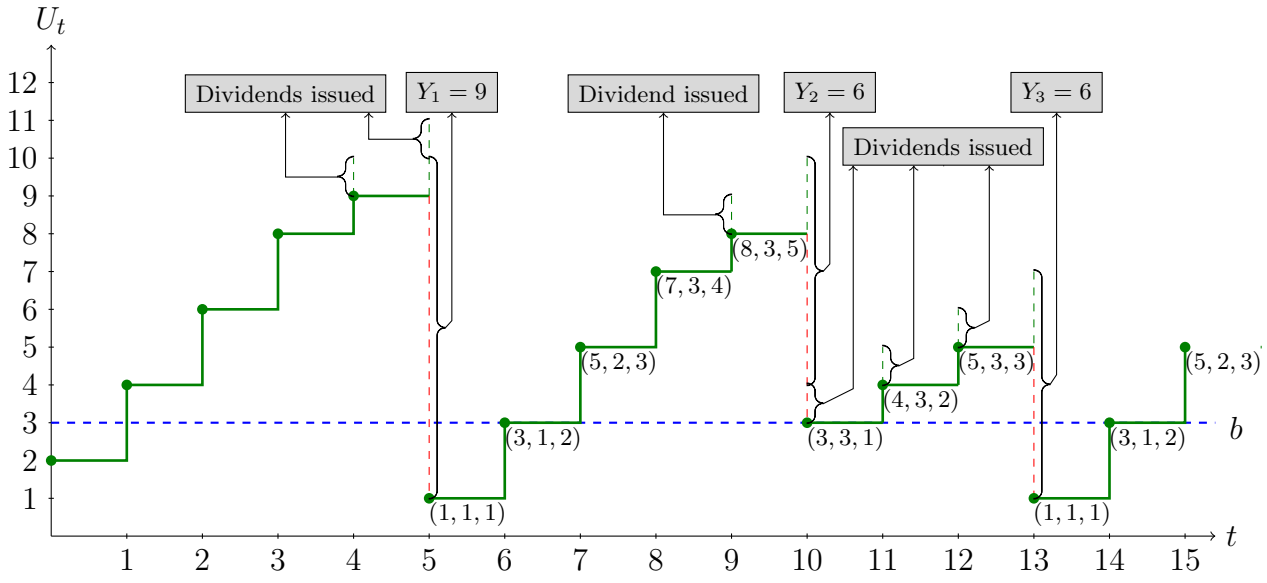

Figure 3.1: A sample evolution of the trivariate stochastic process with $u = 2$, $d = 1$, $c_1 = c_2 = 1$, $c = 2$, $b = 3$, $h = 3$, $k = 5$, $D_5 = 1$, and $0 < p < 1$.

In the special case where $h = 1$, we remark that we can use the bivariate process $\{(U_t, E_t), t = k, k+1, k+2, \ldots\}$ instead, simplifying the above results considerably. Therefore, with the above Markovian setup, we can proceed to employing MAMs.

3.3 The Transition Probability Matrix

From Alfa and Drekic (2007), the interclaim time pmf, a_k, can be expressed in the following discrete phase-type form:

$$a_k = \underline{e}_1 S^{k-1} \underline{s}' \text{ for } k \in \mathbb{N}.$$

Recalling that $A_k = Pr\{W_i > k\}$ for $i \in \{2, 3, 4, \ldots\}$, the $n_a \times n_a$ matrix S and the $1 \times n_a$ vector $\underline{s}$ can be written as follows:

$$S = \begin{array}{c} \\ 1 \\ 2 \\ \vdots \\ n_a - 1 \\ n_a \end{array} \begin{array}{c} \begin{array}{ccccccc} 1 & 2 & 3 & \cdots & n_a - 1 & n_a \end{array} \\ \begin{pmatrix} 0 & A_1/A_0 & 0 & \cdots & 0 & 0 \\ 0 & 0 & A_2/A_1 & \cdots & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & 0 & A_{n_a-1}/A_{n_a-2} \\ 0 & 0 & 0 & \cdots & 0 & 0 \end{pmatrix} \end{array}$$

and

$$\underline{s} = \begin{array}{c} \begin{array}{cccccc} 1 & 2 & \cdots & n_a - 1 & n_a \end{array} \\ \begin{pmatrix} 1 - A_1/A_0 & 1 - A_2/A_1 & \cdots & 1 - A_{n_a-1}/A_{n_a-2} & 1 \end{pmatrix} \end{array}.$$

We remark that because S and $\underline{s}$ will be included in the algorithmic derivations used for calculating the desired ruin probabilities, their dimensions need to both be finite (i.e., $n_a < \infty$).

From the earlier Markovian relationships, we next construct the associated transition probability matrix:

$$P = \begin{array}{c} \\ \vdots \\ -1 \\ 0 \\ 1 \\ \vdots \\ b-1 \\ b \\ b+1 \\ \vdots \end{array} \begin{array}{c} \begin{array}{ccccccccc} \cdots & -1 & 0 & 1 & \cdots & b-1 & b & b+1 & \cdots \end{array} \\ \begin{pmatrix} \ddots & \vdots & \vdots & \vdots & \cdots & \vdots & \vdots & \vdots & \cdots \\ \cdots & \tilde{A}_{-1,-1} & \tilde{A}_{-1,0} & \tilde{A}_{-1,1} & \cdots & \tilde{A}_{-1,b-1} & \tilde{A}_{-1,b} & \tilde{A}_{-1,b+1} & \cdots \\ \cdots & \tilde{A}_{0,-1} & \tilde{A}_{0,0} & \tilde{A}_{0,1} & \cdots & \tilde{A}_{0,b-1} & \tilde{A}_{0,b} & \tilde{A}_{0,b+1} & \cdots \\ \cdots & \tilde{A}_{1,-1} & \tilde{A}_{1,0} & \tilde{A}_{1,1} & \cdots & \tilde{A}_{1,b-1} & \tilde{A}_{1,b} & \tilde{A}_{1,b+1} & \cdots \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \cdots \\ \cdots & \tilde{A}_{b-1,-1} & \tilde{A}_{b-1,0} & \tilde{A}_{b-1,1} & \cdots & \tilde{A}_{b-1,b-1} & \tilde{A}_{b-1,b} & \tilde{A}_{b-1,b+1} & \cdots \\ \cdots & \tilde{B}_{b,-1} & \tilde{B}_{b,0} & \tilde{B}_{b,1} & \cdots & \tilde{B}_{b,b-1} & \tilde{C}_{b,b} & \tilde{C}_{b,b+1} & \cdots \\ \cdots & \tilde{B}_{b+1,-1} & \tilde{B}_{b+1,0} & \tilde{B}_{b+1,1} & \cdots & \tilde{B}_{b+1,b-1} & \tilde{C}_{b+1,b} & \tilde{C}_{b+1,b+1} & \cdots \\ \cdots & \vdots & \vdots & \vdots & \cdots & \vdots & \vdots & \vdots & \ddots \end{pmatrix} \end{array}.$$

We then further break down the above block-matrices of P (each of which are of dimension $hn_a \times hn_a$) into smaller block-matrices of dimension $h \times h$. In what follows, let the (m,n)th block-matrix of an arbitrary matrix A be denoted by $[A]_{m,n}$. In addition, let

$$B_\ell = \begin{cases} \mathbf{0} & \text{if } \ell \in \mathbb{Z}^-; \\ S & \text{if } \ell = 0; \\ \underline{s}' \underline{e}_1 \alpha_\ell & \text{if } \ell \in \mathbb{Z}^+, \end{cases}$$

x and y be the respective surplus level values at times t and $t+1$, and m and n be the respective dividend counter values at times t and $t+1$. Invoking the stationarity property that our Markovian process $\{(U_t, D_t, E_t), t = k, k+1, k+2, \ldots\}$ exhibits, we obtain the following explicit formulas for the decompositions of the above block-matrices of P:

$$[\tilde{A}_{x,z}]_{m,n} = \mathbf{1}_{[n=V_{x,m}(0,1,x+c-z,0,.,.)]} B_{x+c-z}.$$

$$[\tilde{B}_{x,z}]_{m,n} = \begin{cases} \mathbf{1}_{[n=V_{x,m}(0,1,x+c-z,0,.,.)]} B_{x+c-z} & \text{if } m < h; \\ p\mathbf{1}_{[n=V_{x,h}(0,1,x+c-z,0,.,.)]} B_{x+c-z} & \text{if } m = h. \\ +(1-p) \sum_{s^*=c_1}^{\min\{c_2, x+c-b\}} d^*_{s^*}\{x+c\} \mathbf{1}_{[n=V_{x,h}(0,1,x+c-s^*-z,0,s^*,\overline{ClmFrst})]} B_{x+c-s^*-z} & \end{cases}$$

$$[\tilde{C}_{x,z}]_{m,n} = \begin{cases} \mathbf{1}_{[n=V_{x,m}(0,1,x+c-z,0,.,.)]} B_{x+c-z} & \text{if } m < h; \\ p \sum_{y=y_0}^{x+c-z} d^*_{x+c-y-z}\{x+c-y\} \mathbf{1}_{[n=V_{x,h}(0,1,y,0,x+c-y-z,ClmFrst)]} B_y & \text{if } m = h. \\ +(1-p) \sum_{s^*=c_1}^{\min\{c_2, x+c-z\}} d^*_{s^*}\{x+c\} \mathbf{1}_{[n=V_{x,h}(0,1,x+c-s^*-z,0,s^*,\overline{ClmFrst})]} B_{x+c-s^*-z} & \end{cases}$$

We now use the above block-matrices to derive matrix-analytic expressions to calculate the desired trivariate and bivariate probabilities of ruin. To begin, we extract the useful sections of the matrix P.

3.4 Splitting the States and Defining Probability Vectors

As done in Alfa and Drekic (2007) and in Drekic and Mera (2011), we partition the trivariate state space of $\{(U_t, D_t, E_t), t = k, k+1, k+2, \ldots\}$ into two mutually exclusive sub-state spaces: one containing all ruin states and the other containing all non-ruin states. The purpose for defining these two sub-state spaces is to analyze the surplus process up to the time of ruin in terms of two distinct time periods—the time period ranging from time 0 to time $T-1$ (i.e., all non-ruin state transitions), and the one time unit period from time $T-1$ to time T (i.e., all state transitions from non-ruin to ruin)—which will allow us to formulate expressions used for calculating the desired probabilities of ruin.

Letting Q represent the transition probability matrix governing all non-ruin to non-ruin state transitions, and R represent the transition probability matrix governing all non-ruin to ruin state transitions, we have

$$
Q = \begin{array}{c} \\ 0 \\ 1 \\ 2 \\ \vdots \\ b-1 \\ b \\ b+1 \\ \vdots \end{array}
\begin{array}{c}
\begin{array}{ccccccc} 0 & 1 & \cdots & b-1 & b & b+1 & \cdots \end{array} \\
\begin{pmatrix}
\tilde{A}_{0,0} & \tilde{A}_{0,1} & \cdots & \tilde{A}_{0,b-1} & \tilde{A}_{0,b} & \tilde{A}_{0,b+1} & \cdots \\
\tilde{A}_{1,0} & \tilde{A}_{1,1} & \cdots & \tilde{A}_{1,b-1} & \tilde{A}_{1,b} & \tilde{A}_{1,b+1} & \cdots \\
\tilde{A}_{2,0} & \tilde{A}_{2,1} & \cdots & \tilde{A}_{2,b-1} & \tilde{A}_{2,b} & \tilde{A}_{2,b+1} & \cdots \\
\vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots \\
\tilde{A}_{b-1,0} & \tilde{A}_{b-1,1} & \cdots & \tilde{A}_{b-1,b-1} & \tilde{A}_{b-1,b} & \tilde{A}_{b-1,b+1} & \cdots \\
\tilde{B}_{b,0} & \tilde{B}_{b,1} & \cdots & \tilde{B}_{b,b-1} & \tilde{C}_{b,b} & \tilde{C}_{b,b+1} & \cdots \\
\tilde{B}_{b+1,0} & \tilde{B}_{b+1,1} & \cdots & \tilde{B}_{b+1,b-1} & \tilde{C}_{b+1,b} & \tilde{C}_{b+1,b+1} & \cdots \\
\vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots
\end{pmatrix}
\end{array}
$$

and

$$
R = \begin{array}{c} \\ 0 \\ 1 \\ 2 \\ \vdots \\ b-1 \\ b \\ b+1 \\ \vdots \end{array}
\begin{array}{c}
\begin{array}{cccc} -1 & -2 & -3 & \cdots \end{array} \\
\begin{pmatrix}
\tilde{A}_{0,-1} & \tilde{A}_{0,-2} & \tilde{A}_{0,-3} & \cdots \\
\tilde{A}_{1,-1} & \tilde{A}_{1,-1} & \tilde{A}_{1,-3} & \cdots \\
\tilde{A}_{2,-1} & \tilde{A}_{2,-2} & \tilde{A}_{2,-3} & \cdots \\
\vdots & \vdots & \vdots & \ddots \\
\tilde{A}_{b-1,-1} & \tilde{A}_{b-1,-2} & \tilde{A}_{b-1,-3} & \cdots \\
\tilde{B}_{b,-1} & \tilde{B}_{b,-2} & \tilde{B}_{b,-3} & \cdots \\
\tilde{B}_{b+1,-1} & \tilde{B}_{b+1,-2} & \tilde{B}_{b+1,-3} & \cdots \\
\vdots & \vdots & \vdots & \ddots
\end{pmatrix}
\end{array}.
$$

Now, by conditioning on the first claim occurring at time k, and exploiting the homogeneity of the transition probabilities beyond the first claim (since the interclaim times are iid after the first claim), we define the following three vectors associated with our Markov process, which will be used in calculating the desired ruin probabilities.

First of all, let $\underline{\tilde{b}}^{(k)}$ be the *initial* probability vector containing the probabilities of all the non-ruin states the Markov process $\{(U_t, D_t, E_t), t = k, k+1, k+2, \ldots\}$ can assume at time k. Specifically, we have

$$\underline{\tilde{b}}^{(k)} = \begin{array}{c} \begin{array}{cccc} 0 & 1 & 2 & \cdots \end{array} \\ \left(\begin{array}{cccc} \tilde{b}_0^{(k)} & \tilde{b}_1^{(k)} & \tilde{b}_2^{(k)} & \cdots \end{array} \right) \end{array},$$

where

$$\underline{\tilde{b}}_i^{(k)} = \begin{array}{c} \begin{array}{cccc} 1 & 2 & \cdots & h \end{array} \\ \left(\begin{array}{cccc} \underline{b}_{i,1}^{(k)} & \underline{b}_{i,2}^{(k)} & \cdots & \underline{b}_{i,h}^{(k)} \end{array} \right) \end{array} \text{ for } i \in \mathbb{N},$$

with each $\underline{b}_{i,\ell}^{(k)}$ having a dimension of $1 \times n_a$.

Next, let $\underline{\tilde{g}}_n^{(k)}$ be the non-ruin probability vector at time $k+n$ containing the probabilities of all the non-ruin states the Markov process $\{(U_t, D_t, E_t), t = k, k+1, k+2, \ldots\}$ can assume at time $k+n$ without ruin having occurred by time $k+n$. Thus, using the above notation, we have

$$\underline{\tilde{g}}_n^{(k)} = \begin{array}{c} \begin{array}{cccc} 0 & 1 & 2 & \cdots \end{array} \\ \left(\begin{array}{cccc} \tilde{g}_{n,0}^{(k)} & \tilde{g}_{n,1}^{(k)} & \tilde{g}_{n,2}^{(k)} & \cdots \end{array} \right) \end{array} = \underline{\tilde{b}}^{(k)} Q^n \text{ for } n \in \mathbb{N},$$

where

$$\underline{\tilde{g}}_{n,i}^{(k)} = \begin{array}{c} \begin{array}{cccc} 1 & 2 & \cdots & h \end{array} \\ \left(\begin{array}{cccc} \underline{g}_{n,i,1}^{(k)} & \underline{g}_{n,i,2}^{(k)} & \cdots & \underline{g}_{n,i,h}^{(k)} \end{array} \right) \end{array} \text{ for } i \in \mathbb{N},$$

with each $\underline{g}_{n,i,\ell}^{(k)}$ having a dimension of $1 \times n_a$. Note that $\underline{g}_{n,i,\ell}^{(k)} = \underline{0}$ for $i > u + c(k+n)$, as $u + c(k+n)$ represents the value of the surplus process at time $k+n$ in the event that no claims or dividends ever take place. In Section 3.5.1, we will derive formulas that produce

lower cutoff points for i than $u+c(k+n)$, which will be useful in efficiently computing the relevant elements of this vector.

Lastly, let $\underline{\tilde{h}}_n^{(k)}$ be the ruin probability vector at time $k+n$, containing the probabilities of all the ruin states the Markov process $\{(U_t, D_t, E_t), t = k, k+1, k+2, \ldots\}$ can assume at time $k+n$ if ruin occurs at precisely time $k+n$. It follows that

$$\underline{\tilde{h}}_n^{(k)} = \begin{array}{c} \begin{array}{cccc} -1 & -2 & -3 & \cdots \end{array} \\ \left(\begin{array}{cccc} \tilde{h}_{n,-1}^{(k)} & \tilde{h}_{n,-2}^{(k)} & \tilde{h}_{n,-3}^{(k)} & \cdots \end{array} \right) \end{array} = \underline{\tilde{b}}^{(k)} Q^{n-1} R = \underline{\tilde{g}}_{n-1}^{(k)} R \text{ for } n \in \mathbb{Z}^+,$$

where

$$\underline{\tilde{h}}_{n,-j}^{(k)} = \begin{array}{c} \begin{array}{cccc} 1 & 2 & \cdots & h \end{array} \\ \left(\begin{array}{cccc} \underline{h}_{n,-j,1}^{(k)} & \underline{h}_{n,-j,2}^{(k)} & \cdots & \underline{h}_{n,-j,h}^{(k)} \end{array} \right) \end{array} \text{ for } j \in \mathbb{Z}^+,$$

with each $\underline{h}_{n,-j,\ell}^{(k)}$ having a dimension of $1 \times n_a$. We remark that $\underline{h}_{n,-j,\ell}^{(k)} = \underline{0}$ for $j > y_\alpha - \min\{c, b + (c-c_2)_+\}$, with the right-hand side of this inequality being infinite when $y_\alpha = \infty$. The quantity $\min\{c, b + (c-c_2)_+\}$ actually represents the smallest possible surplus prior to ruin value that can be achieved over all time. The first term of this quantity is based on the fact that premiums are earned before any potential claims and dividends. Thus, before the ruining claim, the surplus process could climb to level c from the smallest non-ruin surplus value, which is obviously level 0. On the other hand, if c is sufficiently larger than b, c_2 itself is sufficiently large, and ruin occurs at a dividend paying time unit such that the ruining claim is implemented after the dividend, then the smallest possible surplus prior to ruin value can actually be smaller than c. For example, if $b = 1$, $c = 3$, $c_2 = 4$, $U_{T-1} = 3$, and T is a dividend paying time unit, it is possible for U_{T-} to be equal to 2 by gaining the premium c and issuing a dividend of size c_2 before the ruining claim. Thus, in this case, it is possible to achieve a surplus prior to ruin value which is

less than the premium amount c, and this is captured by the second term of the quantity $\min\{c, b + (c - c_2)_+\}$.

In the above analysis, we have implicitly assumed that it is possible for dividends to be processed before claims (i.e., $p < 1$). If $p = 1$, however, then the smallest possible surplus prior to ruin value is simply equal to c. Therefore, incorporating the value of p, the quantity $\min\{c, b + (c - c_2)_+\}$ can be more accurately expressed as

$$i_{min} = c\mathbf{1}_{[p=1]} + \min\{c, b + (c - c_2)_+\}\mathbf{1}_{[p<1]},$$

so that $\underline{h}^{(k)}_{n,-j} = \underline{0}$ for $j > y_\alpha - i_{min}$.

Although we have identified the required elements needed to calculate our desired ruin probabilities, we face the computational problem of infinite dimensionality. That is, in order to compute the desired ruin probabilities, we need to conduct numerical operations on $\underline{\tilde{b}}^{(k)}_i$, $\underline{\tilde{g}}^{(k)}_n$, $\underline{\tilde{h}}^{(k)}_n$, Q, and R, which is computationally infeasible since the number of elements involved is infinitely many. Hence, the objective becomes to find ways in retaining a finite number of elements from these block-vectors and block-matrices that can be used to compute, exactly or approximately, the desired ruin probabilities. Fortunately, it turns out that we can compute these probabilities exactly using MAMs on the above block-vectors and block-matrices. The next section shows how this is done.

3.5 Efficient Methods for Computing Probability Vectors

3.5.1 Ceiling Values

The first step in tackling the infinite dimensionality of the above vectors is to determine which elements we need to retain and which ones we can afford to discard. As our probability block-vectors are indexed first and foremost with respect to the surplus level i (where each level i itself has h vectors), we realize that beyond a particular surplus level, at a given point in time, the probability values will be 0. For example, as mentioned in the previous section, if we are assessing the surplus process at time $k+n$, then we know that the block-vectors whose ith subscript is greater than $u+c(k+n)$ will be zero vectors, since the surplus process can never exceed level $u+c(k+n)$ at time $k+n$. Hence, all we need to do is to locate the index values in those block-vectors after which the probability values will always be 0. We refer to such index values as *ceiling values*.

Because our surplus processes involve dividend systems operating under a fairly general framework, determining the "optimal" or "best" ceiling value may not always be feasible without making certain assumptions and knowing specific parameter values. This is illustrated in the following three ceiling values we use for assessing our probability vectors.

The first ceiling value we introduce, which we denote by $v_{u,d}(k)$, has the objective of outputting the maximum level the surplus process can achieve at time k starting from an initial surplus of u and an initial dividend counter of d, immediately before the first claim and a possible dividend are implemented. A formula for this ceiling value would involve finding the level the surplus process takes on at time k before assessing any dividends (i.e.,

$u + ck$), and then subtracting from this value all minimum dividend amounts issued up until time $k-1$ (i.e., $TotDiv_{|\mathcal{T}_{u,d}(k-1)|}(c_1)$). Hence, $v_{u,d}(k)$ can simply be expressed as follows:

$$v_{u,d}(k) = u + ck - TotDiv_{|\mathcal{T}_{u,d}(k-1)|}(c_1).$$

Note that this formula yields an optimal ceiling value for the initial probability vector $\underline{b}_{i,\ell}^{(k)}$, in the sense that $\underline{b}_{i,\ell}^{(k)} = \underline{0}$ for all $i > v_{u,d}(k)$ while it is possible that $\underline{b}_{i,\ell}^{(k)} \neq \underline{0}$ for any $i \leq v_{u,d}(k)$. Furthermore, we remark that $v_{u,d}(k) = u+ck$ if $k \leq z_{u,d}^*$, since $|\mathcal{T}_{u,d}(k-1)| = 0$ in this case.

The second ceiling value we use, which we denote by $\tilde{v}_{u,d}(t)$, is a more general version than the previous one, whose objective is to output the maximum level the surplus process can achieve at a general time unit t, starting from an initial surplus of u and an initial dividend counter of d. Therefore, unlike the previous ceiling value above, there is no restriction here on the number of claims that have occurred by time t. This can result in a situation where particular claim sizes occurring at certain time points before time t can lead to a higher surplus value at time t than a situation where claim size and frequency are both kept to a minimum until time t. For instance, suppose that $u = 1$, $d = 1$, $c = 5$, $c_1 = c_2 = 3$, $b = 5$, $h = 2$, $y_0 = 1$, and the dividend counter D_t, while at or above level b, increments by 1 until reaching h, and then resets to 1. Hence, $z_u = 1$, and $z_{u,d}^* = 3$. Now, suppose we wish to calculate $\tilde{v}_{u,d}(3)$. Using the formula above and implementing the dividend at time 3 will yield a value of $1 + 5 \times 3 - 3 = 13$. However, if a claim of size 2 occurred at time 1, we could obtain a surplus value of 14 (assuming no claims occur at times 2 and 3). Thus, a suitable formula for this ceiling value will be required to account for such scenarios.

One obvious non-optimal formula we can use for this ceiling value is to simply set $\tilde{v}_{u,d}(t)$ equal to $u + ct$. In that case, claims and dividends are unaccounted for, and hence it can be considered as a rather *inefficient* ceiling value since there is unnecessary extra computational time used in retaining zero-valued vector elements.

Nonetheless, it turns out that if we make three minor simplifications, we can derive a formula which calculates a more *efficient* ceiling value. We notify readers that while the derivations outlined here and the final expressions given at the end of this section are quite extensive, they are very useful for computational implementation. Bearing that in mind, we state the simplifications underlying our derivations. First of all, we assume that the dividend counter, D_t, is both one-unit incremental and cyclic above the threshold level (i.e., once above level b, D_t increments one unit at a time until reaching h; one time unit after reaching h, it resets to 1 and begins incrementing by 1 again). Secondly, we assign the value of D_{z_u} to be equal to 1, thereby implying that $z^*_{u,d} = z_u + h$. As we will be comparing various sample paths with each other, this simplification enables us to avoid problems arising from asynchronous dividend paying time periods. Moreover, $z^*_{u,d}$ does not actually depend on d now, and hence, neither will $\tilde{v}_{u,d}(t)$. Thirdly, we exclude the consideration of claims for $t \geq z^*_{u,d}$. This simplification is essentially due to the fact that for certain models in our class, claims can disrupt dividend paying time periods.

Before proceeding further, we note that based on the second simplification stated above, we can suppress the argument d and replace $\tilde{v}_{u,d}(t)$ with $\tilde{v}_u(t)$ and use $z_u + h$ instead of $z^*_{u,d}$. Finally, we would like to emphasize that although our results shown on page 57 are based on the above simplifications, they are generally applicable to all the models in our class. Of course, further simplifications can be made and greater ceiling value efficiency

can be attained in accordance to the specifications of the model/models under study.

To derive the formula for $\tilde{v}_u(t)$, we begin by considering values of t satisfying $t < z_u + h$. In this case, the obvious choice for $\tilde{v}_u(t)$ is simply

$$u + ct - \mathbf{1}_{[t \geq n_r]} y_0 - \left\lfloor \frac{(t - n_r)_+}{n_a} \right\rfloor y_0,$$

since there are no dividends to be implemented. However, for values of $t \geq z_u + h$, it will become necessary to incorporate other sample paths alongside the "mainstream" one (i.e., the path that starts at level u at time 0, with dividend frequency and amount kept to a minimum upon achieving level b). These other paths we will consider all share a common feature: the surplus process was brought down to level $b - 1$ at some time point in the past, with the dividend counter set equal to 1 at that time point, and dividend frequency and amount are kept to a minimum beyond that time point. We denote a sample path of this kind as a "$b - 1/t_0$" path, where t_0 denotes the time at which the sample path was brought down to level $b - 1$ (with corresponding dividend counter set equal to 1). In what follows, we compare the surplus level of each sample path for $t \geq z_u + h$.

At time $t = z_u + h$, we need to compare the "mainstream" path with the "$b - 1/z_u$" path (in fact, it was this path that was shown to yield the greater surplus value in the earlier numerical example). Still, we must be cautious in general about when to consider alternate paths as possible routes for the process to take. For instance, the "$b - 1/z_u$" path is not relevant if $u \geq b$ (since, in this case, we know that $z_u = 0$ and we are certain that $U_0 = u \neq b - 1$), and hence, should not be considered as a possible alternate path with which we compare the "mainstream" one. We incorporate this aspect into our final formulas on page 57.

Returning to our analysis at time $t = z_u + h$, we see that the reason the "mainstream" and "$b - 1/z_u$" paths are the only two ones eligible to yield a maximum surplus value at time $z_u + h$ lies in the observation that the only instant the "mainstream" path can be overtaken is at time $z_u + h$ when the dividend is implemented. In other words, by delaying the dividend counter by 1 time unit—through having the surplus process reside at level $b - 1$ at time z_u and setting the dividend counter to 1—and hence, not issuing a dividend at time $z_u + h$, may produce a higher surplus value at time $z_u + h$ than the "mainstream" path. Thus, this other sample path results in the earning of h premiums. Therefore, if we define $Div_i(x)$ to be the restricted value of an unrestricted dividend of size x, given that the surplus process is at level i immediately before the dividend is issued (i.e., $Div_i(x) = \min\{(i - b)_+, x\}$), we have

$$\tilde{v}_u(z_u + h) = \max\left\{u + c(z_u + h) - Div_{u+c(z_u+h)}(c_1), b - 1 + ch\right\}.$$

At time $t = z_u + h + 1$, and assuming for the moment that $h > 1$ (i.e., before the next dividend payment period in the "mainstream" path), we need to compare the surplus values at time t emerging from three particular sample paths: "mainstream", "$b - 1/z_u$", and "$b - 1/z_u + 1$". Note that, when evaluating the third sample path, no dividend is issued from time $z_u + 2$ to time $z_u + h + 1$ since $(z_u + h + 1) - (z_u + 2) + 1 = h$ and $h + 1$ periods are required for a dividend to be issued if starting from level $b - 1$ at time $z_u + 1$ with a dividend counter value of 1. Thus,

$$\tilde{v}_u(z_u + h + 1) = \max\left\{u + c(z_u + h + 1) - Div_{u+c(z_u+h)}(c_1), b - 1 + c(h + 1) - Div_{b-1+c(h+1)}(c_1), b - 1 + ch\right\}.$$

In order to proceed further, we must examine $\tilde{v}_u(t)$ for $t > z_u + h$ under two separate cases. The final results of both cases are shown on page 57.

The first case examines the situation when $c_1 \leq c$. In this case, the above function for $\tilde{v}_u(z_u+h+1)$ simplifies to comparing only between the first and second sample paths (e.g., see Figure 3.2 for an illustration), namely

$$\tilde{v}_u(z_u+h+1) = \max\{u+c(z_u+h+1)-c_1, b-1+c(h+1)-c_1\}.$$

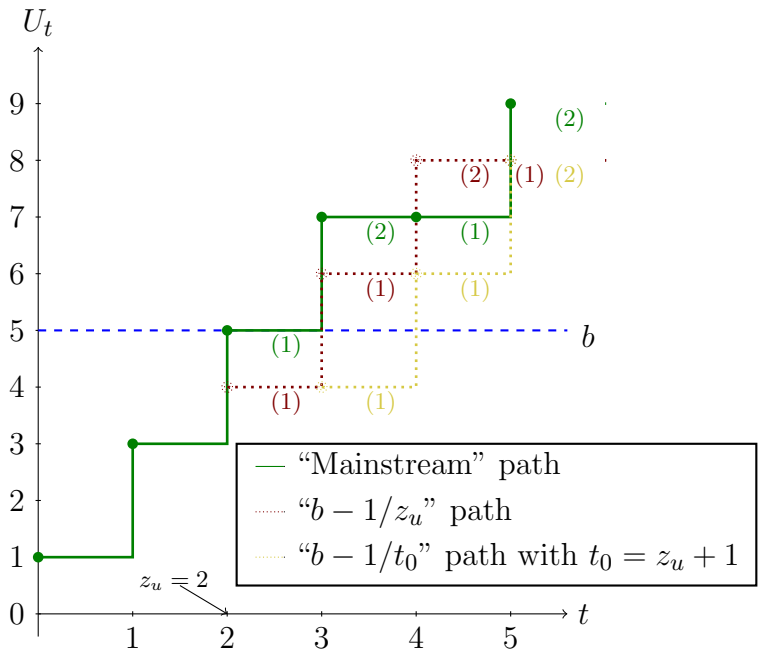

Figure 3.2: Analysis of $\tilde{v}_u(z_u+h+1)$ for a surplus process with $u=1$, $d=1$, $c_1=2\leq c=2$, $b=5$, and $h=2$.

For $t = z_u+h+2$, and assuming for the moment that $h>2$ (i.e., before the next dividend paying time period in the "mainstream" path), the two sample paths chosen above both receive a premium of c. Since $c_1 \leq c$, the "$b-1/z_u$" path is guaranteed to yield a surplus value at least as high as the surplus level at time z_u+h+2 resulting from the "$b-1/z_u+2$" path. Thus,

$$\tilde{v}_u(z_u+h+2) = \max\{u+c(z_u+h+2)-c_1, b-1+c(h+2)-c_1\}.$$

In general, for $t = z_u + h + n$ where $1 \leq n < h$, it follows from the same line of logic that

$$\tilde{v}_u(z_u + h + n) = \max\{u + c(z_u + h + n) - c_1, b - 1 + c(h + n) - c_1\}.$$

If we now consider the situation at $t = z_u + 2h$ (i.e., a dividend will be issued in the "mainstream" path, but no dividend will be issued in the "$b - 1/z_u$" path), it is clear that

$$\tilde{v}_u(z_u + 2h) = \max\{u + c(z_u + 2h) - 2c_1, b - 1 + 2ch - c_1\}.$$

For $t = z_u + 2h + 1$, both sample paths will receive a premium of c and (assuming again that $h > 1$) a dividend will only be issued in the "$b - 1/z_u$" path. Thus,

$$\tilde{v}_u(z_u + 2h + 1) = \max\{u + c(z_u + 2h + 1) - 2c_1, b - 1 + c(2h + 1) - 2c_1\}.$$

Adopting the same essential approach in general, we therefore have, for the case of $t \geq z_u + h$, the following expression:

$$\tilde{v}_u(t) = \max\left\{u + ct - \left\lceil \frac{t - z_u - h + 1}{h} \right\rceil c_1, b - 1 + c(t - z_u) - \left\lceil \frac{t - z_u - h}{h} \right\rceil c_1\right\}.$$

Turning our attention to the second case in which $c_1 > c$, we begin with $t = z_u + h + 1$, again assuming for the moment that $h > 1$ (i.e., before the next dividend paying time period in the "mainstream" path). Since $c_1 > c$, among the "$b - 1/t_0$" sample paths, the one yielding the highest surplus value at time $z_u + h + 1$ is the one that incorporates the most frequent premium payments and no dividends. Since D_{z_u} is assumed to be equal to 1, the "$b - 1/z_u + 1$" path is the one we seek (e.g., see Figure 3.3), so that we obtain

$$\tilde{v}_u(z_u + h + 1) = \max\left\{u + c(z_u + h + 1) - Div_{u+c(z_u+h)}(c_1), b - 1 + ch\right\}.$$

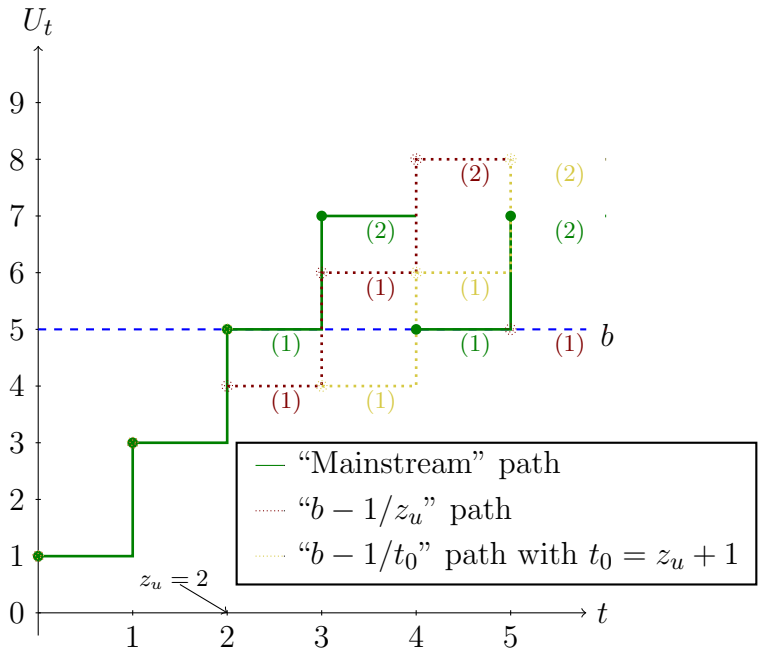

Figure 3.3: Analysis of $\tilde{v}_u(z_u + h + 1)$ for a surplus process with $u = 1$, $d = 1$, $c_1 > c = 2$, $b = 5$, and $h = 2$.

For $t = z_u + h + 2$, and assuming for the moment that $h > 2$ (i.e., before the next dividend paying time period in the "mainstream" path), we compare the "mainstream" path with the "$b - 1/z_u$", "$b - 1/z_u + 1$", and "$b - 1/z_u + 2$" paths. Among the three "$b - 1/t_0$" paths, since $c_1 > c$, the one that yields the greatest premiums without invoking a dividend payment is the "$b - 1/z_u + 2$" path. Mathematically, the number of premiums earned from this path is clearly h, so that we have

$$\tilde{v}_u(z_u + h + 2) = \max\left\{u + c(z_u + h + 2) - Div_{u+c(z_u+h)}(c_1), b - 1 + ch\right\}.$$

In general, for $t = z_u + h + n$ where $1 \leq n < h$, it follows from the same line of logic that

$$\tilde{v}_u(z_u + h + n) = \max\left\{u + c(z_u + h + n) - Div_{u+c(z_u+h)}(c_1), b - 1 + ch\right\}.$$

Let us now consider the surplus process at $t = z_u + 2h$: for the "mainstream" path, the surplus process reaches level $u + c(z_u + h) - Div_{u+c(z_u+h)}(c_1) + ch$ immediately before incorporating a restricted dividend of unrestricted size c_1. If we attempt to place the

above-written surplus level as a subscript of the $Div_i(.)$ function, we will face an issue of lengthy formula writing. This issue will persist for higher values of t, and so, we introduce a recursive function in order to tackle this issue. Specifically, we define

$$a_u(\ell) = \begin{cases} u + c(z_u + h) & \text{if } \ell = 0; \\ \\ a_u(\ell - 1) - Div_{a_u(\ell-1)}(c_1) + ch & \text{if } \ell > 0. \end{cases}$$

Therefore, for $t = z_u + 2h$, we have

$$\tilde{v}_u(z_u + 2h) = \max\left\{a_u(1) - Div_{a_u(1)}(c_1), b - 1 + ch\right\}.$$

Keep in mind that the above formula includes two dividend amounts in the "mainstream" path: one at time $z_u + h$ (incorporated in the term $a_u(1)$) and the other at time $z_u + 2h$.

For $t = z_u + h + n$ where $h \leq n < 2h$, all we need to do for the "mainstream" path is to incorporate the collection of $n - h$ additional premiums, resulting in the following expression:

$$\tilde{v}_u(z_u + h + n) = \max\left\{a_{u,d}(1) - Div_{a_u(1)}(c_1) + c\,(n - h)\,, b - 1 + ch\right\}.$$

Applying the same reasoning for $2h \leq n < 3h$, we get the following expression:

$$\tilde{v}_u(z_u + h + n) = \max\left\{a_u(2) - Div_{a_u(2)}(c_1) + c\,(n - 2h)\,, b - 1 + ch\right\}.$$

Combining the above cases (i.e., for models with $c_1 \leq c$ and models with $c_1 > c$, and all values of t) with the given assumptions, we summarize the results for $\tilde{v}_u(t)$ on page 57.

The third and final ceiling value we use, which we denote by $\tilde{v}_{u,d}(k,n)$ for $n \in \mathbb{N}$, assumes that the first claim occurred at time k, starting from an initial surplus of u and an initial dividend counter of d, and has the objective of outputting an upper bound on the surplus process at time $k+n$. Hence, we can interpret this ceiling value as a conditional version of $\tilde{v}_u(k+n)$, in the sense that a claim is known to have occurred at time k. Specifically, we assess the surplus process at time k (to be viewed as our new starting point) and adjust the initial surplus level and dividend counter accordingly. As a result, we can make use of the previous ceiling value formula.

However, we remark that the general ceiling value, $\tilde{v}_u(t)$, we wish to employ here should pertain to that of an OISAM (since the first claim has been assumed to occur). Thus, to make the distinction, we replace $\tilde{v}_u(t)$ with $\tilde{v}_u^{ord}(t)$. The only changes to be made involve replacing n_r with n_a in the formula of $\tilde{v}_u(t)$ found on page 57. We now proceed to the analysis of $\tilde{v}_{u,d}(k,n)$, under the following six non-overlapping, exhaustive cases:

(1) $k \notin \mathcal{T}_{u,d}(k)$ and $v_{u,d}(k) - y_0 < b$: In this case, the "mainstream" path yields the "optimal" level at time $k+n$, namely:

$$\tilde{v}_{u,d}(k,n) = \tilde{v}^{ord}_{v_{u,d}(k)-y_0}(n).$$

(2) $k \notin \mathcal{T}_{u,d}(k)$ and $v_{u,d}(k) - y_0 \geq b$: In this case, since the "mainstream" path is above level b at time k, it is susceptible to more dividends being paid out than a path starting below level b at time k, which can potentially result in having a lower surplus value at time $k+n$ than that alternate path. Thus, we need to compare the surplus level at time $k+n$ from the "mainstream" path with that from the "$b-1/k$" path. Thus,

$$\tilde{v}_{u,d}(k,n) = \max\left\{\tilde{v}^{ord}_{v_{u,d}(k)-y_0}(n), \tilde{v}^{ord}_{b-1}(n)\right\}.$$

(3) $k \in \mathcal{T}_{u,d}(k)$, $p = 0$, and $v_{u,d}(k) - Div_{v_{u,d}(k)}(c_1) - y_0 < b$: In this case, since $p = 0$, the first claim is implemented after the dividend. Since the resulting surplus level is below the threshold, the "mainstream" path yields the "optimal" level at time $k+n$, namely:

$$\tilde{v}_{u,d}(k,n) = \tilde{v}^{ord}_{v_{u,d}(k)-Div_{v_{u,d}(k)}(c_1)-y_0}(n).$$

(4) $k \in \mathcal{T}_{u,d}(k)$, $p = 0$, and $v_{u,d}(k) - Div_{v_{u,d}(k)}(c_1) - y_0 \geq b$: In this case, since $p = 0$, the first claim is implemented after the dividend. Since the "mainstream" path is above level b at time k, it is again susceptible to more dividends being paid out than a path starting below level b at time k, which can potentially result in having a lower surplus value at time $k + n$ than that alternate path. Therefore, we need to compare the surplus level at time $k+n$ from the "mainstream" path with that from the "$b-1/k$" path. Thus,

$$\tilde{v}_{u,d}(k,n) = \max\{\tilde{v}^{ord}_{v_{u,d}(k)-Div_{v_{u,d}(k)}(c_1)-y_0}(n), \tilde{v}^{ord}_{b-1}(n)\}.$$

(5) $k \in \mathcal{T}_{u,d}(k)$, $p > 0$, and $v_{u,d}(k) - y_0 - Div_{v_{u,d}(k)-y_0}(c_1) < b$: In this case, since $p > 0$, the first claim can either be implemented before or after the dividend. However, since dividends are restricted, it is clear that the first claim being implemented before the dividend is guaranteed to always yield a surplus value at least as high as the situation when the first claim is implemented after the dividend. Hence, we assign the "mainstream" path to be the one where the first claim occurs before the dividend, and thus,

$$\tilde{v}_{u,d}(k,n) = \tilde{v}^{ord}_{v_{u,d}(k)-y_0-Div_{v_{u,d}(k)-y_0}(c_1)}(n).$$

(6) $k \in \mathcal{T}_{u,d}(k)$, $p > 0$, and $v_{u,d}(k) - y_0 - Div_{v_{u,d}(k)-y_0}(c_1) \geq b$: As stated in the above case, since $p > 0$ and dividends are restricted, the first claim being implemented before the dividend will always yield a surplus value at least as high as the situation when the first claim is implemented following the dividend. Hence, we assign the "mainstream" path to be the one where the first claim occurs before the dividend. Since this "mainstream" path is above level b at time k, it is susceptible to more dividends being paid out than a path starting below level b at time k, which can potentially result in having a lower surplus value at time $k+n$ than that alternate path. Therefore, we need to compare the surplus level at time $k+n$ from the "mainstream" path with that from the "$b-1/k$" path. Thus,

$$\tilde{v}_{u,d}(k,n) = \max\{\tilde{v}^{ord}_{v_{u,d}(k)-y_0-Div_{v_{u,d}(k)-y_0}(c_1)}(n), \tilde{v}^{ord}_{b-1}(n)\}.$$

The final expression combining the above six cases is shown on the next page.

$$\tilde{v}_u(t) = \begin{cases} u + ct - \mathbf{1}_{[t \geq n_r]} y_0 - \left\lfloor \dfrac{(t - n_r)_+}{n_a} \right\rfloor y_0 & \text{if } t < z_u + h; \\ \max\{u + ct - c_1, b\} & \text{if } t = z_u + h \text{ and } u \geq b; \\ \max\{u + ct - c_1, b - 1 + ch\} & \text{if } t = z_u + h \text{ and } u < b; \\ \max\left\{u + ct - \left\lceil \dfrac{t - z_u - h + 1}{h} \right\rceil c_1, b - 1 + c(t - z_u) - \left\lceil \dfrac{t - z_u - h}{h} \right\rceil c_1 \right\} & \text{if } t > z_u + h \text{ and } c_1 \leq c; \\ \max\left\{a_u \left(\left\lfloor \dfrac{t - z_u - h}{h} \right\rfloor \right) - Div_{a_u(\lfloor \frac{t - z_u - h}{h} \rfloor)}(c_1) + c \ mod(t - z_u, h), b - 1 + ch \right\} & \text{if } t > z_u + h \text{ and } c_1 > c. \end{cases}$$

$$\tilde{v}_{u,d}(k,n) = \begin{cases} \tilde{v}^{ord}_{v_{u,d}(k) - y_0}(n) & \text{if } k \notin \mathcal{T}_{u,d}(k) \text{ and } v_{u,d}(k) - y_0 < b; \\ \max\left\{\tilde{v}^{ord}_{v_{u,d}(k) - y_0}(n), \tilde{v}^{ord}_{b-1}(n)\right\} & \text{if } k \notin \mathcal{T}_{u,d}(k) \text{ and } v_{u,d}(k) - y_0 \geq b; \\ \tilde{v}^{ord}_{v_{u,d}(k) - Div_{v_{u,d}(k)}(c_1) - y_0}(n) & \text{if } k \in \mathcal{T}_{u,d}(k),\ p = 0, \text{ and } v_{u,d}(k) - Div_{v_{u,d}(k)}(c_1) - y_0 < b; \\ \max\left\{\tilde{v}^{ord}_{v_{u,d}(k) - Div_{v_{u,d}(k)}(c_1) - y_0}(n), \tilde{v}^{ord}_{b-1}(n)\right\} & \text{if } k \in \mathcal{T}_{u,d}(k),\ p = 0, \text{ and } v_{u,d}(k) - Div_{v_{u,d}(k)}(c_1) - y_0 \geq b; \\ \tilde{v}^{ord}_{v_{u,d}(k) - y_0 - Div_{v_{u,d}(k) - y_0}(c_1)}(n) & \text{if } k \in \mathcal{T}_{u,d}(k),\ p > 0, \text{ and } v_{u,d}(k) - y_0 - Div_{v_{u,d}(k) - y_0}(c_1) < b; \\ \max\left\{\tilde{v}^{ord}_{v_{u,d}(k) - y_0 - Div_{v_{u,d}(k) - y_0}(c_1)}(n), \tilde{v}^{ord}_{b-1}(n)\right\} & \text{if } k \in \mathcal{T}_{u,d}(k),\ p > 0, \text{ and } v_{u,d}(k) - y_0 - Div_{v_{u,d}(k) - y_0}(c_1) \geq b. \end{cases}$$

We now apply those ceiling values in selecting the relevant elements from the probability vectors $\underline{b}_i^{(k)}$, $\underline{g}_{n,i}^{(k)}$, and $\underline{h}_{n,-j}^{(k)}$ for computation.

3.5.2 Determining the Values of the Sub-vectors of $\underline{b}_i^{(k)}$

Let Ω_k represent the sample space of all possible non-ruin values that the surplus process can take at time k. By construction, $\Omega_k = \{0, 1, \ldots, v_{u,d}(k) - y_0\}$. Therefore, note that $\underline{b}_{i,\ell}^{(k)} = \underline{0}$ for $i \notin \Omega_k$. For $i \in \Omega_k$, the values of $\underline{b}_{i,\ell}^{(k)}$ can be broken up into two separate cases, which we detail below:

The first case is if $k \notin \mathcal{T}_{u,d}(k)$ (i.e., k is not a dividend paying time unit), so that there are $\left|\mathcal{T}_{u,d}(k-1)\right|$ total dividends. Furthermore, if we assume that the sum of these (restricted) dividends is equal to a value s, then the surplus process will be at level $u+ck-s$ immediately before the claim is implemented. In addition, the claim size will have to be equal to $u+ck-s-i$ in order for the surplus process to be at level i. Finally, the dividend counter will be equal to $V_{u,d}(0, k, u+ck-s-i, s, ., .)$. Combining these elements together, we obtain

$$\sum_{s \in \mathcal{R}_{u,d}(k-1)} f_{u,d}^{(|\mathcal{T}_{u,d}(k-1)|)}(s) \alpha_{u+ck-s-i} \mathbf{1}_{[\ell = V_{u,d}(0,k,u+ck-s-i,s,.,.)]} \cdot \underline{e}_1 \cdot$$

The second case is if $k \in \mathcal{T}_{u,d}(k)$ (i.e., k is a dividend paying time unit). Note that before time k, we have $\left|\mathcal{T}_{u,d}(k-1)\right|$ previous dividends. Furthermore, if we assume that the sum of these (restricted) dividends is equal to a value s, then the surplus process would be at level $u+ck-s$ immediately before the claim or dividend is first implemented. Since k is a dividend paying time unit, we need to consider two cases: the claim being implemented before the dividend and vice-versa.

If the claim is processed before the dividend, then the range of values of the claim size depends on the value of i. Specifically, if $i < b$, no dividend ends up being issued; in other words, a dividend amount of 0 is issued with probability 1. Hence, the claim size range simply consists of the singleton value $u + ck - s - i$. As in the first case, the value of the dividend counter, ℓ, is set equal to $V_{u,d}(0, k, u + ck - s - i, s, ., .)$.

However, if $i \geq b$, a potential non-zero dividend will be issued after the claim. Thus, the claim size will range from y_0 to $u + ck - s - i$. If we assume that this claim size is equal to y, then the dividend counter will be equal to $V_{u,d}(0, k, y, s, u + ck - s - y - i, ClmFrst)$. Note that the special case $i = b$ and $u + ck - s = b$ has a zero probability of occurring because this will imply that a claim of size 0 will be administered, which is impossible since claim sizes are at least equal to y_0, which is positively valued.

If the dividend is processed before the claim, then the range of the restricted dividend amount will depend on both i and the value of the surplus process at time k given by $u + ck - s$. Specifically, if $i < b$, then the restricted dividend amount will range from $\min\{c_1, u + ck - s - b\}$ to $\min\{c_2, u + ck - s - b\}$. However, if $i \geq b$, then the restricted dividend amount will range from $\min\{c_1, u+ck-s-b\}$ to $\min\{c_2, u+ck-s-(i+y_0)\}$. Note that if $c_1 > u+ck-s-(i+y_0)$, then $\min\{c_1, u+ck-s-b\} > \min\{c_2, u+ck-s-(i+y_0)\}$, implying that there will not be any restricted dividend amount small enough to bring the surplus process down to level $i + y_0$.

We can combine the above two situations since the value of i can be used to compare the relative values of $u + ck - s - b$ and $u + ck - s - (i + y_0)$ in the limits of the dividend amount. As a result, the restricted dividend amount will range from $\min\{c_1, u+ck-s-b\}$

to $\min\{c_2, u+ck-s-\max\{b, i+y_0\}\}$. Finally, by assuming a restricted dividend amount equal to s^* at time k, the value of the dividend counter, ℓ, will be equal to $V_{u,d}(0,k,u+ck-s-s^*-i,s,s^*,\overline{ClmFrst})$. Combining these elements together, we obtain

$$\sum_{s\in\mathcal{R}_{u,d}(k-1)} f_{u,d}^{(|\mathcal{T}_{u,d}(k-1)|)}(s)\left\{p\sum_{y=y_0}^{u+ck-s-i}\alpha_y d^*_{u+ck-s-y-i}\{u+ck-s-y\}\mathbf{1}_{[\ell=V_{u,d}(0,k,y,s,u+ck-s-y-i,ClmFrst)]}\right.$$
$$\left.+(1-p)\sum_{s^*=\min\{c_1,u+ck-s-b\}}^{\min\{c_2,u+ck-s-\max\{b,i+y_0\}\}} d^*_{s^*}\{u+ck-s\}\ \alpha_{u+ck-s-s^*-i}\mathbf{1}_{[\ell=V_{u,d}(0,k,u+ck-s-s^*-i,s,s^*,\overline{ClmFrst})]}\right\}\cdot\underline{e}_1.$$

Finally, the result of combining the above two cases together is shown on page 64.

3.5.3 Determining the Values of the Sub-vectors of $\underline{g}_{n,i}^{(k)}$

Note that since $\tilde{\underline{g}}_n^{(k)} = \left(\underline{\tilde{g}}_{n,0}^{(k)}, \underline{\tilde{g}}_{n,1}^{(k)}, \underline{\tilde{g}}_{n,2}^{(k)}, \ldots\right) = \underline{\tilde{b}}^{(k)}Q^n$ for $n\in\mathbb{N}$, we have $\underline{\tilde{g}}_0^{(k)} = \underline{\tilde{b}}^{(k)}$ and $\underline{\tilde{g}}_n^{(k)} = \underline{\tilde{g}}_{n-1}^{(k)}Q$ for $n\in\mathbb{Z}^+$. In particular, this recursive relation can be expressed more explicitly as follows:

$$\left(\underline{\tilde{g}}_{n,0}^{(k)}, \underline{\tilde{g}}_{n,1}^{(k)}, \underline{\tilde{g}}_{n,2}^{(k)}, \ldots\right) = \left(\underline{\tilde{g}}_{n-1,0}^{(k)}, \underline{\tilde{g}}_{n-1,1}^{(k)}, \underline{\tilde{g}}_{n-1,2}^{(k)}, \ldots\right)\times$$
$$\begin{array}{c} \\ 0\\1\\2\\\vdots\\b-1\\b\\b+1\\\vdots\end{array}\!\!\begin{array}{c}\begin{array}{ccccccc}0 & 1 & \cdots & b-1 & b & b+1 & \cdots\end{array}\\\begin{pmatrix}\tilde{A}_{0,0} & \tilde{A}_{0,1} & \cdots & \tilde{A}_{0,b-1} & \tilde{A}_{0,b} & \tilde{A}_{0,b+1} & \cdots\\ \tilde{A}_{1,0} & \tilde{A}_{1,1} & \cdots & \tilde{A}_{1,b-1} & \tilde{A}_{1,b} & \tilde{A}_{1,b+1} & \cdots\\ \tilde{A}_{2,0} & \tilde{A}_{2,1} & \cdots & \tilde{A}_{2,b-1} & \tilde{A}_{2,b} & \tilde{A}_{2,b+1} & \cdots\\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots\\ \tilde{A}_{b-1,0} & \tilde{A}_{b-1,1} & \cdots & \tilde{A}_{b-1,b-1} & \tilde{A}_{b-1,b} & \tilde{A}_{b-1,b+1} & \cdots\\ \tilde{B}_{b,0} & \tilde{B}_{b,1} & \cdots & \tilde{B}_{b,b-1} & \tilde{C}_{b,b} & \tilde{C}_{b,b+1} & \cdots\\ \tilde{B}_{b+1,0} & \tilde{B}_{b+1,1} & \cdots & \tilde{B}_{b+1,b-1} & \tilde{C}_{b+1,b} & \tilde{C}_{b+1,b+1} & \cdots\\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots\end{pmatrix}\end{array}.$$

Noting that $\underline{\tilde{g}}_{n-1,i}^{(k)}$ has zero-valued elements for $i > \tilde{v}_{u,d}(k,n-1)$, the above equation simplifies to give the following result:

$$\underline{\tilde{g}}_{n,i}^{(k)} = \begin{cases}\underline{\tilde{b}}_i^{(k)} & \text{if } n=0;\\ \displaystyle\sum_{r=0}^{b-1}\underline{\tilde{g}}_{n-1,r}^{(k)}\tilde{A}_{r,i} + \sum_{r=b}^{\tilde{v}_{u,d}(k,n-1)}\underline{\tilde{g}}_{n-1,r}^{(k)}\left(\tilde{B}_{r,i}\mathbf{1}_{[i<b]}+\tilde{C}_{r,i}\mathbf{1}_{[i\geq b]}\right) & \text{if } n\in\mathbb{Z}^+.\end{cases}$$

The next step is to break down the above block-matrices into smaller sub-blocks. Specifically, for $n \in \mathbb{Z}^+$, we have

$$\begin{aligned}
\underline{\tilde{g}}_{n,i}^{(k)} &= \left(\underline{g}_{n,i,1}^{(k)}, \underline{g}_{n,i,2}^{(k)}, \ldots, \underline{g}_{n,i,h}^{(k)}\right) \\
&= \sum_{r=0}^{b-1} \left(\underline{g}_{n-1,r,1}^{(k)}, \underline{g}_{n-1,r,2}^{(k)}, \ldots, \underline{g}_{n-1,r,h}^{(k)}\right) \tilde{A}_{r,i} \\
&+ \sum_{r=b}^{\tilde{v}_{u,d}(k,n-1)} \left(\underline{g}_{n-1,r,1}^{(k)}, \underline{g}_{n-1,r,2}^{(k)}, \ldots, \underline{g}_{n-1,r,h}^{(k)}\right) \left(\tilde{B}_{r,i}\mathbf{1}_{[i<b]} + \tilde{C}_{r,i}\mathbf{1}_{[i\geq b]}\right).
\end{aligned}$$

Equating components for $\ell \in \{1, 2, \ldots, h\}$, we immediately obtain

$$\underline{g}_{n,i,\ell}^{(k)} = \begin{cases} \underline{b}_{i,\ell}^{(k)} & \text{if } n = 0; \\ \sum_{r=0}^{b-1} \sum_{m=1}^{h} \underline{g}_{n-1,r,m}^{(k)} [\tilde{A}_{r,i}]_{m,\ell} & \\ + \sum_{r=b}^{\tilde{v}_{u,d}(k,n-1)} \sum_{m=1}^{h} \underline{g}_{n-1,r,m}^{(k)} \left([\tilde{B}_{r,i}]_{m,\ell}\mathbf{1}_{[i<b]} + [\tilde{C}_{r,i}]_{m,\ell}\mathbf{1}_{[i\geq b]}\right) & \text{if } n \in \mathbb{Z}^+. \end{cases}$$

Since $[\tilde{A}_{r,i}]_{m,\ell} = \mathbf{1}_{[\ell=V_{r,m}(0,1,r+c-i,0,.,.)]} B_{r+c-i}$, the first term in the second case above vanishes for $r < i - c$, since $B_i = \mathbf{0}$ for $i < 0$. Thus, we can replace $r = 0$, the lower index of the first summation (found in the first term in the second case above), with $r = \max\{0, i - c\}$. By decomposing $[\tilde{B}_{r,i}]_{m,\ell}$ and $[\tilde{C}_{r,i}]_{m,\ell}$ in the second term, and examining the B_i matrices within them, we find that these matrices are equal to $\mathbf{0}$ for $i - c < 0$. Thus, similar to the first summation in the first term, we can replace the lower index of the first summation in the second term with $r = \max\{b, i - c\}$.

Substituting the explicit expressions of the block-matrices given on page 39 into the above equation for $\underline{g}^{(k)}_{n,i,\ell}$, we obtain the resulting expression shown on page 64.

3.5.4 Determining the Values of the Sub-vectors of $\underline{h}^{(k)}_{n,-j}$

Note that since $\underline{\tilde{h}}^{(k)}_n = \left(\underline{\tilde{h}}^{(k)}_{n,-1}, \underline{\tilde{h}}^{(k)}_{n,-2}, \underline{\tilde{h}}^{(k)}_{n,-3}, \ldots\right) = \underline{\tilde{b}}^{(k)} Q^{n-1} R$ for $n \in \mathbb{Z}^+$, it clearly follows that $\underline{\tilde{h}}^{(k)}_n = \underline{\tilde{g}}^{(k)}_{n-1} R$ for $n \in \mathbb{Z}^+$. As done in the previous sub-section, we start by writing out this recursive formula more explicitly as follows:

$$\left(\underline{\tilde{h}}^{(k)}_{n,-1}, \underline{\tilde{h}}^{(k)}_{n,-2}, \underline{\tilde{h}}^{(k)}_{n,-3}, \ldots\right) = \left(\underline{\tilde{g}}^{(k)}_{n-1,0}, \underline{\tilde{g}}^{(k)}_{n-1,1}, \underline{\tilde{g}}^{(k)}_{n-1,2}, \ldots\right) \times \begin{array}{c} \\ 0 \\ 1 \\ 2 \\ \vdots \\ b-1 \\ b \\ b+1 \\ \vdots \end{array} \begin{array}{c} \begin{array}{cccc} -1 & -2 & -3 & \cdots \end{array} \\ \begin{pmatrix} \tilde{A}_{0,-1} & \tilde{A}_{0,-2} & \tilde{A}_{0,-3} & \cdots \\ \tilde{A}_{1,-1} & \tilde{A}_{1,-1} & \tilde{A}_{1,-3} & \cdots \\ \tilde{A}_{2,-1} & \tilde{A}_{2,-2} & \tilde{A}_{2,-3} & \cdots \\ \vdots & \vdots & \vdots & \ddots \\ \tilde{A}_{b-1,-1} & \tilde{A}_{b-1,-2} & \tilde{A}_{b-1,-3} & \cdots \\ \tilde{B}_{b,-1} & \tilde{B}_{b,-2} & \tilde{B}_{b,-3} & \cdots \\ \tilde{B}_{b+1,-1} & \tilde{B}_{b+1,-2} & \tilde{B}_{b+1,-3} & \cdots \\ \vdots & \vdots & \vdots & \ddots \end{pmatrix} \end{array}.$$

As stated in the previous sub-section, since $\underline{\tilde{g}}^{(k)}_{n-1,i}$ has zero-valued elements for $i > \tilde{v}_{u,d}(k, n-1)$, we obtain the following recursive result:

$$\underline{\tilde{h}}^{(k)}_{n,-j} = \sum_{r=0}^{b-1} \underline{\tilde{g}}^{(k)}_{n-1,r} \tilde{A}_{r,-j} + \sum_{r=b}^{\tilde{v}_{u,d}(k,n-1)} \underline{\tilde{g}}^{(k)}_{n-1,r} \tilde{B}_{r,-j} \text{ for } n \in \mathbb{Z}^+.$$

The next step is to break down the above block-matrices into smaller sub-blocks. Specifically, for $n \in \mathbb{Z}^+$, we have

$$\underline{\tilde{h}}^{(k)}_{n,-j} = \left(\underline{h}^{(k)}_{n,-j,1}, \underline{h}^{(k)}_{n,-j,2}, \ldots, \underline{h}^{(k)}_{n,-j,h}\right)$$

$$= \sum_{r=0}^{b-1} \left(\underline{g}^{(k)}_{n-1,r,1}, \underline{g}^{(k)}_{n-1,r,2}, \ldots, \underline{g}^{(k)}_{n-1,r,h}\right) \tilde{A}_{r,-j}$$

$$+ \sum_{r=b}^{\tilde{v}_{u,d}(k,n-1)} \left(\underline{g}^{(k)}_{n-1,r,1}, \underline{g}^{(k)}_{n-1,r,2}, \ldots, \underline{g}^{(k)}_{n-1,r,h}\right) \tilde{B}_{r,-j}.$$

Equating components for $\ell \in \{1, 2, \ldots, h\}$, we obtain

$$\underline{h}^{(k)}_{n,-j,\ell} = \sum_{r=0}^{b-1} \sum_{m=1}^{h} \underline{g}^{(k)}_{n-1,r,m} [\tilde{A}_{r,-j}]_{m,\ell} + \sum_{r=b}^{\tilde{v}_{u,d}(k,n-1)} \sum_{m=1}^{h} \underline{g}^{(k)}_{n-1,r,m} [\tilde{B}_{r,-j}]_{m,\ell} \text{ for } n \in \mathbb{Z}^+.$$

Once again, substituting the explicit expressions of the block-matrices given on page 39 into the above equation for $\underline{h}^{(k)}_{n,-j,\ell}$, we obtain the resulting expression shown on the next page.

$$\underline{b}_{i,\ell}^{(k)} = \begin{cases} \displaystyle\sum_{s\in\mathcal{R}_{u,d}(k-1)} f_{u,d}^{\left(\left|\mathcal{T}_{u,d}(k-1)\right|\right)}(s)\alpha_{u+ck-s-i}\mathbf{1}_{[\ell=V_{u,d}(0,k,u+ck-s-i,s,.,.)]}\cdot\underline{e}_1 & \text{if } k\notin\mathcal{T}_{u,d}(k); \\ \displaystyle\sum_{s\in\mathcal{R}_{u,d}(k-1)} f_{u,d}^{\left(\left|\mathcal{T}_{u,d}(k-1)\right|\right)}(s)\left\{p\sum_{y=y_0}^{u+ck-s-i}\alpha_y d^*_{u+ck-s-y-i}\{u+ck-s-y\}\mathbf{1}_{[\ell=V_{u,d}(0,k,y,s,u+ck-s-y-i,ClmFrst)]}\right. & \text{if } k\in\mathcal{T}_{u,d}(k). \\ \qquad\left.+(1-p)\sum_{s^*=\min\{c_1,u+ck-s-b\}}^{\min\{c_2,u+ck-s-\max\{b,i+y_0\}\}} d^*_{s^*}\{u+ck-s\}\ \alpha_{u+ck-s-s^*-i}\mathbf{1}_{[\ell=V_{u,d}(0,k,u+ck-s-s^*-i,s,s^*,\overline{ClmFrst})]}\right\}\cdot\underline{e}_1 & \end{cases}$$

$$\underline{g}_{n,i,\ell}^{(k)} = \begin{cases} \underline{b}_{i,\ell}^{(k)} & \text{if } n=0; \\ \displaystyle\sum_{r=\max\{0,i-c\}}^{\tilde{v}_{u,d}(k,n-1)}\sum_{m=1}^{h-1}\underline{\tilde{g}}_{n-1,r,m}^{(k)}\left(\mathbf{1}_{[\ell=V_{r,m}(0,1,r+c-i,0,.,.)]}B_{r+c-i}\right) & \text{if } n\in\mathbb{Z}^+. \\ +\displaystyle\sum_{r=\max\{0,i-c\}}^{b-1}\underline{\tilde{g}}_{n-1,r,h}^{(k)}\left(\mathbf{1}_{[\ell=V_{r,h}(0,1,r+c-i,0,.,.)]}B_{r+c-i}\right) & \\ +\displaystyle\sum_{r=\max\{b,i-c\}}^{\tilde{v}_{u,d}(k,n-1)}\underline{\tilde{g}}_{n-1,r,h}^{(k)}\left(p\mathbf{1}_{[\ell=V_{r,h}(0,1,r+c-i,0,.,.)]}B_{r+c-i}+(1-p)\sum_{s^*=\min\{c_1,r+c-b\}}^{\min\{c_2,r+c-b\}} d^*_{s^*}\{r+c\}\mathbf{1}_{[\ell=V_{r,h}(0,1,r+c-s^*-i,0,s^*,\overline{ClmFrst})]}B_{r+c-s^*-i}\right)\mathbf{1}_{[i<b]} & \\ +\displaystyle\sum_{r=\max\{b,i-c\}}^{\tilde{v}_{u,d}(k,n-1)}\underline{\tilde{g}}_{n-1,r,h}^{(k)}\left(p\sum_{y=0}^{r+c-i} d^*_{r+c-y-i}\{r+c-y\}\mathbf{1}_{[\ell=V_{r,h}(0,1,y,0,r+c-y-i,ClmFrst)]}B_y\right. & \\ \qquad\left.+(1-p)\sum_{s^*=\min\{c_1,r+c-b\}}^{\min\{c_2,r+c-i\}} d^*_{s^*}\{r+c\}\mathbf{1}_{[\ell=V_{r,h}(0,1,r+c-s^*-i,0,s^*,\overline{ClmFrst})]}B_{r+c-s^*-i}\right)\mathbf{1}_{[i\geq b]} & \end{cases}$$

$$\underline{h}_{n,-j,\ell}^{(k)} = \sum_{r=0}^{\tilde{v}_{u,d}(k,n-1)}\sum_{m=1}^{h-1}\underline{\tilde{g}}_{n-1,r,m}^{(k)}\left(\mathbf{1}_{[\ell=V_{r,m}(0,1,r+c+j,0,.,.)]}B_{r+c+j}\right) + \sum_{r=0}^{b-1}\underline{\tilde{g}}_{n-1,r,h}^{(k)}\left(\mathbf{1}_{[\ell=V_{r,h}(0,1,r+c+j,0,.,.)]}B_{r+c+j}\right)$$
$$+\sum_{r=b}^{\tilde{v}_{u,d}(k,n-1)}\underline{\tilde{g}}_{n-1,r,h}^{(k)}\left(p\mathbf{1}_{[\ell=V_{r,h}(0,1,r+c+j,0,.,.)]}B_{r+c+j}+(1-p)\sum_{s^*=\min\{c_1,r+c-b\}}^{\min\{c_2,r+c-b\}} d^*_{s^*}\{r+c\}\mathbf{1}_{[\ell=V_{r,h}(0,1,r+c-s^*+j,0,s^*,\overline{ClmFrst})]}B_{r+c-s^*+j}\right).$$

3.6 Determining the Conditional Trivariate and Bivariate Ruin Probabilities

For $n \in \mathbb{Z}^+$, we define three conditional ruin probabilities of interest. First of all, let $\psi_{n,i,j}^{(k)}(u,d)$ be the probability of ruin occurring at time $k+n$, such that the surplus prior to ruin and the deficit at ruin are equal to i and j, respectively, given that the first claim occurred at time k. This conditional trivariate ruin probability is mathematically equal to $Pr\{T = k+n, U_{T-} = i, |U_T| = j | U_k \in \Omega_k, U_0 = u, D_0 = d\}$. Let $\omega_{n,i}^{(k)}(u,d)$ be the probability of ruin occurring at time $k+n$, such that the surplus prior to ruin is equal to i, given that the first claim occurred at time k. This conditional bivariate ruin probability is mathematically equal to $Pr\{T = k+n, U_{T-} = i | U_k \in \Omega_k, U_0 = u, D_0 = d\}$. Finally, let $\phi_{n,j}^{(k)}(u,d)$ be the probability of ruin occurring at time $k+n$, such that the deficit at ruin is equal to j, given that the first claim occurred at time k. This conditional bivariate ruin probability is mathematically equal to $Pr\{T = k+n, |U_T| = j \ | U_k \in \Omega_k, U_0 = u, D_0 = d\}$.

It is worthwhile to note that ruin cannot occur if any of the following holds true:

(1) The maximum possible claim size y_α does not exceed i_{min} (i.e, the ruining claim must exceed the smallest surplus prior to ruin value).

(2) $i > \tilde{v}_{u,d}(k,n)$ (i.e, the surplus prior to ruin at time $k+n$ cannot exceed the ceiling value).

(3) $j > y_\alpha - i_{min}$ (i.e., the deficit at ruin cannot exceed the difference between the largest claim size and smallest surplus prior to ruin value).

Now, we turn our attention to deriving algorithmic expressions for each of the above conditional ruin probabilities.

3.6.1 Calculating $\psi^{(k)}_{n,i,j}(u,d)$

Keeping in mind that ruin does not occur in the first $n-1$ transitions from time k, we examine $\psi^{(k)}_{n,i,j}(u,d)$ under the following four non-overlapping, exhaustive cases:

Case 1: $c \leq c_2$ and $c \leq b$

In this case, $\min\{c, b+(c-c_2)_+\} = c$. Now, we examine the conditional trivariate ruin probability, $\psi^{(k)}_{n,i,j}(u,d)$, under five non-overlapping, exhaustive sub-cases:

(1) $i < c$: This is an impossible scenario, and hence, $\psi^{(k)}_{n,i,j}(u,d) = 0$.

(2) $c \leq i < b$:

At time $k+n-1$, the process must be at level $i-c$, without the possibility of a dividend being issued at time $k+n$. Hence, we simply have

$$\psi^{(k)}_{n,i,j}(u,d) = \sum_{\ell=1}^{h} \underline{g}^{(k)}_{n-1,i-c,\ell}\underline{s}'\alpha_{i+j}.$$

(3) $i = b$:

Here, we have two possible scenarios to consider at time $k+n$: one where no dividend is issued, and one where a dividend is issued immediately before the ruining claim. Note that the first scenario is identical to the above sub-case, and thus, we have $\sum_{\ell=1}^{h} \underline{g}^{(k)}_{n-1,b-c,\ell}\underline{s}'\alpha_{b+j}$ contributing to $\psi^{(k)}_{n,b,j}(u,d)$. As for the second scenario, since dividend amounts are restricted, and we are examining the surplus process at level b, it is possible that the original (unrestricted) dividend amount could have been large enough to bring the surplus process below b. Thus, by considering all possible values of the surplus process at time $k+n-1$, we ultimately have

$(1-p)\sum_{s=c}^{c_2}\sum_{s^*=\max\{c_1,s\}}^{c_2} d_{s^*}\ \underline{g}^{(k)}_{n-1,b+s-c,h}\underline{s}'\alpha_{b+j}$ contributing to $\psi^{(k)}_{n,b,j}(u,d)$.

Hence, combining the above two scenarios, we get

$$\psi^{(k)}_{n,b,j}(u,d)=\left(\sum_{\ell=1}^{h}\underline{g}^{(k)}_{n-1,b-c,\ell}+(1-p)\sum_{s=c}^{c_2}\sum_{s^*=\max\{c_1,s\}}^{c_2} d_{s^*}\ \underline{g}^{(k)}_{n-1,b+s-c,h}\right)\underline{s}'\alpha_{b+j}.$$

(4) $b<i<b+c$:

As in the previous sub-case, we have two possible scenarios to consider at time $k+n$: one where no dividend is issued, and one where a dividend is issued immediately before the ruining claim. We remark that the first scenario is identical to the second sub-case, and thus, we have $\sum_{\ell=1}^{h}\underline{g}^{(k)}_{n-1,i-c,\ell}\underline{s}'\alpha_{i+j}$ contributing to $\psi^{(k)}_{n,i,j}(u,d)$. Concerning the second scenario, and unlike the second scenario in the previous sub-case, since the surplus process is above level b, dividend amounts are not capped. Thus, by considering all possible values of the surplus process at time $k+n-1$, this scenario yields $(1-p)\sum_{s^*=\max\{c_1,b+c-i\}}^{c_2} d_{s^*}\ \underline{g}^{(k)}_{n-1,i+s^*-c,h}\underline{s}'\alpha_{i+j}$ contributing to $\psi^{(k)}_{n,i,j}(u,d)$. The reason for having $\max\{c_1,b+c-i\}$ as the lower index of s^*, as opposed to just c_1, is that we need to insure that the surplus process at time $k+n-1$ is at or above level b (i.e., $i+s^*-c\geq b$); otherwise, a dividend at time $k+n$ would not be possible.

Hence, combining the above two scenarios, we get

$$\psi^{(k)}_{n,i,j}(u,d)=\left(\sum_{\ell=1}^{h}\underline{g}^{(k)}_{n-1,i-c,\ell}+(1-p)\sum_{s^*=\max\{c_1,b+c-i\}}^{c_2} d_{s^*}\ \underline{g}^{(k)}_{n-1,i+s^*-c,h}\right)\underline{s}'\alpha_{i+j}.$$

(5) $i \geq b + c$:

Here, we actually have three possible scenarios to consider at time $k + n$: one where no dividend is issued, one where a potential dividend would have been issued had it not been for the ruining claim, and one where a dividend is issued immediately before the ruining claim. For the first scenario, we have $\sum_{\ell=1}^{h-1} \underline{g}^{(k)}_{n-1,i-c,\ell} \underline{s}' \alpha_{i+j}$ contributing to $\psi^{(k)}_{n,i,j}(u, d)$. Note that unlike in the previous sub-cases, we do not include here the dividend counter value h in the summation since, at time $k + n - 1$, the surplus process is at a level high enough for a dividend to be issued should the dividend counter value be equal to h, which is not in the scenario we are considering here. In the case of the second scenario, we simply have $p\underline{g}^{(k)}_{n-1,i-c,h} \underline{s}' \alpha_{i+j}$ contributing to $\psi^{(k)}_{n,i,j}(u, d)$. Finally, for the third scenario, by considering all possible values of the surplus process at time $k + n - 1$, we have $(1 - p) \sum_{s^*=c_1}^{c_2} d_{s^*} \ \underline{g}^{(k)}_{n-1,i+s^*-c,h} \underline{s}' \alpha_{i+j}$ contributing to $\psi^{(k)}_{n,i,j}(u, d)$. Note that, unlike the previous sub-case, s^* has a minimum value of c_1 since $b + c - i$ is guaranteed to be non-positive for $i \geq b + c$.

Hence, combining the above three scenarios, we get

$$\psi^{(k)}_{n,i,j}(u, d) = \left(\sum_{\ell=1}^{h-1} \underline{g}^{(k)}_{n-1,i-c,\ell} + p\underline{g}^{(k)}_{n-1,i-c,h} + (1 - p) \sum_{s^*=c_1}^{c_2} d_{s^*} \ \underline{g}^{(k)}_{n-1,i+s^*-c,h} \right) \underline{s}' \alpha_{i+j}.$$

Therefore, combining all of the above sub-cases, we have the following expression:

$$\psi_{n,i,j}^{(k)}(u,d)=\begin{cases}0 & \text{if } i<c;\\ \sum_{\ell=1}^{h}\underline{g}_{n-1,i-c,\ell}^{(k)}\underline{s}'\alpha_{i+j} & \text{if } c\le i<b;\\ \left(\sum_{\ell=1}^{h}\underline{g}_{n-1,b-c,\ell}^{(k)}+(1-p)\sum_{s=c}^{c_2}\sum_{s^*=\max\{c_1,s\}}^{c_2}d_{s^*}\ \underline{g}_{n-1,b+s-c,h}^{(k)}\right)\underline{s}'\alpha_{b+j} & \text{if } i=b;\\ \left(\sum_{\ell=1}^{h}\underline{g}_{n-1,i-c,\ell}^{(k)}+(1-p)\sum_{s^*=\max\{c_1,b+c-i\}}^{c_2}d_{s^*}\ \underline{g}_{n-1,i+s^*-c,h}^{(k)}\right)\underline{s}'\alpha_{i+j} & \text{if } b<i<b+c;\\ \left(\sum_{\ell=1}^{h-1}\underline{g}_{n-1,i-c,\ell}^{(k)}+p\underline{g}_{n-1,i-c,h}^{(k)}+(1-p)\sum_{s^*=c_1}^{c_2}d_{s^*}\ \underline{g}_{n-1,i+s^*-c,h}^{(k)}\right)\underline{s}'\alpha_{i+j} & \text{if } i\ge b+c.\end{cases}$$

Case 2: $c \le c_2$ and $c > b$

In this case, $\min\{c, b+(c-c_2)_+\} = b$. Similar to the approach used above, we examine the conditional trivariate ruin probability, $\psi_{n,i,j}^{(k)}(u,d)$, under five non-overlapping, exhaustive sub-cases:

(1) $i < b$: This is an impossible scenario, and hence, $\psi_{n,i,j}^{(k)}(u,d) = 0$.

(2) $i = b$:

This sub-case is valid only if a dividend is issued immediately before the ruining claim. Since dividend amounts are restricted, and we are examining the surplus process at level b, it is possible that the original (unrestricted) dividend amount could have been large enough to bring the process below b. Hence, by considering all possible values of the surplus process at time $k+n-1$, we have

$$\psi_{n,b,j}^{(k)}(u,d)=(1-p)\sum_{s=c}^{c_2}\sum_{s^*=\max\{c_1,s\}}^{c_2}d_{s^*}\ \underline{g}_{n-1,b+s-c,h}^{(k)}\underline{s}'\alpha_{b+j}.$$

(3) $b < i < c$:

As in the previous sub-case, this sub-case is valid only if a dividend is issued immediately before the ruining claim. Unlike the above sub-case, however, since the surplus process is above level b, dividend amounts are not capped. Hence, by considering all possible values of the surplus process at time $k+n-1$, we have

$$\psi^{(k)}_{n,i,j}(u,d) = (1-p) \sum_{s^*=\max\{c_1, b+c-i\}}^{c_2} d_{s^*}\ \underline{g}^{(k)}_{n-1,i+s^*-c,h}\underline{s}'\alpha_{i+j}.$$

(4) $c \leq i < b+c$:

Here, we have two possible scenarios to consider at time $k+n$: one where no dividend is issued, and one where a dividend is issued immediately before the ruining claim. Note that the first scenario is identical to the second sub-case in Case 1, and thus, we have $\sum_{\ell=1}^{h} \underline{g}^{(k)}_{n-1,i-c,\ell}\underline{s}'\alpha_{i+j}$ contributing to $\psi^{(k)}_{n,i,j}(u,d)$. The second scenario, on the other hand, is identical to the previous sub-case, and thus, we have
$(1-p) \sum_{s^*=\max\{c_1, b+c-i\}}^{c_2} d_{s^*}\ \underline{g}^{(k)}_{n-1,i+s^*-c,h}\underline{s}'\alpha_{i+j}$ contributing to $\psi^{(k)}_{n,i,j}(u,d)$.

Hence, combining the above two scenarios, we get

$$\psi^{(k)}_{n,i,j}(u,d) = \left(\sum_{\ell=1}^{h} \underline{g}^{(k)}_{n-1,i-c,\ell} + (1-p) \sum_{s^*=\max\{c_1, b+c-i\}}^{c_2} d_{s^*}\ \underline{g}^{(k)}_{n-1,i+s^*-c,h} \right) \underline{s}'\alpha_{i+j}.$$

(5) $i \geq b+c$:

Here, we have three possible scenarios to consider at time $k+n$: one where no dividend is issued, one where a potential dividend would have been issued had it not been for the ruining claim, and one where a dividend is issued immediately before the ruining claim. For the first scenario, as explained in the fifth sub-case of Case 1, we have $\sum_{\ell=1}^{h-1} \underline{g}^{(k)}_{n-1,i-c,\ell}\underline{s}'\alpha_{i+j}$ contributing to $\psi^{(k)}_{n,i,j}(u,d)$. In the case of the second

scenario, we simply have $p\underline{g}^{(k)}_{n-1,i-c,h}\underline{s}'\alpha_{i+j}$ contributing to $\psi^{(k)}_{n,i,j}(u,d)$. Finally, for the third scenario, by considering all possible values of the surplus process at time $k+n-1$, we have $(1-p)\sum_{s^*=c_1}^{c_2} d_{s^*}\ \underline{g}^{(k)}_{n-1,i+s^*-c,h}\underline{s}'\alpha_{i+j}$ contributing to $\psi^{(k)}_{n,i,j}(u,d)$. Note that, unlike the previous sub-case, s^* has a minimum value of c_1, since $b+c-i$ is again non-positive for $i \geq b+c$.

Hence, combining the above three scenarios, we get

$$\psi^{(k)}_{n,i,j}(u,d) = \left(\sum_{\ell=1}^{h-1} \underline{g}^{(k)}_{n-1,i-c,\ell} + p\underline{g}^{(k)}_{n-1,i-c,h} + (1-p)\sum_{s^*=c_1}^{c_2} d_{s^*}\ \underline{g}^{(k)}_{n-1,i+s^*-c,h}\right)\underline{s}'\alpha_{i+j}.$$

Therefore, combining all of the above sub-cases, we have the following expression:

$$\psi^{(k)}_{n,i,j}(u,d) = \begin{cases} 0 & \text{if } i < b; \\ (1-p)\sum_{s=c}^{c_2}\sum_{s^*=\max\{c_1,s\}}^{c_2} d_{s^*}\ \underline{g}^{(k)}_{n-1,b+s-c,h}\underline{s}'\alpha_{b+j} & \text{if } i = b; \\ (1-p)\sum_{s^*=\max\{c_1,b+c-i\}}^{c_2} d_{s^*}\ \underline{g}^{(k)}_{n-1,i+s^*-c,h}\underline{s}'\alpha_{i+j} & \text{if } b < i < c; \\ \left(\sum_{\ell=1}^{h} \underline{g}^{(k)}_{n-1,i-c,\ell} + (1-p)\sum_{s^*=\max\{c_1,b+c-i\}}^{c_2} d_{s^*}\ \underline{g}^{(k)}_{n-1,i+s^*-c,h}\right)\underline{s}'\alpha_{i+j} & \text{if } c \leq i < b+c; \\ \left(\sum_{\ell=1}^{h-1} \underline{g}^{(k)}_{n-1,i-c,\ell} + p\underline{g}^{(k)}_{n-1,i-c,h} + (1-p)\sum_{s^*=c_1}^{c_2} d_{s^*}\ \underline{g}^{(k)}_{n-1,i+s^*-c,h}\right)\underline{s}'\alpha_{i+j} & \text{if } i \geq b+c. \end{cases}$$

We remark that in the case of $p = 1$, the above expression for $\psi^{(k)}_{n,i,j}(u,d)$ yields a value of 0 for all $i < c$, in agreement with our earlier observation that the minimum surplus prior to ruin value is equal to c in this case.

Case 3: $c > c_2$ and $c \leq b + (c - c_2)_+$

In this case, $\min\{c, b + (c - c_2)_+\} = c$. Next, we examine the conditional trivariate ruin probability, $\psi^{(k)}_{n,i,j}(u, d)$, under four non-overlapping, exhaustive sub-cases:

(1) $i < c$: This is an impossible scenario, and hence, $\psi^{(k)}_{n,i,j}(u, d) = 0$.

(2) $c \leq i < b + c - c_2$:

At time $k+n-1$, we observe that the process was at level $i-c$, without the possibility of a dividend being issued at time $k + n$. Furthermore, note that for this case, unlike the previous two, the surplus process cannot be brought down to level b by a dividend (immediately before implementing the ruining claim) since $c > c_2$. Hence, we simply have

$$\psi^{(k)}_{n,i,j}(u, d) = \sum_{\ell=1}^{h} \underline{g}^{(k)}_{n-1,i-c,\ell} \underline{s}' \alpha_{i+j}.$$

(3) $b + c - c_2 \leq i < b + c$:

Here, we have two possible scenarios to consider at time $k+n$: one where no dividend is issued, and one where a dividend is issued immediately before the ruining claim. Note that the first scenario is identical to the previous sub-case, and thus, we have $\sum_{\ell=1}^{h} \underline{g}^{(k)}_{n-1,i-c,\ell} \underline{s}' \alpha_{i+j}$ contributing to $\psi^{(k)}_{n,i,j}(u, d)$. The second scenario, on the other hand, is identical to the second scenario of the fourth sub-case of Case 2, and thus, we have $(1 - p) \sum_{s^*=\max\{c_1, b+c-i\}}^{c_2} d_{s^*} \ \underline{g}^{(k)}_{n-1,i+s^*-c,h} \underline{s}' \alpha_{i+j}$ contributing to $\psi^{(k)}_{n,i,j}(u, d)$.

Hence, combining the above two scenarios, we get

$$\psi^{(k)}_{n,i,j}(u, d) = \left(\sum_{\ell=1}^{h} \underline{g}^{(k)}_{n-1,i-c,\ell} + (1 - p) \sum_{s^*=\max\{c_1, b+c-i\}}^{c_2} d_{s^*} \ \underline{g}^{(k)}_{n-1,i+s^*-c,h} \right) \underline{s}' \alpha_{i+j}.$$

(4) $i \geq b + c$:

Here, we have three possible scenarios to consider at time $k + n$: one where no dividend is issued, one where a potential dividend would have been issued had it not been for the ruining claim, and one where a dividend is issued immediately before the ruining claim. For the first scenario, as explained in the fifth sub-case of Case 1, we have $\sum_{\ell=1}^{h-1} \underline{g}^{(k)}_{n-1,i-c,\ell} \underline{s}' \alpha_{i+j}$ contributing to $\psi^{(k)}_{n,i,j}(u,d)$. In the case of the second scenario, we simply have $p\underline{g}^{(k)}_{n-1,i-c,h} \underline{s}' \alpha_{i+j}$ contributing to $\psi^{(k)}_{n,i,j}(u,d)$. Finally, for the third scenario, by considering all possible values of the surplus process at time $k + n - 1$, we have $(1-p) \sum_{s^*=c_1}^{c_2} d_{s^*}\ \underline{g}^{(k)}_{n-1,i+s^*-c,h} \underline{s}' \alpha_{i+j}$ contributing to $\psi^{(k)}_{n,i,j}(u,d)$. Note that, unlike the previous sub-case, s^* has a minimum value of c_1, since $b+c-i$ is non-positive for $i \geq b + c$.

Hence, combining all of the above three scenarios, we get

$$\psi^{(k)}_{n,i,j}(u,d) = \left(\sum_{\ell=1}^{h-1} \underline{g}^{(k)}_{n-1,i-c,\ell} + p\underline{g}^{(k)}_{n-1,i-c,h} + (1-p) \sum_{s^*=c_1}^{c_2} d_{s^*}\ \underline{g}^{(k)}_{n-1,i+s^*-c,h} \right) \underline{s}' \alpha_{i+j}.$$

Therefore, combining all of the above sub-cases, we have the following expression:

$$\psi^{(k)}_{n,i,j}(u,d) = \begin{cases} 0 & \text{if } i < c; \\ \sum_{\ell=1}^{h} \underline{g}^{(k)}_{n-1,i-c,\ell} \underline{s}' \alpha_{i+j} & \text{if } c \leq i < b + c - c_2; \\ \left(\sum_{\ell=1}^{h} \underline{g}^{(k)}_{n-1,i-c,\ell} + (1-p) \sum_{s^*=\max\{c_1, b+c-i\}}^{c_2} d_{s^*}\ \underline{g}^{(k)}_{n-1,i+s^*-c,h} \right) \underline{s}' \alpha_{i+j} & \text{if } b + c - c_2 \leq i, \\ & \text{and } i < b + c; \\ \left(\sum_{\ell=1}^{h-1} \underline{g}^{(k)}_{n-1,i-c,\ell} + p\underline{g}^{(k)}_{n-1,i-c,h} + (1-p) \sum_{s^*=c_1}^{c_2} d_{s^*}\ \underline{g}^{(k)}_{n-1,i+s^*-c,h} \right) \underline{s}' \alpha_{i+j} & \text{if } i \geq b + c. \end{cases}$$

Case 4: $c > c_2$ and $c > b + (c - c_2)_+$

In this case, $\min\{c, b+(c-c_2)_+\} = b+c-c_2$. Now, we examine the conditional trivariate ruin probability, $\psi^{(k)}_{n,i,j}(u,d)$, under four non-overlapping, exhaustive sub-cases:

(1) $i < b + c - c_2$: This is an impossible scenario, and hence, $\psi^{(k)}_{n,i,j}(u,d) = 0$. Note that for this case, unlike the previous ones, the surplus prior to ruin is always greater than b at time $k+n$. This can be demonstrated by showing that the smallest value it can take is greater than b: suppose that at time $k+n-1$, the surplus process is at the lowest non-ruined value possible, which is level 0. Then, in the next time period (i.e, at time $k+n$), the surplus process will increase by a premium, and hence will be at level c. Now, suppose a dividend of maximum amount c_2 is issued immediately before the ruining claim.; then, the surplus prior to ruin will be equal to $b+c-c_2$. Since $c > c_2$, therefore the minimum surplus prior to ruin value is guaranteed to be greater than b.

(2) $b + c - c_2 \leq i < c$:

This sub-case is valid only if a dividend is issued immediately before the ruining claim. Hence, by considering all possible values of the surplus process at time $k+n-1$, we simply have

$$\psi^{(k)}_{n,i,j}(u,d) = (1-p) \sum_{s^*=\max\{c_1, b+c-i\}}^{c_2} d_{s^*}\ \underline{g}^{(k)}_{n-1,i+s^*-c,h}\underline{s}'\alpha_{i+j}.$$

The reason for having $\max\{c_1, b+c-i\}$ as the lower index of s^*, as opposed to just c_1, is that we need to insure that the surplus process at time $k+n-1$ is at or above level b (i.e., $i+s^*-c \geq b$); otherwise, a dividend at time $k+n$ would not be possible.

(3) $c \leq i < b + c$:

Here, we have two possible scenarios to consider at time $k + n$: one where no dividend is issued, and one where a dividend is issued immediately before the ruining claim. Note that the the first scenario is identical to the second sub-case in Case 3, and thus, we have $\sum_{\ell=1}^{h} \underline{g}^{(k)}_{n-1,i-c,\ell} \underline{s}' \alpha_{i+j}$ contributing to $\psi^{(k)}_{n,i,j}(u, d)$. The second scenario, on the other hand, is identical to the previous sub-case, and thus, we have $(1-p) \sum_{s^*=\max\{c_1, b+c-i\}}^{c_2} d_{s^*} \ \underline{g}^{(k)}_{n-1,i+s^*-c,h} \underline{s}' \alpha_{i+j}$ contributing to $\psi^{(k)}_{n,i,j}(u, d)$.

Hence, combining the above two scenarios, we get

$$\psi^{(k)}_{n,i,j}(u, d) = \left(\sum_{\ell=1}^{h} \underline{g}^{(k)}_{n-1,i-c,\ell} + (1-p) \sum_{s^*=\max\{c_1, b+c-i\}}^{c_2} d_{s^*} \ \underline{g}^{(k)}_{n-1,i+s^*-c,h} \right) \underline{s}' \alpha_{i+j}.$$

(4) $i \geq b + c$:

Here, we have three possible scenarios at to consider time $k + n$: one where no dividend is issued, one where a potential dividend would have been issued had it not been for the ruining claim, and one where a dividend is issued immediately before the ruining claim. For the first scenario, as explained in the fifth sub-case of Case 1, we have $\sum_{\ell=1}^{h-1} \underline{g}^{(k)}_{n-1,i-c,\ell} \underline{s}' \alpha_{i+j}$ contributing to $\psi^{(k)}_{n,i,j}(u, d)$. In the case of the second scenario, we simply have $p \underline{g}^{(k)}_{n-1,i-c,h} \underline{s}' \alpha_{i+j}$ contributing to $\psi^{(k)}_{n,i,j}(u, d)$. Finally, for the third scenario, by considering all possible values of the surplus process at time $k + n - 1$, in this scenario, we have $(1-p) \sum_{s^*=c_1}^{c_2} d_{s^*} \ \underline{g}^{(k)}_{n-1,i+s^*-c,h} \underline{s}' \alpha_{i+j}$ contributing to $\psi^{(k)}_{n,i,j}(u, d)$. Note that, unlike the previous sub-case, s^* has a minimum value of c_1, since $b + c - i$ is non-positive for $i \geq b + c$.

Hence, combining the above three scenarios, we get

$$\psi_{n,i,j}^{(k)}(u,d) = \left(\sum_{\ell=1}^{h-1} \underline{g}_{n-1,i-c,\ell}^{(k)} + p\underline{g}_{n-1,i-c,h}^{(k)} + (1-p) \sum_{s^*=c_1}^{c_2} d_{s^*} \ \underline{g}_{n-1,i+s^*-c,h}^{(k)} \right) \underline{s}' \alpha_{i+j}.$$

Therefore, combining all of the above sub-cases, we have the following expression:

$$\psi_{n,i,j}^{(k)}(u,d) = \begin{cases} 0 & \text{if } i < b+c-c_2; \\ (1-p) \sum\limits_{s^*=\max\{c_1, b+c-i\}}^{c_2} d_{s^*} \ \underline{g}_{n-1,b+s^*-c,h}^{(k)} \underline{s}' \alpha_{i+j} & \text{if } b+c-c_2 \leq i < c; \\ \left(\sum\limits_{\ell=1}^{h} \underline{g}_{n-1,i-c,\ell}^{(k)} + (1-p) \sum\limits_{s^*=\max\{c_1, b+c-i\}}^{c_2} d_{s^*} \ \underline{g}_{n-1,i+s^*-c,h}^{(k)} \right) \underline{s}' \alpha_{i+j} & \text{if } c \leq i < b+c; \\ \left(\sum\limits_{\ell=1}^{h-1} \underline{g}_{n-1,i-c,\ell}^{(k)} + p\underline{g}_{n-1,i-c,h}^{(k)} + (1-p) \sum\limits_{s^*=c_1}^{c_2} d_{s^*} \ \underline{g}_{n-1,i+s^*-c,h}^{(k)} \right) \underline{s}' \alpha_{i+j} & \text{if } i \geq b+c. \end{cases}$$

Once again, we comment that in the case of $p = 1$, the above expression for $\psi_{n,i,j}^{(k)}(u,d)$ correctly yields a value of 0 for all $i < c$, as the minimum surplus prior to ruin value is equal to c in this case.

3.6.2 Calculating $\omega_{n,i}^{(k)}(u,d)$

A straightforward way to calculate this conditional bivariate ruin probability is by summing $\psi_{n,i,j}^{(k)}(u,d)$ over all possible values of j. Since the maximum claim size is y_α and the surplus prior to ruin is set equal to i, the maximum value that the deficit at ruin, j, can therefore be is $y_\alpha - i$. Thus, we have

$$\omega_{n,i}^{(k)}(u,d) = \sum_{j=1}^{y_\alpha - i} \psi_{n,i,j}^{(k)}(u,d).$$

Looking at the results in the previous sub-section, and observing that the j's are only found in the claim size pmf, we end up with expressions for $\omega_{n,i}^{(k)}(u,d)$ identical to those for $\psi_{n,i,j}^{(k)}(u,d)$, with the exception that all the α_{i+j}'s are now replaced with Λ_i's (i.e., the claim size tail probability function). Specifically, we get the following results:

Case 1: $c \leq c_2$ **and** $c \leq b$

$$\omega_{n,i,j}^{(k)}(u,d) = \begin{cases} 0 & \text{if } i < c; \\ \sum_{\ell=1}^{h} \underline{g}_{n-1,i-c,\ell}^{(k)} \underline{s}' \Lambda_i & \text{if } c \leq i < b; \\ \left(\sum_{\ell=1}^{h} \underline{g}_{n-1,b-c,\ell}^{(k)} + (1-p) \sum_{s=c}^{c_2} \sum_{s^*=\max\{c_1,s\}}^{c_2} d_{s^*} \ \underline{g}_{n-1,b+s-c,h}^{(k)} \right) \underline{s}' \Lambda_b & \text{if } i = b; \\ \left(\sum_{\ell=1}^{h} \underline{g}_{n-1,i-c,\ell}^{(k)} + (1-p) \sum_{s^*=\max\{c_1,b+c-i\}}^{c_2} d_{s^*} \ \underline{g}_{n-1,i+s^*-c,h}^{(k)} \right) \underline{s}' \Lambda_i & \text{if } b < i < b+c; \\ \left(\sum_{\ell=1}^{h-1} \underline{g}_{n-1,i-c,\ell}^{(k)} + p \underline{g}_{n-1,i-c,h}^{(k)} + (1-p) \sum_{s^*=c_1}^{c_2} d_{s^*} \ \underline{g}_{n-1,i+s^*-c,h}^{(k)} \right) \underline{s}' \Lambda_i & \text{if } i \geq b+c. \end{cases}$$

Case 2: $c \leq c_2$ **and** $c > b$

$$\omega_{n,i}^{(k)}(u,d) = \begin{cases} 0 & \text{if } i < b; \\ (1-p) \sum_{s=c}^{c_2} \sum_{s^*=\max\{c_1,s\}}^{c_2} d_{s^*} \ \underline{g}_{n-1,b+s-c,h}^{(k)} \underline{s}' \Lambda_b & \text{if } i = b; \\ \left((1-p) \sum_{s^*=\max\{c_1,b+c-i\}}^{c_2} d_{s^*} \ \underline{g}_{n-1,i+s^*-c,h}^{(k)} \right) \underline{s}' \Lambda_i & \text{if } b < i < c; \\ \left(\sum_{\ell=1}^{h} \underline{g}_{n-1,i-c,\ell}^{(k)} + (1-p) \sum_{s^*=\max\{c_1,b+c-i\}}^{c_2} d_{s^*} \ \underline{g}_{n-1,i+s^*-c,h}^{(k)} \right) \underline{s}' \Lambda_i & \text{if } c \leq i < b+c; \\ \left(\sum_{\ell=1}^{h-1} \underline{g}_{n-1,i-c,\ell}^{(k)} + p \underline{g}_{n-1,i-c,h}^{(k)} + (1-p) \sum_{s^*=c_1}^{c_2} d_{s^*} \ \underline{g}_{n-1,i+s^*-c,h}^{(k)} \right) \underline{s}' \Lambda_i & \text{if } i \geq b+c. \end{cases}$$

Case 3: $c > c_2$ **and** $c \leq b + (c - c_2)_+$

$$\omega_{n,i}^{(k)}(u,d) = \begin{cases} 0 & \text{if } i < c; \\ \sum_{\ell=1}^{h} \underline{g}_{n-1,i-c,\ell}^{(k)} \underline{s}' \Lambda_i & \text{if } c \leq i < b + c - c_2; \\ \left(\sum_{\ell=1}^{h} \underline{g}_{n-1,i-c,\ell}^{(k)} + (1-p) \sum_{s^*=\max\{c_1, b+c-i\}}^{c_2} d_{s^*} \ \underline{g}_{n-1,i+s^*-c,h}^{(k)} \right) \underline{s}' \Lambda_i & \text{if } b + c - c_2 \leq i < b + c; \\ \left(\sum_{\ell=1}^{h-1} \underline{g}_{n-1,i-c,\ell}^{(k)} + p \underline{g}_{n-1,i-c,h}^{(k)} + (1-p) \sum_{s^*=c_1}^{c_2} d_{s^*} \ \underline{g}_{n-1,i+s^*-c,h}^{(k)} \right) \underline{s}' \Lambda_i & \text{if } i \geq b + c. \end{cases}$$

Case 4: $c > c_2$ **and** $c > b + (c - c_2)_+$

$$\omega_{n,i}^{(k)}(u,d) = \begin{cases} 0 & \text{if } i < b + c - c_2; \\ (1-p) \sum_{s^*=\max\{c_1, b+c-i\}}^{c_2} d_{s^*} \ \underline{g}_{n-1,b+s^*-c,h}^{(k)} \underline{s}' \Lambda_i & \text{if } b + c - c_2 \leq i < c; \\ \left(\sum_{\ell=1}^{h} \underline{g}_{n-1,i-c,\ell}^{(k)} + (1-p) \sum_{s^*=\max\{c_1, b+c-i\}}^{c_2} d_{s^*} \ \underline{g}_{n-1,i+s^*-c,h}^{(k)} \right) \underline{s}' \Lambda_i & \text{if } c \leq i < b + c; \\ \left(\sum_{\ell=1}^{h-1} \underline{g}_{n-1,i-c,\ell}^{(k)} + p \underline{g}_{n-1,i-c,h}^{(k)} + (1-p) \sum_{s^*=c_1}^{c_2} d_{s^*} \ \underline{g}_{n-1,i+s^*-c,h}^{(k)} \right) \underline{s}' \Lambda_i & \text{if } i \geq b + c. \end{cases}$$

3.6.3 Calculating $\phi_{n,j}^{(k)}(u,d)$

One method to calculate this conditional bivariate ruin probability is by summing $\psi_{n,i,j}^{(k)}(u,d)$ over all possible values of i. Since the maximum claim size is y_α and the deficit at ruin is set equal to j, the maximum value that the surplus prior to ruin, i, can therefore be is $y_\alpha - j$. Thus, we have

$$\phi_{n,j}^{(k)}(u,d) = \sum_{i=i_{min}}^{y_\alpha - j} \psi_{n,i,j}^{(k)}(u,d).$$

An alternative method to calculate $\phi_{n,j}^{(k)}(u,d)$ is to use the ruin probability vector $\underline{\tilde{h}}_{n,-j}^{(k)}$ directly. Knowing that the elapsed claim time counter, E_t, must be equal to 1 at time $t = k + n$ since a claim (i.e., the ruining one) occurs, we have

$$\phi_{n,j}^{(k)}(u,d) = \sum_{\ell=1}^{h} \underline{h}_{n,-j,\ell}^{(k)} \underline{e}_1'.$$

The next section shows the derivations of the unconditional versions of the above probabilities.

3.7 Determining the Unconditional Trivariate and Bivariate Ruin Probabilities

We are now in a position to obtain explicit expressions for $\psi_{\tau,i,j}(u,d)$, $\omega_{\tau,i}(u,d)$, and $\phi_{\tau,j}(u,d)$ from their conditional counterparts. The final results are shown on page 86, and the methodology is shown in the following sub-sections.

3.7.1 Calculating $\psi_{\tau,i,j}(u,d)$

Using the law of total probability, we have

$$\begin{aligned}
\psi_{\tau,i,j}(u,d) &= Pr\{T = \tau, U_{T^-} = i, |U_T| = j \mid U_0 = u, D_0 = d\} \\
&= \sum_{k=1}^{n_r} r_k Pr\{T = \tau, U_{T^-} = i, |U_T| = j \mid U_0 = u, D_0 = d, W_1 = k\} \\
&= \sum_{k=1}^{\min\{\tau, n_r\}} r_k Pr\{T = \tau, U_{T^-} = i, |U_T| = j \mid U_0 = u, D_0 = d, W_1 = k\}.
\end{aligned}$$

Now, if ruin occurs after n_r time units (i.e., $\tau > n_r$), then, by construction

$$\begin{aligned}\psi_{\tau,i,j}(u,d) &= \sum_{k=1}^{n_r} r_k Pr\{T=\tau, U_{T^-}=i, |U_T|=j \ | U_0=u, D_0=d, W_1=k\} \\ &= \sum_{k=1}^{n_r} r_k \psi^{(k)}_{\tau-k,i,j}(u,d).\end{aligned}$$

If, however, ruin occurs at or before time n_r (i.e., $\tau \leq n_r$), then either the first claim is the cause of ruin, or the first claim occurs at some time $k \in \{1,2,\ldots,\tau-1\}$ and ruin is subsequently experienced $\tau - k$ time units later. Mathematically, this amounts to

$$\begin{aligned}\psi_{\tau,i,j}(u,d) = &\sum_{k=1}^{\tau} r_k Pr\{T=\tau, U_{T^-}=i, |U_T|=j \ | U_0=u, D_0=d, W_1=k\} \\ = &\sum_{k=1}^{\tau-1} r_k Pr\{T=\tau, U_{T^-}=i, |U_T|=j \ | U_0=u, D_0=d, W_1=k\} \\ &+ r_\tau Pr\{T=\tau, U_{T^-}=i, |U_T|=j \ | U_0=u, D_0=d, W_1=\tau\} \\ = &\sum_{k=1}^{\tau-1} r_k \psi^{(k)}_{\tau-k,i,j}(u,d) + r_\tau Pr\{T=\tau, U_{T^-}=i, |U_T|=j \ | U_0=u, D_0=d, W_1=\tau\}.\end{aligned}$$

As done in Alfa and Drekic (2007), we can combine these two cases together to obtain

$$\psi_{\tau,i,j}(u,d) = \sum_{k=1}^{\min\{\tau-1,n_r\}} r_k \psi^{(k)}_{\tau-k,i,j}(u,d) + r_\tau Pr\{T=\tau, U_{T^-}=i, |U_T|=j \ | U_0=u, D_0=d, W_1=\tau\}.$$

The second component, $Pr\{T=\tau, U_{T^-}=i, |U_T|=j \ | U_0=u, D_0=d, W_1=\tau\}$, can be further assessed under the following two non-overlapping, exhaustive cases: the ruining first claim occurs during a non-dividend paying time period, and the ruining first claim occurs during a dividend paying time period. Under the first case, the above probability expression is equal to

$$\sum_{s\in\mathcal{R}_{u,d}(\tau-1)} f_{u,d}^{(|\mathcal{T}_{u,d}(\tau-1)|)}(s)\mathbf{1}_{[i=u+c\tau-s]}\alpha_{u+c\tau-s+j}.$$

Since the indicator function is equal to 1 only when $s = u+c\tau-i$, the previous probability expression simplifies to become

$$f_{u,d}^{(|\mathcal{T}_{u,d}(\tau-1)|)}(u+c\tau-i)\alpha_{i+j}.$$

Under the second case, by taking into account all possible dividends prior to and including time τ, the same probability expression is now given by

$$\sum_{s\in\mathcal{R}_{u,d}(\tau-1)} f_{u,d}^{(|\mathcal{T}_{u,d}(\tau-1)|)}(s)\Bigg\{p\mathbf{1}_{[i=u+c\tau-s]}\alpha_{u+c\tau-s+j}$$
$$+(1-p)\sum_{s^*=\min\{c_1,u+c\tau-s-b\}}^{\min\{c_2,u+c\tau-s-b\}}\mathbf{1}_{[i=u+c\tau-s-s^*]}d_{s^*}^*\{u+c\tau-s\}\ \alpha_{u+c\tau-s-s^*+j}\Bigg\}.$$

Since the indicator functions are equal to 1 only when $s = u+c\tau-i$ for the first one and $s^* = u+c\tau-s-i$ for the second one, the above probability expression simplifies to become

$$\left(pf_{u,d}^{(|\mathcal{T}_{u,d}(\tau-1)|)}(u+c\tau-i)+(1-p)\sum_{s\in\mathcal{R}_{u,d}(\tau-1)} f_{u,d}^{(|\mathcal{T}_{u,d}(\tau-1)|)}(s)d_{u+c\tau-s-i}^*\{u+c\tau-s\}\right)\alpha_{i+j}.$$

However, an identity exists in that $\sum_{s\in\mathcal{R}_{u,d}(\tau-1)} f_{u,d}^{(|\mathcal{T}_{u,d}(\tau-1)|)}(s)d_{u+c\tau-s-i}^*\{u+c\tau-s\}$ is actually equal to $f_{u,d}^{(|\mathcal{T}_{u,d}(\tau)|)}(u+c\tau-i)$. This is because evaluating dividend amounts that will result in the surplus process being at level i at time τ can be calculated by either accounting for all possible total dividend amounts up to and including time $\tau-1$, and then ensuring that the dividend amount at time τ is equal to the value that will bring the surplus process down to level i, or by simply accounting for all dividend values from time 1 to time τ that will yield a total amount such that the surplus process is brought down

from level $u + c\tau$ to level i. Thus, the above probability expression further simplifies to become

$$\left(pf_{u,d}^{(|\mathcal{T}_{u,d}(\tau-1)|)}(u + c\tau - i) + (1-p)f_{u,d}^{(|\mathcal{T}_{u,d}(\tau)|)}(u + c\tau - i)\right)\alpha_{i+j},$$

and the combination of the above two cases of $\psi_{\tau,i,j}(u,d)$ is shown on page 86.

3.7.2 Calculating $\omega_{\tau,i}(u,d)$

Note that

$$\omega_{\tau,i}(u,d) = \sum_{j=1}^{y_\alpha - i} \psi_{\tau,i,j}(u,d)$$

and

$$\omega_{n,i}^{(k)}(u,d) = \sum_{j=1}^{y_\alpha - i} \psi_{n,i,j}^{(k)}(u,d).$$

Using these two equations, we examine $\omega_{\tau,i}(u,d)$ under two non-overlapping, exhaustive cases: the ruining first claim occurs during a non-dividend paying time period, and the ruining first claim occurs during a dividend paying time period.

Under the first case, we have

$$\omega_{\tau,i,j}(u,d) = \sum_{j=1}^{y_\alpha - i} \psi_{\tau,i,j}(u,d)$$

$$= \sum_{j=1}^{y_\alpha - i} \left(\sum_{k=1}^{\min\{\tau-1,n_r\}} r_k \psi^{(k)}_{\tau-k,i,j}(u,d) + r_\tau f_{u,d}^{\left(\left|\mathcal{T}_{u,d}(\min\{\tau-1,n_r\})\right|\right)}(u+c\tau-i)\alpha_{i+j} \right)$$

$$= \sum_{k=1}^{\min\{\tau-1,n_r\}} r_k \sum_{j=1}^{y_\alpha - i} \psi^{(k)}_{\tau-k,i,j}(u,d) + r_\tau f_{u,d}^{\left(\left|\mathcal{T}_{u,d}(\min\{\tau-1,n_r\})\right|\right)}(u+c\tau-i) \sum_{j=1}^{y_\alpha - i} \alpha_{i+j}$$

$$= \sum_{k=1}^{\min\{\tau-1,n_r\}} r_k \omega^{(k)}_{\tau-k,i}(u,d) + r_\tau f_{u,d}^{\left(\left|\mathcal{T}_{u,d}(\min\{\tau-1,n_r\})\right|\right)}(u+c\tau-i)\Lambda_i.$$

Note that since all summands are non-negative, there is no issue in switching the order of summations, by invoking Tonelli's theorem (e.g., see Billingsley (1995), Theorem 18.3).

Under the second case, by evaluating all possible dividends prior to and including time τ, we have

$$\omega_{\tau,i,j}(u,d) = \sum_{j=1}^{y_\alpha - i} \psi_{\tau,i,j}(u,d)$$

$$= \sum_{j=1}^{y_\alpha - i} \left\{ \sum_{k=1}^{\min\{\tau-1,n_r\}} r_k \psi^{(k)}_{\tau-k,i,j}(u,d) + r_\tau \left(p f_{u,d}^{\left(\left|\mathcal{T}_{u,d}(\tau-1)\right|\right)}(u+c\tau-i) + (1-p) f_{u,d}^{\left(\left|\mathcal{T}_{u,d}(\tau)\right|\right)}(u+c\tau-i) \right) \alpha_{i+j} \right\}$$

$$= \sum_{k=1}^{\min\{\tau-1,n_r\}} r_k \sum_{j=1}^{y_\alpha - i} \psi^{(k)}_{\tau-k,i,j}(u,d) + r_\tau \left(p f_{u,d}^{\left(\left|\mathcal{T}_{u,d}(\tau-1)\right|\right)}(u+c\tau-i) + (1-p) f_{u,d}^{\left(\left|\mathcal{T}_{u,d}(\tau)\right|\right)}(u+c\tau-i) \right) \sum_{j=1}^{y_\alpha - i} \alpha_{i+j}$$

$$= \sum_{k=1}^{\min\{\tau-1,n_r\}} r_k \omega^{(k)}_{\tau-k,i}(u,d) + r_\tau \left(p f_{u,d}^{\left(\left|\mathcal{T}_{u,d}(\tau-1)\right|\right)}(u+c\tau-i) + (1-p) f_{u,d}^{\left(\left|\mathcal{T}_{u,d}(\tau)\right|\right)}(u+c\tau-i) \right) \Lambda_i.$$

As in the above case, all summands are again non-negative, and so, there is no issue in switching the order of summations, by invoking Tonelli's theorem. The combination of the above two cases of $\omega_{\tau,i}(u,d)$ is shown on page 86.

3.7.3 Calculating $\phi_{\tau,j}(u,d)$

Note that

$$\phi_{\tau,i}(u,d) = \sum_{i=i_{min}}^{\min\{y_\alpha - j, \tilde{v}_u(\tau)\}} \psi_{\tau,i,j}(u,d)$$

and

$$\phi^{(k)}_{n,i}(u,d) = \sum_{i=i_{min}}^{\min\{y_\alpha - j, \tilde{v}_u(n)\}} \psi^{(k)}_{n,i,j}(u,d).$$

Using these two equations, we examine $\phi_{\tau,j}(u,d)$ under two non-overlapping, exhaustive cases: the ruining first claim occurs during a non-dividend paying time period, and the ruining first claim occurs during a dividend paying time period.

Under the first case, we have

$$\begin{aligned}
\phi_{\tau,j}(u,d) &= \sum_{i=i_{min}}^{\min\{y_\alpha - j, \tilde{v}_u(\tau)\}} \psi_{\tau,i,j}(u,d) \\
&= \sum_{i=i_{min}}^{\min\{y_\alpha - j, \tilde{v}_u(\tau)\}} \left(\sum_{k=1}^{\min\{\tau-1, n_r\}} r_k \psi^{(k)}_{\tau-k,i,j}(u,d) + r_\tau f_{u,d}^{(|\mathcal{T}_{u,d}(\tau-1)|)}(u + c\tau - i)\alpha_{i+j} \right) \\
&= \sum_{k=1}^{\min\{\tau-1, n_r\}} r_k \sum_{i=i_{min}}^{\min\{y_\alpha - j, \tilde{v}_u(\tau)\}} \psi^{(k)}_{\tau-k,i,j}(u,d) + r_\tau \sum_{i=i_{min}}^{\min\{y_\alpha - j, \tilde{v}_u(\tau)\}} f_{u,d}^{(|\mathcal{T}_{u,d}(\tau-1)|)}(u + c\tau - i)\alpha_{i+j} \\
&= \sum_{k=1}^{\min\{\tau-1, n_r\}} r_k \phi^{(k)}_{\tau-k,j}(u,d) + r_\tau \sum_{i=i_{min}}^{\min\{y_\alpha - j, \tilde{v}_u(\tau)\}} f_{u,d}^{(|\mathcal{T}_{u,d}(\tau-1)|)}(u + c\tau - i)\alpha_{i+j}.
\end{aligned}$$

Note that since all summands are non-negative, there is no issue in switching the order of summations, by again invoking Tonelli's theorem.

Under the second case, by evaluating all possible dividends prior to and including time τ, we have

$$\begin{aligned}
\phi_{\tau,j}(u,d) &= \sum_{i=i_{min}}^{\min\{y_\alpha - j, \tilde{v}_u(\tau)\}} \psi_{\tau,i,j}(u,d) \\
&= \sum_{i=i_{min}}^{\min\{y_\alpha - j, \tilde{v}_u(\tau)\}} \left\{ \sum_{k=1}^{\min\{\tau-1, n_r\}} r_k \psi^{(k)}_{\tau-k,i,j}(u,d) + r_\tau \left(p f_{u,d}^{(|\mathcal{T}_{u,d}(\tau-1)|)}(u + c\tau - i) \right.\right. \\
&\qquad \left.\left. + (1-p) f_{u,d}^{(|\mathcal{T}_{u,d}(\tau)|)}(u + c\tau - i) \right) \alpha_{i+j} \right\} \\
&= \sum_{k=1}^{\min\{\tau-1, n_r\}} r_k \sum_{i=i_{min}}^{\min\{y_\alpha - j, \tilde{v}_u(\tau)\}} \psi^{(k)}_{\tau-k,i,j}(u,d) \\
&\quad + r_\tau \sum_{i=i_{min}}^{\min\{y_\alpha - j, \tilde{v}_u(\tau)\}} \left(p f_{u,d}^{(|\mathcal{T}_{u,d}(\tau-1)|)}(u + c\tau - i) + (1-p) f_{u,d}^{(|\mathcal{T}_{u,d}(\tau)|)}(u + c\tau - i) \right) \alpha_{i+j} \\
&= \sum_{k=1}^{\min\{\tau-1, n_r\}} r_k \phi^{(k)}_{\tau-k,i,j}(u,d) \\
&\quad + r_\tau \sum_{i=i_{min}}^{\min\{y_\alpha - j, \tilde{v}_u(\tau)\}} \left(p f_{u,d}^{(|\mathcal{T}_{u,d}(\tau-1)|)}(u + c\tau - i) + (1-p) f_{u,d}^{(|\mathcal{T}_{u,d}(\tau)|)}(u + c\tau - i) \right) \alpha_{i+j}.
\end{aligned}$$

As in the above case, all summands are again non-negative, and so, there is no issue in switching the order of summations, by invoking Tonelli's theorem. The next page summarizes the derivations of the unconditional trivariate and bivariate ruin probabilities.

This concludes our algorithmic derivations using MAMs. Now, we turn our attention to deriving the finite-ruin time based Gerber-Shiu function and the moments of the total dividends paid by a finite time horizon or before ruin occurs, whichever happens first, using the first claim conditioning approach.

$$\psi_{\tau,i,j}(u,d) = \begin{cases} \sum_{k=1}^{\min\{\tau-1,n_r\}} r_k \psi^{(k)}_{\tau-k,i,j}(u,d) + r_\tau f_{u,d}^{\left(|\mathcal{T}_{u,d}(\tau-1)|\right)}(u+c\tau-i)\alpha_{i+j} & \text{if } \tau \notin \mathcal{T}_{u,d}(\tau); \\ \sum_{k=1}^{\min\{\tau-1,n_r\}} r_k \psi^{(k)}_{\tau-k,i,j}(u,d) + r_\tau \left(p f_{u,d}^{\left(|\mathcal{T}_{u,d}(\tau-1)|\right)}(u+c\tau-i) + \ (1-p) f_{u,d}^{\left(|\mathcal{T}_{u,d}(\tau)|\right)}(u+c\tau-i) \right) \alpha_{i+j} & \text{if } \tau \in \mathcal{T}_{u,d}(\tau). \end{cases}$$

$$\omega_{\tau,i}(u,d) = \begin{cases} \sum_{k=1}^{\min\{\tau-1,n_r\}} r_k \omega^{(k)}_{\tau-k,i}(u,d) + r_\tau f_{u,d}^{\left(|\mathcal{T}_{u,d}(\tau-1)|\right)}(u+c\tau-i)\Lambda_i & \text{if } \tau \notin \mathcal{T}_{u,d}(\tau); \\ \sum_{k=1}^{\min\{\tau-1,n_r\}} r_k \omega^{(k)}_{\tau-k,i}(u,d) + r_\tau \left(p f_{u,d}^{\left(|\mathcal{T}_{u,d}(\tau-1)|\right)}(u+c\tau-i) + \ (1-p) f_{u,d}^{\left(|\mathcal{T}_{u,d}(\tau)|\right)}(u+c\tau-i) \right) \Lambda_i & \text{if } \tau \in \mathcal{T}_{u,d}(\tau). \end{cases}$$

$$\phi_{\tau,j}(u,d) = \begin{cases} \sum_{k=1}^{\min\{\tau-1,n_r\}} r_k \phi^{(k)}_{\tau-k,j}(u,d) + r_\tau \sum_{i=i_{min}}^{\min\{y_\alpha-j,\tilde{v}_u(\tau)\}} f_{u,d}^{\left(|\mathcal{T}_{u,d}(\tau-1)|\right)}(u+c\tau-i)\alpha_{i+j} & \text{if } \tau \notin \mathcal{T}_{u,d}(\tau); \\ \sum_{k=1}^{\min\{\tau-1,n_r\}} r_k \phi^{(k)}_{\tau-k,j}(u,d) + r_\tau \sum_{i=i_{min}}^{\min\{y_\alpha-j,\tilde{v}_u(\tau)\}} \left(p f_{u,d}^{\left(|\mathcal{T}_{u,d}(\tau-1)|\right)}(u+c\tau-i) + (1-p) f_{u,d}^{\left(|\mathcal{T}_{u,d}(\tau)|\right)}(u+c\tau-i) \right) \alpha_{i+j} & \text{if } \tau \in \mathcal{T}_{u,d}(\tau). \end{cases}$$

Chapter 4

Algorithmic Expressions for Ruin-related Quantities Using the First Claim Conditioning Method

4.1 Introduction

One of the main advantages of assuming that interclaim times (not counting the time until the first claim, which is potentially different from the others) are iid is that the claim instants form a sequence of renewal points for the underlying stochastic process of interest. That is, at a claim-occurring time unit, say time t, we can reset t to 0, and effectively start the process anew. As a result, since interclaim times are iid beyond the first claim, DISAMs eventually behave like OISAMs. For this reason, we consider the DISAM class and condition on the time until the first claim occurs and obtain expressions for the following ruin-related quantities of interest, namely:

$$\widetilde{G}_{v,m}(u,d) = E[v^T w(U_{T^-}, |U_T|)\mathbf{1}_{[T\leq m]}|U_0 = u, D_0 = d]$$

and

$$\widetilde{\mathcal{D}}^r_m(u,d) = E[S^r_{\min\{T,m\}}|U_0 = u, D_0 = d].$$

For these two quantities of interest, note that we constrain the analysis to a finite time horizon, $m \in \mathbb{Z}^+$. That is, the surplus process is analyzed only up to time m. We, of course, still conform to the assumption that the time of ruin, T, is a stopping time of the process, implying that we do not examine the process beyond the time of ruin, even if ruin occurs before time m. This will be evident in the derivation of the two ruin-related quantities of interest. As indicated in Chapter 2, we can relax the finite time horizon constraint under certain conditions to the more general form, where the only constraint is $T < \infty$.

In addition to conditioning on $W_1 = k$, we also need to account for other factors that will affect the surplus level of the process. Specifically, we will need to account for the size of the first claim, as well as all possible (restricted) dividend sums before time k, whether or not time k is a dividend-issuing time unit, and, if so, whether or not the claim is processed before the dividend. This can be done systematically using the law of total probability; and so, our recursive formulas will involve multiple summations over the possible variables that can influence the value of the surplus process.

We remark that a very useful feature of employing the first claim conditioning approach, unlike the situation with MAMs in the previous chapter, is that the distributions of the time until the first claim occurs and the ordinary interclaim times need not have finite maximum support values (i.e., n_r and n_a can be infinite). We now proceed to applying this method on the remaining two aforementioned performance measures of interest.

4.2 Determining the Finite-ruin Time Based Gerber-Shiu Function

With $\widetilde{G}_{v,m}(u,d)$ serving as the finite-ruin time based Gerber-Shiu function for a DISAM, we also need to introduce $G_{v,m}(u,d)$ as its OISAM counterpart. By conditioning on the time of the first claim (i.e., $W_1 = k$), we can split $\widetilde{G}_{v,m}(u,d)$ into five non-overlapping, exhaustive cases, all of which will be combined together into one final expression at the end:

(1) **Ruining first claim occurs, during a non-dividend paying time period, by time horizon m:**

In this case, there is a total of $|\mathcal{T}_{u,d}(k)|$ dividends, which are issued prior to the ruining first claim. If we assume that the sum of these (restricted) dividends is equal to a value s, then the surplus process will be at level $u + ck - s$ immediately before the ruining claim is implemented, which itself must range in value from $u+ck-s+1$ to y_α. Thus, using the law of total probability, we readily obtain

$$\sum_{k \notin \mathcal{T}_{u,d}(\min\{n_r,m\})} v^k \sum_{s \in \mathcal{R}_{u,d}(k)} f_{u,d}^{(|\mathcal{T}_{u,d}(k)|)}(s) \left\{ \sum_{y=u+ck-s+1}^{y_\alpha} \alpha_y w(u+ck-s, |u+ck-s-y|) \right\} r_k.$$

(2) **Ruining first claim occurs, during a dividend paying time period, by time horizon m:**

Before time k, $|\mathcal{T}_{u,d}(k-1)|$ dividends in total are issued. If we assume that the sum of these (restricted) dividends is equal to a value s, then the surplus process will be at level $u + ck - s$ immediately before the claim or dividend is first implemented. There are two sub-cases to consider at this point: either the claim is implemented

before the dividend, or vice-versa. In the first sub-case, no dividend at time k ends up being issued. Therefore, the size of the ruining claim, y, will range in value from $u+ck-s+1$ to y_α. In the second sub-case, where the dividend gets implemented first, the restricted dividend amount will range in value from $\min\{c_1, u+ck-s-b\}$ to $\min\{c_2, u+ck-s-b\}$. Hence, if we assume that the restricted dividend amount at time k is equal to a value s^*, then the size of the subsequent ruining claim, y, must range in value from $u+ck-s-s^*+1$ to y_α. Incorporating these elements via the law of total probability, we obtain

$$\sum_{k\in\mathcal{T}_{u,d}(\min\{n_r,m\})} v^k \sum_{s\in\mathcal{R}_{u,d}(k-1)} f_{u,d}^{(|\mathcal{T}_{u,d}(k-1)|)}(s) \left\{ p \sum_{y=u+ck-s+1}^{y_\alpha} \alpha_y w(u+ck-s, |u+ck-s-y|) \right.$$

$$\left. +(1-p) \sum_{s^*=\min\{c_1,u+ck-s-b\}}^{\min\{c_2,u+ck-s-b\}} d_{s^*}^*\{u+ck-s\} \sum_{y=u+ck-s-s^*+1}^{y_\alpha} \alpha_y w(u+ck-s-s^*, |u+ck-s-s^*-y|) \right\} r_k.$$

However, due to the identity on page 81, it follows that

$$\sum_{s\in\mathcal{R}_{u,d}(k-1)} f_{u,d}^{(|\mathcal{T}_{u,d}(k-1)|)}(s) \sum_{s^*=\min\{c_1,u+ck-s-b\}}^{\min\{c_2,u+ck-s-b\}} d_{s^*}^*\{u+ck-s\} = \sum_{s\in\mathcal{R}_{u,d}(k)} f_{u,d}^{(|\mathcal{T}_{u,d}(k)|)}(s),$$

implying that the above expression becomes

$$\sum_{k\in\mathcal{T}_{u,d}(\min\{n_r,m\})} v^k \left\{ p \sum_{s\in\mathcal{R}_{u,d}(k-1)} f_{u,d}^{(|\mathcal{T}_{u,d}(k-1)|)}(s) \sum_{y=u+ck-s+1}^{y_\alpha} \alpha_y w(u+ck-s, |u+ck-s-y|) \right.$$

$$\left. +(1-p) \sum_{s\in\mathcal{R}_{u,d}(k)} f_{u,d}^{(|\mathcal{T}_{u,d}(k)|)}(s) \sum_{y=u+ck-s+1}^{y_\alpha} \alpha_y w(u+ck-s, |u+ck-s-y|) \right\} r_k.$$

(3) **Non-ruining first claim occurs, during a non-dividend paying time period, by time horizon m:**

In this case, there is a total of $|\mathcal{T}_{u,d}(k)|$ dividends, which are issued prior to the non-ruining first claim. If we assume that the sum of these (restricted) dividends is equal to a value s, then the surplus process will be at level $u + ck - s$ immediately before the claim is implemented, and its size, y, will range in value from y_0 to $u + ck - s$. Therefore, since ruin does not occur at this claim instant, we can view the surplus process as "renewed" at time k, with corresponding new time horizon $m - k$, new initial surplus value equal to $u + ck - s - y$, and new initial dividend counter value equal to $V_{u,d}(0, k, y, s, ., .)$. In addition, following this first claim, the behaviour of our DISAM now reverts to that of an OISAM. Thus, using the law of total probability, we end up with

$$\sum_{k \notin \mathcal{T}_{u,d}(\min\{n_r, m\})} v^k \sum_{s \in \mathcal{R}_{u,d}(k)} f_{u,d}^{(|\mathcal{T}_{u,d}(k)|)}(s) \left\{ \sum_{y=y_0}^{u+ck-s} \alpha_y G_{v,m-k}\left(u + ck - s - y, V_{u,d}(0, k, y, s, ., .)\right) \right\} r_k.$$

(4) **Non-ruining first claim occurs, during a dividend paying time period, before time horizon m:**

Before time k, $|\mathcal{T}_{u,d}(k-1)|$ dividends in total are issued. If we assume that the sum of these (restricted) dividends is equal to a value s, then the surplus process will be at level $u + ck - s$ immediately before the claim or dividend is first implemented. Once again, there are two sub-cases to consider at this point: either the claim is implemented before the dividend, or vice-versa. In the first sub-case, the non-ruining claim size, y, will range in value from y_0 to $u + ck - s$, and so, the restricted dividend amount will range in value from $\min\{c_1, (u + ck - s - y - b)_+\}$ to $\min\{c_2, (u + ck - s - y - b)_+\}$. Note that the $(.)_+$ function is employed to account for the possibility

that the claim size, y, is large enough to bring the surplus process below level b, thereby resulting in a restricted dividend amount equal to 0. In the second sub-case, the restricted dividend amount will range in value from $\min\{c_1, u+ck-s-b\}$ to $\min\{c_2, u+ck-s-b\}$. Letting s^* denote the restricted dividend amount at time k, the non-ruining claim size, y, must therefore range in value from y_0 to $u+ck-s-s^*$. In either sub-case, ruin did not occur, and similar to the previous case, the surplus process can be viewed as "renewed" at time k, with corresponding new time horizon $m-k$, new initial surplus value equal to $u+ck-s-s^*-y$, and new initial dividend counter value equal to $V_{u,d}(0,k,y,s,s^*,Order)$, where $Order \in \{ClmFrst, \overline{ClmFrst}\}$. Moreover, following this first claim, the behaviour of our DISAM again reverts to that of an OISAM. Thus, using the law of total probability, we obtain

$$\sum_{k\in\mathcal{T}_{u,d}(\min\{n_r,m\})} v^k \sum_{s\in\mathcal{R}_{u,d}(k-1)} f_{u,d}^{\left(|\mathcal{T}_{u,d}(k-1)|\right)}(s)$$

$$\times \left\{ p \sum_{y=y_0}^{u+ck-s} \alpha_y \sum_{s^*=\min\{c_1,(u+ck-s-y-b)_+\}}^{\min\{c_2,(u+ck-s-y-b)_+\}} d^*_{s^*}\{u+ck-s-y\} G_{v,m-k}\left(u+ck-s-s^*-y, V_{u,d}(0,k,y,s,s^*,ClmFrst)\right) \right.$$

$$\left. + (1-p) \sum_{s^*=\min\{c_1,u+ck-s-b\}}^{\min\{c_2,u+ck-s-b\}} d^*_{s^*}\{u+ck-s\} \sum_{y=y_0}^{u+ck-s-s^*} \alpha_y G_{v,m-k}\left(u+ck-s-s^*-y, V_{u,d}(0,k,y,s,s^*,\overline{ClmFrst})\right) \right\} r_k.$$

(5) **First claim occurs beyond time horizon m (i.e., $k > m$):**

The conditional finite-ruin time based Gerber-Shiu function is clearly equal to 0 since $\mathbf{1}_{[T\le m]} = 0$ in this case.

In cases (3) and (4), the OISAM counterpart $G_{v,m}(u,d)$ appears in the derived formulas. However, we note that we can apply the same methodology on $G_{v,m}(u,d)$ except that quantities n_r and r_k are replaced by n_a and a_k, respectively. The following page combines all the above cases together into a single formula.

$$\widetilde{G}_{v,m}(u,d) = \sum_{k\notin\mathcal{T}_{u,d}(\min\{n_r,m\})} v^k \sum_{s\in\mathcal{R}_{u,d}(k)} f_{u,d}^{\left(\left|\mathcal{T}_{u,d}(k)\right|\right)}(s)\left\{\sum_{y=y_0}^{u+ck-s}\alpha_y G_{v,m-k}(u+ck-s-y,V_{u,d}(0,k,y,s,.,.)) + \sum_{y=u+ck-s+1}^{y_\alpha}\alpha_y w(u+ck-s,|u+ck-s-y|)\right\} r_k$$

$$+\ p\sum_{k\in\mathcal{T}_{u,d}(\min\{n_r,m\})} v^k \sum_{s\in\mathcal{R}_{u,d}(k-1)} f_{u,d}^{\left(\left|\mathcal{T}_{u,d}(k-1)\right|\right)}(s)\left\{\sum_{y=y_0}^{u+ck-s}\alpha_y \sum_{s^*=\min\{c_1,(u+ck-s-y-b)_+\}}^{\min\{c_2,(u+ck-s-y-b)_+\}} d_{s^*}^*\{u+ck-s-y\}G_{v,m-k}\left(u+ck-s-s^*-y,V_{u,d}(0,k,y,s,s^*,ClmFrst)\right)\right.$$
$$\left. + \sum_{y=u+ck-s+1}^{y_\alpha}\alpha_y w(u+ck-s,|u+ck-s-y|)\right\} r_k$$

$$+\ (1-p)\sum_{k\in\mathcal{T}_{u,d}(\min\{n_r,m\})} v^k \left\{\sum_{s\in\mathcal{R}_{u,d}(k-1)} f_{u,d}^{\left(\left|\mathcal{T}_{u,d}(k-1)\right|\right)}(s) \sum_{s^*=\min\{c_1,u+ck-s-b\}}^{\min\{c_2,u+ck-s-b\}} d_{s^*}^*\{u+ck-s\}\sum_{y=y_0}^{u+ck-s-s^*}\alpha_y G_{v,m-k}\left(u+ck-s-s^*-y,V_{u,d}(0,k,y,s,s^*,\overline{ClmFrst})\right)\right.$$
$$\left. + \sum_{s\in\mathcal{R}_{u,d}(k)} f_{u,d}^{\left(\left|\mathcal{T}_{u,d}(k)\right|\right)}(s)\sum_{y=u+ck-s+1}^{y_\alpha}\alpha_y w(u+ck-s,|u+ck-s-y|)\right\} r_k,$$

where

$$G_{v,m}(u,d) = \sum_{k\notin\mathcal{T}_{u,d}(\min\{n_a,m\})} v^k \sum_{s\in\mathcal{R}_{u,d}(k)} f_{u,d}^{\left(\left|\mathcal{T}_{u,d}(k)\right|\right)}(s)\left\{\sum_{y=y_0}^{u+ck-s}\alpha_y G_{v,m-k}(u+ck-s-y,V_{u,d}(0,k,y,s,.,.)) + \sum_{y=u+ck-s+1}^{y_\alpha}\alpha_y w(u+ck-s,|u+ck-s-y|)\right\} a_k$$

$$+\ p\sum_{k\in\mathcal{T}_{u,d}(\min\{n_a,m\})} v^k \sum_{s\in\mathcal{R}_{u,d}(k-1)} f_{u,d}^{\left(\left|\mathcal{T}_{u,d}(k-1)\right|\right)}(s)\left\{\sum_{y=y_0}^{u+ck-s}\alpha_y \sum_{s^*=\min\{c_1,(u+ck-s-y-b)_+\}}^{\min\{c_2,(u+ck-s-y-b)_+\}} d_{s^*}^*\{u+ck-s-y\}G_{v,m-k}\left(u+ck-s-s^*-y,V_{u,d}(0,k,y,s,s^*,ClmFrst)\right)\right.$$
$$\left. + \sum_{y=u+ck-s+1}^{y_\alpha}\alpha_y w(u+ck-s,|u+ck-s-y|)\right\} a_k$$

$$+\ (1-p)\sum_{k\in\mathcal{T}_{u,d}(\min\{n_a,m\})} v^k \left\{\sum_{s\in\mathcal{R}_{u,d}(k-1)} f_{u,d}^{\left(\left|\mathcal{T}_{u,d}(k-1)\right|\right)}(s) \sum_{s^*=\min\{c_1,u+ck-s-b\}}^{\min\{c_2,u+ck-s-b\}} d_{s^*}^*\{u+ck-s\}\sum_{y=y_0}^{u+ck-s-s^*}\alpha_y G_{v,m-k}\left(u+ck-s-s^*-y,V_{u,d}(0,k,y,s,s^*,\overline{ClmFrst})\right)\right.$$
$$\left. + \sum_{s\in\mathcal{R}_{u,d}(k)} f_{u,d}^{\left(\left|\mathcal{T}_{u,d}(k)\right|\right)}(s)\sum_{y=u+ck-s+1}^{y_\alpha}\alpha_y w(u+ck-s,|u+ck-s-y|)\right\} a_k.$$

4.3 Determining the Moments of the Total Dividends before a Finite Time Horizon

With $\widetilde{\mathcal{D}}_m^r(u,d)$ serving as the rth moment of the total dividends issued by a finite time horizon of $m \in \mathbb{Z}^+$ or before ruin occurs, whichever happens first, for a DISAM, we similarly need to introduce $\mathcal{D}_m^r(u,d)$ to be its OISAM counterpart. By once again conditioning on the time of the first claim (i.e., $W_1 = k$), we can split $\widetilde{\mathcal{D}}_m^r(u,d)$ into five non-overlapping, exhaustive cases, all of which will be combined together into one final expression at the end:

(1) **Ruining first claim occurs, during a non-dividend paying time period, by time horizon m:**

In this case, there is a total of $|\mathcal{T}_{u,d}(k)|$ dividends, which are issued prior to the ruining first claim. If we assume that the sum of these (restricted) dividends is equal to a value s, then the surplus process will be at level $u + ck - s$ immediately before the claim is implemented, which itself must range in value from $u + ck - s + 1$ to y_α. Thus, using the law of total probability, we readily obtain

$$\sum_{k \notin \mathcal{T}_{u,d}(\min\{n_r, m\})} \sum_{s \in \mathcal{R}_{u,d}(k)} f_{u,d}^{(|\mathcal{T}_{u,d}(k)|)}(s) s^r \Lambda_{u+ck-s} \; r_k.$$

(2) **Ruining first claim occurs, during a dividend paying time period, by time horizon m:**

Before time k, $|\mathcal{T}_{u,d}(k-1)|$ dividends in total are issued. If we assume that the sum of these (restricted) dividends is equal to a value s, then the surplus process will be at level $u + ck - s$ immediately before the claim or dividend is first implemented.

There are two sub-cases to consider at this point: either the claim is implemented before the dividend, or vice-versa. In the first sub-case, no dividend at time k ends up being issued. Therefore, the size of the ruining claim, y, will range in value from $u + ck - s + 1$ to y_α. In the second sub-case, where the dividend gets implemented first, the restricted dividend amount will range in value from $\min\{c_1, u + ck - s - b\}$ to $\min\{c_2, u + ck - s - b\}$. Hence, if we assume that the restricted dividend amount at time k is equal to a value s^*, then the size of the subsequent ruining claim, y, must range in value from $u + ck - s - s^* + 1$ to y_α. Incorporating these elements via the law of total probability, we obtain

$$\sum_{k \in \mathcal{T}_{u,d}(\min\{n_r, m\})} \sum_{s \in \mathcal{R}_{u,d}(k-1)} f_{u,d}^{(|\mathcal{T}_{u,d}(k-1)|)}(s) \left\{ p s^r \Lambda_{u+ck-s} \right.$$

$$\left. + (1-p) \sum_{s^* = \min\{c_1, u+ck-s-b\}}^{\min\{c_2, u+ck-s-b\}} d_{s^*}^*\{u + ck - s\} (s + s^*)^r \Lambda_{u+ck-s-s^*} \right\} r_k.$$

However, as stated in the previous section, we again employ the identity

$$\sum_{s \in \mathcal{R}_{u,d}(k-1)} f_{u,d}^{(|\mathcal{T}_{u,d}(k-1)|)}(s) \sum_{s^* = \min\{c_1, u+ck-s-b\}}^{\min\{c_2, u+ck-s-b\}} d_{s^*}^*\{u + ck - s\} = \sum_{s \in \mathcal{R}_{u,d}(k)} f_{u,d}^{(|\mathcal{T}_{u,d}(k)|)}(s)$$

to the above expression and ultimately obtain

$$\sum_{k \in \mathcal{T}_{u,d}(\min\{n_r, m\})} \left\{ p \sum_{s \in \mathcal{R}_{u,d}(k-1)} f_{u,d}^{(|\mathcal{T}_{u,d}(k-1)|)}(s) s^r \Lambda_{u+ck-s} + (1-p) \sum_{s \in \mathcal{R}_{u,d}(k)} f_{u,d}^{(|\mathcal{T}_{u,d}(k)|)}(s) s^r \Lambda_{u+ck-s} \right\} r_k.$$

(3) **Non-ruining first claim occurs, during a non-dividend paying time period, by time horizon m:**

In this case, there is a total of $|\mathcal{T}_{u,d}(k)|$ dividends, which are issued prior to the non-ruining first claim. If we assume that the sum of these (restricted) dividends is equal to a value s, then the surplus process will be at level $u + ck - s$ immediately before the claim is implemented, and its size, y, will range in value from y_0 to $u + ck - s$. Therefore, since ruin does not occur at this claim instant, we can view the surplus process as "renewed" at time k, with corresponding new time horizon $m - k$, new initial surplus value equal to $u + ck - s - y$, and new initial dividend counter value equal to $V_{u,d}(0, k, y, s, ., .)$. In addition, following this first claim, the behaviour of our DISAM now reverts to that of an OISAM. Thus, using the law of total probability, we end up with

$$\sum_{k \notin \mathcal{T}_{u,d}(\min\{n_r, m\})} \sum_{s \in \mathcal{R}_{u,d}(k)} f_{u,d}^{(|\mathcal{T}_{u,d}(k)|)}(s) \left\{ \sum_{y=y_0}^{u+ck-s} \alpha_y \right.$$

$$\left. \times E\left[\left\{ s + S_{\min\{T, m-k\}} \right\}^r | U_0 = u + ck - s - y, D_0 = V_{u,d}(0, k, y, s, ., .) \right] \right\} r_k.$$

Using the binomial theorem on the above expectation function (as well as the convention that $0^0 = 1$), we obtain

$$\sum_{k \notin \mathcal{T}_{u,d}(\min\{n_r, m\})} \sum_{s \in \mathcal{R}_{u,d}(k)} f_{u,d}^{(|\mathcal{T}_{u,d}(k)|)}(s) \sum_{y=y_0}^{u+ck-s} \alpha_y \sum_{\omega=0}^{r} \binom{r}{\omega} s^{\omega} \mathcal{D}_{m-k}^{r-\omega}(u + ck - s - y, V_{u,d}(0, k, y, s, ., .)) r_k.$$

(4) **Non-ruining first claim occurs, during a dividend paying time period, before time horizon m:**

Before time k, $|\mathcal{T}_{u,d}(k-1)|$ dividends in total are issued. If we assume that the sum of these (restricted) dividends is equal to a value s, then the surplus process will be

at level $u + ck - s$ immediately before the claim or dividend is first implemented. Once again, there are two sub-cases to consider at this point: either the claim is implemented before the dividend, or vice-versa. In the first sub-case, the non-ruining claim size, y, will range in value from y_0 to $u+ck-s$, and so, the restricted dividend amount will range in value from $\min\{c_1, (u + ck - s - y - b)_+\}$ to $\min\{c_2, (u + ck - s - y - b)_+\}$. Note that the $(.)_+$ function is employed to account for the possibility that the claim size, y, is large enough to bring the surplus process below level b, thereby resulting in a restricted dividend amount equal to 0. In the second sub-case, the restricted dividend amount will range in value from $\min\{c_1, u + ck - s - b\}$ to $\min\{c_2, u + ck - s - b\}$. Letting s^* denote the restricted dividend amount at time k, the non-ruining claim size, y, must therefore range in value from y_0 to $u + ck - s - s^*$. In either sub-case, ruin did not occur, and similar to the previous case, the surplus process can be viewed as "renewed" at time k, with corresponding new time horizon $m - k$, new initial surplus value equal to $u + ck - s - s^* - y$, and new initial dividend counter value equal to $V_{u,d}(0, k, y, s, s^*, Order)$, where $Order \in \{ClmFrst, \overline{ClmFrst}\}$. Moreover, following this first claim, the behaviour of our DISAM again reverts to that of an OISAM. Thus, using the law of total probability, we obtain

$$
\begin{aligned}
&\sum_{k\in\mathcal{T}_{u,d}(\min\{n_r,m\})} \sum_{s\in\mathcal{R}_{u,d}(k-1)} f_{u,d}^{(|\mathcal{T}_{u,d}(k-1)|)}(s) \\
&\times \left\{ p \sum_{y=y_0}^{u+ck-s} \alpha_y \sum_{s^*=\min\{c_1,(u+ck-s-y-b)_+\}}^{\min\{c_2,(u+ck-s-y-b)_+\}} d_{s^*}^*\{u + ck - s - y\} \right. \\
&\times E\left[\left\{s + s^* + S_{\min\{T,m-k\}}\right\}^r | U_0 = u + ck - s - s^* - y, D_0 = V_{u,d}(0, k, y, s, s^*, ClmFrst)\right] \\
&+ (1-p) \sum_{s^*=\min\{c_1,u+ck-s-b\}}^{\min\{c_2,u+ck-s-b\}} d_{s^*}^*\{u + ck - s\} \sum_{y=y_0}^{u+ck-s-s^*} \alpha_y \\
&\left. \times E\left[\left\{s + s^* + S_{\min\{T,m-k\}}\right\}^r | U_0 = u + ck - s - s^* - y, D_0 = V_{u,d}(0, k, y, s, s^*, \overline{ClmFrst})\right] \right\} r_k.
\end{aligned}
$$

As done previously, applying the binomial theorem within the above expectation functions yields

$$\begin{aligned}
&\sum_{k\in\mathcal{T}_{u,d}(\min\{n_r,m\})}\sum_{s\in\mathcal{R}_{u,d}(k-1)} f_{u,d}^{(|\mathcal{T}_{u,d}(k-1)|)}(s)\\
&\times\Bigg\{p\sum_{y=y_0}^{u+ck-s}\alpha_y\sum_{s^*=\min\{c_1,(u+ck-s-y-b)_+\}}^{\min\{c_2,(u+ck-s-y-b)_+\}} d_{s^*}^*\{u+ck-s-y\}\\
&\times\sum_{\omega=0}^{r}\binom{r}{\omega}(s+s^*)^{\omega}\mathcal{D}_{m-k}^{r-\omega}(u+ck-s-s^*-y,V_{u,d}(0,k,y,s,s^*,ClmFrst))\\
&+(1-p)\sum_{s^*=\min\{c_1,u+ck-s-b\}}^{\min\{c_2,u+ck-s-b\}} d_{s^*}^*\{u+ck-s\}\sum_{y=y_0}^{u+ck-s-s^*}\alpha_y\\
&\times\sum_{\omega=0}^{r}\binom{r}{\omega}(s+s^*)^{\omega}\mathcal{D}_{m-k}^{r-\omega}(u+ck-s-s^*-y,V_{u,d}(0,k,y,s,s^*,\overline{ClmFrst}))\Bigg\}r_k.
\end{aligned}$$

(5) **First claim occurs beyond time horizon m (i.e., $k>m$):**

In this case, there is a total of $|\mathcal{T}_{u,d}(m)|$ dividends. Thus, we immediately obtain

$$\sum_{k=m+1}^{n_r}\sum_{s\in\mathcal{R}_{u,d}(m)} f_{u,d}^{(|\mathcal{T}_{u,d}(m)|)}(s)s^r r_k = R_m\sum_{s\in\mathcal{R}_{u,d}(m)} f_{u,d}^{(|\mathcal{T}_{u,d}(m)|)}(s)s^r.$$

In cases (3) and (4), the OISAM counterpart $\mathcal{D}_m^r(u,d)$ appears in the derived formulas. However, we note that we can apply the same methodology on $\mathcal{D}_m^r(u,d)$ except that quantities n_r, r_k, and R_m are replaced by n_a, a_k, and A_m, respectively. The following two pages combine all the above cases together into a single formula.

$$\widetilde{\mathcal{D}}^r_m(u,d) = \sum_{k\notin\mathcal{T}_{u,d}(\min\{n_r,m\})}\sum_{s\in\mathcal{R}_{u,d}(k)} f_{u,d}^{\left(|\mathcal{T}_{u,d}(k)|\right)}(s)\left\{\sum_{y=y_0}^{u+ck-s}\alpha_y\sum_{\omega=0}^{r}\binom{r}{\omega}s^{\omega}\mathcal{D}^{r-\omega}_{m-k}(u+ck-s-y,V_{u,d}(0,k,y,s,.,.))+s^r\Lambda_{u+ck-s}\right\}r_k$$

$$+\ p\sum_{k\in\mathcal{T}_{u,d}(\min\{n_r,m\})}\sum_{s\in\mathcal{R}_{u,d}(k-1)} f_{u,d}^{\left(|\mathcal{T}_{u,d}(k-1)|\right)}(s)\left\{\sum_{y=y_0}^{u+ck-s}\alpha_y\sum_{s^*=\min\{c_1,(u+ck-s-y-b)_+\}}^{\min\{c_2,(u+ck-s-y-b)_+\}} d^*_{s^*}\{u+ck-s-y\}\right.$$
$$\left.\times\sum_{\omega=0}^{r}\binom{r}{\omega}(s+s^*)^{\omega}\mathcal{D}^{r-\omega}_{m-k}(u+ck-s-s^*-y,V_{u,d}(0,k,y,s,s^*,ClmFrst))+s^r\Lambda_{u+ck-s}\right\}r_k$$

$$+\ (1-p)\sum_{k\in\mathcal{T}_{u,d}(\min\{n_r,m\})}\left\{\sum_{s\in\mathcal{R}_{u,d}(k-1)} f_{u,d}^{\left(|\mathcal{T}_{u,d}(k-1)|\right)}(s)\sum_{s^*=\min\{c_1,u+ck-s-b\}}^{\min\{c_2,u+ck-s-b\}} d^*_{s^*}\{u+ck-s\}\sum_{y=y_0}^{u+ck-s-s^*}\alpha_y\right.$$
$$\left.\times\sum_{\omega=0}^{r}\binom{r}{\omega}(s+s^*)^{\omega}\mathcal{D}^{r-\omega}_{m-k}(u+ck-s-s^*-y,V_{u,d}(0,k,y,s,s^*,\overline{ClmFrst}))+\sum_{s\in\mathcal{R}_{u,d}(k)} f_{u,d}^{\left(|\mathcal{T}_{u,d}(k)|\right)}(s)s^r\Lambda_{u+ck-s}\right\}r_k$$

$$+\ R_m\sum_{s\in\mathcal{R}_{u,d}(m)} f_{u,d}^{(|\mathcal{T}_{u,d}(m)|)}(s)s^r,$$

where

$$\mathcal{D}_m^r(u,d) = \sum_{k\notin\mathcal{T}_{u,d}(\min\{n_a,m\})}\sum_{s\in\mathcal{R}_{u,d}(k)} f_{u,d}^{\left(\left|\mathcal{T}_{u,d}(k)\right|\right)}(s)\left\{\sum_{y=y_0}^{u+ck-s}\alpha_y\sum_{\omega=0}^{r}\binom{r}{\omega}s^{\omega}\mathcal{D}_{m-k}^{r-\omega}(u+ck-s-y,V_{u,d}(0,k,y,s,.,.))+s^r\Lambda_{u+ck-s}\right\}a_k$$

$$+\ p\sum_{k\in\mathcal{T}_{u,d}(\min\{n_a,m\})}\sum_{s\in\mathcal{R}_{u,d}(k-1)} f_{u,d}^{\left(\left|\mathcal{T}_{u,d}(k-1)\right|\right)}(s)\left\{\sum_{y=y_0}^{u+ck-s}\alpha_y\sum_{s^*=\min\{c_1,(u+ck-s-y-b)_+\}}^{\min\{c_2,(u+ck-s-y-b)_+\}} d_{s^*}^*\{u+ck-s-y\}\right.$$
$$\left.\times\sum_{\omega=0}^{r}\binom{r}{\omega}(s+s^*)^{\omega}\mathcal{D}_{m-k}^{r-\omega}(u+ck-s-s^*-y,V_{u,d}(0,k,y,s,s^*,ClmFrst))+s^r\Lambda_{u+ck-s}\right\}a_k$$

$$+\ (1-p)\sum_{k\in\mathcal{T}_{u,d}(\min\{n_a,m\})}\left\{\sum_{s\in\mathcal{R}_{u,d}(k-1)} f_{u,d}^{\left(\left|\mathcal{T}_{u,d}(k-1)\right|\right)}(s)\sum_{s^*=\min\{c_1,u+ck-s-b\}}^{\min\{c_2,u+ck-s-b\}} d_{s^*}^*\{u+ck-s\}\sum_{y=y_0}^{u+ck-s-s^*}\alpha_y\right.$$
$$\left.\times\sum_{\omega=0}^{r}\binom{r}{\omega}(s+s^*)^{\omega}\mathcal{D}_{m-k}^{r-\omega}(u+ck-s-s^*-y,V_{u,d}(0,k,y,s,s^*,\overline{ClmFrst}))+\sum_{s\in\mathcal{R}_{u,d}(k)} f_{u,d}^{\left(\left|\mathcal{T}_{u,d}(k)\right|\right)}(s)s^r\Lambda_{u+ck-s}\right\}a_k$$

$$+\ A_m\sum_{s\in\mathcal{R}_{u,d}(m)} f_{u,d}^{(|\mathcal{T}_{u,d}(m)|)}(s)s^r.$$

Chapter 5

Analysis and Comparison of Six DISAMs Extracted from the General Class

5.1 Introduction

The framework used in this thesis derives its generality from that fact that certain functions defined for characterizing particular aspects of the surplus process, $V_{u,d}(.)$ being the central one, are represented in a general way. The objective of this part of the thesis is to shed additional light on the computational aspects of the DISAM class by comparing six different types of models belonging to this class and providing numerical calculations for particular examples. The difference between these six models lie in how D_t behaves after reaching level h—which we classify as *Above Modes*—and how it behaves below the threshold level—which we classify as *Below Modes*.

5.2 Specifying the Above and Below Modes

We introduce the following two Above Modes:

(1) **Consecutive:** While at or above level b, the dividend counter increments one unit at a time until it reaches h. Once the dividend counter is equal to h, it remains equal to h in future time periods until a large enough claim brings the surplus process below level b, where the corresponding Below Mode then takes effect. This type of Above Mode was explored in Drekic and Mera (2011) for the case $h = 1$.

(2) **Cyclic (a.k.a. Periodic):** While at or above level b, the dividend counter increments one unit at a time until it reaches h. Once the dividend counter is equal to h, it is reset to 1 in the next time period (provided that the surplus process remains above level b at this time). In the recent literature, such types of dividend systems have been explored in a variety of continuous-time insurance risk models (e.g., see Albrecher et al. (2011) and Avanzi et al. (2013)).

We also introduce the following three Below Modes:

(1) **Countdown:** While below level b, the dividend counter decrements one unit at a time until it reaches 1. It then remains at 1 until the surplus process reaches or crosses level b. However, if the process reaches or crosses level b and is immediately brought below level b by a resulting claim, the countdown process is not interrupted (assuming the dividend counter is greater than 1). On the other hand, if the surplus process is already above the threshold level and is brought down below level b, the dividend counter freezes at that value and begins counting down at the next time unit—unless, of course, the surplus process goes above level b again at the next time unit. Figures 5.1 and 5.2 compare two sample paths of a surplus process under a Consecutive Countdown model and under a Cyclic Countdown model, both subjected to the same claim occurrence times and claim sizes.

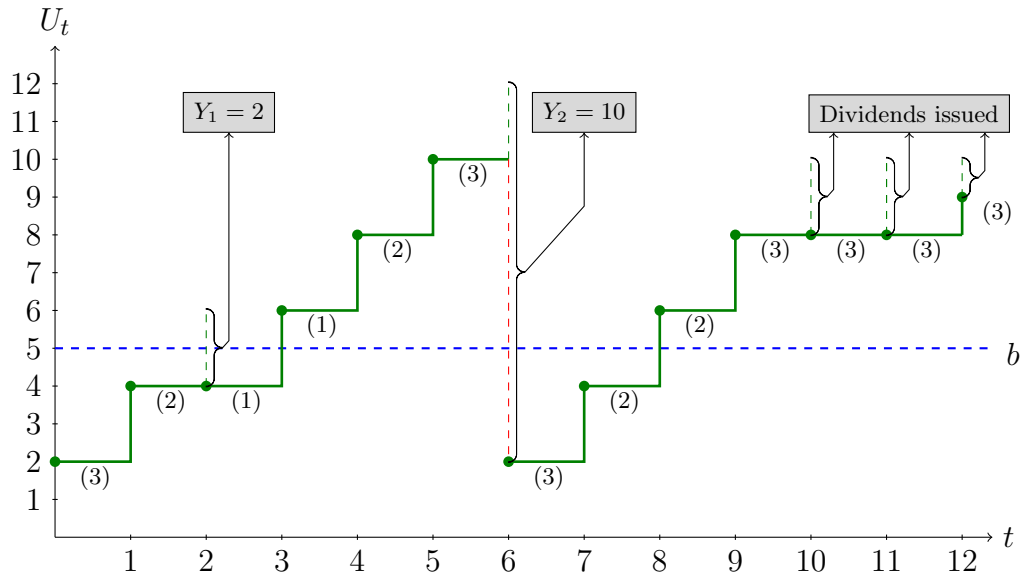

Figure 5.1: A sample path of a Consecutive Countdown surplus process with $u = 2$, $d = 3$, $c_1 = 1$, $c_2 = 2$, $c = 2$, $b = 5$, $h = 3$, and $p = 1$.

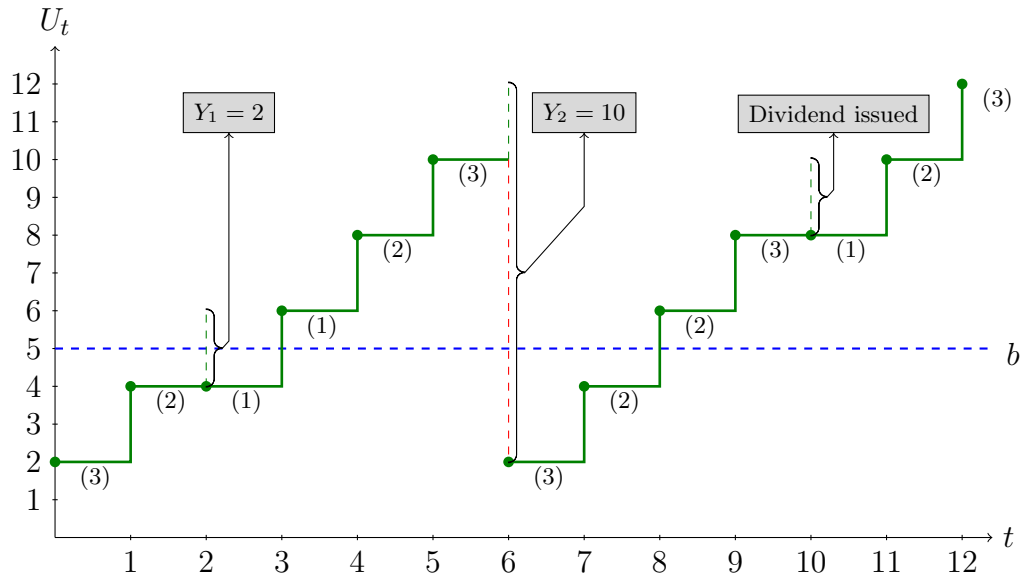

Figure 5.2: A sample path of a Cyclic Countdown surplus process with $u = 2$, $d = 3$, $c_1 = 1$, $c_2 = 2$, $c = 2$, $b = 5$, $h = 3$, and $p = 1$.

(2) **Freeze:** While below level b, the dividend counter does not change its value. However, it does potentially change its value one time unit after the surplus process reaches or crosses level b. Moreover, if the surplus process is already above the threshold level and is subsequently brought down below level b, the dividend counter freezes at that value until it crosses level b again. Of course, for a Consecutive Freeze model, this implies that the once the dividend counter reaches h, it will remain permanently equal to h, and so, thereon, a dividend will always be paid one time period after reaching or crossing level b (provided that there is no claim significant enough to bring the surplus process back below level b). Figures 5.3 and 5.4 compare two sample paths of a surplus process under a Consecutive Freeze model and under a Cyclic Freeze model, both subjected to the same claim occurrence times and claim sizes.

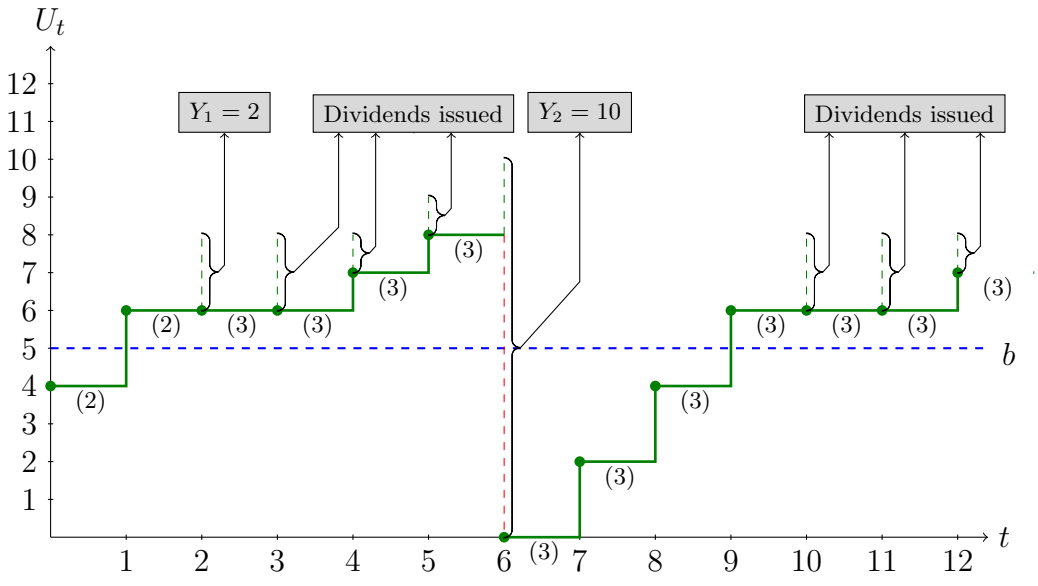

Figure 5.3: A sample path of a Consecutive Freeze surplus process with $u = 4$, $d = 2$, $c_1 = 1$, $c_2 = 2$, $c = 2$, $b = 5$, $h = 3$, and $p = 1$.

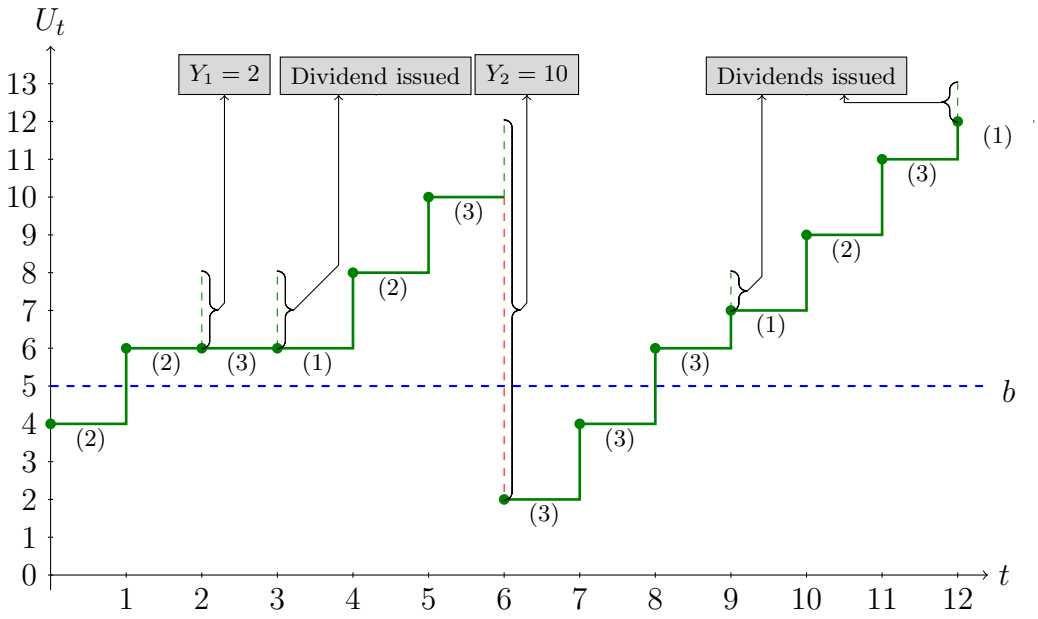

Figure 5.4: A sample path of a Cyclic Freeze surplus process with $u = 2$, $d = 2$, $c_1 = 1$, $c_2 = 2$, $c = 2$, $b = 5$, $h = 3$, and $p = 1$.

(3) **Reset:** The simplest of the three Below Modes in that the dividend counter is always equal to 1 whenever the surplus process resides below level b. Figures 5.5 and 5.6 compare two sample paths of a surplus process under a Consecutive Reset model and under a Cyclic Reset model, both subjected to the same claim occurrence times and claim sizes.

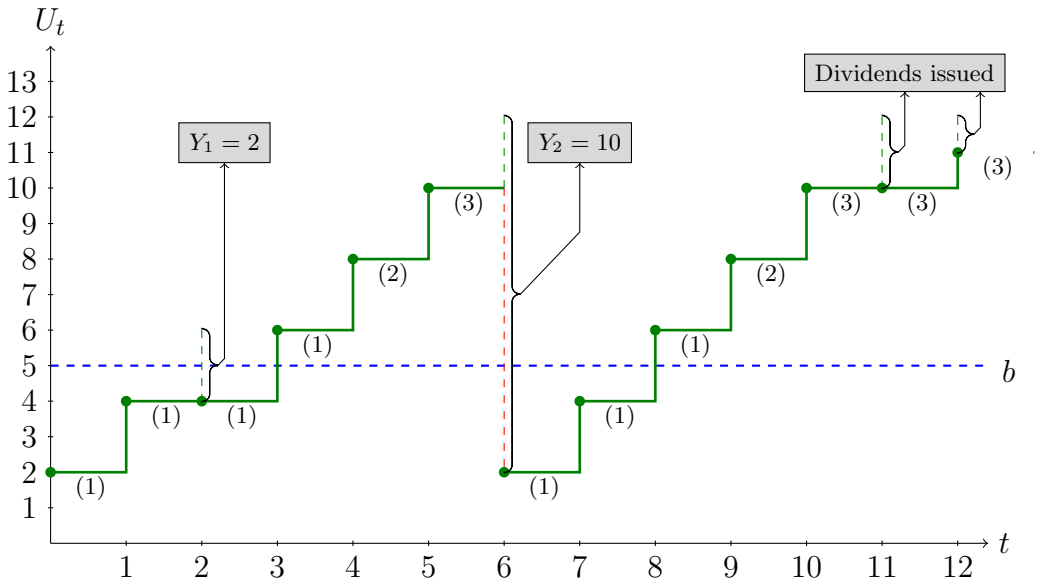

Figure 5.5: A sample path of a Consecutive Reset surplus process with $u = 2$, $d = 1$, $c_1 = 1$, $c_2 = 2$, $c = 2$, $b = 5$, $h = 3$, and $p = 1$.

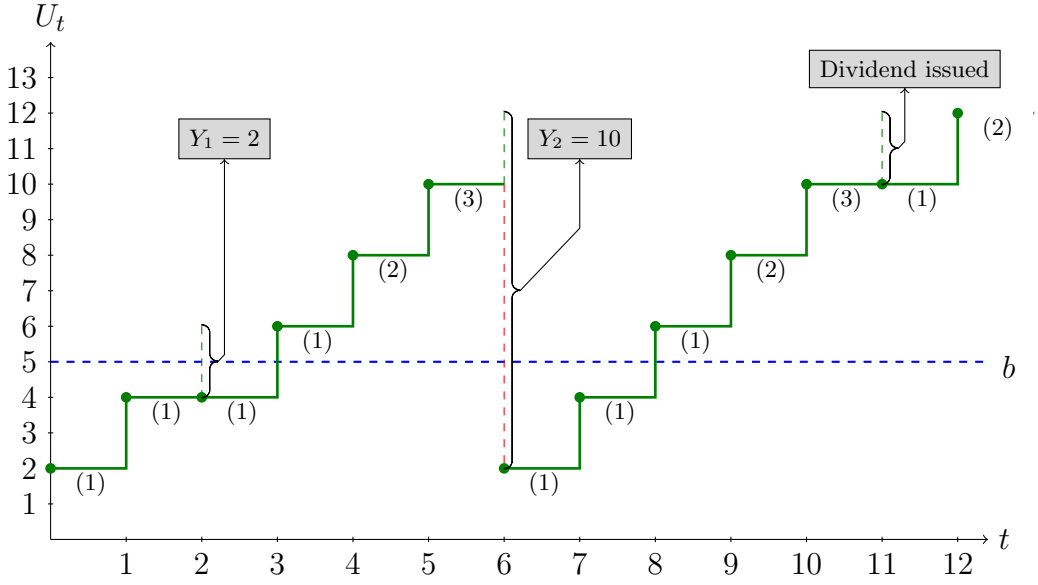

Figure 5.6: A sample path of a Cyclic Reset surplus process with $u = 2$, $d = 1$, $c_1 = 1$, $c_2 = 2$, $c = 2$, $b = 5$, $h = 3$, and $p = 1$.

Of course, for the special case $h = 1$, all the above models collapse to one type. We next proceed to provide formulas for the functions given in Section 2.4.2 for each of the six models.

5.3 Function Derivations

Categorizing models according to the two aforementioned Above Mode types (i.e., Below Mode type is kept general here), we obtain the following expressions:

- Assuming no claims have occurred during the time period over which the first n dividends are issued, we have

$$P_{z^*_{u,d}}(n) = \begin{cases} n & \text{if Consecutive;} \\ \\ nh & \text{if Cyclic,} \end{cases}$$

$$TotDiv_n(x) = \begin{cases} 0 & \text{if } n = 0; \\ \\ \min\left\{x, u + cz^*_{u,d} - b\right\} & \text{if } n = 1; \\ \\ TotDiv_{n-1}(x) + \min\left\{x, u + c\left[z^*_{u,d} + (n-1)\right] - TotDiv_{n-1}(x) - b\right\} & \text{if } n > 1 \\ & \text{and Consecutive;} \\ \\ TotDiv_{n-1}(x) + \min\left\{x, u + c\left[z^*_{u,d} + (n-1)h\right] - TotDiv_{n-1}(x) - b\right\} & \text{if } n > 1 \\ & \text{and Cyclic,} \end{cases}$$

and

$$f_{u,d}^{(n)}(s) = \begin{cases} \mathbf{1}_{[s=0]} & \text{if } n = 0; \\ d_s^*\{u + cz_{u,d}^*\} & \text{if } n = 1; \\ \displaystyle\sum_{w=0}^{s} d_{s-w}^*\{u + c[z_{u,d}^* + (n-1)] - w\} f_{u,d}^{(n-1)}(w) & \text{if } n > 1 \text{ and Consecutive}; \\ \displaystyle\sum_{w=0}^{s} d_{s-w}^*\{u + c[z_{u,d}^* + (n-1)h] - w\} f_{u,d}^{(n-1)}(w) & \text{if } n > 1 \text{ and Cyclic}. \end{cases}$$

- Assuming no claims have occurred up to time t, we have

$$\mathcal{T}_{u,d}(t) = \begin{cases} \left\{z_{u,d}^* + (n-1) : n = 1, 2, \ldots, t - z_{u,d}^* + 1\right\} & \text{if Consecutive}; \\ \left\{z_{u,d}^* + (n-1)h : n = 1, 2, \ldots, \left\lfloor \dfrac{t - z_{u,d}^*}{h} \right\rfloor + 1\right\} & \text{if Cyclic} \end{cases}$$

and

$$\mathcal{R}_{u,d}(t) = \begin{cases} \left\{TotDiv_{(t-z_{u,d}^*+1)_+}(c_1), \ldots, TotDiv_{(t-z_{u,d}^*+1)_+}(c_2)\right\} & \text{if Consecutive}; \\ \left\{TotDiv_{\left\lfloor \frac{t-z_{u,d}^*}{h} \right\rfloor + 1}(c_1), \ldots, TotDiv_{\left\lfloor \frac{t-z_{u,d}^*}{h} \right\rfloor + 1}(c_2)\right\} & \text{if Cyclic}. \end{cases}$$

- $v_{u,d}(k) = \begin{cases} u + ck - TotDiv_{(k-z^*_{u,d})_+}(c_1) & \text{if Consecutive;} \\ \\ u + ck - TotDiv_{\left(\left\lfloor \frac{k-1-z^*_{u,d}}{h} \right\rfloor + 1\right)_+}(c_1) & \text{if Cyclic.} \end{cases}$

Categorizing models according to the three aforementioned Below Mode types (i.e., Above Mode type is kept general here), we end up with

$$z^*_{u,d} = \begin{cases} z_u + h - (d - 1 - (z_u - 1)_+)_+ & \text{if Countdown;} \\ \\ z_u + h - (d-1) & \text{if Freeze;} \\ \\ z_u + h - \mathbf{1}_{[u \geq b]}(d-1) & \text{if Reset.} \end{cases}$$

We note that if our Below Mode is Reset, d must be equal to 1 if $u < b$. The function $V_{u,d}(0, t, y, s, s^*, Order)$, however, is specific for each combination of Above and Below Mode types, which we indicate in the following pages:

(1) **The Consecutive Countdown Model:**

$$V_{u,d}(0,t,y,s,s^*,Order) = \begin{cases} \max\{d-t,1\} & \text{if } t < z_u, \\ & \text{or } t = z_u \text{ and } y > u + ct - b; \\ \max\{d-(z_u-1)_+,1\} & \text{if } t = z_u \text{ and } y \le u + ct - b; \\ \max\{d-(z_u-1)_+,1\} + (t - z_u) & \text{if } z_u < t < z^*_{u,d} \text{ and } y \le u + ct - b; \\ \max\{d-(z_u-1)_+,1\} + (t - z_u - 1) & \text{if } z_u < t < z^*_{u,d} \text{ and } y > u + ct - b; \\ h & \text{if } t \ge z^*_{u,d}. \end{cases}$$

(2) **The Consecutive Freeze Model:**

$$V_{u,d}(0,t,y,s,s^*,Order) = \begin{cases} d & \text{if } t \le z_u; \\ d + (t - z_u) & \text{if } z_u < t < z^*_{u,d} \text{ and } y \le u + ct - b; \\ d + (t - z_u - 1) & \text{if } z_u < t < z^*_{u,d} \text{ and } y > u + ct - b; \\ h & \text{if } t \ge z^*_{u,d}. \end{cases}$$

(3) **The Consecutive Reset Model:**

$$V_{u,d}(0,t,y,s,s^*,Order) = \begin{cases} 1 & \text{if } t \le z_u, \\ & \text{or } z_u < t < z^*_{u,d} \text{ and } y > u + ct - b, \\ & \text{or } t \ge z^*_{u,d} \text{ and } y > u + ct - s - \mathbf{1}_{[Order=\overline{ClmFirst}]}s^* - b; \\ d + (t - z_u) & \text{if } z_u < t < z^*_{u,d} \text{ and } y \le u + ct - b; \\ h & \text{if } t \ge z^*_{u,d},\ y \le u + ct - s - \mathbf{1}_{[Order=\overline{ClmFirst}]}s^* - b. \end{cases}$$

(4) **The Cyclic Countdown Model:**

$$V_{u,d}(0,t,y,s,s^*,Order) = \begin{cases} \max\{d-t,1\} & \text{if } t < z_u, \\ & \text{or } t = z_u \text{ and } y > u + ct - b; \\ \max\{d-(z_u-1)_+,1\} & \text{if } t = z_u \text{ and } y \le u + ct - b; \\ \max\{d-(z_u-1)_+,1\} + (t - z_u) & \text{if } z_u < t < z^*_{u,d} \text{ and } y \le u + ct - b; \\ \max\{d-(z_u-1)_+,1\} + (t - z_u - 1) & \text{if } z_u < t < z^*_{u,d} \text{ and } y > u + ct - b; \\ 1 & \text{if } t \ge z^*_{u,d},\ y \le u + ct - s - \mathbf{1}_{[Order=\overline{ClmFirst}]}s^* - b, \\ & \text{and } \dfrac{t - z^*_{u,d}}{h} \in \mathbb{N}; \\ mod(t - z^*_{u,d}, h) + 1 & \text{if } t \ge z^*_{u,d},\ y \le u + ct - s - b, \text{ and } \dfrac{t - z^*_{u,d}}{h} \notin \mathbb{N}; \\ h & \text{if } t \ge z^*_{u,d},\ y > u + ct - s - \mathbf{1}_{[Order=\overline{ClmFirst}]}s^* - b, \\ & \text{and } \dfrac{t - z^*_{u,d}}{h} \in \mathbb{N}; \\ mod(t - z^*_{u,d}, h) & \text{if } t \ge z^*_{u,d},\ y > u + ct - s - b, \text{ and } \dfrac{t - z^*_{u,d}}{h} \notin \mathbb{N}. \end{cases}$$

(5) **The Cyclic Freeze Model:**

$$V_{u,d}(0,t,y,s,s^*,Order) = \begin{cases} d & \text{if } t \le z_u; \\ d + (t - z_u) & \text{if } z_u < t < z^*_{u,d} \text{ and } y \le u + ct - b; \\ d + (t - z_u - 1) & \text{if } z_u < t < z^*_{u,d} \text{ and } y > u + ct - b; \\ 1 & \text{if } t \ge z^*_{u,d},\ y \le u + ct - s - \mathbf{1}_{[Order=\overline{ClmFirst}]}s^* - b, \\ & \text{and } \dfrac{t - z^*_{u,d}}{h} \in \mathbb{N}; \\ mod(t - z^*_{u,d}, h) + 1 & \text{if } t \ge z^*_{u,d},\ y \le u + ct - s - b, \text{ and } \dfrac{t - z^*_{u,d}}{h} \notin \mathbb{N}; \\ h & \text{if } t \ge z^*_{u,d},\ y > u + ct - s - \mathbf{1}_{[Order=\overline{ClmFirst}]}s^* - b, \\ & \text{and } \dfrac{t - z^*_{u,d}}{h} \in \mathbb{N}; \\ mod(t - z^*_{u,d}, h) & \text{if } t \ge z^*_{u,d},\ y > u + ct - s - b, \text{ and } \dfrac{t - z^*_{u,d}}{h} \notin \mathbb{N}. \end{cases}$$

(6) **The Cyclic Reset Model:**

$$V_{u,d}(0,t,y,s,s^*,Order) = \begin{cases} 1 & \text{if } t \leq z_u, \\ & \text{or } z_u < t < z^*_{u,d} \text{ and } y > u + ct - b, \\ & \text{or } t \geq z^*_{u,d} \text{ and } \dfrac{t - z^*_{u,d}}{h} \in \mathbb{N}, \\ & \text{or } t \geq z^*_{u,d},\ y > u + ct - s - b, \text{ and } \dfrac{t - z^*_{u,d}}{h} \notin \mathbb{N}; \\ d + (t - z_u) & \text{if } z_u < t < z^*_{u,d} \text{ and } y \leq u + ct - b; \\ mod(t - z^*_{u,d}, h) + 1 & \text{if } t \geq z^*_{u,d},\ y \leq u + ct - s - b, \text{ and } \dfrac{t - z^*_{u,d}}{h} \notin \mathbb{N}. \end{cases}$$

We remark that unlike Cases (4) and (5), the $V_{u,d}(.)$ function in Case (6) does not depend on the $Order$ variable since it is guaranteed to be equal to 1 at a dividend paying time unit. Furthermore, as noted in Section 2.4.2, the parameters of the $V_{u,d}(.)$ function need to be correctly specified; otherwise the function will be invalid (e.g., if $s > 0$ and $t < z^*_{u,d}$).

Using the formulas above, we now proceed to specifying the parameters of the models we investigate and demonstrating numerical results of particular ruin-related performance measures of interest.

5.4 Numerical Examples

In what follows, we consider some of the same parametric models originally examined by Drekic and Mera (2011). In particular, we assume:

- $u = 10$, $d = 1$, $b = 50$, and $c = 5$.
- $n_a = 10$ and $a_\omega = 1/10$ for $\omega = 1, 2, \ldots, 10$, so that $E[W_2] = 5.5$.
- We consider three different types of pmfs for r_ω:

(1) Delayed: $n_r = 1$, and thus, $r_1 = 1$.

(2) Ordinary: $n_r = n_a = 10$ and $r_\omega = a_\omega = 1/10$ for $\omega = 1, 2, \ldots, 10$.

(3) Stationary: r_ω is the stationary (or equilibrium) pmf of a_ω given by the formula $r_\omega = \frac{A_{\omega-1}}{E[W_2]}$ for $\omega = 1, 2, \ldots, n_a$ (e.g., see Pavlova and Willmot (2004)). Hence, using the above formula for a_ω, we immediately obtain $r_\omega = \frac{11-\omega}{55}$ for $\omega = 1, 2, \ldots, 10$, so that $E[W_1] = 4$.

- $\alpha_y = \left(1 + \frac{y-1}{30}\right)^{-4} - \left(1 + \frac{y}{30}\right)^{-4}$ for $y \in \mathbb{Z}^+$, so that $y_0 = 1$, $y_\alpha = \infty$, and $E[Y_1] \approx 10.5111$. We remark that the pmf α_y is a discretized version of a Pareto distribution.

- For Consecutive models, let $c_1 = 2$ and $c_2 = 4$ with $d_2 = 2/5$, $d_3 = 1/5$, and $d_4 = 2/5$. As for Cyclic models, let $c_1 = c_2 = \infty$. Models of this kind are known as *Parisian* (e.g., see Dassios and Wu (2009)): a term originally coined in Chesney and Jean-Picqué (1997), used to describe a particular kind of European barrier option where the option becomes worthless if the underlying asset has been consistently above (up-and-out) or below (down-and-out) a pre-specified threshold level for a certain amount of time.

We calculate the following performance measures for each of the six model types under different combinations of $h \in \{1, 2, 3, 4, 5\}$ and $p \in \{0, 0.5, 1\}$:

- The expected surplus prior to ruin and expected deficit at ruin by a finite time horizon of 100 units, denoted by I_{100} and J_{100}, respectively. These two performance measures are calculated using $\widetilde{G}_{v,m}(10, 1)$ with $v = 1$ and $m = 100$, and setting the penalty function accordingly: $w(U_{T^-}, |U_T|) = U_{T^-}$ for calculating I_{100} and $w(U_{T^-}, |U_T|) = |U_T|$ for calculating J_{100}.

- The cumulative trivariate ruin probability denoted by $\Psi_{\tau,i,j}(10,1)$, whose value is equal to $\sum_{t=1}^{\tau} \sum_{i_0=i_{min}}^{\min\{\tilde{v}_{10}(t),i\}} \sum_{j_0=1}^{j} \psi_{t,i_0,j_0}(10,1)$. This performance measure is calculated via the methodology in Chapter 3 as well as by using $\tilde{G}_{1,\tau}(10,1)$ with the penalty function set equal to $w(U_{T^-},|U_T|) = \mathbf{1}_{[U_{T^-}\leq\, i]}\mathbf{1}_{[|U_T|\leq\, j]}$. We calculate this performance measure for $\tau = 100$, $i = \lceil I_{100} \rceil$, and $j = \lceil J_{100} \rceil$.

- The cumulative univariate ruin probability denoted by $\Gamma_\tau(10,1)$, whose value is calculated using $\tilde{G}_{1,\tau}(10,1)$ and setting the penalty function equal to $w(U_{T^-},|U_T|) = 1$. We calculate this performance measure for $\tau = 100$.

- The expected value and standard deviation of total dividends by a finite time horizon of 100 units or before ruin occurs, whichever happens first. These two performance measures are calculated using $\widetilde{\mathcal{D}}^r_m(10,1)$ with $m = 100$ and $r = 1, 2$.

In the following tables, let (a1) represent a Consecutive Countdown model, (a2) represent a Consecutive Freeze model, (a3) represent a Consecutive Reset model, (b1) represent a Cyclic Countdown model, (b2) represent a Cyclic Freeze model, and (b3) represent a Cyclic Reset model. All results are displayed to at least 3 significant digits. Both approaches (i.e., using MAMs and the first claim conditioning method) were available to calculate certain quantities (such as the cumulative trivariate probabilities), thereby serving as a way of verifying the numerical results obtained. We note however, for the examples chosen here, the approach based on MAMs tended to be slower overall. All our results were calculated via the C++ programming language, using the Boost and OpenMP libraries in our program.

Based on the numerical results in Tables 5.1 through 5.6, we give a brief summary of our findings:

- Overall, as h and/or p increase/increases (keeping all other parameter values and distributions fixed), the expected deficit at ruin value, the cumulative univariate ruin probability, and the standard deviation of total dividends (before a finite time horizon) all decrease. We note that this same trend also occurs as $E[W_1]$ increases.

- For models with either $h = 1$ or $h = 2$ and $p = 0$, the Below Mode type actually ends up being irrelevant since the dividend counter is always equal to 1 below level b (recall that we set $d = 1$ in all our examples). As a result, Consecutive models (i.e., (a1), (a2), and (a3)) as well as Cyclic models (i.e., (b1), (b2), and (b3)) are identical in behaviour (keeping all other parameter values and distributions fixed), and hence yield the same performance measure values in all six tables.

- In Table 5.1, we observe that for Consecutive models (i.e., (a1), (a2), and (a3)), the expected surplus prior to ruin shows a clear decreasing trend as $E[W_1]$, h, and/or p increase/increases. This trend is also seen in Cyclic models (i.e., (b1), (b2), and (b3)) when varying h and p. However, this is not consistently the case when comparing across models with different distributions for W_1, as demonstrated by simply comparing the ($h = 1$, $p = 0$) values for distribution (1), model (b1) with distribution (2), model (b1). Nonetheless, the trend becomes more consistent as h increases.

  An explanation for this trend is based on the antagonistic nature of the two random variables within the Gerber-Shiu function, namely $w(U_{T^-}, |U_T|) = U_{T^-}$ and $\mathbf{1}_{[T \leq 100]}$. For models whose surplus process yields higher surplus values, U_{T^-} is more likely to be larger in value than what it would be in models whose surplus process yields lower surplus values. On the other hand, models whose surplus process yields higher values also have lower ruin probabilities (i.e., $\mathbf{1}_{[T \leq 100]} = 0$ more often). And so, the

trend we see in Table 5.1 reflects the quantity that has the greater influence. For the examples given, we find that the quantity $\mathbf{1}_{[T\leq 100]}$ becomes more pronounced for larger values of $E[W_1]$, h, and p (when comparing between models of the same Below Mode type).

- In Table 5.2, we noted earlier that the expected deficit at ruin shows a clear decreasing trend as $E[W_1]$, h, and/or p increase/increases. More generally, the trend reflects the notion that as the probability of ruin decreases, the expected deficit at ruin also decreases. A mathematical explanation for this phenomenon is based on realizing that the two random variables within the Gerber-Shiu function, $w(U_{T^-}, |U_T|) = |U_T|$ and $\mathbf{1}_{[T\leq 100]}$, are complementary in the sense that models whose surplus process yields relatively high surplus values have smaller ruin probabilities (i.e., $\mathbf{1}_{[T\leq 100]} = 0$ more often), and hence, will more likely yield lower expected deficit at ruin values when compared to those found in models yielding lower surplus values.

- In Table 5.3, we remark that the values of $\lceil I_{100} \rceil$ and $\lceil J_{100} \rceil$ are not static and vary depending on the specific model being considered. Furthermore, we note that the results for Consecutive models (i.e., (a1), (a2), and (a3)) do not have any noticeable "jumps" in terms of their values, whereas Cyclic models (i.e., (b1), (b2), and (b3)) do have noticeable jumps (particularly when $h = 2$ and $h = 3$), reflecting greater variability in the values of $\lceil I_{100} \rceil$ and $\lceil J_{100} \rceil$ for these models.

- As expected, there is a decreasing trend in the results of Table 5.4 as $E[W_1]$, h, and/or p increase/increases since the surplus process is allowed to take on higher values as these parameters are increased. We remark that Reset models (i.e., (a3) and (b3)) yield the lowest ruin probabilities while Freeze models (i.e., (a2) and (b2)) yield the highest (keeping all other parameter values and distributions fixed). This

makes intuitive sense since the surplus process for Reset models incorporates less frequent dividends than the other two Below Mode counterparts, and hence their surplus processes are less likely to become ruined.

- In Tables 5.5 and 5.6, we observe that for Consecutive models (i.e., (a1), (a2), and (a3)), the expected values and standard deviations of total dividends (before a finite time horizon) decrease as $E[W_1]$, h, and/or p increase/increases. This also holds true in the case of Cyclic models (i.e., (b1), (b2), and (b3)) for the latter performance measure. However, for the expected value of total dividends, this decreasing trend shows up for larger values of h (i.e., $h \geq 3$). For instance, when comparing Cyclic models with $p = 0.5$, we find that the expected total dividends are less for models with $h = 5$ than those with $h = 4$, which themselves are less than those with $h = 3$ (keeping all other parameter values and distributions fixed).

We also note here that the Reset models (i.e., (a3) and (b3)) yield the lowest expected total dividend values while Freeze models (i.e., (a2) and (b2)) yield the highest. However, this is not generalizable beyond the parameter value and distribution choices given here because of the antagonistic nature between dividend frequency and the likelihood of ruin. An example disproving the generality can be constructed by considering a Cyclic model where $u = 2$, $c = 2$, $b = 5$, $h = 4$, and $p = 0.5$, having delayed pmf r_ω as specified by (1), and with W_2 and Y_1 following the same distributions as above. For $m = 200$, we calculated $\tilde{D}^1_{200}(2, 1) \approx 6.297$ under the Freeze Mode and $\tilde{D}^1_{200}(2, 1) \approx 6.306$ under the Reset Mode, demonstrating that it is possible for insurance companies adopting Reset Mode dividend systems to yield higher expected total dividends than those adopting Freeze Mode ones (keeping all other parameter values and distributions fixed).

Table 5.1: Expected surplus prior to ruin values for $T \leq 100$

		$h=1$			$h=2$			$h=3$			$h=4$			$h=5$		
Distr.	Model	$p=0$	$p=0.5$	$p=1$	$p=0$	$p=0.5$	$p=1$	$p=0$	$p=0.5$	$p=1$	$p=0$	$p=0.5$	$p=1$	$p=0$	$p=0.5$	$p=1$
(1)	(a1)	10.771	10.569	10.363	10.466	10.302	10.132	10.229	10.085	9.936	10.026	9.895	9.760	9.841	9.721	9.597
	(a2)	10.771	10.569	10.363	10.597	10.411	10.221	10.417	10.244	10.068	10.238	10.076	9.912	10.060	9.909	9.756
	(a3)	10.771	10.569	10.363	10.407	10.251	10.091	10.117	9.988	9.854	9.871	9.759	9.643	9.655	9.555	9.453
	(b1)	15.220	14.751	14.220	14.701	14.263	13.765	14.248	13.821	13.360	13.844	13.434	12.997	13.428	13.069	12.663
	(b2)	15.220	14.751	14.220	14.701	14.268	13.777	14.261	13.837	13.380	13.866	13.458	13.025	13.455	13.097	12.691
	(b3)	15.220	14.751	14.220	14.701	14.260	13.763	14.243	13.812	13.345	13.837	13.421	12.980	13.402	13.046	12.644
(2)	(a1)	9.825	9.566	9.302	9.438	9.227	9.011	9.136	8.953	8.764	8.878	8.712	8.541	8.645	8.492	8.334
	(a2)	9.825	9.566	9.302	9.608	9.369	9.126	9.378	9.159	8.935	9.152	8.947	8.737	8.928	8.736	8.540
	(a3)	9.825	9.566	9.302	9.360	9.162	8.957	8.991	8.827	8.658	8.680	8.537	8.391	8.406	8.279	8.149
	(b1)	15.483	14.889	14.216	14.828	14.271	13.638	14.250	13.709	13.124	13.738	13.218	12.664	13.211	12.755	12.240
	(b2)	15.483	14.889	14.216	14.828	14.278	13.654	14.266	13.729	13.149	13.767	13.249	12.699	13.246	12.791	12.277
	(b3)	15.483	14.889	14.216	14.828	14.268	13.635	14.244	13.697	13.105	13.729	13.202	12.641	13.178	12.726	12.217
(3)	(a1)	10.240	9.997	9.748	9.873	9.674	9.470	9.588	9.413	9.233	9.342	9.184	9.021	9.119	8.974	8.824
	(a2)	10.240	9.997	9.748	10.032	9.807	9.577	9.816	9.606	9.393	9.599	9.404	9.205	9.385	9.203	9.018
	(a3)	10.240	9.997	9.748	9.801	9.614	9.420	9.452	9.295	9.134	9.155	9.019	8.879	8.894	8.773	8.650
	(b1)	15.620	15.056	14.417	14.996	14.469	13.870	14.453	13.937	13.379	13.964	13.467	12.940	13.460	13.025	12.535
	(b2)	15.620	15.056	14.417	14.996	14.475	13.885	14.469	13.956	13.402	13.991	13.497	12.973	13.492	13.059	12.569
	(b3)	15.620	15.056	14.417	14.996	14.466	13.867	14.448	13.926	13.361	13.955	13.452	12.918	13.428	12.997	12.512

Table 5.2: Expected deficit at ruin values for $T \leq 100$

		$h=1$			$h=2$			$h=3$			$h=4$			$h=5$		
Distr.	Model	$p=0$	$p=0.5$	$p=1$	$p=0$	$p=0.5$	$p=1$	$p=0$	$p=0.5$	$p=1$	$p=0$	$p=0.5$	$p=1$	$p=0$	$p=0.5$	$p=1$
(1)	(a1)	7.532	7.408	7.281	7.344	7.246	7.144	7.203	7.119	7.032	7.085	7.010	6.932	6.981	6.912	6.842
	(a2)	7.532	7.408	7.281	7.425	7.312	7.197	7.318	7.214	7.109	7.213	7.118	7.021	7.112	7.024	6.934
	(a3)	7.532	7.408	7.281	7.308	7.216	7.120	7.136	7.062	6.984	6.995	6.931	6.865	6.873	6.817	6.760
	(b1)	10.409	10.060	9.664	10.023	9.707	9.349	9.702	9.402	9.078	9.426	9.143	8.843	9.147	8.905	8.631
	(b2)	10.409	10.060	9.664	10.023	9.711	9.357	9.711	9.413	9.091	9.441	9.160	8.861	9.165	8.923	8.649
	(b3)	10.409	10.060	9.664	10.023	9.706	9.347	9.698	9.396	9.068	9.421	9.135	8.832	9.130	8.890	8.619
(2)	(a1)	5.538	5.378	5.216	5.298	5.172	5.042	5.118	5.011	4.900	4.969	4.873	4.774	4.837	4.750	4.660
	(a2)	5.538	5.378	5.216	5.403	5.258	5.111	5.266	5.135	5.001	5.134	5.013	4.889	5.006	4.894	4.779
	(a3)	5.538	5.378	5.216	5.251	5.133	5.011	5.032	4.937	4.839	4.852	4.772	4.688	4.698	4.627	4.550
	(b1)	9.196	8.753	8.251	8.708	8.307	7.850	8.297	7.917	7.507	7.947	7.589	7.208	7.595	7.287	6.939
	(b2)	9.196	8.753	8.251	8.708	8.312	7.860	8.309	7.931	7.523	7.967	7.610	7.231	7.617	7.310	6.962
	(b3)	9.196	8.753	8.251	8.708	8.304	7.848	8.293	7.909	7.494	7.941	7.579	7.194	7.572	7.267	6.925
(3)	(a1)	6.044	5.895	5.742	5.818	5.700	5.577	5.649	5.547	5.441	5.506	5.415	5.321	5.380	5.298	5.212
	(a2)	6.044	5.895	5.742	5.916	5.780	5.641	5.787	5.663	5.535	5.662	5.546	5.429	5.539	5.433	5.324
	(a3)	6.044	5.895	5.742	5.775	5.663	5.548	5.567	5.477	5.383	5.396	5.319	5.240	5.249	5.182	5.113
	(b1)	9.523	9.103	8.627	9.059	8.679	8.247	8.674	8.311	7.919	8.339	7.997	7.634	8.001	7.708	7.377
	(b2)	9.523	9.103	8.627	9.059	8.683	8.257	8.685	8.324	7.935	8.357	8.017	7.656	8.023	7.730	7.399
	(b3)	9.523	9.103	8.627	9.059	8.677	8.245	8.670	8.304	7.907	8.333	7.988	7.620	7.980	7.690	7.363

Table 5.3: Cumulative trivariate ruin probabilities of the form $\Psi_{100,\lceil I_{100}\rceil,\lceil J_{100}\rceil}(10,1)$

		$h=1$			$h=2$			$h=3$			$h=4$			$h=5$		
Distr.	Model	$p=0$	$p=0.5$	$p=1$	$p=0$	$p=0.5$	$p=1$	$p=0$	$p=0.5$	$p=1$	$p=0$	$p=0.5$	$p=1$	$p=0$	$p=0.5$	$p=1$
(1)	(a1)	0.00701	0.00694	0.00687	0.00690	0.00685	0.00680	0.00683	0.00679	0.00547	0.00677	0.00546	0.00500	0.00502	0.00500	0.00497
	(a2)	0.00701	0.00694	0.00687	0.00695	0.00689	0.00683	0.00689	0.00684	0.00678	0.00684	0.00679	0.00547	0.00679	0.00547	0.00544
	(a3)	0.00701	0.00694	0.00687	0.00688	0.00684	0.00679	0.00680	0.00504	0.00502	0.00502	0.00500	0.00498	0.00498	0.00496	0.00494
	(b1)	0.13654	0.13349	0.12578	0.13343	0.12588	0.01422	0.12589	0.01432	0.01391	0.01437	0.01403	0.01130	0.01405	0.01296	0.01112
	(b2)	0.13654	0.13349	0.12578	0.13343	0.12589	0.01423	0.12590	0.01433	0.01393	0.01438	0.01404	0.01288	0.01407	0.01298	0.01114
	(b3)	0.13654	0.13349	0.12578	0.13343	0.12588	0.01422	0.12588	0.01431	0.01390	0.01436	0.01402	0.01129	0.01403	0.01295	0.01111
(2)	(a1)	0.00199	0.00193	0.00187	0.00190	0.00186	0.00181	0.00184	0.00139	0.00136	0.00121	0.00119	0.00117	0.00119	0.00117	0.00115
	(a2)	0.00199	0.00193	0.00187	0.00194	0.00189	0.00183	0.00189	0.00184	0.00139	0.00185	0.00139	0.00120	0.00139	0.00120	0.00117
	(a3)	0.00199	0.00193	0.00187	0.00188	0.00184	0.00139	0.00139	0.00121	0.00118	0.00119	0.00117	0.00115	0.00115	0.00114	0.00113
	(b1)	0.02241	0.01924	0.01849	0.01918	0.01861	0.00634	0.01861	0.00645	0.00599	0.00650	0.00612	0.00508	0.00615	0.00518	0.00447
	(b2)	0.02241	0.01924	0.01849	0.01918	0.01862	0.00635	0.01863	0.00646	0.00601	0.00652	0.00614	0.00509	0.00617	0.00520	0.00448
	(b3)	0.02241	0.01924	0.01849	0.01918	0.01861	0.00633	0.01861	0.00644	0.00598	0.00649	0.00611	0.00506	0.00612	0.00516	0.00446
(3)	(a1)	0.00308	0.00246	0.00240	0.00243	0.00239	0.00235	0.00237	0.00234	0.00230	0.00233	0.00230	0.00226	0.00229	0.00174	0.00172
	(a2)	0.00308	0.00246	0.00240	0.00302	0.00242	0.00237	0.00242	0.00238	0.00233	0.00238	0.00234	0.00230	0.00234	0.00231	0.00227
	(a3)	0.00308	0.00246	0.00240	0.00242	0.00238	0.00234	0.00234	0.00231	0.00228	0.00229	0.00226	0.00172	0.00173	0.00171	0.00170
	(b1)	0.03293	0.03218	0.02825	0.03077	0.02837	0.00804	0.02837	0.00815	0.00716	0.00820	0.00728	0.00611	0.00784	0.00700	0.00590
	(b2)	0.03293	0.03218	0.02825	0.03077	0.02837	0.00805	0.02839	0.00816	0.00717	0.00822	0.00783	0.00612	0.00786	0.00702	0.00592
	(b3)	0.03293	0.03218	0.02825	0.03077	0.02836	0.00803	0.02837	0.00814	0.00715	0.00820	0.00727	0.00609	0.00728	0.00619	0.00589

Table 5.4: Cumulative univariate ruin probabilities of the form $\Gamma_{100}(10,1)$

		$h=1$			$h=2$			$h=3$			$h=4$			$h=5$		
Distr.	Model	$p=0$	$p=0.5$	$p=1$	$p=0$	$p=0.5$	$p=1$	$p=0$	$p=0.5$	$p=1$	$p=0$	$p=0.5$	$p=1$	$p=0$	$p=0.5$	$p=1$
(1)	(a1)	0.37517	0.36978	0.36426	0.36696	0.36283	0.35855	0.36106	0.35761	0.35402	0.35631	0.35329	0.35015	0.35221	0.34952	0.34671
	(a2)	0.37517	0.36978	0.36426	0.37049	0.36568	0.36076	0.36599	0.36166	0.35723	0.36177	0.35782	0.35378	0.35776	0.35415	0.35046
	(a3)	0.37517	0.36978	0.36426	0.36542	0.36157	0.35756	0.35825	0.35525	0.35210	0.35256	0.35008	0.34748	0.34782	0.34571	0.34351
	(b1)	0.50791	0.48951	0.46870	0.48760	0.47147	0.45313	0.47137	0.45641	0.44022	0.45791	0.44407	0.42935	0.44463	0.43295	0.41976
	(b2)	0.50791	0.48951	0.46870	0.48760	0.47166	0.45353	0.47183	0.45694	0.44083	0.45863	0.44485	0.43018	0.44547	0.43379	0.42058
	(b3)	0.50791	0.48951	0.46870	0.48760	0.47138	0.45305	0.47122	0.45611	0.43975	0.45768	0.44370	0.42883	0.44378	0.43223	0.41924
(2)	(a1)	0.21538	0.20844	0.20133	0.20488	0.19958	0.19409	0.19732	0.19294	0.18838	0.19130	0.18747	0.18348	0.18612	0.18270	0.17913
	(a2)	0.21538	0.20844	0.20133	0.20945	0.20328	0.19696	0.20369	0.19817	0.19254	0.19835	0.19331	0.18818	0.19328	0.18867	0.18398
	(a3)	0.21538	0.20844	0.20133	0.20286	0.19794	0.19281	0.19371	0.18989	0.18590	0.18649	0.18334	0.18005	0.18047	0.17780	0.17501
	(b1)	0.38414	0.36078	0.33434	0.35846	0.33790	0.31451	0.33767	0.31868	0.29813	0.32060	0.30303	0.28432	0.30375	0.28892	0.27218
	(b2)	0.38414	0.36078	0.33434	0.35846	0.33815	0.31503	0.33826	0.31937	0.29891	0.32153	0.30403	0.28539	0.30483	0.29000	0.27322
	(b3)	0.38414	0.36078	0.33434	0.35846	0.33779	0.31441	0.33747	0.31830	0.29753	0.32031	0.30255	0.28366	0.30266	0.28801	0.27151
(3)	(a1)	0.25044	0.24395	0.23731	0.24058	0.23559	0.23042	0.23347	0.22929	0.22495	0.22772	0.22408	0.22028	0.22278	0.21953	0.21615
	(a2)	0.25044	0.24395	0.23731	0.24484	0.23903	0.23310	0.23945	0.23420	0.22885	0.23434	0.22957	0.22469	0.22951	0.22515	0.22070
	(a3)	0.25044	0.24395	0.23731	0.23871	0.23406	0.22923	0.23006	0.22642	0.22262	0.22318	0.22018	0.21705	0.21745	0.21491	0.21226
	(b1)	0.41091	0.38877	0.36371	0.38648	0.36705	0.34496	0.36705	0.34892	0.32931	0.35070	0.33396	0.31613	0.33458	0.32046	0.30452
	(b2)	0.41091	0.38877	0.36371	0.38648	0.36729	0.34545	0.36759	0.34957	0.33005	0.35157	0.33490	0.31713	0.33561	0.32148	0.30550
	(b3)	0.41091	0.38877	0.36371	0.38648	0.36694	0.34487	0.36686	0.34856	0.32874	0.35043	0.33350	0.31551	0.33356	0.31960	0.30389

Table 5.5: Expected values of total dividends by 100 time units or before ruin occurs, whichever happens first

		$h=1$			$h=2$			$h=3$			$h=4$			$h=5$		
Distr.	Model	$p=0$	$p=0.5$	$p=1$	$p=0$	$p=0.5$	$p=1$	$p=0$	$p=0.5$	$p=1$	$p=0$	$p=0.5$	$p=1$	$p=0$	$p=0.5$	$p=1$
(1)	(a1)	160.867	161.059	161.255	159.983	160.183	160.389	159.137	159.325	159.521	158.262	158.438	158.622	157.345	157.507	157.677
	(a2)	160.867	161.059	161.255	160.718	160.851	160.987	160.339	160.429	160.521	159.775	159.836	159.897	159.061	159.098	159.136
	(a3)	160.867	161.059	161.255	159.648	159.877	160.117	158.411	158.646	158.899	157.134	157.369	157.629	155.824	156.057	156.317
	(b1)	182.156	183.569	185.101	182.849	183.976	185.196	183.154	184.064	185.013	183.120	183.875	184.646	182.968	183.546	184.156
	(b2)	182.156	183.569	185.101	182.849	183.982	185.213	183.163	184.084	185.048	183.144	183.912	184.700	183.019	183.607	184.228
	(b3)	182.156	183.569	185.101	182.849	183.973	185.183	183.144	184.048	184.989	183.091	183.838	184.602	182.934	183.495	184.086
(2)	(a1)	204.774	205.048	205.328	203.729	204.004	204.287	202.711	202.960	203.219	201.635	201.868	202.111	200.505	200.719	200.943
	(a2)	204.774	205.048	205.328	204.653	204.845	205.041	204.225	204.353	204.483	203.545	203.635	203.725	202.675	202.734	202.793
	(a3)	204.774	205.048	205.328	203.308	203.618	203.943	201.798	202.104	202.434	200.215	200.521	200.857	198.587	198.888	199.225
	(b1)	231.375	233.233	235.250	232.310	233.816	235.449	232.816	234.015	235.266	232.824	233.824	234.854	232.681	233.448	234.265
	(b2)	231.375	233.233	235.250	232.310	233.821	235.466	232.825	234.037	235.306	232.850	233.865	234.912	232.740	233.520	234.350
	(b3)	231.375	233.233	235.250	232.310	233.812	235.432	232.805	233.997	235.239	232.788	233.779	234.795	232.643	233.388	234.175
(3)	(a1)	194.479	194.717	194.962	193.432	193.680	193.937	192.432	192.667	192.911	191.399	191.617	191.844	190.311	190.510	190.719
	(a2)	194.479	194.717	194.962	194.311	194.481	194.653	193.873	193.991	194.111	193.214	193.294	193.376	192.371	192.422	192.473
	(a3)	194.479	194.717	194.962	193.032	193.314	193.610	191.562	191.852	192.164	190.046	190.334	190.652	188.485	188.769	189.086
	(b1)	219.943	221.666	223.536	220.803	222.180	223.672	221.169	222.309	223.499	221.184	222.128	223.092	221.049	221.771	222.535
	(b2)	219.943	221.666	223.536	220.803	222.187	223.690	221.180	222.332	223.539	221.212	222.171	223.157	221.109	221.842	222.617
	(b3)	219.943	221.666	223.536	220.803	222.176	223.656	221.158	222.291	223.472	221.149	222.085	223.042	221.010	221.712	222.453

Table 5.6: Standard deviations of total dividends by 100 time units or before ruin occurs, whichever happens first

		$h=1$			$h=2$			$h=3$			$h=4$			$h=5$		
Distr.	Model	$p=0$	$p=0.5$	$p=1$	$p=0$	$p=0.5$	$p=1$	$p=0$	$p=0.5$	$p=1$	$p=0$	$p=0.5$	$p=1$	$p=0$	$p=0.5$	$p=1$
(1)	(a1)	113.563	113.315	113.061	112.257	112.130	111.998	111.177	111.104	111.028	110.178	110.134	110.088	109.205	109.179	109.152
	(a2)	113.563	113.315	113.061	112.826	112.618	112.407	112.002	111.824	111.643	111.119	110.962	110.803	110.183	110.045	109.904
	(a3)	113.563	113.315	113.061	111.955	111.867	111.776	110.585	110.572	110.564	109.330	109.359	109.399	108.141	108.198	108.268
	(b1)	139.049	138.725	138.330	137.922	137.590	137.199	136.831	136.501	136.131	135.801	135.470	135.102	134.796	134.482	134.125
	(b2)	139.049	138.725	138.330	137.922	137.606	137.234	136.864	136.545	136.186	135.858	135.535	135.178	134.871	134.560	134.203
	(b3)	139.049	138.725	138.330	137.922	137.581	137.182	136.812	136.470	136.087	135.763	135.420	135.043	134.726	134.408	134.052
(2)	(a1)	89.729	89.182	88.618	88.099	87.752	87.391	86.817	86.575	86.323	85.701	85.513	85.318	84.653	84.503	84.346
	(a2)	89.729	89.182	88.618	88.611	88.170	87.717	87.473	87.115	86.750	86.361	86.056	85.744	85.251	84.989	84.723
	(a3)	89.729	89.182	88.618	87.793	87.495	87.184	86.269	86.099	85.925	84.972	84.871	84.771	83.804	83.749	83.700
	(b1)	117.964	116.725	115.291	115.899	114.765	113.470	114.011	113.003	111.898	112.431	111.495	110.489	110.943	110.124	109.219
	(b2)	117.964	116.725	115.291	115.899	114.787	113.515	114.059	113.062	111.967	112.510	111.578	110.576	111.036	110.216	109.305
	(b3)	117.964	116.725	115.291	115.899	114.753	113.449	113.986	112.962	111.841	112.387	111.435	110.414	110.850	110.036	109.137
(3)	(a1)	96.938	96.505	96.060	95.436	95.168	94.889	94.235	94.048	93.853	93.159	93.019	92.874	92.140	92.031	91.919
	(a2)	96.938	96.505	96.060	95.952	95.597	95.232	94.931	94.633	94.329	93.895	93.640	93.381	92.845	92.626	92.404
	(a3)	96.938	96.505	96.060	95.139	94.915	94.683	93.680	93.561	93.440	92.401	92.343	92.290	91.233	91.215	91.206
	(b1)	124.285	123.339	122.243	122.510	121.646	120.656	120.923	120.117	119.233	119.499	118.747	117.928	118.151	117.488	116.751
	(b2)	124.285	123.339	122.243	122.510	121.666	120.698	120.966	120.171	119.296	119.569	118.823	118.015	118.237	117.574	116.830
	(b3)	124.285	123.339	122.243	122.510	121.635	120.638	120.901	120.080	119.180	119.457	118.691	117.865	118.067	117.406	116.676

Chapter 6

Conclusions and Future Research Ideas

The main goal of this thesis is to derive algorithmic formulas for computing particular ruin-related performance measures in our generally-defined DISAM class. Although we have also displayed the versatility of our class with specific model examples in the previous chapter, our focus is primarily on model class set-up and algorithmic derivation. And so, we consider the results presented here as merely an introduction to a vast landscape of future research.

For instance, the examples given in the previous chapter could be further examined. Optimization studies could be made, possibly resulting in establishing general criteria regarding how the parameters in these models should be selected for achieving optimal values of particular performance measures. Questions such as "*Does increasing h consistently cause $\Psi_{\tau,i,j}(u,d)$ to decrease and vice-versa?*" and "*Can we optimize $\widetilde{\mathcal{D}}^r_m(u,d)$ without significantly increasing the value of $\Psi_{\tau,i,j}(u,d)$?*" can be addressed.

Another area of study might involve changing the behaviour of the dividend counter

D_t, other than those considered in the previous chapter. For instance, what would the results look like for a model with a dividend counter, which is one-unit decreasing once it reaches h, counting back down to 1 and then increasing again? There may also be interest in models where D_t increments by more than one unit, or possibly even non-linearly. For example, it may be desirable to have D_t jump by two units once U_t reaches or crosses level b and continues to do so until it reaches h. There are a myriad of possible models that can be explored by just altering the nature of the dividend counter D_t. Of course, the functions for which we made specific assumptions on the behaviour of D_t (i.e., $\tilde{v}_{u,d}(t)$ and $\tilde{v}_{u,d}(k,n)$) would need to be examined again.

Another performance measure often encountered in the literature (but excluded from our analysis) is the calculation of the expected (i.e., $r = 1$) discounted total dividends prior to ruin (e.g, see Jie-hua and Wei (2010) for their analysis of a compound binomial DISAM under stochastic interest rates). In our DISAM class, the main challenge in calculating this performance measure stems from the random nature of the individual dividend amounts. This problem becomes feasible if dividend amounts are deterministic or Parisian in structure. In these two cases, dividend amounts can be individually discounted. Note, however, that this becomes more complicated for $r > 1$ since the summation of the individual amounts will no longer be linear.

There may also be further generalizations and structural changes that can be made to the model class. For example, it might be of interest to explore what happens if level b varies according to time. For instance, if the surplus process has been above the threshold level for an extended period of time, it may be desirable to consistently increase the threshold level b as the insurance company may be more confident in issuing dividends when its

surplus is at higher values. This may also be explored under the opposite situation: if the surplus process has been below level b for a substantially extended period of time, it may be desirable to consistently decrease the threshold level b, as shareholder satisfaction may become of significant importance (e.g., failing to pay dividends exposes the insurance company to reputation decline, which may lead to significant loss of business). Finding the floor value level b can take will depend on how much risk of ruin an insurance company would be willing to tolerate (e.g., $\min\{b \in \mathbb{N} : Pr\{T < \infty\} \leq q\}$ for $0 < q < 1$). Alongside a varying threshold level, it may also be of interest to explore what happens if a barrier (or cap) were placed on the surplus process, beyond which any excess is taken away or perhaps paid as an additional dividend. This has a realistic appeal since, as mentioned in Chapter 2, infinite reserves are generally characterized as unrealistic (e.g., see Seal (1969), p. 122).

Another direction one may pursue is developing a continuous-time analogue for the DISAM class, alongside the corresponding performance measures derived in this thesis, and seeing, through the use of numerical examples, how well the discrete framework can be used to approximate its continuous counterpart. This type of investigation was explored, for instance, in the last numerical example of Alfa and Drekic (2007) under their simple DISAM.

On the applicability side, the parameters used and distributions chosen in our examples are constrained by both the computational power at our disposal and our software writing ability. We produced our results using C++. Although we enhanced the speed of our calculations through dynamic programming methods such as multi-threading and function *memoization* (i.e., functions that store previous calculations), it is very likely that software engineers can further enhance the computational speed of our algorithms. By doing so, it

would be more feasible to calculating values of performance measures that are constrained by computational speed such as: (1) large values of τ to calculate $\Gamma_\tau(u,d)$, (2) calculating the *tail value of risk* at particular percentiles (i.e., $E[T|T > VaR_q(T)]$ for some percentile q, where $VaR_q(T) = \min\{t \in \mathbb{Z}^+ : Pr\{T > t\} \leq 1-q\}$), and (3) working with distributional models that have unrestricted dividends covering a larger range of values (e.g., the binomial distribution with $c_1 = 0$ and $c_2 = 30$).

Furthermore, another possible area of exploration is to find out if the finite-ruin time based Gerber-Shiu function approach of Chapter 4 is consistently faster than the one based on MAMs from Chapter 3. A situation we foresee when MAMs might significantly outperform the Gerber-Shiu approach is if matrix calculations were to be made under the *Compute Unified Device Architecture* (a.k.a. CUDA) framework, which is exceptionally effective in parallelizing matrix manipulation routines (e.g., see Cook (2013)).

In conclusion, we find that incorporating algorithmic analyses in insurance risk models is the natural next step in the ruin theoretic literature. As societies evolve, and as the needs of each of them become more complex, so must the necessary mechanisms and tools used, that keep them functioning and thriving, do the same. Although insurance risk models only are a minor part of the picture in assessing the financial health of insurance companies, their increasing complexities are nonetheless coming to play a more crucial role in effectively evaluating the well-being of these necessary agents of society. Thus, it is of the utmost interest to further generalize these models and take them to the furthest frontier of applicability possible.

References

Soohan Ahn, Joseph H.T. Kim, and Vaidyanathan Ramaswami. A new class of models for heavy tailed distributions in finance and insurance risk. *Insurance: Mathematics and Economics*, 51(1):43–52, 2012.

Hansjörg Albrecher and Onno J. Boxma. On the discounted penalty function in a Markov-dependent risk model. *Insurance: Mathematics and Economics*, 37(3):650–672, 2005.

Hansjörg Albrecher, Eric C.K. Cheung, and Stefan Thonhauser. Randomized observation periods for the compound Poisson risk model: Dividends. *ASTIN Bulletin*, 41(2):645–672, 2011.

Attahiru S. Alfa and Steve Drekic. Algorithmic analysis of the Sparre Andersen model in discrete time. *ASTIN Bulletin*, 37(2):293–317, 2007.

Attahiru S. Alfa and Marcel F. Neuts. Modelling vehicular traffic using the discrete time Markovian arrival process. *Transportation Science*, 29(2):109–117, 1995.

Simon Anastasiadis and Stefanka Chuvoka. Multivariate insurance models: An overview. *Insurance: Mathematics and Economics*, 51(1):222–227, 2012.

Benjamin Avanzi. Strategies for dividend distribution: A review. *North American Actuarial Journal*, 13(2):217–251, 2009.

Benjamin Avanzi and Bernard Wong. On a mean reverting dividend strategy with Brownian motion. *Insurance: Mathematics and Economics*, 51(2):229–238, 2012.

Benjamin Avanzi, Eric C.K. Cheung, Bernard Wong, and Jae-Kyung Woo. On a periodic dividend barrier strategy in the dual model with continuous monitoring of solvency. *Insurance: Mathematics and Economics*, 52(1):98–113, 2013.

Kenneth Baclawski. *Introducation to Probability with R.* Chapman & Hall/CRC, Boca Raton, 2008.

Andrei L. Badescu and David Landriault. Applications of fluid flow matrix analytic methods in ruin theory. *Revista de la Real Academia de Ciencias Exactas, Fisicas y Naturales. Serie A. Matematicas*, 103(2):353–372, 2009.

Zhen-hua Bao. A note on the compound binomial model with randomized dividend strategy. *Applied Mathematics and Computation*, 194(1):276–286, 2007.

Benjamin Baumgartner and Riccardo Gatto. A bootstrap test for the probability of ruin in the compound Poisson risk process. *ASTIN Bulletin*, 40(1):241–245, 2010.

Patrick Billingsley. *Probability and Measure (3rd edition).* John Wiley & Sons, Inc., New York, 1995.

Lijun Bo, Renming Song, Dan Tang, Yongjin Wang, and Yuewei Yang. Lévy risk model with two-sided jumps and a barrier dividend strategy. *Insurance: Mathematics and Economics*, 50(2):280–291, 2012.

Lijun Bo, Renming Song, Dan Tang, Yongjin Wang, and Yuewei Yang. Erratum to "Lévy risk model with two-sided jumps and a barrier dividend strategy". *Insurance: Mathematics and Economics*, 52(1):124–125, 2013.

Shixue Cheng, Hans U. Gerber, and Elias S.W. Shiu. Discounted probabilities and ruin theory in the compound binomial model. *Insurance: Mathematics and Economics*, 26(2–3):239–250, 2000.

Marc Chesney and Monique Jean-Picqué. Brownian excursions and Parisian barrier options. *Advances in Applied Probability*, 29(1):165–184, 1997.

Ralf Christ and Josef Steinebach. Estimating the adjustment coefficient in an $ARMA(p, q)$ risk model. *Insurance: Mathematics and Economics*, 17(2):149–161, 1995.

Shane Cook. *CUDA Programming: A Developer's Guide to Parallel Computing with GPUs*. Elsevier Inc., Waltham, 2013.

Hélène Cossette, David Landriault, and Étienne Marceau. Compound binomial risk model in a Markovian environment. *Insurance: Mathematics and Economics*, 35(2):425–443, 2004.

Hélène Cossette, David Landriault, and Étienne Marceau. Ruin probabilities in the discrete time renewal risk model. *Insurance: Mathematics and Economics*, 38(2):309–323, 2006.

Hélène Cossette, Étienne Marceau, and Véronique Maume-Deschamps. Discrete-time risk models on time series for count random variables. *ASTIN Bulletin*, 40(1):123–150, 2010.

Claudia Czado, Rainer Kastenmeier, Eike Christian, and Aleksey Min. A mixed copula model for insurance claims and sizes. *Scandinavian Actuarial Journal*, (4):278–305, 2012.

Angelos Dassios and Shanle Wu. On barrier strategy dividends with Parisian implementation delay for classical surplus processes. *Insurance: Mathematics and Economics*, 45(2): 195–202, 2009.

Bruno de Finetti. Su un' impastazione alternativa della teoria colletiva del rischio. *Transactions of the XVth International Congress of Actuaries*, 2:433–443, 1957.

David C.M. Dickson. Some stable algorithms in ruin theory and their applications. *ASTIN Bulletin*, 25(2):153–175, 1995.

Alfredo D. Edígio dos Reis. On the moments of ruin and recovery times. *Insurance: Mathematics and Economics*, 27(3):331–343, 2000.

Steve Drekic and Ana Maria Mera. Ruin analysis of a threshold strategy in a discrete-time Sparre Andersen model. *Methodology and Computing in Applied Probability*, 13(4): 723–747, 2011.

Steve Drekic, David C.M. Dickson, David Stanford, and Gordon E. Willmot. On the distribution of the deficit at ruin when claims are phase-type. *Scandinavian Actuarial Journal*, (2):105–120, 2004.

François Dufresne and Hans U. Gerber. Three methods to calculate the probability of ruin. *ASTIN Bulletin*, 19(1):71–90, 1989.

Charles H. Edwards. *The Historical Development of Calculus*. Springer-Verlag Inc., New York, 1979.

José Garrido and Xiaowen Zhou. A finite-time Gerber-Shiu function. *Department of Mathematics and Statistics, Concordia University, Working paper*, 2010.

Hans U. Gerber. Mathematical fun with the compound binomial process. *ASTIN Bulletin*, 18(2):161–168, 1988.

Hans U. Gerber and Elias S.W. Shiu. On the time value of ruin. *North American Actuarial Journal*, 2(1):48–78, 1998.

Jan Grandell. *Mixed Poisson Processes*. Chapman & Hall, London, 1997.

Xie Jie-hua and Zou Wei. A risk model with paying dividends and random environment. *Insurance: Mathematics and Economics*, 42(2):717–726, 2008.

Xie Jie-hua and Zou Wei. Expected present value of total dividends in a delayed claims risk model under stochastic interest rates. *Insurance: Mathematics and Economics*, 46(2): 415–422, 2010.

Samuel Karlin and Howard E. Taylor. *A First Course in Stochastic Processes (2nd edition)*. Academic Press, New York, 1975.

Alexey Kuznetsov and Manuel Morales. Computing the finite-time expected discounted penalty function for a family of Lévy risk processes. *Scandinavian Actuarial Journal*, 31 pages, in press.

David Landriault. Randomized dividends in the compound binomial model with a general premium rate. *Scandanavian Actuarial Journal*, (1):1–15, 2008.

David Landriault. Interview, December 2011.

Shuanming Li. The time of recovery and the maximum severity of ruin in a Sparre Andersen model. *North American Actuarial Journal*, 12(4):413–427, 2008.

Shuanming Li, Yi Lu, and José Garrido. A review of discrete-time risk models. *Royal Spanish Academy of Sciences, Series A: Mathematics*, 103(2):321–337, 2009.

X. Sheldon Lin and Kristina P. Sendova. The compound Poisson risk model with multiple thresholds. *Insurance: Mathematics and Economics*, 42(2):617–627, 2008.

Éva O. Mihálykó and Csaba Mihálykó. Mathematical investigation of the Gerber-Shiu function in the case of dependent inter-claim time and claim size. *Insurance: Mathematics and Economics*, 48(3):378–383, 2011.

Ilie-Radu Mitric, Andrei L. Badescu, and David A. Stanford. On the absolute ruin problem in a Sparre Andersen risk model with constant interest. *Insurance: Mathematics and Economics*, 50(1):167–178, 2012.

Marcel F. Neuts. *Matrix-Geometric Solutions in Stochastic Models: An Algorithmic Approach*. Johns Hopkins University Press, Baltimore, 1981.

Kristina P. Pavlova and Gordon E. Willmot. The discrete stationary renewal risk model and the Gerber-Shiu discounted penalty function. *Insurance: Mathematics and Economics*, 35(2):267–277, 2004.

Sidney I. Resnick. *A Probability Path*. Birkhäuser, Boston, 1999.

Hilary Seal. *Stochastic Theory of a Risk Business*. John Wiley & Sons, Inc., New York, 1969.

David A. Stanford, Krzysztof J. Stroiński, and Karen Lee. Ruin probabilities based at claim instants for some non-Poisson claim processes. *Insurance: Mathematics and Economics*, 26(2–3):251–267, 2000.

David A. Stanford, Kaiqi Yu, and Jiandong Ren. Erlangian approximation to finite time ruin probabilities in perturbed risk models. *Scandinavian Actuarial Journal*, (1):38–58, 2011.

Jiyang Tan and Xiangqun Yang. The compound binomial model with randomized decisions on paying dividends. *Insurance: Mathematics and Economics*, 39(1):1–18, 2006.

Qihe Tang and Zhongyi Yuan. A hybrid estimate for the finite-time ruin probability in a bivariate autoregressive model with application to portfolio optimization. *North American Actuarial Journal*, 16(3):378–397, 2012.

Oliver Westall. *The Historian and the Business of Insurance*. Manchester University Press, Manchester, 1984.

Gordon E. Willmot. A note on a class of delayed renewal risk processes. *Insurance: Mathematics and Economics*, 34(2):251–257, 2004.

Gordon E. Willmot. Interview, December 2011.

Gordon E. Willmot and Jae-Kyung Woo. On the analysis of a general class of dependent risk processes. *Insurance: Mathematics and Economics*, 51(1):134–141, 2012.

Xueyuan Wu and Shuanming Li. On a discrete-time Sparre Andersen model with phase-type claims. *Department of Economics, University of Melbourne, Working paper 08-169*, 2008.

Xueyuan Wu and Shuanming Li. Matrix-form recursions for a family of compound distributions. *ASTIN Bulletin*, 40(1):351–368, 2010.

K.C. Yuen and J.Y. Guo. Ruin probabilities for time-correlated claims in the compound binomial model. *Insurance: Mathematics and Economics*, 29(1):47–57, 2001.

Appendix: C++ Code

Here we provide the computer code used to produce the numerical results in Chapter 5. We remark that this Appendix will only be provided during the thesis display period. For the final version of the thesis, we will include a software component in the form of a CD.

```
//0GeneralModel.cpp
#include "1ModelClass.h"
#include "2PreliminaryFunctions.h"

#include <cmath> //Incorporate ceiling and floor functions
#include <ctime> //Time functions
#include <fstream>//To export results to a CSV file
#include <iomanip>
#include <iostream>
#include <omp.h>//For parallel programming
#include <vector> //Vector and matrix algebra
using namespace std;
////////////////////////////////////////////////////////////////////////////
string ModelTypeString(int Type)
{
    switch(Type)
    {
case 0: return "CNTDWN";
break;
case 1: return "FRZN";
break;
default: return "STOVR";
    }
}
////////////////////////////////////////////////////////////////////////////
string DelayedTypeString(int Type)
{
    switch(Type)
    {
case 0: return "Del";
break;
case 1: return "Ord";
break;
default: return "Sta";
    }
```

```
}
////////////////////////////////////////////////////////////////////////////
int main()
{
cout<<"Welcome to Basil's Program!!"<<endl<<endl;

    int t1=0, t2=0, t3=0, t4=0;//Timing variables to see how fast segments of code run
    Model *Ut;
    int u=10, c=5, b=50, c1=2, c2=4, l0=1, nr=10, na=10, n=100, ED=10, ES=10;
    double *a=new double[na], *r=new double[nr],*d=new double[c2-c1+1];

cout<<"n= ";
cin>>n;
    for (int i=0; i<na; ++i) a[i]=1/double(na);//subsequent interclaim time pmf

    d[0]=0.4; d[1]=0.2; d[2]=0.4;

    double Def, Sur, Tri, Uni, ExpT, STDT;
matrix<double> DefMat(18,15), SurMat(18,15), TriMat(18,15), UniMat(18,15), ExpTMat(18,15), STDTMat(18,15);
    t1=clock();
    ////////////////////////////////////////////////////////////////////////////
int proc, numThreads=omp_get_num_procs();

cout<<"There are "<<numThreads<<" threads :)"<<endl;
cout<<"The maximum number of threads you can use here is"<<min(15,numThreads)<<endl;
cout<<"Enter number of threads you want to use: ";
cin>>proc;
while(proc<=0 || proc>min(15,numThreads))
{
cout<<"Oops! "<<proc<<" isn't valid"<<endl;
cout<<"Enter number of threads you want to use: ";
cin>>proc;
}
omp_set_num_threads(proc);
cout<<"This program will use "<<proc<<" threads"<<endl<<endl;
```

```
    //Numerical output part


for(int row=0; row<18; ++row)
    {
int DelayedType=floor(row/6.0);
int ModelType=row%3;
        #pragma omp parallel for schedule (dynamic)
        for(int col=0; col<15; ++col)
        {
            t3=clock();
            switch(DelayedType)
{
 case 0://Delayed
 nr=1;r[0]=1;
 break;
 case 1://Ordinary
nr=10;
for(int i=0; i<nr; ++i) r[i]=1/double(nr);//first claim time ordinary pmf
break;
 default://Stationary
nr=10;
for (int i=0; i<nr; ++i) r[i]=(11-i-1)/55.0;//first claim time stationary pmf
            }
int h=floor(col/3.0)+1;
            double p=(col%3)/2.0;


            for(int Penalty=0; Penalty<=3; ++Penalty)//Da loop where all the magic is :)
            {
               Ut=new Model;//The surplus process
               Ut->ParInitialize(u,c,b,c1,c2,nr,na,h,10,p);
               Ut->DistrInitialize(r,a,d);
   Ut->ModelType=ModelType;
               Ut->PenaltyType=Penalty;
               switch(Penalty)
               {
```

```
                case 0:
   if(row%6<3) Def=Ut->GRedPrem(n,u,l0,true);
   else Def=Ut->GParisian(n,u,l0,true);

   ED=ceil(Def);
   DefMat(row,col)=Def;
       break;
                case 1:
   if(row%6<3)  Sur=Ut->GRedPrem(n,u,l0,true);
   else  Sur=Ut->GParisian(n,u,l0,true);

                     ES=ceil(Sur);
                     SurMat(row,col)=Sur;
                     break;
                case 2:
Ut->Deficit=ED;
                      Ut->Surplus=ES;
if(row%6<3) Tri=Ut->GRedPrem(n, u, l0, true);
else Tri=Ut->GParisian(n, u, l0, true);

                      TriMat(row,col)=Tri;
break;
  default:
if(row%6<3) Uni=Ut->GRedPrem(n, u, l0, true);
else Uni=Ut->GParisian(n, u, l0, true);

                      UniMat(row,col)=Uni;

if(row%6<3) ExpT=Ut->DRedPrem(n,u,l0,1,true);
else ExpT=Ut->DParisian(n,u,l0,1,true);

ExpTMat(row,col)=ExpT;//Place here since ExpT is insensitive to PenltyType.
double ExpT2;
if(row%6<3) ExpT2=Ut->DRedPrem(n,u,l0,2,true);
else ExpT2=Ut->DParisian(n,u,l0,2,true);
```

```
                        STDT=sqrt(ExpT2-pow(ExpT,2.0));
STDTMat(row,col)=STDT;//Place here since STDT is insensitive to PenltyType.
               }
   delete Ut;//Necessary for large values of n; otherwise memory capacity will finish
            }
            t4=clock();
            cout<<endl<<"Time for "<<DelayedTypeString(DelayedType)<<", "<<ModelTypeString(ModelType);
            cout<<", h="<<h<<", p="<<p<<"-->"<<double(t4-t3)/CLOCKS_PER_SEC<<" seconds."<<endl;
         }
     }
//Writing onto csv files
ofstream ExpDef, ExpSur, Triv, Univ, ExpTotD, STDTotD;
    ExpDef.open("ExpDef.csv");
    ExpSur.open("ExpSur.csv");
    Triv.open("Triv.csv");
Univ.open("Univ.csv");
    ExpTotD.open("ExpTotD.csv");
    STDTotD.open("STDTotD.csv");


for(int row=0; row<20; ++row)
{
if(row<2)
{
ExpDef<<","<<","; ExpSur<<","<<","; Triv<<","<<","; Univ<<","<<","; ExpTotD<<","<<","; STDTotD<<","<<",";
for(int col=0; col<15; ++col)
{
if(row==0)
{
int h=floor(col/3.0)+1;
ExpDef<<"h="<<h<<","; ExpSur<<"h="<<h<<","; Triv<<"h="<<h<<","; Univ<<"h="<<h<<","; ExpTotD<<"h="<<h<<","; STDTotD<<"h="<<h<<",";
}
else
{
double p=(col%3)/2.0;
```

```
ExpDef<<"p="<<p<<","; ExpSur<<"p="<<p<<","; Triv<<"p="<<p<<","; Univ<<"p="<<p<<","; ExpTotD<<"p="<<p<<","; STDTotD<<"p="<<p<<",";
}
}
}
else
{
int DelayedType=floor((row-2)/6.0);
int ModelType=(row-2)%3;

ExpDef<<DelayedTypeString(DelayedType)<<","<<ModelTypeString(ModelType)<<",";
ExpSur<<DelayedTypeString(DelayedType)<<","<<ModelTypeString(ModelType)<<",";
Triv<<DelayedTypeString(DelayedType)<<","<<ModelTypeString(ModelType)<<",";
Univ<<DelayedTypeString(DelayedType)<<","<<ModelTypeString(ModelType)<<",";
ExpTotD<<DelayedTypeString(DelayedType)<<","<<ModelTypeString(ModelType)<<",";
STDTotD<<DelayedTypeString(DelayedType)<<","<<ModelTypeString(ModelType)<<",";

for(int col=0; col<15; ++col)
            {
               ExpDef<<setprecision(13)<<DefMat(row-2,col)<<",";
   ExpSur<<setprecision(13)<<SurMat(row-2,col)<<",";
   Triv<<setprecision(15)<<TriMat(row-2,col)<<",";
   Univ<<setprecision(15)<<UniMat(row-2,col)<<",";
   ExpTotD<<setprecision(12)<<ExpTMat(row-2,col)<<",";
   STDTotD<<setprecision(12)<<STDTMat(row-2,col)<<",";

            }
}
ExpDef<<endl; ExpSur<<endl; Triv<<endl; Univ<<endl;ExpTotD<<endl; STDTotD<<endl;
}

    STDTotD.close();
    ExpTotD.close();
    Triv.close();
Univ.close();
    ExpSur.close();
```

```
ExpDef.close();
    t2=clock();

    cout<<"Total Time-->"<<double(t2-t1)/CLOCKS_PER_SEC<<" seconds."<<endl;
system("pause");
    return 0;
}
////////////////////////////////////////////////////////////////////////////
////////////////////////////////////////////////////////////////////////////

//1ModelClass.h
#include <boost/numeric/ublas/io.hpp>
#include <boost/numeric/ublas/matrix.hpp>
#include <boost/numeric/ublas/vector.hpp>

#include <cmath> //Incorporate ceiling and floor functions
#include <ctime> //Time functions
#include <iostream>
#include <vector> //Vector and matrix algebra

using namespace boost::numeric::ublas;
using namespace std;
////////////////////////////////////////////////////////////////////////////
class Model
{
    void MemoizeSetupParameters();
void MemoizeSetupStorage();

void StorageSet(std::vector<std::vector<std::vector<std::vector<double> > > >&,int,int,int,int,int,int,int,int);//f

void StorageSet(std::vector<std::vector<std::vector<std::vector<double> > > >&,int,int,int,int,int,int);//Gerber-Shiu
void StorageSet(std::vector<std::vector<std::vector<std::vector<std::vector<double> > > > >&,int,int,int,int,int,int,int,int);//Dividend Moments

int fMaxuStorage, fMaxlOStorage, fMaxnStorage, fMaxsStorage;
std::vector<std::vector<std::vector<std::vector<double> > > > fStorage;
```

```
void fStorageExpand(int, int, int, int);

int GRedPremMaxmStorage, GRedPremMaxuStorage, GRedPremMaxlStorage;
std::vector<std::vector<std::vector<std::vector<double> > > > GRedPremStorage;
void GRedPremStorageExpand(int,int,int,bool);

int DRedPremMaxmStorage, DRedPremMaxuStorage, DRedPremMaxlStorage, DRedPremMaxrStorage;
std::vector<std::vector<std::vector<std::vector<std::vector<double> > > > > DRedPremStorage;
void DRedPremStorageExpand(int,int,int,int,bool);

int GParisianMaxmStorage, GParisianMaxuStorage, GParisianMaxlStorage;
std::vector<std::vector<std::vector<std::vector<double> > > > GParisianStorage;
void GParisianStorageExpand(int,int,int,bool);

int DParisianMaxmStorage, DParisianMaxuStorage, DParisianMaxlStorage, DParisianMaxrStorage;
std::vector<std::vector<std::vector<std::vector<std::vector<double> > > > > DParisianStorage;
void DParisianStorageExpand(int,int,int,int,bool);


    public:
int u, c, b, c1, c2, nr, na, h, l0;
    double p, *del, *ord, *dd, vv, ExpY;
int lines;//Temporary just use for checking code
    int ModelType;
int PenaltyType;
int Surplus, Deficit;

Model();
~Model();
    void ParInitialize(int,int,int,int,int,int,int,int,int,double);
    void DistrInitialize(double[], double[], double[]);//Waiting time and Dividend Payment
double r(int);
double a(int);
double d(int);
double dstar(int,int);
    double alpha(int);
```

```
double Lambda(int);
    double R(int);
double A(int);

    //Nomenclature Functions
    int z(int);
int zstar(int,int);
double f(int,int,int,int);

double GRedPrem(int,int,int,bool);
double DRedPrem(int,int,int,int,bool);

double GParisian(int,int,int,bool);
double DParisian(int,int,int,int,bool);
};
////////////////////////////////////////////////////////////////////////////
////////////////////////////////////////////////////////////////////////////
//1ModelClass.cpp
//#define _SCL_SECURE_NO_WARNINGS//Removes warnings
#include "1ModelClass.h"
#include "2PreliminaryFunctions.h"

////////////////////////////////////////////////////////////////////////////
//Functions of Model
Model::Model()
{
   u=2;c=2;b=5;c1=1;c2=2;nr=10;na=10;h=2;l0=1;p=1;vv=1;ModelType=2;PenaltyType=3;Surplus=10;Deficit=10;
   del=new double[nr]; ord=new double[na]; dd=new double[c2-c1+1];
   MemoizeSetupParameters();

   for(int i=0; i<c2-c1+1; i++) dd[i]=1/double(c2-c1+1);
   for(int i=0; i<na; i++)
   {
   del[i]=1/double(nr); ord[i]=1/double(na);
   }
```

```
   ExpY=5.5;
}

void Model::ParInitialize(int uu, int cc, int bb, int cc1, int cc2, int nnr, int nna, int hh, int ll0, double pp)
{
b=bb;
u=uu;c=cc;c1=cc1;c2=cc2;nr=nnr;na=nna;h=hh;l0=ll0;p=pp;
std::vector<double> temp;

ExpY=10.511104947394237;
MemoizeSetupStorage();
}

void Model::DistrInitialize(double rr[], double aa[], double ddd[])//Waiting time and Dividend Payment
{
   del=new double[nr]; ord=new double[na]; dd=new double[c2-c1+1];
   for(int i=0; i<nr; i++) del[i]=rr[i];
   for(int i=0; i<na; i++) ord[i]=aa[i];
   for(int i=0; i<c2-c1+1; i++) dd[i]=ddd[i];
}

double Model::r(int x)
{
if(x>=1 && x<=nr) return del[x-1];
else return 0;
}

double Model::a(int x)
{
if(x>=1 && x<=na) return ord[x-1];
else return 0;
}

double Model::d(int x)
{
```

```
   if (x<c1 || x>c2) return 0;
   else return dd[x-c1];
}

double Model::dstar(int l,int x)
{
  if(l>=min(c1,max(x-b,0))&& l<min(c2,max(x-b,0))) return d(l);
  else if (l==min(c2,max(x-b,0)))
  {
       double sum=0;
       for(int w=max(c1,min(c2,max(x-b,0)));w<=c2;w++) sum+=d(w);//The max(c1,...) part eliminates the index problem
       return sum;
   }
  else return 0;
}

double Model::alpha(int i)
{
return pow(1+(i-1)/30.0,-4)-pow(1+i/30.0,-4);
}
double Model::Lambda(int i)
{
double sum=0;
for(int j=1; j<=i; j++) sum=sum+alpha(j);
return 1-sum;
}

double Model::R(int k)
{
   double sum=0;
   if(k>=nr) return 0;
   if(k<=0) return 1;
   else
   {
       for(int i=k+1; i<=nr; i++) sum+=r(i);
```

```
       return sum;
   }
}

double Model::A(int k)
{
   double sum=0;
   if(k>=na) return 0;
   if(k<=0) return 1;
   else
   {
      for(int i=k+1; i<=na; i++) sum+=a(i);
      return sum;
   }
}

int Model::z(int uu)
{
   if(c==0)
   {
      if(uu<b) return -1;//Indicating that z(u)="Infinity";
      else return 0;
   }
   else
   {
      double u_double=uu, b_double=b, c_double=c, z_double=ceil(max(b_double-u_double,0.0)/c_double);
      return int(z_double);
   }
}

int Model::zstar(int uu, int l10)
{
    switch(ModelType)
    {
      //Countdown
```

```
      case 0://Consecutive
           return z(uu)+h-max(ll0-1-max(z(uu)-1,0),0);
      break;
      //Freeze
      case 1://Consecutive
           return z(uu)+h-(ll0-1);
      break;
      //Reset
      default://Consecutive
           if(uu<b) return z(uu)+h;
           else return h-(ll0-1);
    }
}

double Model::f(int uu, int ll0, int n, int ss)//Remember that s is used as a MAM probability std::vector
{
int zstr=zstar(uu,ll0);
if(uu<0 || n<0) return 0;

fStorageExpand(uu,ll0,n,ss);
if(fStorage[uu][ll0-1][n][ss]!=-1) return fStorage[uu][ll0-1][n][ss];
else
{
double ff;
if(n==0)
{
if(ss==0) ff=1;
else ff=0;

fStorage[uu][ll0-1][n][ss]=ff;
return ff;
}
else if(n==1)
{
ff=dstar(ss,uu+c*zstr);
```

```
fStorage[uu][ll0-1][n][ss]=ff;
return ff;
}
else
{
ff=0;
for(int w=0; w<=ss; ++w) ff=ff+dstar(ss-w,uu+c*(zstr+n-1)-w)*f(uu,ll0,n-1,w);

fStorage[uu][ll0-1][n][ss]=ff;
return ff;
}
}
}

double Model::GRedPrem(int m, int uu, int ll0, bool delayed)
{
if(m<=0 || u<0) return 0;
else
{
GRedPremStorageExpand(m,uu,ll0,delayed);
if(GRedPremStorage[m-1][uu][ll0-1][0]!=-1 && delayed==false) return GRedPremStorage[m-1][uu][ll0-1][0];
else if(GRedPremStorage[m-1][uu][ll0-1][1]!=-1 && delayed==true) return GRedPremStorage[m-1][uu][ll0-1][1];
else
{
int min_n_m;

if(delayed==true) min_n_m=min(nr,m);
else min_n_m=min(na,m);


int zz=z(uu), zstr=zstar(uu,ll0);
double sum=0;
for(int k=1; k<=min_n_m; ++k)//First summation
{
```

```
double x;
if(delayed==true) x=del[k-1];
else x=ord[k-1];
if(k<zstr)//If k is not a dividend paying instant
{
for(int jj=1; jj<=uu+c*k; ++jj)
{
int lll;
switch(ModelType)
{
case 0:
if(k<zz||(k==zz && jj>uu+c*zz-b)) lll=max(1,ll0-k);
else if (k==zz && jj<=uu+c*zz-b) lll=max(1,ll0-max(zz-1,0));
else lll=max(1,ll0-max(zz-1,0))+(k-zz)-I(jj>uu+c*k-b);
break;
case 1:
if(k<=zz) lll=ll0;
else lll=ll0+(k-zz)-I(jj>uu+c*k-b);
break;
default:
if(k<=zz||jj>uu+c*k-b) lll=1;
else lll=ll0+(k-zz);
}
sum=sum+alpha(jj)*GRedPrem(m-k,uu+c*k-jj,lll,false)*x;
}
switch(PenaltyType)
{
case 0:
sum=sum+(ExpY-(uu+c*k)*Lambda(uu+c*k))*x;
for(int jj=1; jj<=uu+c*k; ++jj) sum=sum-jj*alpha(jj)*x;
break;
case 1: sum=sum+Lambda(uu+c*k)*(uu+c*k)*x;
break;
case 2:
if(uu+c*k<=Surplus) sum=sum+(Lambda(uu+c*k)-Lambda(uu+c*k+Deficit))*x;
```

```
break;
default:sum=sum+Lambda(uu+c*k)*x;
}

}
else//If k is a dividend paying instant.
{
for(int ss=(k-zstr)*c1; ss<=(k-zstr)*c2; ++ss)
{
if(p!=0)
{
for(int jj=1; jj<=uu+c*k-ss; ++jj)
{
if(ModelType==2 && jj>uu+c*k-ss-b) sum=sum+f(uu,ll0,max(k-zstr,0),ss)*p*alpha(jj)*GRedPrem(m-k,uu+c*k-ss-jj,1,false)*x;
else
{
for(int sstar=min(c1,max(uu+c*k-ss-jj-b,0)); sstar<=min(c2,max(uu+c*k-ss-jj-b,0)); ++sstar)
sum=sum+f(uu,ll0,max(k-zstr,0),ss)*p*alpha(jj)*dstar(sstar,uu+c*k-ss-jj)*GRedPrem(m-k,uu+c*k-ss-jj-sstar,h,false)*x;
}

}
switch(PenaltyType)
{
case 0:
sum=sum+f(uu,ll0,max(k-zstr,0),ss)*p*(ExpY-Lambda(uu+c*k-ss)*(uu+c*k-ss))*x;
for(int jj=1; jj<=uu+c*k-ss; ++jj) sum=sum-f(uu,ll0,max(k-zstr,0),ss)*p*jj*alpha(jj)*x;
break;
case 1: sum=sum+f(uu,ll0,max(k-zstr,0),ss)*p*Lambda(uu+c*k-ss)*(uu+c*k-ss)*x;
break;
case 2:
if(uu+c*k-ss<=Surplus) sum=sum+f(uu,ll0,max(k-zstr,0),ss)*p*(Lambda(uu+c*k-ss)-Lambda(uu+c*k-ss+Deficit))*x;
break;
default: sum=sum+f(uu,ll0,max(k-zstr,0),ss)*p*Lambda(uu+c*k-ss)*x;
}
}
```

```
if(p!=1)
{
for(int sstar=c1; sstar<=c2; ++sstar)
{
for(int jj=1; jj<=uu+c*k-ss-sstar; ++jj)
{
if(ModelType==2 && jj>uu+c*k-ss-sstar-b) sum=sum+f(uu,ll0,max(k-zstr,0),ss)*d(sstar)*(1-p)*alpha(jj)*GRedPrem(m-k,uu+c*k-ss-sstar-jj,1,false)*x;
else sum=sum+f(uu,ll0,max(k-zstr,0),ss)*d(sstar)*(1-p)*alpha(jj)*GRedPrem(m-k,uu+c*k-ss-sstar-jj,h,false)*x;
}

switch(PenaltyType)
{
case 0:
sum=sum+f(uu,ll0,max(k-zstr,0),ss)*d(sstar)*(1-p)*(ExpY-(uu+c*k-ss-sstar)*Lambda(uu+c*k-ss-sstar))*x;
for(int jj=1; jj<=uu+c*k-ss-sstar; ++jj) sum=sum-f(uu,ll0,max(k-zstr,0),ss)*d(sstar)*(1-p)*jj*alpha(jj)*x;
break;
case 1: sum=sum+f(uu,ll0,max(k-zstr,0),ss)*d(sstar)*(1-p)*Lambda(uu+c*k-ss-sstar)*(uu+c*k-ss-sstar)*x;
break;
case 2:
if(uu+c*k-ss-sstar<=Surplus) sum=sum+f(uu,ll0,max(k-zstr+1,0),ss+sstar)*(1-p)*(Lambda(uu+c*k-ss-sstar)-Lambda(uu+c*k-ss-sstar+Deficit))*x;
break;
default: sum=sum+f(uu,ll0,max(k-zstr,0),ss)*d(sstar)*(1-p)*Lambda(uu+c*k-ss-sstar)*x;
}
}
}
}
}
}
if(delayed==false) GRedPremStorage[m-1][uu][ll0-1][0]=sum;
else GRedPremStorage[m-1][uu][ll0-1][1]=sum;
return sum;
}
}
}
```

```
double Model::DRedPrem(int m, int uu, int ll0, int rr, bool delayed)
{
if(m<1 || uu<0) return 0;
else
{
DRedPremStorageExpand(m,uu,ll0,rr,delayed);
if(DRedPremStorage[m-1][uu][ll0-1][rr][0]!=-1 && delayed==false) return DRedPremStorage[m-1][uu][ll0-1][rr][0];
else if(DRedPremStorage[m-1][uu][ll0-1][rr][1]!=-1 && delayed==true) return DRedPremStorage[m-1][uu][ll0-1][rr][1];
else
{
int min_n_m;

if(delayed==true) min_n_m=min(nr,m);
else min_n_m=min(na,m);

int zz=z(uu), zstr=zstar(uu,ll0);
double sum=0;
for(int k=1; k<=min_n_m; ++k)//First summation
{
double x;
if(delayed==true) x=del[k-1];
else x=ord[k-1];


if(k<zstr)//If k is not a dividend paying instant
{
for(int jj=1; jj<=uu+c*k; ++jj)
{
int lll;
switch(ModelType)
{
case 0:
if(k<zz || (k==zz && jj>uu+c*zz-b)) lll=max(1,ll0-k);
else if (k==zz && jj<=uu+c*zz-b) lll=max(1,ll0-max(zz-1,0));
else lll=max(1,ll0-max(zz-1,0))+(k-zz)-I(jj>uu+c*k-b);
```

```
break;
case 1:
if(k<=zz) lll=ll0;
else lll=ll0+(k-zz)-I(jj>uu+c*k-b);
break;
default:
if(k<=zz||jj>uu+c*k-b) lll=1;
else lll=ll0+(k-zz);
}
sum=sum+alpha(jj)*DRedPrem(m-k,uu+c*k-jj,lll,rr,false)*x;
}
}

else///If k is a dividend paying instant. Keep in mind that neither b nor h can be infinite if this case holds
{
for(int ss=(k-zstr)*c1; ss<=(k-zstr)*c2; ++ss)
{
for(int jj=1; jj<=uu+c*k-ss; ++jj)
{
for(int sstar=min(c1,max(uu+c*k-ss-jj-b,0)); sstar<=min(c2,max(uu+c*k-ss-jj-b,0)); ++sstar)
{
for(int w=0; w<=rr; w++)
{
if(rr-w==0)
{
if(w==0) sum=sum+f(uu,ll0,k-zstr,ss)*p*alpha(jj)*dstar(sstar,uu+c*k-ss-jj)*x;
else sum=sum+f(uu,ll0,k-zstr,ss)*p*alpha(jj)*dstar(sstar,uu+c*k-ss-jj)*Binomial(rr,w)*pow(double(ss+sstar),double(w))*x;
}
else
{
int lll;
if(ModelType==2 && jj>uu+c*k-ss-b) lll=1;
else lll=h;

if(w==0) sum=sum+f(uu,ll0,k-zstr,ss)*p*alpha(jj)*dstar(sstar,uu+c*k-ss-jj)*DRedPrem(m-k,uu+c*k-ss-jj-sstar,lll,rr-w,false)*x;
```

```
else
sum=sum+f(uu,ll0,k-zstr,ss)*p*alpha(jj)*dstar(sstar,uu+c*k-ss-jj)*Binomial(rr,w)*pow(double(ss+sstar),double(w))*DRedPrem(m-k,uu+c*k-ss-jj-sstar,lll,rr-w,false)*x;
}
}
}//end of for(int jj=1; jj<=jMax; jj++)
}

if(rr==0) sum=sum+f(uu,ll0,k-zstr,ss)*p*Lambda(uu+c*k-ss)*x;
else sum=sum+f(uu,ll0,k-zstr,ss)*p*Lambda(uu+c*k-ss)*pow(double(ss),double(rr))*x;

for(int sstar=min(c1,max(uu+c*k-ss-b,0)); sstar<=min(c2,max(uu+c*k-ss-b,0)); ++sstar)
{
for(int jj=1; jj<=uu+c*k-ss-sstar; ++jj)
{
for(int w=0; w<=rr; ++w)
{
if(rr-w==0)
{
if (w==0) sum=sum+f(uu,ll0,k-zstr,ss)*(1-p)*dstar(sstar,uu+c*k-ss)*alpha(jj)*x;//Avoid the 0^0 issue
else sum=sum+f(uu,ll0,k-zstr,ss)*(1-p)*dstar(sstar,uu+c*k-ss)*alpha(jj)*Binomial(rr,w)*pow(double(ss+sstar),double(w))*x;
}
else
{
int lll;
if(ModelType==2 && jj>uu+c*k-ss-sstar-b) lll=1;
else lll=h;
if (w==0) sum=sum+f(uu,ll0,k-zstr,ss)*(1-p)*dstar(sstar,uu+c*k-ss)*alpha(jj)*Binomial(rr,w)*DRedPrem(m-k,uu+c*k-ss-sstar-jj,lll,rr-w,false)*x;//Avoid the 0^0 issue
else
sum=sum+f(uu,ll0,k-zstr,ss)*(1-p)*dstar(sstar,uu+c*k-ss)*alpha(jj)*Binomial(rr,w)*pow(double(ss+sstar),double(w))*DRedPrem(m-k,uu+c*k-ss-sstar-jj,lll,rr-w,false)*x;
}
}
}
if (rr==0) sum=sum+f(uu,ll0,k-zstr,ss)*(1-p)*dstar(sstar,uu+c*k-ss)*Lambda(uu+c*k-ss-sstar)*x;//Avoid the 0^0 issue
else sum=sum+f(uu,ll0,k-zstr,ss)*(1-p)*dstar(sstar,uu+c*k-ss)*Lambda(uu+c*k-ss-sstar)*pow(double(ss+sstar),double(rr))*x;
}
```

```
}
}
}

for(int ss=max((m-zstr+1)*c1,0); ss<=max((m-zstr+1)*c2,0); ++ss)
{
if(rr==0)//Avoid the 0^0 issue
{
if(delayed==false) sum=sum+A(m)*f(uu,ll0,max(m-zstr+1,0),ss);
else sum=sum+R(m)*f(uu,ll0,max(m-zstr+1,0),ss);
}
else
{
if(delayed==false) sum=sum+A(m)*f(uu,ll0,max(m-zstr+1,0),ss)*pow(double(ss),double(rr));
else sum=sum+R(m)*f(uu,ll0,max(m-zstr+1,0),ss)*pow(double(ss),double(rr));
}
}

if(delayed==false) DRedPremStorage[m-1][uu][ll0-1][rr][0]=sum;
else DRedPremStorage[m-1][uu][ll0-1][rr][1]=sum;
return sum;
}
}
}
////////////////////////////////////////////////////////////////////////////
double Model::GParisian(int m, int uu, int ll0, bool delayed)
{
if(m<=0 || uu<0) return 0;
else
{
GParisianStorageExpand(m,uu,ll0,delayed);
if(GParisianStorage[m-1][uu][ll0-1][0]!=-1 && delayed==false) return GParisianStorage[m-1][uu][ll0-1][0];
else if(GParisianStorage[m-1][uu][ll0-1][1]!=-1 && delayed==true) return GParisianStorage[m-1][uu][ll0-1][1];
else
{
```

```
int min_n_m;

if(delayed==true) min_n_m=min(nr,m);
else min_n_m=min(na,m);

int zz=z(uu), zstr=zstar(uu,ll0);
double sum=0;
for(int k=1; k<=min_n_m; ++k)//First summation
{

double x; bool DivTime; int vmax;
if(delayed==true) x=del[k-1];
else x=ord[k-1];

if(k>=zstr && (k-zstr)%h==0)
{
if(k==zstr) vmax=uu+c*k;
else vmax=b+c*h;
DivTime=true;
}
else
{
if(k<zstr) vmax=uu+c*k;
else vmax=b+c*((k-zstr)%h);
DivTime=false;
}

if(DivTime==false)//If k is not a dividend paying instant
{
for(int jj=1; jj<=vmax; ++jj)
{
int lll;
switch(ModelType)
{
case 0:
```

```
if(k<zz||(k==zz && jj>vmax-b)) lll=max(1,ll0-k);
else if (k==zz && jj<=vmax-b) lll=max(1,ll0-max(zz-1,0));
else if (k>zz && k<zstr) lll=max(1,ll0-max(zz-1,0))+(k-zz)-I(jj>vmax-b);
else lll=(k-zstr)%h+1-I(jj>vmax-b);
break;
case 1:
if(k<=zz) lll=ll0;
else if(k>zz && k<zstr) lll=ll0+(k-zz)-I(jj>vmax-b);
else lll=(k-zstr)%h+1-I(jj>vmax-b);
break;
default:
if(k<=zz || jj>vmax-b ||(k>=zstr && (k-zstr)%h==0)) lll=1;
else if(k>zz && k<zstr) lll=ll0+(k-zz);
else lll=(k-zstr)%h+1;
}
sum=sum+alpha(jj)*GParisian(m-k,vmax-jj,lll,false)*x;
}
switch(PenaltyType)
{
case 0:
sum=sum+(ExpY-vmax*Lambda(vmax))*x;
for(int jj=1; jj<=vmax; ++jj) sum=sum-jj*alpha(jj)*x;
break;
case 1: sum=sum+Lambda(vmax)*(vmax)*x;
break;
case 2:
if(vmax<=Surplus) sum=sum+(Lambda(vmax)-Lambda(vmax+Deficit))*x;
break;
default:sum=sum+Lambda(vmax)*x;
}


}
else//If k is a dividend paying instant.
{
if(p!=0)
```

```
{
for(int jj=1; jj<=vmax; ++jj)
{

if(jj<=vmax-b) sum=sum+p*alpha(jj)*GParisian(m-k,b,1,false)*x;
else
{
if(ModelType==2) sum=sum+p*alpha(jj)*GParisian(m-k,vmax-jj,1,false)*x;
else sum=sum+p*alpha(jj)*GParisian(m-k,vmax-jj,h,false)*x;
}
}
switch(PenaltyType)
{
case 0:
sum=sum+p*(ExpY-Lambda(vmax)*(vmax))*x;
for(int jj=1; jj<=vmax; ++jj) sum=sum-p*jj*alpha(jj)*x;//This can be merged with above!!!!!!!!!
break;
case 1: sum=sum+p*Lambda(vmax)*(vmax)*x;
break;
case 2:
if(vmax<=Surplus) sum=sum+p*(Lambda(vmax)-Lambda(vmax+Deficit))*x;
break;
default: sum=sum+p*Lambda(vmax)*x;
}
}
if(p!=1)
{
for(int jj=1; jj<=b; ++jj) sum=sum+(1-p)*alpha(jj)*GParisian(m-k,b-jj,1,false)*x;

switch(PenaltyType)
{
case 0:
sum=sum+(1-p)*(ExpY-Lambda(b)*b)*x;
for(int jj=1; jj<=b; ++jj) sum=sum-(1-p)*jj*alpha(jj)*x;
break;
```

```
case 1: sum=sum+(1-p)*Lambda(b)*b*x;
break;
case 2:
if(b<=Surplus) sum=sum+(1-p)*(Lambda(b)-Lambda(b+Deficit))*x;
break;
default: sum=sum+(1-p)*Lambda(b)*x;
}
}
}
}
if(delayed==false) GParisianStorage[m-1][uu][ll0-1][0]=sum;
else GParisianStorage[m-1][uu][ll0-1][1]=sum;
return sum;
}
}
}

double Model::DParisian(int m, int uu, int ll0, int rr, bool delayed)
{
if(m<=0 || uu<0) return 0;
else if (rr==0) return 1;
else
{
DParisianStorageExpand(m,uu,ll0,rr,delayed);
if(DParisianStorage[m-1][uu][ll0-1][rr][0]!=-1 && delayed==false) return DParisianStorage[m-1][uu][ll0-1][rr][0];
else if(DParisianStorage[m-1][uu][ll0-1][rr][1]!=-1 && delayed==true) return DParisianStorage[m-1][uu][ll0-1][rr][1];
else
{
int min_n_m;

if(delayed==true) min_n_m=min(nr,m);
else min_n_m=min(na,m);

int zz=z(uu), zstr=zstar(uu,ll0);
```

```
double sum=0;
for(int k=1; k<=min_n_m; ++k)//First summation
{
double x; bool DivTime; int vmax;
if(delayed==true) x=del[k-1];
else x=ord[k-1];

if(k>=zstr && (k-zstr)%h==0)
{
if(k==zstr) vmax=uu+c*k;
else vmax=b+c*h;
DivTime=true;
}
else
{
if(k<zstr) vmax=uu+c*k;
else vmax=b+c*((k-zstr)%h);
DivTime=false;
}

int TotDivpriork;
if(k-1<zstr) TotDivpriork=0;
else TotDivpriork=uu+c*zstr-b+floor(double(k-1-zstr)/h)*c*h;

int TotDivatk;
if(k<zstr) TotDivatk=0;
else TotDivatk=uu+c*zstr-b+floor(double(k-zstr)/h)*c*h;

if(DivTime==false)//If k is not a dividend paying instant
{
for(int jj=1; jj<=vmax; ++jj)
{
int lll;
switch(ModelType)
```

```
{
case 0:
if(k<zz||(k==zz && jj>vmax-b)) lll=max(1,ll0-k);
else if (k==zz && jj<=vmax-b) lll=max(1,ll0-max(zz-1,0));
else if (k>zz && k<zstr) lll=max(1,ll0-max(zz-1,0))+(k-zz)-I(jj>vmax-b);
else lll=(k-zstr)%h+1-I(jj>vmax-b);
break;
case 1:
if(k<=zz) lll=ll0;
else if(k>zz && k<zstr) lll=ll0+(k-zz)-I(jj>vmax-b);
else lll=(k-zstr)%h+1-I(jj>vmax-b);
break;
default:
if(k<=zz||jj>vmax-b) lll=1;
else if(k>zz && k<zstr) lll=ll0+(k-zz);
else lll=(k-zstr)%h+1;
}
for(int w=0; w<=rr; ++w)
{
if(rr-w==0) sum=sum+alpha(jj)*pow(double(TotDivpriork),double(w))*x;
else
{
if(w==0) sum=sum+alpha(jj)*DParisian(m-k,vmax-jj,lll,rr-w,false)*x;
else sum=sum+alpha(jj)*Binomial(rr,w)*pow(double(TotDivpriork),double(w))*DParisian(m-k,vmax-jj,lll,rr-w,false)*x;
}
}
}
sum=sum+Lambda(vmax)*pow(double(TotDivpriork),double(rr))*x;
}


else///If k is a dividend paying instant.
{
if(p!=0)
{
for(int jj=1; jj<=vmax; ++jj)
```

```
{
for(int w=0; w<=rr; ++w)
{
if(rr-w==0)
{
if(w==0) sum=sum+p*alpha(jj)*x;
else
{
if(vmax-jj>=b) sum=sum+p*alpha(jj)*pow(double(TotDivpriork+vmax-jj-b),double(w))*x;
else sum=sum+p*alpha(jj)*pow(double(TotDivpriork),double(w))*x;
}
}
else
{
if(vmax-jj>=b)
{
if(w==0) sum=sum+p*alpha(jj)*DParisian(m-k,b,1,rr-w,false)*x;
else sum=sum+p*alpha(jj)*Binomial(rr,w)*pow(double(TotDivpriork+vmax-jj-b),double(w))*DParisian(m-k,b,1,rr-w,false)*x;
}
else
{
int lll;
if(ModelType==2) lll=1;
else lll=h;

if(w==0) sum=sum+p*alpha(jj)*DParisian(m-k,vmax-jj,lll,rr-w,false)*x;
else sum=sum+p*alpha(jj)*Binomial(rr,w)*pow(double(TotDivpriork),double(w))*DParisian(m-k,vmax-jj,lll,rr-w,false)*x;
}
}
}
}
sum=sum+p*Lambda(vmax)*pow(double(TotDivpriork),double(rr))*x;
}
if(p!=1)
{
```

```
for(int jj=1; jj<=b; ++jj)
{
for(int w=0; w<=rr; ++w)
{
if(rr-w==0)
{
if(w==0) sum=sum+(1-p)*alpha(jj)*x;
else sum=sum+(1-p)*alpha(jj)*pow(double(TotDivatk),double(w))*x;
}
else
{
if(w==0) sum=sum+(1-p)*alpha(jj)*DParisian(m-k,b-jj,1,rr-w,false)*x;
else sum=sum+(1-p)*alpha(jj)*Binomial(rr,w)*pow(double(TotDivatk),double(w))*DParisian(m-k,b-jj,1,rr-w,false)*x;
}
}
}
sum=sum+(1-p)*Lambda(b)*pow(double(TotDivatk),double(rr))*x;
}
}
}
int TotDivatm;
if(m<zstr) TotDivatm=0;
else TotDivatm=uu+c*zstr-b+floor(double(m-zstr)/h)*c*h;

double x;
if(delayed==false) x=A(m);
else x=R(m);

if(rr==0) sum=sum+x;//Avoid the 0^0 issue
else sum=sum+x*pow(double(TotDivatm),double(rr));

if(delayed==false) DParisianStorage[m-1][uu][ll0-1][rr][0]=sum;
else DParisianStorage[m-1][uu][ll0-1][rr][1]=sum;
return sum;
}
```

```
}
}
////////////////////////////////////////////////////////////////////////////
////////////////////////////////////////////////////////////////////////////
//1ModelClassMemoize.cpp
#define _SCL_SECURE_NO_WARNINGS//Removes warnings
#include "1ModelClass.h"
#include "2PreliminaryFunctions.h"

////////////////////////////////////////////////////////////////////////////
//Destructor
Model::~Model(){}

//Memoization Functions

void Model::MemoizeSetupParameters()
{
fMaxuStorage=u; fMaxlOStorage=lO; fMaxnStorage=0; fMaxsStorage=0;//Initial values at minimium

GRedPremMaxmStorage=1;GRedPremMaxuStorage=0;GRedPremMaxlStorage=h; //Initial values at minimum  GRedPremMaxmStorage=1
DRedPremMaxmStorage=1;DRedPremMaxuStorage=0;DRedPremMaxlStorage=h; DRedPremMaxrStorage=0;//Initial values at minimum

GParisianMaxmStorage=1;GParisianMaxuStorage=0;GParisianMaxlStorage=h; //Initial values at minimum  GParisianMaxmStorage=1
DParisianMaxmStorage=1;DParisianMaxuStorage=0;DParisianMaxlStorage=h; DParisianMaxrStorage=0;//Initial values at minimum
}

void Model::MemoizeSetupStorage()
{
StorageSet(fStorage,0,1,0,0,fMaxuStorage,fMaxlOStorage,fMaxnStorage,fMaxsStorage);

StorageSet(GRedPremStorage,1,0,1,GRedPremMaxmStorage,GRedPremMaxuStorage,GRedPremMaxlStorage);
StorageSet(DRedPremStorage,1,0,1,0,DRedPremMaxmStorage,DRedPremMaxuStorage,DRedPremMaxlStorage,DRedPremMaxrStorage);

StorageSet(GParisianStorage,1,0,1,GParisianMaxmStorage,GParisianMaxuStorage,GParisianMaxlStorage);
StorageSet(DParisianStorage,1,0,1,0,DParisianMaxmStorage,DParisianMaxuStorage,DParisianMaxlStorage,DParisianMaxrStorage);
```

```
}

//f
void Model::StorageSet(std::vector<std::vector<std::vector<std::vector<double> > > >&M,int Mina1,int Mina2,int Mina3,int Mina4,int Maxa1,int Maxa2,int Maxa3,int Maxa4)
{
for(int a1=Mina1; a1<=Maxa1; a1++)
{
std::vector<std::vector<std::vector<double> > > dim3;
for(int a2=Mina2; a2<=Maxa2; a2++)
{
std::vector<std::vector<double> > dim2;
for(int a3=Mina3; a3<=Maxa3; a3++)
{
std::vector<double> dim1;
for(int a4=Mina4; a4<=Maxa4; a4++) dim1.push_back(-1);

dim2.push_back(dim1);
}
dim3.push_back(dim2);
}
M.push_back(dim3);
}
}

void Model::StorageSet(std::vector<std::vector<std::vector<std::vector<double> > > >&M,int Mina1,int Mina2,int Mina3,int Maxa1,int Maxa2,int Maxa3)
{
for(int a1=Mina1; a1<=Maxa1; a1++)
{
std::vector<std::vector<std::vector<double> > > dim3;
for(int a2=Mina2; a2<=Maxa2; a2++)
{
std::vector<std::vector<double> > dim2;
for(int a3=Mina3; a3<=Maxa3; a3++)
{
std::vector<double> dim1;
```

```
dim1.push_back(-1);//Delayed=false
dim1.push_back(-1);//Delayed=true

dim2.push_back(dim1);
}
dim3.push_back(dim2);
}
M.push_back(dim3);
}
}

void Model::StorageSet(std::vector<std::vector<std::vector<std::vector<double> > > > >&M,int Mina1,int Mina2,int Mina3,int Mina4, int Maxa1,int Maxa2,int Maxa3,int Maxa4)
{
for(int a1=Mina1; a1<=Maxa1; a1++)
{
std::vector<std::vector<std::vector<std::vector<double> > > > dim4;
for(int a2=Mina2; a2<=Maxa2; a2++)
{
std::vector<std::vector<std::vector<double> > > dim3;
for(int a3=Mina3; a3<=Maxa3; a3++)
{
std::vector<std::vector<double> > dim2;
for(int a4=Mina4; a4<=Maxa4; a4++)
{
std::vector<double> dim1;
dim1.push_back(-1);//Delayed=false
dim1.push_back(-1);//Delayed=true

dim2.push_back(dim1);
}
dim3.push_back(dim2);
}
dim4.push_back(dim3);
}
M.push_back(dim4);
```

```
}
}

void Model::fStorageExpand(int uu, int ll0, int nn, int ss)
{
if(uu>fMaxuStorage || ll0>fMaxl0Storage || nn>fMaxnStorage || ss>fMaxsStorage)//Expansion required
{
int Minu=fMaxuStorage, Minl0=fMaxl0Storage, Minn=fMaxnStorage, Mins=fMaxsStorage;

if(uu>fMaxuStorage) fMaxuStorage=uu;
if(ll0>fMaxl0Storage) fMaxl0Storage=ll0;
if(nn>fMaxnStorage) fMaxnStorage=nn;
if(ss>fMaxsStorage) fMaxsStorage=ss;

std::vector<std::vector<std::vector<std::vector<double> > > > temp;
for(int uuu=0; uuu<=fMaxuStorage; uuu++)
{
std::vector<std::vector<std::vector<double> > > dim3;
for(int lll0=1; lll0<=fMaxl0Storage; lll0++)
{
std::vector<std::vector<double> > dim2;
for(int nnn=0; nnn<=fMaxnStorage; nnn++)
{
std::vector<double> dim1;
for(int sss=0; sss<=fMaxsStorage; sss++)
{
if(uuu<=Minu && lll0<=Minl0 && nnn<=Minn && sss<=Mins) dim1.push_back(fStorage[uuu][lll0-1][nnn][sss]);
else dim1.push_back(-1);
}
dim2.push_back(dim1);
}
dim3.push_back(dim2);
}
temp.push_back(dim3);
}
```

```
fStorage=temp;
}
}

void Model::GRedPremStorageExpand(int m, int uu, int ll0,bool delayed)
{
if(m>GRedPremMaxmStorage || uu>GRedPremMaxuStorage || ll0>GRedPremMaxlStorage)//Expansion required
{
int Minm=GRedPremMaxmStorage, Minu=GRedPremMaxuStorage, Minl=GRedPremMaxlStorage;

if(m>GRedPremMaxmStorage) GRedPremMaxmStorage=m;
if(uu>GRedPremMaxuStorage) GRedPremMaxuStorage=uu;
if(ll0>GRedPremMaxlStorage) GRedPremMaxlStorage=ll0;



std::vector<std::vector<std::vector<std::vector<double> > > > temp;
for(int mm=1; mm<=GRedPremMaxmStorage; mm++)
{
std::vector<std::vector<std::vector<double> > > dim3;
for(int uuu=0; uuu<=GRedPremMaxuStorage; uuu++)
{
std::vector<std::vector<double> > dim2;
for(int lll0=1; lll0<=GRedPremMaxlStorage; lll0++)
{
std::vector<double> dim1;
if(mm<=Minm && uuu<=Minu && lll0<=Minl)
{
dim1.push_back(GRedPremStorage[mm-1][uuu][lll0-1][0]);
dim1.push_back(GRedPremStorage[mm-1][uuu][lll0-1][1]);
}
else
{
dim1.push_back(-1);//Delayed=false
dim1.push_back(-1);//Delayed=true
}
```

```
dim2.push_back(dim1);
}
dim3.push_back(dim2);
}
temp.push_back(dim3);
}
GRedPremStorage=temp;
}
}

void Model::DRedPremStorageExpand(int m, int uu, int ll0, int rr, bool delayed)
{
if(m>DRedPremMaxmStorage || uu>DRedPremMaxuStorage || ll0>DRedPremMaxlStorage || rr>DRedPremMaxrStorage)//Expansion required
{
int Minm=DRedPremMaxmStorage, Minu=DRedPremMaxuStorage, Minl=DRedPremMaxlStorage, Minr=DRedPremMaxrStorage;

if(m>DRedPremMaxmStorage) DRedPremMaxmStorage=m;
if(uu>DRedPremMaxuStorage) DRedPremMaxuStorage=uu;
if(ll0>DRedPremMaxlStorage) DRedPremMaxlStorage=ll0;
if(rr>DRedPremMaxrStorage) DRedPremMaxrStorage=rr;

std::vector<std::vector<std::vector<std::vector<std::vector<double> > > > > temp;
for(int mm=1; mm<=DRedPremMaxmStorage; mm++)
{
std::vector<std::vector<std::vector<std::vector<double > > > > dim4;
for(int uuu=0; uuu<=DRedPremMaxuStorage; uuu++)
{
std::vector<std::vector<std::vector<double> > > dim3;
for(int lll0=1; lll0<=DRedPremMaxlStorage; lll0++)
{
std::vector<std::vector<double> > dim2;
for(int rrr=0; rrr<=DRedPremMaxrStorage; rrr++)
{
std::vector<double> dim1;
if(mm<=Minm && uuu<=Minu && lll0<=Minl && rrr<=Minr)
```

```
{
dim1.push_back(DRedPremStorage[mm-1][uuu][lll0-1][rrr][0]);
dim1.push_back(DRedPremStorage[mm-1][uuu][lll0-1][rrr][1]);
}
else
{
dim1.push_back(-1);//Delayed=false
dim1.push_back(-1);//Delayed=true
}
dim2.push_back(dim1);
}
dim3.push_back(dim2);
}
dim4.push_back(dim3);
}
temp.push_back(dim4);
}
DRedPremStorage=temp;
}
}

void Model::GParisianStorageExpand(int m, int uu, int ll0,bool delayed)
{
if(m>GParisianMaxmStorage || uu>GParisianMaxuStorage || ll0>GParisianMaxlStorage)//Expansion required
{
int Minm=GParisianMaxmStorage, Minu=GParisianMaxuStorage, Minl=GParisianMaxlStorage;

if(m>GParisianMaxmStorage) GParisianMaxmStorage=m;
if(uu>GParisianMaxuStorage) GParisianMaxuStorage=uu;
if(ll0>GParisianMaxlStorage) GParisianMaxlStorage=ll0;



std::vector<std::vector<std::vector<std::vector<double> > > > temp;
for(int mm=1; mm<=GParisianMaxmStorage; mm++)
{
```

```
std::vector<std::vector<std::vector<double> > > dim3;
for(int uuu=0; uuu<=GParisianMaxuStorage; uuu++)
{
std::vector<std::vector<double> > dim2;
for(int lll0=1; lll0<=GParisianMaxlStorage; lll0++)
{
std::vector<double> dim1;
if(mm<=Minm && uuu<=Minu && lll0<=Minl)
{
dim1.push_back(GParisianStorage[mm-1][uuu][lll0-1][0]);
dim1.push_back(GParisianStorage[mm-1][uuu][lll0-1][1]);
}
else
{
dim1.push_back(-1);//Delayed=false
dim1.push_back(-1);//Delayed=true
}
dim2.push_back(dim1);
}
dim3.push_back(dim2);
}
temp.push_back(dim3);
}
GParisianStorage=temp;
}
}

void Model::DParisianStorageExpand(int m, int uu, int ll0, int rr, bool delayed)
{
if(m>DParisianMaxmStorage || uu>DParisianMaxuStorage || ll0>DParisianMaxlStorage || rr>DParisianMaxrStorage)//Expansion required
{
int Minm=DParisianMaxmStorage, Minu=DParisianMaxuStorage, Minl=DParisianMaxlStorage, Minr=DParisianMaxrStorage;

if(m>DParisianMaxmStorage) DParisianMaxmStorage=m;
if(uu>DParisianMaxuStorage) DParisianMaxuStorage=uu;
```

```
if(ll0>DParisianMaxlStorage) DParisianMaxlStorage=ll0;
if(rr>DParisianMaxrStorage) DParisianMaxrStorage=rr;

std::vector<std::vector<std::vector<std::vector<std::vector<double> > > > > temp;
for(int mm=1; mm<=DParisianMaxmStorage; mm++)
{
std::vector<std::vector<std::vector<std::vector<double > > > > dim4;
for(int uuu=0; uuu<=DParisianMaxuStorage; uuu++)
{
std::vector<std::vector<std::vector<double> > > dim3;
for(int lll0=1; lll0<=DParisianMaxlStorage; lll0++)
{
std::vector<std::vector<double> > dim2;
for(int rrr=0; rrr<=DParisianMaxrStorage; rrr++)
{
std::vector<double> dim1;
if(mm<=Minm && uuu<=Minu && lll0<=Minl && rrr<=Minr)
{
dim1.push_back(DParisianStorage[mm-1][uuu][lll0-1][rrr][0]);
dim1.push_back(DParisianStorage[mm-1][uuu][lll0-1][rrr][1]);
}
else
{
dim1.push_back(-1);//Delayed=false
dim1.push_back(-1);//Delayed=true
}
dim2.push_back(dim1);
}
dim3.push_back(dim2);
}
dim4.push_back(dim3);
}
temp.push_back(dim4);
}
DParisianStorage=temp;
```

```
}
}
////////////////////////////////////////////////////////////////////////////
////////////////////////////////////////////////////////////////////////////
//2PreliminaryFunctions.h
#include <cmath> //Incorporate ceiling and floor functions
#include <iostream>
#include <vector> //Vector and matrix algebra

using namespace std;
////////////////////////////////////////////////////////////////////////////
int Binomial (int n, int x);//Assuming n and x are integers
bool I(bool A);
////////////////////////////////////////////////////////////////////////////

//2PreliminaryFunctions.cpp
#include "2PreliminaryFunctions.h"

////////////////////////////////////////////////////////////////////////////
////////////////////////////////////////////////////////////////////////////
int Binomial (int n, int x)//Assuming n and x are integers
{
    if (n<x) return 0;
    else if (x==0||n==1) return 1;
    else return Binomial(n-1,x)+Binomial(n-1,x-1);
}
////////////////////////////////////////////////////////////////////////////
bool I(bool A)
{
 return A;
}
```