MÄLARDALENS HÖGSKOLA
Institutionen för Matematik och Fysik

Code: MdH.IMa.Mat.0076 (2006) 10p-AF

MASTER THESIS IN MATHEMATICS /APPLIED MATHEMATICS

A Java applet for simulation of economy with Borrowers under costly defaults

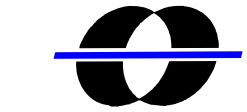
by

Basil Wakid Hassan

Magisterarbete i matematik / tillämpad matematik

DEPARTMENT OF MATHEMATICS AND PHYSICS

MÄLARDALEN UNIVERSITY
SE-721 23 VÄSTERÅS, SWEDEN



MÄLARDALENS HÖGSKOLA

DEPARTEMENT OF MATHEMATICS AND PHYSICS

Master thesis in mathematics / applied mathematics

Date:
2007-01-23

Projectname:
A Java applet for simulation of economy with borrowers under costly defaults

Author:
Basi Wakid Hassan

Supervisor:
Dr. Anatoliy Malyarenko

Examiner:
Prof. Dmitrii Silvestrov

Comprising:
10 points

Dedicated with love
To my wife Hanan, my son Mustafa,
My daughters Saba and Yasmin,
And to the memory of my father,
My mother

Abstract

This master thesis focuses on the simulation of economy with borrowers under costly defaults. The asset-value dynamics in our model exhibits its stochastic mean and volatility. A Java program is developed as a tool for calculation.

Acknowledgements

This study has drawn on the aptitudes, advice and encouragement of more people that I can possibly acknowledge. I would, however, like to recognize the contributions of many who have helped.

First, I want to thank my wife, for her support, interest, encouragement, and children who also have experienced the hefty time demands of this project and have understood and helped me in numerous ways.

I greatly appreciate the assistance of my supervisor Dr. Malyarenko Anatoliy who was the major developer of the study guide and contributed greatly to this project.

I want to thank Prof. Silvestrov Dmitrii, for his role played by examiner this study and appreciate for his efforts and encouragements.

I am very grateful to all my teachers, and other colleagues and administrative staff of Malardalen University, particularly grateful to department of Mathematics and physics for their kind help.

Table of Content

Table of Content 5
1. Introduction 8
1.1. Background 8
1.2. Objective 9
2. Mathematical &Background Reviewing of Basak-Shapiro Credit Risk model 10
2.1. Basak-Shapiro Model of the debt contract and default cost 11
2.2. Basak-Shapiro Optimization problem when debt maturity coincides with Planning horizon $(T = T')$ 12
2.3. Reviewing Mathematical and Background of The study to get the solution 14
3. Analysis and Inferences 18
3.1. Effect of model parameters on optimal wealth 18
3.1.1. Effect of Increase Model Parameters values 19
3.1.2. Effect of Wealth relative Frequency Histogram 23
3.1.3. Effect of Investment Policy 24
3.2. Sensitivity analysis of risk exposure function to change in model's parameters 27
3.2.1. Effect of debt-contract parameter's on risk exposure function 28
3.2.2. Effect of default-costs parameter' on risk exposure function 31
4. Java Applet for Basak-Shapiro model 35
4.1. Introduction 35
4.2. Applet: User Guide and Parameters 36
4.2.1. J Pane 36
4.2.1.1. Input panel 37
4.2.1.2. Control panel 38
4.2.1.3. Output panel 38
4.2.2. Debt function trajectory 39
4.2.3. Wealth function 40
4.2.4. Risk investment function 40
4.2.5. Numerical results 41
4.2.6. Default Relative Frequency Histogram 41
4.2.7. Wealth Relative Frequency Histogram 42
4.2.8. Joint Histogram 42
4.2.9. Final Numerical Results 43
4.2.10. Risk Exposure function 44
5. Conclusion 45
6. References 47
7. Appendix 48

List of Figures

Figure 1 Debt value function with defaults region boundaries 13
Figure 2 wealth function 18
Figure 3 debt value function..... 18
Figure 4 wealth function when (F) increase..... 19
Figure 5 debt value function when(F) increase..... 19
Figure 6 wealth function when(β) increased..... 20
Figure 7 debt value function when(β) increased..... 20
Figure 8 Wealth function when(β) reach to limit of one..... 20
Figure 9 debt value function when(β) reach to limit of one..... 21
Figure 10 Wealth function when (λ) increased..... 21
Figure 11 debt value function when (λ) increased..... 22
Figure 12 Wealth function when (ϕ) increased..... 22
Figure 13 debt value functions when (ϕ) increased..... 22
Figure 14 wealth histogram when (F) is very small..... 23
Figure 15 wealth histogram..... 24
Figure 16 risky investment function for simulated trajectories that lead to no default positions 25
Figure 17 risk exposure function for simulated trajectories that lead to no default positions ... 25
Figure 18 debt value function for simulated trajectories that lead to no default positions 26
Figure 19 Java applet viewer: Risk exposure function 27
Figure 20 for F = 2 28
Figure 21 for β = 0.6..... 29
Figure 22 for F = 0.7 30
Figure 23 for β = 0.4..... 30
Figure 24 for ϕ = 0.5 32
Figure 25 for λ = 0.9..... 32
Figure 26 for ϕ = 0.015 33
Figure 27 for λ = 0.01..... 34
Figure 28 Java Applet Viewer: BasakShapiro2 credit risk model 36
Figure 29 error messages for retaining rate value..... 37
Figure 30 control panel 38
Figure 31 output panels..... 38
Figure 32 output panels..... 39
Figure 33 debt value functions..... 39
Figure 34 wealth function 40
Figur 35 risk investment policy..... 40
Figure 36 numerical results..... 41
Figur 37 default relative frequency histogram..... 41
Figure 38 wealth relative frequency histogram..... 42
Figure 39 joint histogram..... 42
Figure 40 risk exposure function..... 44

List of tables

Table 1 for F = 2 28
Table 2 for β = 0.6 29
Table 3 for F = 0.7 30
Table 4 for β = 0.4 31
Table 5 for ϕ = 0.5 32
Table 6 for λ = 0.9 33
Table 7 for ϕ = 0.015 34
Table 8 for λ = 0.01 34

1. Introduction

1.1. Background

Risk managers are responsible to monitored risk carefully due to their damage affects on businesses. They should be able to identify various risk exposure, measured and controlled. Generally, risk defined as volatility of unexpected outputs. Firm face various types of risk such as business and no business risk. The most interest risk is, in this study, financial risk, specifically credit risk. These risks focus on potential losses in financial market due to fluctuations in interest rate or defaults on financial commitments.

Commonly, risk manager's aims to measure sources of risk, as precisely as possible, purpose to controlling and suitably price risks. So that risk regard as the magnitude of possible loss or volatility, as measured by standard deviation of the possible income or revenue of portfolio (investment or trading) over specific horizon [1].

Financial risks can be classified broadly by several categories such as, market risks, credit risks, liquidity risks, operational risks and legal risk.

The source of risks can be mainly comes from two factors:

Default risk, is define as the objective assessment of default probability for the counterparty combined with the loss given default.

Market risk or credit exposure, is defined as the probable future loss that may be happened due to a decrease in market value. Generally this occurs because the observed market rates subject to changes – e.g. yield curves, implied volatilities, and spot exchange rates.

The quantification of credit risk recently gains great attentions and become a large subject area. Increasing borrowing transactions specifically through private sector in recent years and option to default, as increased default probability of debt contracts, led to raise the need for the credit risk models whose estimate the dynamic of the borrower asset value and credit risk as part of their debt.

Often, default occurs at maturity and when the asset value of borrower less than the face value of a debt contract. Because the dynamic of asset price managed by the borrower so the probability of a credit risk arise.

Credit risk defined as financial loss due to borrowers may be unwilling or unable to fulfill their contractual obligations. This loss encompasses exposure or amount at risk, represented here by the face value of the instrument or debt. And retaining rate represents the fraction recovered given default, or one minus the loss given default [2].

Basak-Shapiro models take a zero-coupon debt contract as borrowers leverage instrument that may default occur upon maturity date.

This study considers only the state where the date of repay the debt is coincides with borrowers planning horizon.

Borrowers optimal terminal net worth and debt value function and the others functions are bounded in three state of the economy area: no default, default and between as resistance of default. The models illustrate a different economics behavior of borrower optimal wealth function across the above regions. Costs of default affect the optimal policy of the borrower with different manner through the horizon of default region.

1.2. Objective

Our objective is to use Basak-Shapiro model, as abase of this study, for the purpose of modeling debt contract and costs of default that affects optimal policies of the borrower and analysis the associated investment and consumption policies.

Principally, the aim of this study is to creating an applet by trying to write a program in Java language. This program will truly enable end user dealing easily with numerical and graphical outputs of this model. To achieve this goal the study will adopt different methods than were used in Basak-Shapiro model for estimation formulas and depicting the figures.

So that, the study intending to design and realizes an applet layout that best display the above mentioned requirements used specially for this model. Our goal also to make this program capable to meet the end user needs and provide a wide flexibility in sensitivity analysis plus performing the necessary calculation for a decision making.

For accomplish these objectives, the study will figure out the appropriate codes and mathematics and write a program in Java language that display an applet with a number of panels, buttons, labels, progress bar, field among others, suitable for this model.

2. Mathematical &Background Reviewing of Basak-Shapiro Credit Risk model

The purpose of this section is to describe and reviewing Basak-Shapiro models which used as background of this study and facilitate the following of this model [3].

The financial market in this model is a finite –horizon, [0-T '] which containing a single consumption good.
The model defines the uncertainty by filtration process of the probability space $[\Omega, F, \{F_t\}, P]$ which has also an N-dimensional Brownian motion representing by the following [4]: $w(t) = [w_1(t), ..., w_N(t)]^T, t \in [0, T']$. The stochastic processes in this model are adapted to $\{F_t\}$, where $t \in [0, T']$ The financial market containing N+1 investment instrument one instantaneously risk-free and the others are risky. The dynamics of the investment net return have the following vector:

$$\begin{pmatrix} r(t)dt \\ \\ \mu(t)dt + \sigma(t)dw(t) \end{pmatrix}, \qquad (1)$$

r = interest rate
μ = drift coefficients
$(\mu, ..., \mu_N)^T$, with the volatility matrix $\sigma \equiv \{\sigma_{ij,}\ i = 1, ..., N;\ j = 1, ..., N\}$ which might be path dependent.
If the market is complete and free of arbitrage it will exists unique state price density process (ζ) which have the following differential equation:

$$d\zeta(t) = -\zeta(t)[r(t)dt + \mathrm{K}(t)^T dw(t)], \quad \ldots\ldots\ldots\ldots \qquad (2)$$
$$\zeta(0) = 1.$$

In the equation (2) $\mathrm{K}(t)$ representing the market price of risk and equal to $\kappa(t) \equiv \sigma(t)^{-1}[\mu(t) - r(t)\bar{I}]$, and $\bar{I} \equiv (1, ..., 1)^T$.
The Arrow-Debreu price is measured by $\zeta(T', \omega)$ per each unit of probability and consumption good taken in the stat $\omega \in \Omega$ at time T'.
A zero-coupon bond which matures at date T [5] is a contract that guarantees the holder to receive one Unit of money (dollar, kronor, sterling …..) at the date T. The borrower in this model is restrict by this instrument to get the initial wealth of W(0), which is equal to the net of borrowing proceeds that the borrower will get it at the time = 0.

At the end of the nonnegative planning-horizon wealth the borrower will receive a terminal net worth W (T '). The borrower chose to invest the borrowing proceed in an investment policy $(\boldsymbol{\theta}) \equiv [\theta_1(t), ...\theta_N(t)]^T$ Symbolize the vector of fractions of wealth that the borrower intends to invest it in each risky investment opportunity. The wealth process W (t) before the debt-maturity date satisfies the following equation

$$dW(t) = W(t)\{r(t) + \theta(t)^T[\mu(t) - r(t)\mathbf{1}]\}\, dt + W(t)\theta(t)^T \sigma(t) dW(t). \qquad (3)$$

The known value $W(0)$ is interpreted as the initial wealth while the value $W(T')$ represents the terminal net worth. Before debt-maturity the total value of the assets $V(t)$, managed by borrower, is

$$V(t) = W(t) + D(t).$$

Where $D(t)$ equal the value of the debt.

The objective function $v[W(T')]$ which represents the expected value that the borrowers wish to maximize is assumed twice continuously differentiable severely concave, severely increasing with $\lim_{x\to 0} v'(x) = \infty$, and $\lim_{x\to\infty} v'(x) = 0$.

2.1. Basak-Shapiro Model of the debt contract and default cost

The main goal of Basak-Shapiro models is to verify if the default costs (when it occurs) have an affect on the optimal policies of the borrower. The default able debt contract has two characteristics:

First, when the default occurs, the lender can calculate only a fraction of the borrower's assets which is reproduced in variation from the absolute priority rule.

Second, the declaration of the default is happened in spite of that the borrower have enough assets to service the debt. The debt contract between borrower and lender has the following Structure:

Assumption1. The payoff of a zero-coupon debt contract is

$$D(T) = \min\{(1-\beta)V(T), F\}.$$

F Is the face value of the debt contract and maturity date is $T \le T'$, β representing the retaining rate and equal to $0 \le \beta \le 1$.

When the borrower is enabling to fully repay the face value at debt-maturity date $T \le T'$, the default occurs. In this case the debt value will be equal to: $D(T) = (1-\beta)V(T) < F$, and the *default boundary* will be equal to $\frac{\beta F}{(1-\beta)}$ and this is involving that the borrower's solvency should be $W(T) \ge \frac{\beta F}{(1-\beta)}$. Where $\beta v(t)$ represent the value that the borrower keeps hold of it

Upon default .The borrowers assets divide into two parts, tangible part of asset, $(1-\beta)V(T)$ which guarantee the repaying of the debt when it is liquidizes, and the intangible part $\beta v(T)$ such as human resources and administration and organizational skills and knowledge's. Since the claim's covered by tangible assets upon default the borrower then can be able to use the intangible assets to generate future cash flows. The debt maturity date (T) is fixed ahead so the default perhaps take place only in this deterministic date, or coincide with planning horizon date (T').

Assumption2. At maturity date **T**, and upon the default occur, the borrower incurs the following defaults costs

$$C(T) = \begin{cases} \Phi + \lambda[F - D(T)], & D(T) < F, \\ 0, & D(T) = F. \end{cases}$$

As it stated before the default happened when $[D(T) < F]$, so $\Phi \ge 0$ represent the fixed cost of the default that the borrower incurs and $\lambda \ge 0$ equal to the proportional costs with the above

total cost of default $C(T)$ .Default cost may be interpreted as direct or in-direct cost because both cost component may include these expenses. $[F - D(T)]$ Equal the amount of default and λ represent the proportional cost per unit of default so the total variable costs will be the multiplication result.
In case we don't have a debt, the investor depend on the self financing, the benchmark investment model (hence-forth B) will be used for comparison. The optimal solution of this model is B-model $W^B(T')$ with F=0 and (V=W).

2.2. Basak-Shapiro Optimization problem when debt maturity coincides with Planning horizon $(T = T')$

The model gives us a solution of the optimization problem of the borrower which is limited by the coincidences of the two dates, debt contract maturity and planning horizon date $(T = T')$. According to default cost, it is quit possible that the borrower declare his or her default ness at time T. The solution gets the following characteristics:

A. The optimization problem is stated as following:

Try to find

$$\max_{W(T)} E[\nu(W(T))].$$

Subject to the constraint

$$E\{\zeta(T)[W(T) + C(T)]\} \le W(0). \tag{4}$$

The borrower s budget constraint shows that the initial wealth W (0), resulted from the proceeds of net borrowing process, must be enough to cover time T values (terminal wealth plus probable costs).
In what follows we suppose that $(T = T')$, $\nu(W) = W$, r(t)=r and **k**(t) = **k**.

Basak and Shapiro propose the following proposition to find the optimal solution:
When debt maturity date agree with the borrower's planning horizon $(T = T')$, the optimal solution of the borrower's optimal net worth will be

$$\text{W*(t)} = \begin{cases} I[y\zeta(T)] & if \zeta(T) < \zeta_* \quad : nodefault, \\ \dfrac{\beta F}{1-\beta} & if \zeta_* \le \zeta(T) < \zeta^* \quad : default - resis\tan ce \\ I[\dfrac{\beta y}{\beta - \lambda(1-\beta)}\zeta^*(T)] & if \zeta^* \le \zeta(T) \quad : default \end{cases} \tag{5}$$

Where **I** (.) is the inverse function of $\nu'(.), \zeta_* \equiv \nu'[\beta F/(1-\beta)]/y$ and $\zeta^* \ge \zeta_*, y \ge 0$ solve the following system: $\nu[I(x\zeta^*)] - \nu[I(y\zeta_*)] = x\zeta^*[I(y\zeta^*) - I(y\zeta_*) + \Phi]$
with $x \equiv \beta y/[\beta + \lambda(1-\beta)], E\{\zeta(T)[W^*(T;t) + C(W^*(T;t))]\} = W(0)$. And the benchmark borrower's optimal terminal net worth is $W^B(T) = I[y^B\zeta(T)]$, where y^B solves $E\{\zeta(T)I(y^B\zeta(T))\} = W^B(0)$.

And they depict the borrower's time-T optimal wealth $W^*(T)$ with relation to the time-T B-policy, at time $(T = T')$, with respect to the price of the consumption good $\zeta(T)$ as independent variable. They found that the borrower show three separate prototypes of economic behavior which can be charted into three regions of the states space:

A. No default
B. Default resistance. The optimal wealth is constant and does not change by increasing the price of $\zeta(T)$.
C. Default.

Interval $[0, \zeta_*)$ represents no default region, the good state, where ζ_* coming just after the terminated of no default region. Since $\zeta(T) < \zeta_*$ the borrower fulfils all his obligations toward the lender and don't defaulting. The intermediate interval $[\zeta_* \leq \zeta(T) < \zeta^*)$ the area in which the borrower resist to announce default ness and called default resistance region or (resistance).While the Interval $[\zeta^* \leq \zeta(T)$ consists of default region, the bad states, in which the borrowers no more can keep his financial obligations toward the lender.

The resistance region, between the two lines, as shown in figure 1 below emerge due to undesirability of costly default as we can consider it as financial distress costs that is leading to obliterate the business and loss the resources.
The minimum level of the optimal wealth should be kept by the borrowers as a buffer against the default is correspond to the value of the default boundary, $W^*(T) > \frac{\beta F}{1-\beta}$, and this indicator will be used to conclude the default state from the estimated data in this study.

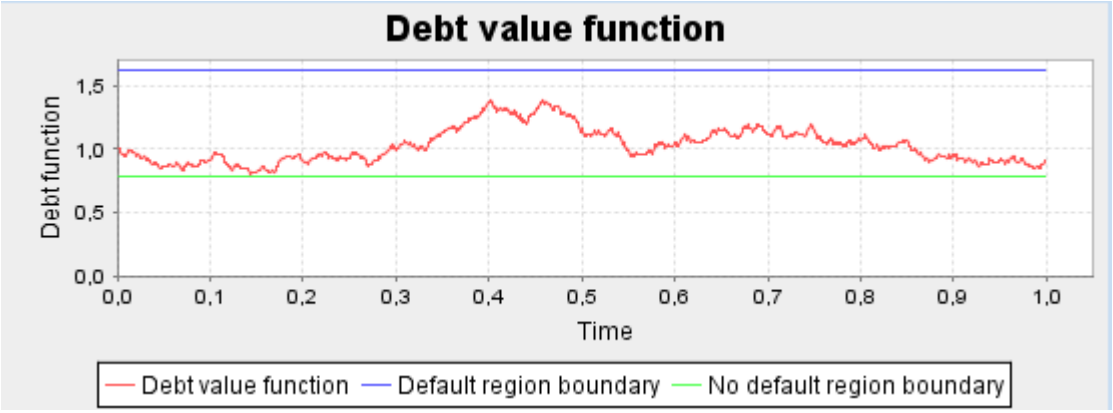


Figure 1 Debt value function with defaults region boundaries

2.3. Reviewing Mathematical and Background of The study to get the solution

We start by introducing an auxiliary variable $z = y\zeta^*$ as the solution to the following equation and take into consideration that $\gamma = 1$:

$$\ln\left[\frac{\beta+\lambda(-\beta)}{\beta y\zeta^*}\right]+\frac{[\beta F-\Phi(1-\beta)]\beta y\zeta^*}{(1-\beta)[\beta+\lambda(1-\beta)]}=1+\ln\frac{\beta F}{(1-\beta)}$$

$$\ln\left[\frac{\beta+\lambda(-\beta)}{\beta z}\right]+\frac{[\beta F-\Phi(1-\beta)]\beta z}{(1-\beta)[\beta+\lambda(1-\beta)]}=1+\ln\frac{\beta F}{(1-\beta)}$$

$$\ln(\beta+\lambda(1-\beta)-\ln\beta z+\frac{[\beta F-\Phi(1-\beta)]\beta z}{(1-\beta)[\beta+\lambda(1-\beta)]}=1+\ln\beta F-\ln(1-\beta)$$

$$\ln(\beta+\lambda(1-\beta)-\ln\beta-\ln z+\frac{[\beta F-\Phi(1-\beta)]\beta z}{(1-\beta)[\beta+\lambda(1-\beta)]}=1+\ln\beta F-\ln(1-\beta)$$

$$-\ln z+\frac{[\beta F-\Phi(1-\beta)]\beta z}{(1-\beta)[\beta+\lambda(1-\beta)]}=1+\ln\beta F+\ln\beta-\ln(1-\beta)-\ln(\beta+\lambda(1-\beta)$$

$$-\ln z+\frac{[\beta F-\Phi(1-\beta)]\beta z}{(1-\beta)[\beta+\lambda(1-\beta)]}=1+\ln\left(\frac{\beta^2 F}{(1-\beta)}\right)-\ln(\beta+\lambda(1-\beta)$$

$$-\ln z+\frac{[\beta F-\Phi(1-\beta)]\beta z}{(1-\beta)[\beta+\lambda(1-\beta)]}=1+\ln\left(\frac{\beta^2 F}{(1-\beta)(\beta+\lambda(1-\beta)}\right)$$

$$\frac{\beta[\beta F-\Phi(1-\beta)]z}{(1-\beta)[\beta+\lambda(1-\beta)]}-\ln z=1+\ln\left(\frac{\beta^2 F}{(1-\beta)(\beta+\lambda(1-\beta)}\right) \quad (6)$$

In Basak and Shapiro model's they propose isoelastic objective function that is equal to $\nu(W)=W^{1-\gamma}/(1-\gamma), where \gamma>0,$ conditioned by state prices are lognormal distributed and associated with constant interest rate and risk market price. Under these conditions they derived an explicit direct expression to calculate the borrower's optimal wealth and investment policy prior the debt maturity and planning horizon as stated below:

$$W^*(t) = \frac{X(T-t)}{\left[y\zeta(t)^{\frac{1}{\gamma}} \right]} + \left\{ \frac{\beta F}{1-\beta} e^{-r(T-t)} N\left[-d_2(\zeta_*)\right] - \frac{X(T-t)N\left[-d_1\zeta_*)\right]}{\left[y\zeta(t)^{\frac{1}{\gamma}} \right]} \right\} -$$

$$\left\{ \left(\frac{\beta F}{1-\beta} - \Phi \right) e^{-r(T-t)} N\left[-d_2(\zeta^*)\right] - \frac{X(T-t)N\left[-d_1(\zeta^*)\right]}{\left\{ \beta y\zeta(t) / \left[\beta + \lambda(1-\beta)\right] \right\}^{\frac{1}{\gamma}}} \right\} \frac{\beta}{\beta + \lambda(1-\beta)} \qquad (7)$$

Here we have t<T, N (.) represents the standard-normal cumulative distribution function.

Where $\ln x(s) \equiv \frac{(1-\gamma)}{\gamma\left[r + \frac{\|k\|^2}{2\gamma} \right] s}$

And when $\gamma = 1$

So,

Ln x(s) = 0

X(s) = 1

X (T-t) = 1

And

$$\frac{X(T-t)}{\left[y\zeta(t) \right]^{\frac{1}{\gamma}}} = \frac{1}{y\zeta(t)} = \frac{\zeta^*}{z\zeta(t)} \qquad (8)$$

And $z = y\zeta^*$.

Next, introduce two auxiliary functions:

$$d_2(x) = \frac{\ln(x/\zeta(t) + (r - \|k\|^2/2)(T-t)}{\|k\|\sqrt{T-t}}$$

$$d_1(x) = d_2(x) + \|k\|\sqrt{T-t}$$

$\zeta_* = \left(\frac{1}{y} \right) \left[\frac{1-\beta}{\beta F} \right]^{\gamma}$, y and ζ^* solve

$$\frac{\gamma}{(1-\gamma)} \left[\frac{\beta + \lambda(1-\beta)}{\beta y\zeta^*} \right]^{(1-\gamma)/\gamma} + \frac{\left[\beta F - \Phi(1-\beta)\right]\beta y\zeta^*}{(1-\beta)\left[\beta + \lambda(1-\beta)\right]} = \frac{1}{(1-\gamma)} \left(\frac{\beta F}{1-\beta} \right)^{1-\gamma}$$

And $W^*(0, y) = W(0)$. Where

$$\zeta_* = \frac{(1-\beta)}{\beta z F} \zeta^*.$$

With our simplification, formula (7) from [8] has the form

$$W^*(t)=\frac{\zeta^*}{z\zeta(t)}+\frac{\beta F}{1-\beta}e^{-r(T-t)}N\left[-d_2(\zeta(x)\right]-\frac{\zeta^*}{z\zeta(t)}N(-d_1(\zeta_*))$$

$$-\left(\frac{\beta F}{1-\beta}-\Phi\right)\frac{\beta}{\beta+\lambda(1-\beta)}e^{-r(T-t)}N\left[-d_2(\zeta^*)\right]$$

$$+\frac{x(T-t)}{\left[\dfrac{\beta y\zeta(t)}{\beta+\lambda(1-\beta)}\right]^{\frac{1}{\gamma}}}*\frac{\beta}{\beta+\lambda(1-\beta)}N\left[-d_1(\zeta^*)\right]$$

$$+\frac{x(T-t)}{\left[(\beta y\zeta(t))/(\beta+\lambda(1-\beta))\right]^{\frac{1}{\gamma}}}*\frac{\beta}{\beta+\lambda(1-\beta)}N\left[-d_1(\zeta^{*)}\right]$$

$$W^*(t)=\frac{\zeta^*}{z\zeta(t)}+\frac{\beta F}{1-\beta}e^{-r(T-t)}N\left(-d_2(\zeta_*)\right)-\frac{\zeta^*}{z\zeta(t)}N\left(-d_1(\zeta_*)\right)$$

$$-\left(\frac{\beta F}{1-\beta}-\Phi\right)\frac{\beta}{\beta+\lambda(1-\beta)}e^{-r(T-t)}N\left(-d_2(\zeta^*)\right)+\frac{\zeta^*}{z\zeta(t)}N\left(-d_1(\zeta^*)\right) \qquad (9)$$

Now, the obvious equality $W^*(0,y)=W(0)$ becomes an equation with respect to ζ^*.
Solving this equation, we obtain the values of ζ^* and ζ_*.

When t = 0, so $\zeta(t)=\zeta(0)=1$.
And $W^*(0)=W(0)=1$.
Then the auxiliary functions are:

$$d_2(x)=\frac{\ln(x)+(r-\|k\|^2/2)T}{\|k\|\sqrt{T}}.$$

$$d_1(x)=d_2(x)+\|k\|\sqrt{T}\ .$$

From equation (9) we get, when t = 0, then the following:

$$W^*(0)=\frac{\zeta^*}{z}+\frac{\beta F}{1-\beta}e^{-rT}N\left[-d_2(\zeta_*)\right]-\frac{\zeta^*}{z}N(-d_1(\zeta_*))$$

$$-\left(\frac{\beta F}{1-\beta}-\Phi\right)\frac{\beta}{\beta+\lambda(1-\beta)}e^{-rT}N\left(-d_2(\zeta^*)\right)+\frac{\zeta^*}{z}N\left(-d_1(\zeta^*)\right)=\mathbf{1} \qquad (10)$$

Simplifying equation (10) we get:

$$W^*(0) = \frac{\zeta^*}{z} + \frac{\beta F}{1-\beta} e^{-rT} N\left[-d_2\left(\frac{1-\beta}{\beta z F} \zeta^* \right)\right] - \frac{\zeta^*}{z} N(-d_1\left(\frac{1-\beta}{\beta z F} \zeta^* \right)$$

$$-\left(\frac{\beta F}{1-\beta} - \Phi \right) \frac{\beta}{\beta + \lambda(1-\beta)} e^{-rT} N\left(-d_2(\zeta^*)\right) + \frac{\zeta^*}{z} N\left(-d_1(\zeta^*)\right) = \mathbf{1} \qquad (11)$$

In the equation (11) the only unknown is ζ^*. So we found an explicit direct expression to solve the equation for ζ^* and use this value to get the value of equation (10) as mentioned before.

The fraction of wealth invested in the risky investment opportunities is:

$$\theta^*(t) = m^*(t)\theta^B(t),$$

(12)

Where $\theta^B = \frac{1}{\gamma}\left[\sigma(t)^T\right]^{-1} k$, which is equal to

$$\theta^B = \left[\sigma(t)^T\right]^{-1} k \ , \qquad (13)$$

The following equation calculates the risk exposure of the borrower:

$$m^*(t) = 1 - \frac{e^{-r(T-t)}}{W^*(t)} \left[\frac{\beta F}{1-\beta} N\left(-d_2(\zeta_*)\right) - \left(\frac{\beta F}{1-\beta} - \phi \right) \frac{\beta}{\beta + \lambda(1-\beta)} N\left(-d_2(\zeta^*)\right) \right.$$

$$\left. - \frac{\beta}{\beta + \lambda(1-\beta)} \left(\frac{\beta F}{1-\beta} - \phi - \frac{\beta + \lambda(1-\beta)}{\beta z} \right) \frac{\varphi\left(d_2(\zeta^*)\right)}{\|k\| \sqrt{T-t}} \right]. \qquad (14)$$

In equation (14) $\varphi(\cdot)$ represent the standard-normal probability distribution function.
Basak and Shapiro proposed exposure of the risky investment with relation to benchmarks policy is bounded by:
If $\phi = 0$, then $0 \le m^*(t) \le 1$.
And if $\phi > 0$, then $0 \le m^*(t) > 1$

3. Analysis and Inferences

3.1. Effect of model parameters on optimal wealth

Figures 2and 3 below illustrate Java applet viewer for wealth function and debt value function

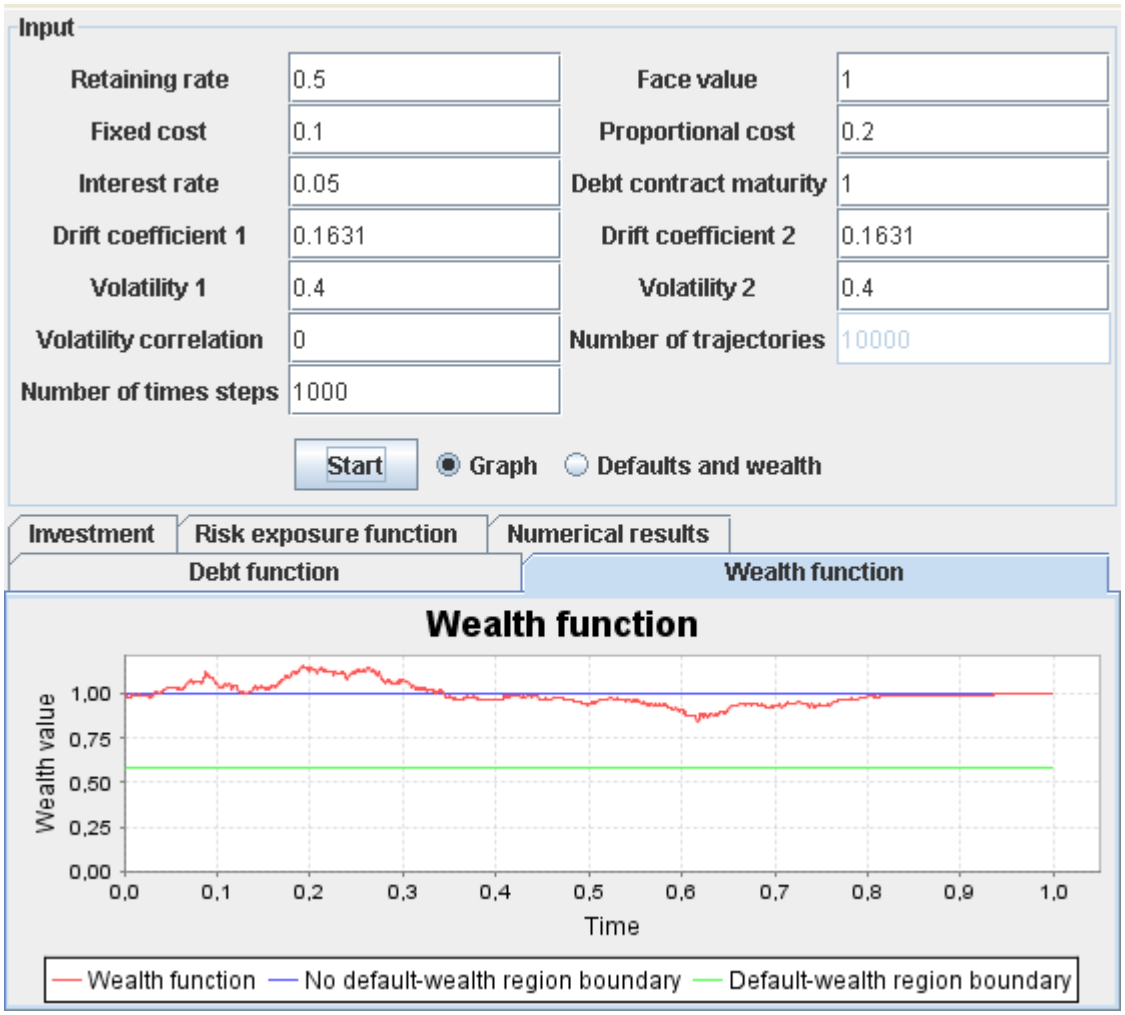


Figure 2 wealth function

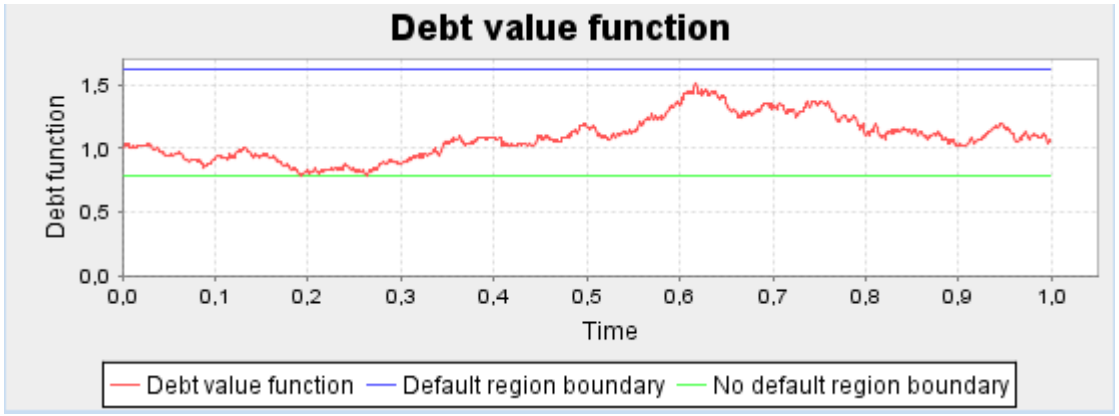


Figure 3 debt value functions

In order to demonstrate the effects of the parameters $F, \beta, \lambda, \phi,$ on the optimal wealth We shall take the following cases:

3.1.1. Effect of Increase Model Parameters values

The figure 2 above shows the wealth function $W^*(t)$ with respect to times periods and no default and default boundaries, $W^*(\zeta_*)$ and $W^*(\zeta^*)$, while figure 3 depict the debt value function with respect to the time periods and region boundaries, ζ_* and ζ^* . Parameters used in portray the figures is shown in the input panel. The program gives the user a high flexibility to inspect the dependency of the optimal solution on the parameters $F, \beta, \lambda, \phi,$ which it is obvious seen below when we shall try to change the parameters values and demonstrate the results?

As the face value (F) increased from 1 to 2, or the retaining rate (β) increased from 0.5 to 0.6. The no default-Wealth and default-Wealth region boundaries, $W^*(\zeta_*)$ and $W^*(\zeta^*)$, increased consequently. But the opposite effect then happened on the no default and default region boundaries, ζ_* and ζ^*, which makes them decrease and led to reduce the size of the resistance region as shown clearly in figures 4 , 5 and 6,7 below :

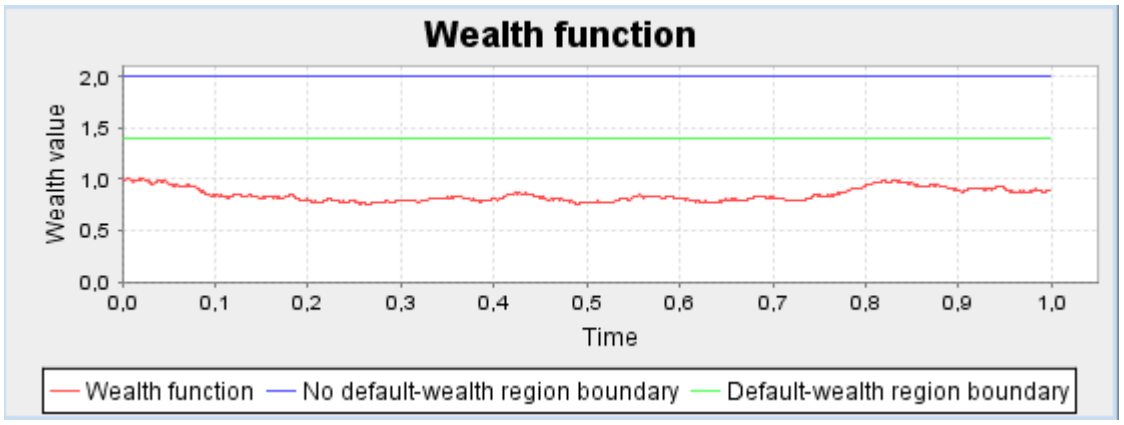


Figure 4 wealth function when (F) increase

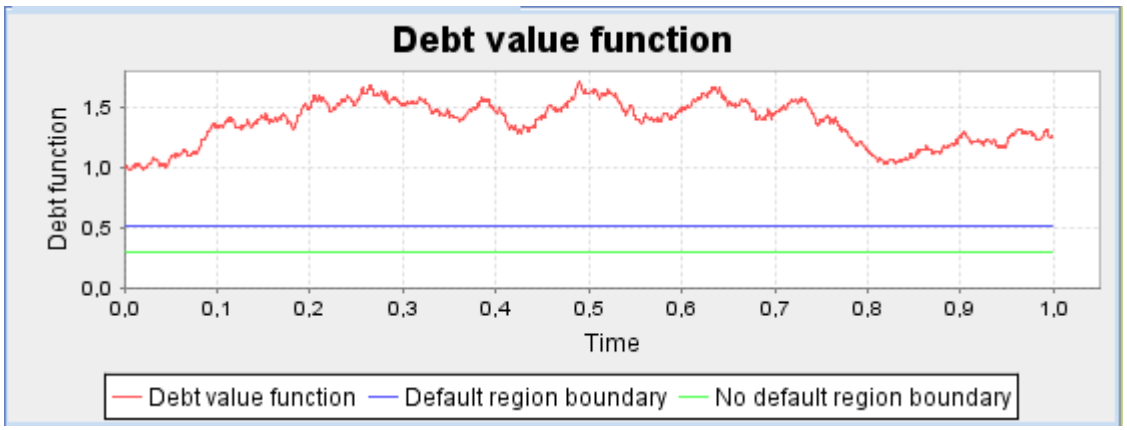


Figure 5 debt value function when (F) increase

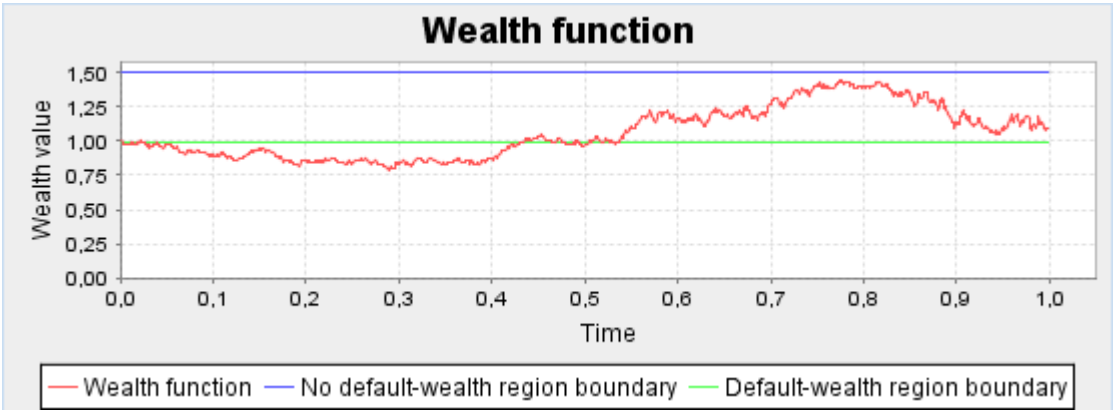


Figure 6 wealth function when (β) increased

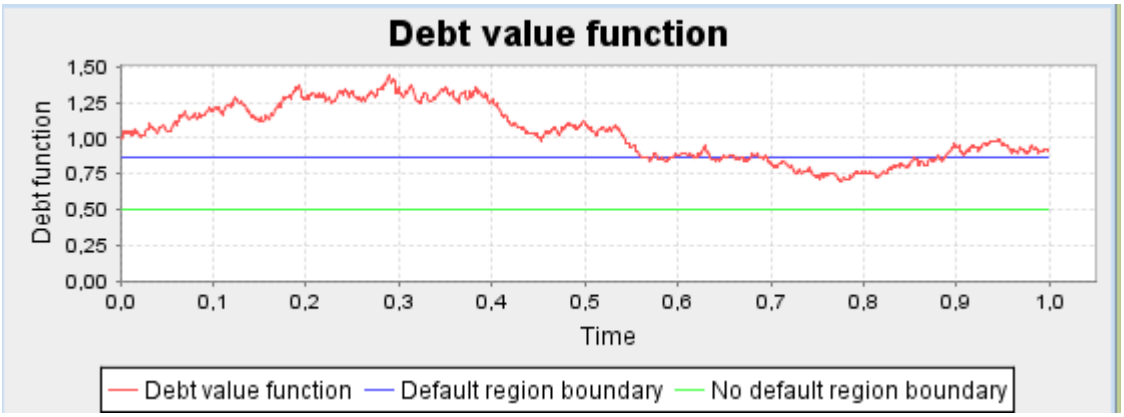


Figure 7 debt value function when (β) increased

As the retaining rate β reach to limit of one (as we calculate it here with 0.99999) the no default-wealth region boundary = 99999, and the default-wealth region boundary = 4077.5. While region boundaries, ζ_* and ζ^*, will be equals to 0.00001 and 0.00006 respectively. Inspecting the figures 8 and 9 below give as an idea about that:

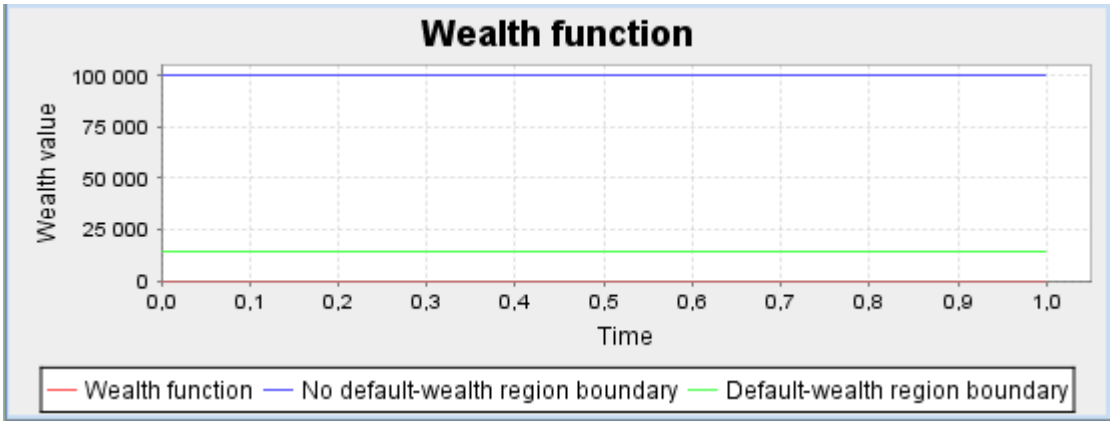


Figure 8 Wealth function when (β) reach to limit of one

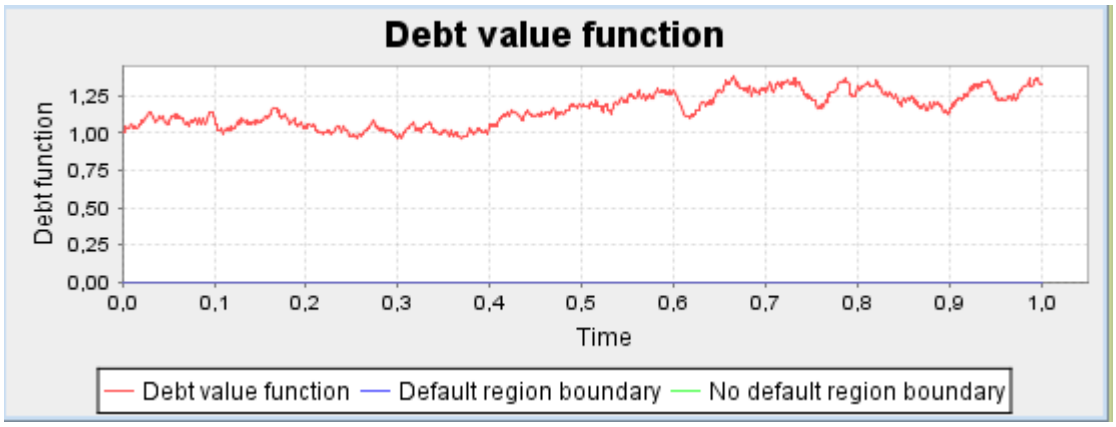


Figure 9 debt value function when (β) reach to limit of one

The effect of reaching the retaining rate limits make the optimal wealth value to be very small, around 1.35, and this is indeed explain the reason behind disappearing of wealth function from the Figure 8.In addition to that the probability of default, thus, will be (1) that's mean a surly default and explain also the vanish of the resistance region in Figure 9 which in fact leads to extend the default region over all the sates and incur the maximal cost of default in spite of the value of the debt, D(0), equal to zero.

If we increase the proportional cost parameter, λ, from 0.2 to 1.0 then we succeed to decrease the probability of default from 0.1432 down till 0.0098, in addition getting an increment of the region boundary value ζ^* from 1.61 up till 2.45 and lower ζ_* from 0.79 down till 0.72, resulting to expand the resistance region in both directions and raise the optimal wealth $W^*(T)$ value from 1.1 to 1.3 and sustain it to be over no default-wealth region boundary which guarantee a good result at the bad states. Figures 10, 11 illustrate the above results.

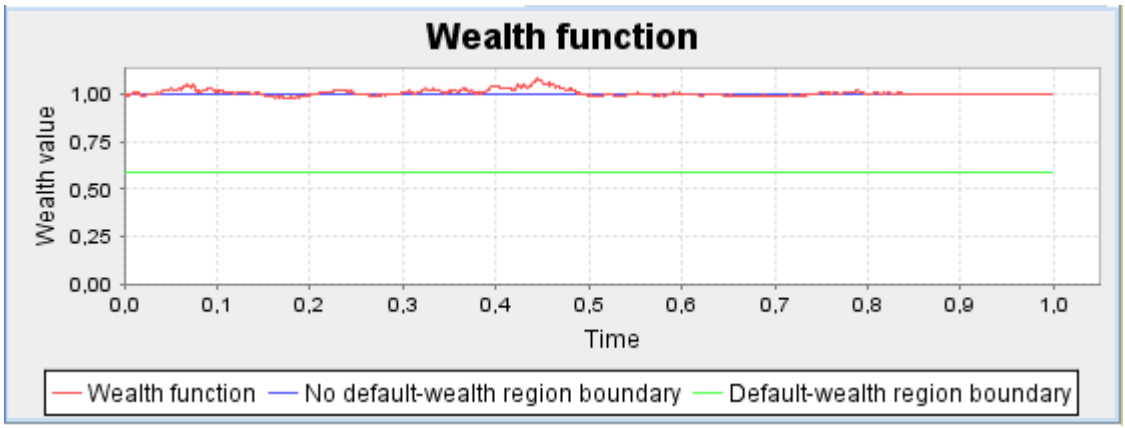


Figure 10 Wealth function when (λ) increased

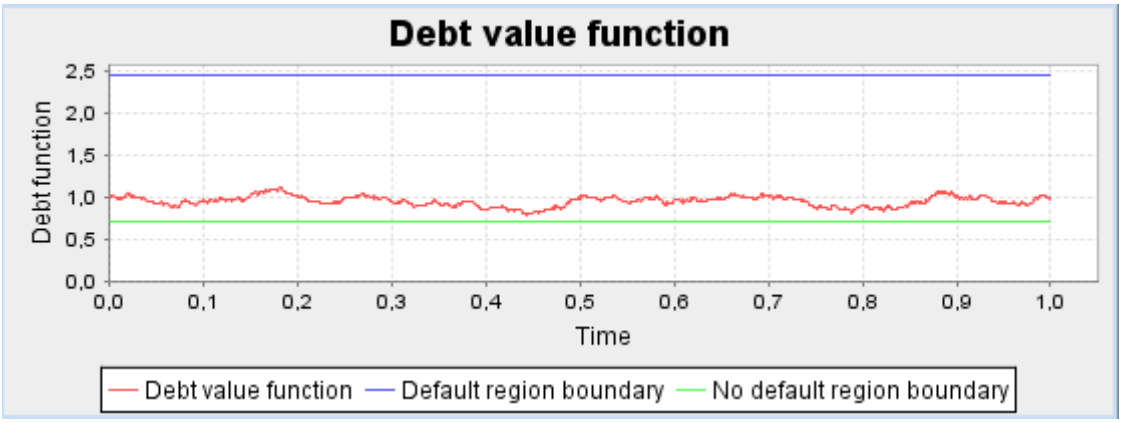


Figure 11 debt value functions when (λ) increased

The effect of increasing the fixed costs, ϕ, from 0.1 to 0.5 reflected directly by extends the resistance region $\zeta_{*}\zeta^{*}$ from (0.79, 1.61) till (0.71, 4.6) respectively and produce a sharp reduction in the probability of default as it's reach's till 0.0001 in addition to eliminating the burden of fixed default costs, which is shown in Figures 12 and 13 below:

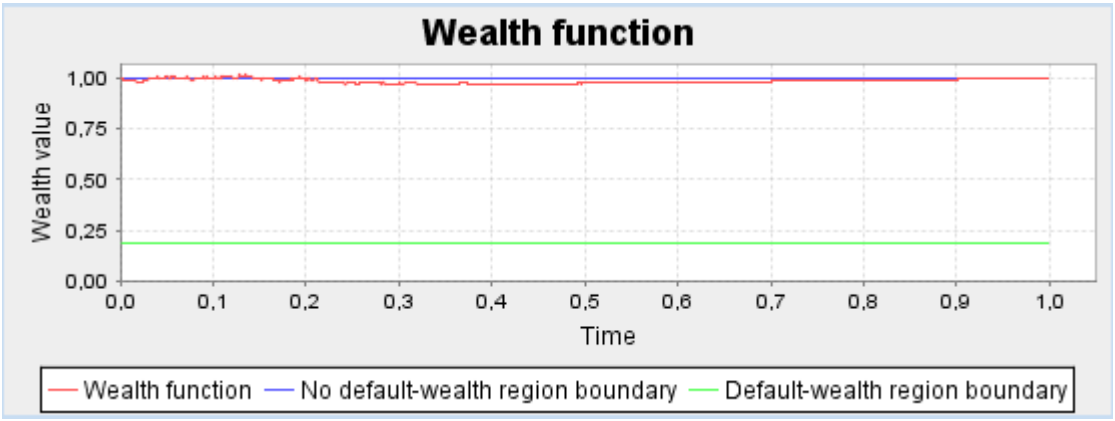


Figure 12 Wealth function when (ϕ) increased

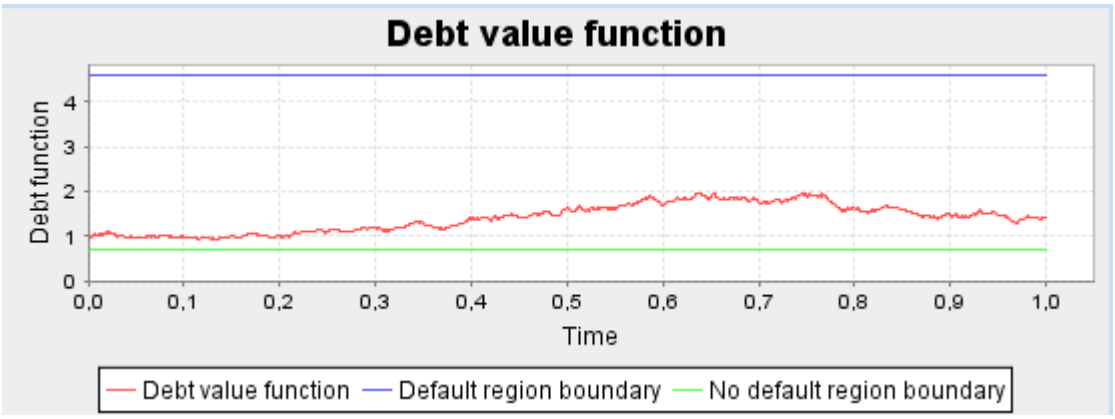


Figure 13 debt value functions when (ϕ) increased

The discontinuity in optimal wealth function disappear when the value of the fixed costs reach to very small amount,0.000001, participated in shrinking of both region boundaries ζ_*, ζ^* to be(0.89, 1.07) and default boundaries $W^*(\zeta_*)$ and $W^*(\zeta^*)$ to be (1, 0.9986) as it is shown in Figure 14 clearly .

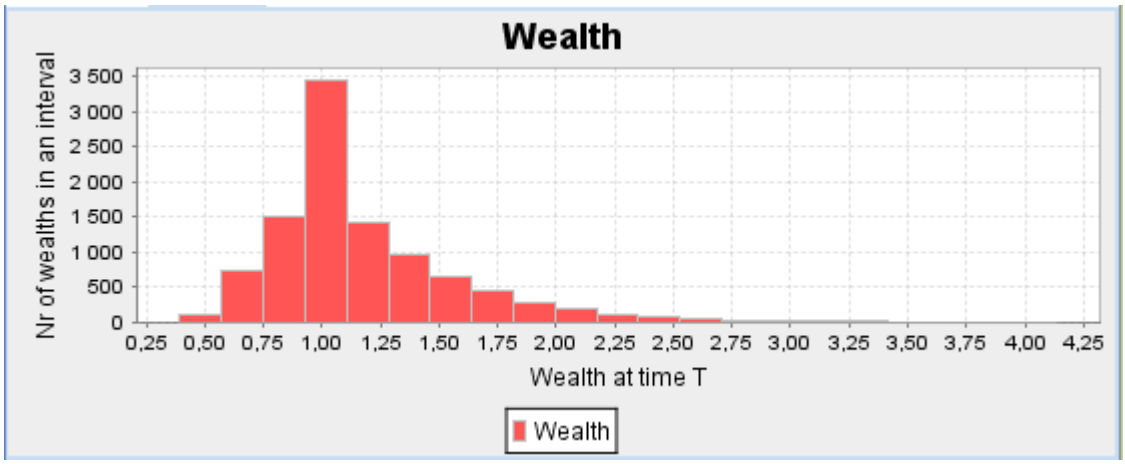


Figure 14 wealth histograms when (F) is very small

3.1.2. Effect of Wealth relative Frequency Histogram

Let know examine the wealth relative frequency histogram which represent the number of trajectories, 10000, simulated to calculate the optimal wealth's value $W^*(T)$ at time T.
A chart is build by subdividing the axis of optimal wealth at time T into intervals of equal width.
Over each interval construct a rectangle, such that the height of the rectangle is relative to the part of the total number of the wealth baths (trajectories) falling in each cell. The relative frequencies (fraction of total number of wealth baths), compute for each interval, are shown in Figure 15.
Our objective of depicting the histogram is to make inferences, and the accompanying wealth histograms are sufficient for our objectives.
The probabilistic interpretation that can be derived from the frequency of wealth at time T histogram is that the likelihood that it will fall in a given interval is proportional to the area under the histogram lying over that interval [6].
Correspondingly the wealth histogram analogous to the shape of the probability density function of the terminal wealth policies that was mentioned in Basak and Shapiro models.

The area of the high rectangle in the Figure 15 represents a similar proportional expression
To the probability mass build up in the optimal wealth at time T. Consequently it is clear that this rectangle constructed at the interval contains the mid point = 1, correspondent to the value of the no default-wealth region boundary $W^*(\zeta_*) = \dfrac{\beta F}{(1-\beta)}$. The gap between the rectangles in

the wealth histogram correspond to discontinuity with no states cover optimal wealth values between $W^*(\zeta_*)$ and $W^*(\zeta^*) = \boldsymbol{I}\left\{\frac{\beta y \zeta^*}{[\beta + \lambda(1-\beta)]}\right\}$.

Viewing Figure 15 shows the rectangles in the no default-wealth region move to the right with decreasing area of these rectangles meaning more wealth with less probability.
Interval that consist of the value of the mean wealth = 1.12223 reflect the heights probability. Whereas intervals hold the default-wealth region $W^*(\zeta^*) = 0.5875$ with two rectangles decreases when moving to the left meaning less wealth with less probability.

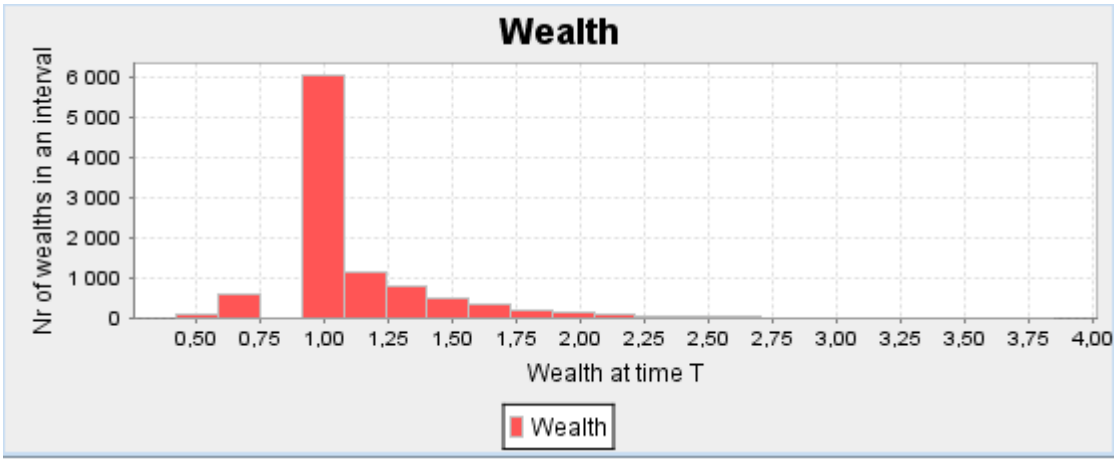


Figure 15 wealth histogram

3.1.3. Effect of Investment Policy

Often it is possible to gives a concise and helpful description of a situation by drawing a graph. For example, Figure 16 describes investment values of the fraction of wealth's invested in each opportunity (here we have first opportunity) at various times up till debt maturity.

The graph shows that the longer the time the more the borrower's risky investment functions $\theta^*(t)$ .So the fractions of the wealth in risky investment grow larger as time passes.
But there is maximum size that value cannot exceed nearby maturity. The maximum size reflect the restrictions imposed, when default been likely, by the obligations fulfillment of the borrow debt contract at maturity. This enforces the borrowers and discourages him from taking additional risk and the function decay down till almost risk less position.

This policy is adapted when the program simulated trajectories that lead to no default positions and the program can shows also opposite graphs that lead to default positions (of course this is happen when the user tries to simulate several trajectories by pushing the start button) which may gives the user a high flexibility and advantage when he or she working within this program.

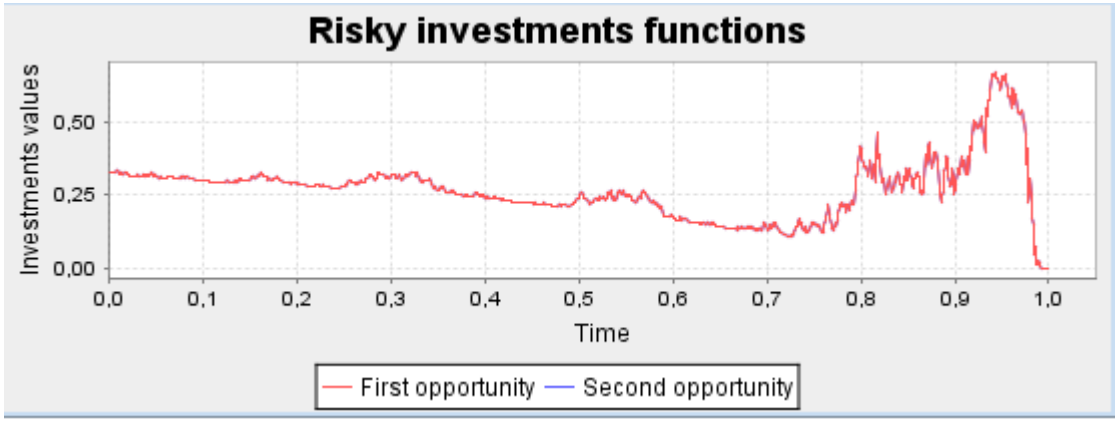


Figure 16 risky investment functions for simulated trajectories that lead to no default positions

In Figure 17 the program have been drawn the graph that depicts the risk exposure function $m^*(t)$.

As we can see the function fluctuated through the time. The graph describes a change that is taking place. The amount of money invested in the risky opportunities is changing as time passes. Program provides mathematical tools to study each of these changes in a quantitative way.

From comparing the two Figures 16 and 17 we can illustrates the relationship between the value of amount invested in the risky opportunity and the degree of risk that borrowers encounter. Where amount invested in the risky opportunity proportional with borrowers risk exposure, as the two graphs shows that clearly.

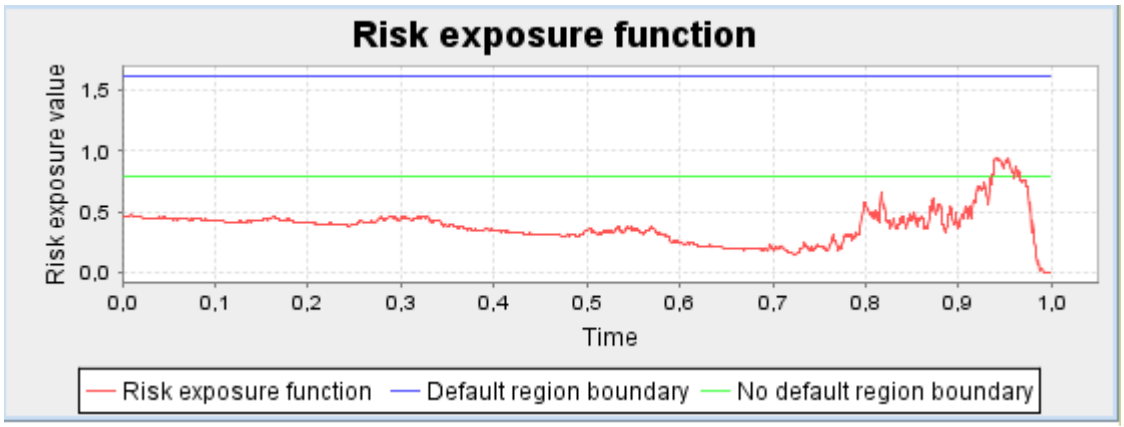


Figure 17 risk exposure function for simulated trajectories that lead to no default positions

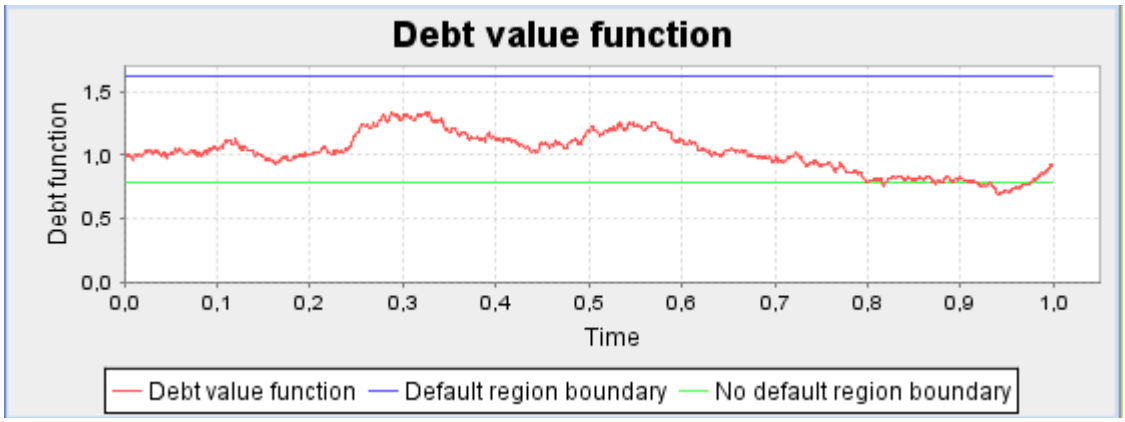


Figure 18 debt value functions for simulated trajectories that lead to no default positions

The simulated trajectories that depicts the graphs of the functions $\theta^*(t), m^*(t)$ and $\zeta(t)$ reveal the borrowers optimal policy in the times interval [0.0 - 0.8] characterized by a declining of the portion invested in risk opportunity (opportunities) because the borrowers seeking to secure the default-resistance level and guarantee the debt value function $\zeta(t)$ left between default boundaries $\zeta_*(t), \zeta^*(t)$ as shown in Figures 16,17and 18 above.

As the time pass, specifically in a time interval [0.8 – 0.95], the borrower's seek to rise the risk portion of invested wealth and consequently led to grow the borrowers risk exposure.
The need of taking more risk generated from necessitates covering potential default cost as well as default-resistance costs.

Of course anticipating favorable economic conditions induce for more investment in a risky opportunities but this will not continuous until the end.
As time headed for maturity, of the debt contract, nearby and threaten of default is very likely. The borrower prefers risk less investment that may be helps him at least to fulfill his obligations or cover the potential default costs.

3.2. Sensitivity analysis of risk exposure function to change in model's parameters

One of unique or distinct feature the program that can provide till users is his ability at conduct the sensitivity analysis among different parameters used in the model and displays the mutual effects and result graphically and numerically.
Figure 19 display the parameters used in calculating risk exposure function and the associated graph of this function, for a trajectory have been chose randomly.

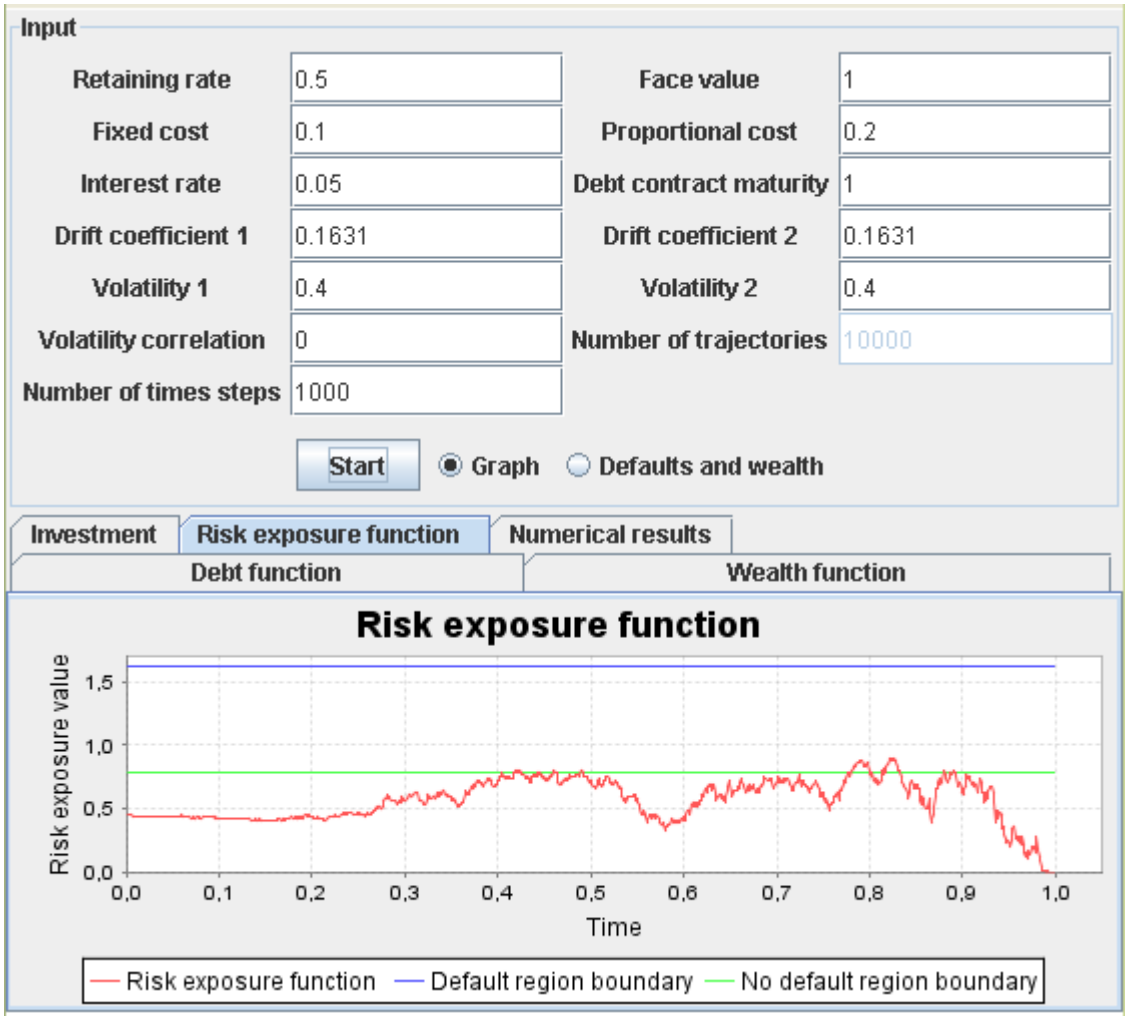


Figure 19 Java applet viewer: Risk exposure function

As it mentioned before when debt maturity coincides with borrower's planning horizon $(T = T')$.Then values of $\gamma = 1$ and $W(0) = 1, \|k\| = 0.4$. Then: $\zeta_* = 0.79047, \zeta^* = 1.61444$
Additionally, effected parameters on the exposure functions can be divided into two types:

First: debt-contract parameters (F, β)
Second: default-costs parameters (ϕ, λ)
Herewith, we are going to demonstrating these effects as following:

3.2.1. Effect of debt-contract parameter's on risk exposure function

To illustrate the effects of changes in values of (F, β), Figures 20, 21, 22 and 23 and schedules (1, 2, 3, and 4) display the situation when increasing and decreasing the values of (F) by 2 and 0.7 respectively

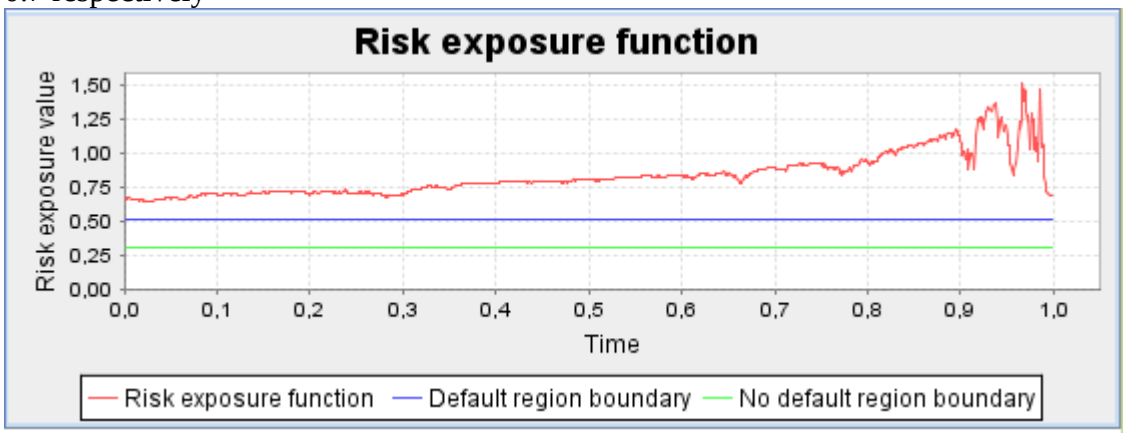


Figure 20 for F = 2

Default-wealth region ...	1.40186	Default region boundary	0.51263
No default-wealth regi...	2	No default region boun...	0.29943
Optimal wealth	1.16711	Probability of default	1
Wealth mean	1.16695	Default mean	0.001
Wealth variance	0.11048	Default variance	0
Wealth quantile 10%	0.8256	Default quantile 10%	0.001
Wealth quantile 25%	0.9305	Default quantile 25%	0.001
Wealth quantile 50%	1.09554	Default quantile 50%	0.001
Wealth quantile 75%	1.31037	Default quantile 75%	0.001
Wealth quantile 90%	1.54434	Default quantile 90%	0.001

Table 1 for F = 2

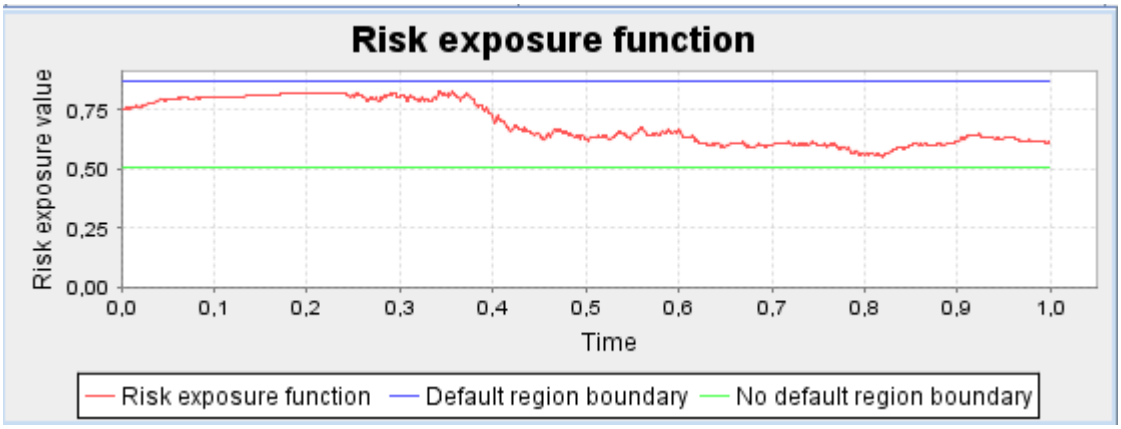


Figure 21 for β = 0.6

Default-wealth region ...	0.98673	**Default region boundary**	0.86859
No default-wealth regi...	1.5	**No default region boun...**	0.50416
Optimal wealth	1.23238	**Probability of default**	1
Wealth mean	1.23204	**Default mean**	0.001
Wealth variance	0.13134	**Default variance**	0
Wealth quantile 10%	0.78392	**Default quantile 10%**	0.001
Wealth quantile 25%	0.92295	**Default quantile 25%**	0.001
Wealth quantile 50%	1.13285	**Default quantile 50%**	0.001
Wealth quantile 75%	1.5	**Default quantile 75%**	0.001
Wealth quantile 90%	1.5	**Default quantile 90%**	0.001

Table 2 for β = 0.6

Increasing F form 1 to 2 and β from 0.5 to 0.6 correlated with lower in default boundary till $\zeta_* = 0.29943, \zeta^* = 0.51263$ in case changing the value of F and till $\zeta_* = 0.50416, \zeta^* = 0.86859$ if changing β.

Subsequently, the probability of default rises sharply and reaches to one. These changes also have an impact on shrinking the resistance region which is displayed in the above Figures and schedules.

Additionally, decrease the value of F till 0.7 and β till 0.4; generate an opposite effect on the risk exposure function and the above mentioned parameters that participated in build up this function as displayed in Figures 22, 23 and schedules 3 and 4.

Risk exposure function generally become sensitive to the reductions in the values of F and β. The likelihood of default almost vanish(equal 0.0012 When decreasing F and 0.001 with β decreasing) which contribute in enhance the financial position's of the borrower's by lowering the risk and allowing avoidance of penalties of default through wideness the resistance region.

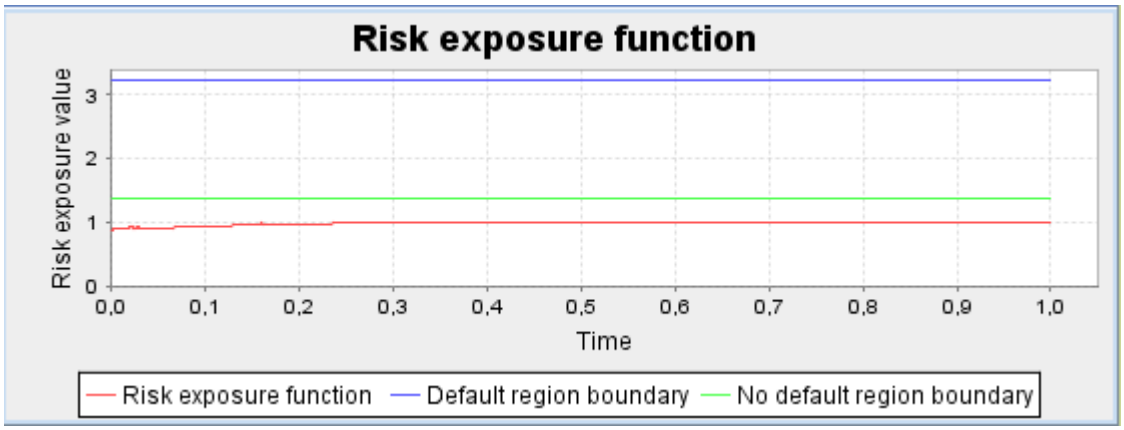


Figure 22 for F = 0.7

Default-wealth region ...	0.36091	**Default region boundary**	3.22687
No default-wealth regi...	0.7	**No default region boun...**	1.38646
Optimal wealth	1.20993	**Probability of default**	0.0012
Wealth mean	1.20956	**Default mean**	0.88182
Wealth variance	0.22958	**Default variance**	0.00923
Wealth quantile 10%	0.7	**Default quantile 10%**	0.651
Wealth quantile 25%	0.84143	**Default quantile 25%**	0.763
Wealth quantile 50%	1.10217	**Default quantile 50%**	0.919
Wealth quantile 75%	1.44876	**Default quantile 75%**	0.944
Wealth quantile 90%	1.84163	**Default quantile 90%**	0.944

Table 3 for F = 0.7

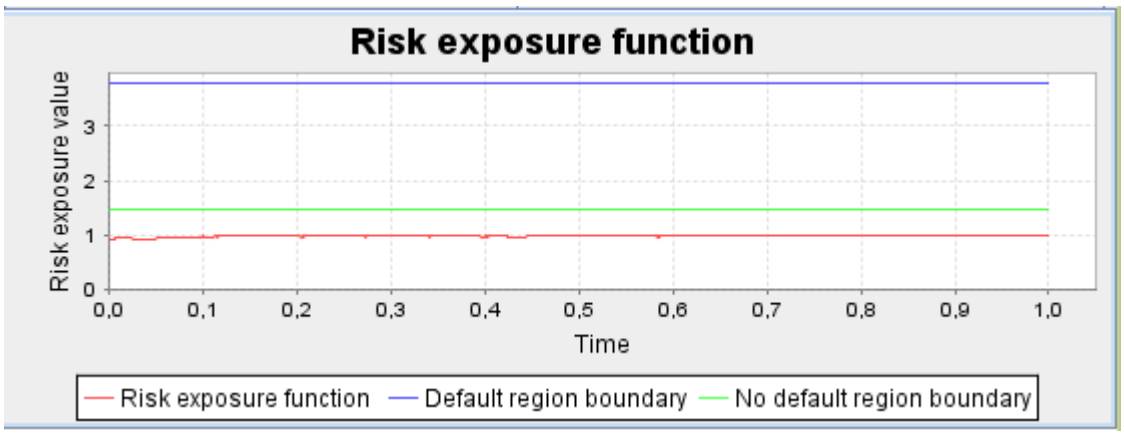


Figure 23 for β = 0.4

Default-wealth region ...	0.33665	Default region boundary	3.77887
No default-wealth regi...	0.66667	No default region boun...	1.46788
Optimal wealth	1.21905	Probability of default	0.0001
Wealth mean	1.21862	Default mean	□
Wealth variance	0.24055	Default variance	-0
Wealth quantile 10%	0.67004	Default quantile 10%	0.808
Wealth quantile 25%	0.853	Default quantile 25%	0.808
Wealth quantile 50%	1.11244	Default quantile 50%	0.808
Wealth quantile 75%	1.45547	Default quantile 75%	0.808
Wealth quantile 90%	1.85875	Default quantile 90%	0.808

Table 4 for β = 0.4

3.2.2. Effect of default-costs parameter' on risk exposure function

The source of risk is not only restricted with debt-contract parameter's, but it's expand to comprise other's sources.
Risk also occurs from vary default-costs parameters, specifically, default fixed cost ϕ and default proportional cost λ.
Performing sensitivity analysis via this program provides a cushion against default costs risk.

Figures 24, 25 illustrate the increasing of default costs and shows outcomes of these increments on the risk exposure function.
Rising fixed cost ϕ from 0.10 till 0.5. And proportional cost λ from 0.2 till 0.9 gives us immediate impact on risk exposure function reflected by reducing the probability of default till 0.0001 and extending resistance region by changing the default boundaries
from $\zeta_* = 0.79047, \zeta^* = 1.61444$ till $\zeta_* = 0.7144, \zeta^* = 4.59221$.

The above changing, in fact, improves the borrower's financial positions by reducing the risk and likewise default on financial obligations.

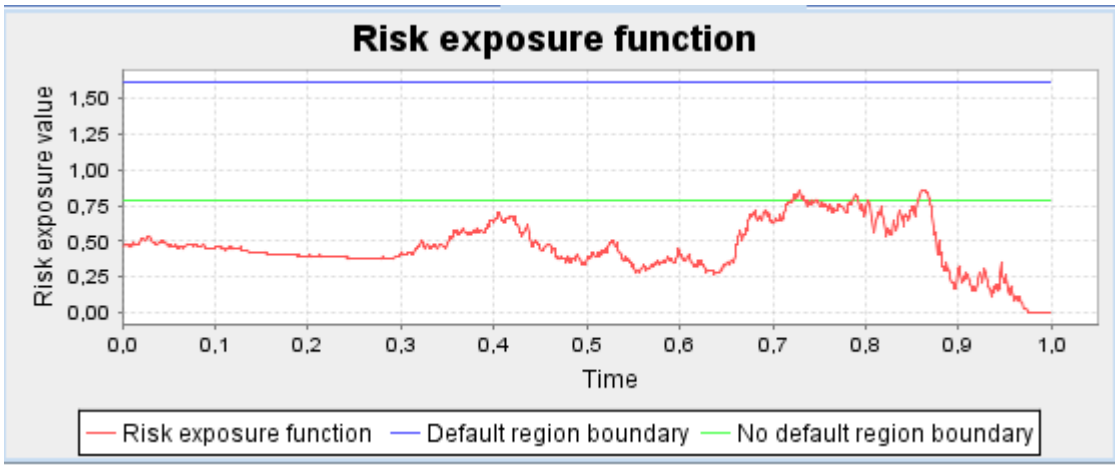


Figure 24 for ϕ = 0.5

Default-wealth region ...	0.18668	**Default region boundary**	4.59221
No default-wealth regi...	1	**No default region boun...**	0.7144
Optimal wealth	1.09307	**Probability of default**	0.0001
Wealth mean	1.09283	**Default mean**	□
Wealth variance	0.04611	**Default variance**	-0
Wealth quantile 10%	1	**Default quantile 10%**	0.989
Wealth quantile 25%	1	**Default quantile 25%**	0.989
Wealth quantile 50%	1	**Default quantile 50%**	0.989
Wealth quantile 75%	1.05427	**Default quantile 75%**	0.989
Wealth quantile 90%	1.35004	**Default quantile 90%**	0.989

Table 5 for ϕ = 0.5

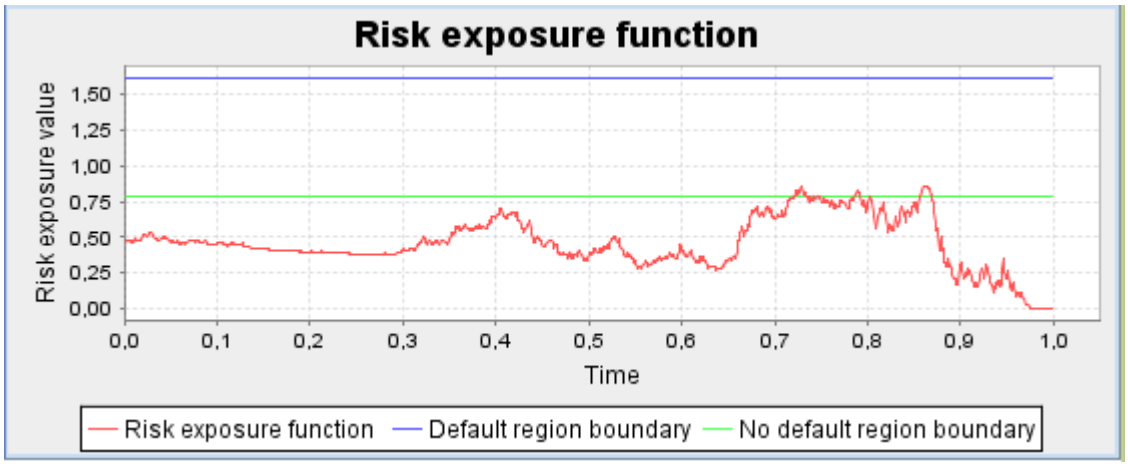


Figure 25 for λ = 0.9

Default-wealth region ...	0.58753	Default region boundary	2.33542
No default-wealth regi...	1	No default region boun...	0.72218
Optimal wealth	1.10562	Probability of default	0.0145
Wealth mean	1.10536	Default mean	0.79575
Wealth variance	0.05912	Default variance	0.02132
Wealth quantile 10%	1	Default quantile 10%	0.568
Wealth quantile 25%	1	Default quantile 25%	0.682
Wealth quantile 50%	1	Default quantile 50%	0.835
Wealth quantile 75%	1.08173	Default quantile 75%	0.917
Wealth quantile 90%	1.39015	Default quantile 90%	0.962

Table 6 for λ = 0.9

Figures 26, 27 and schedules 7, 8 shows adverse effects on risk exposure function, comparisons with increment cases. When the values of the parameter's ϕ and λ falling.

Probability of default well rise from 0.0001 till 0.4765 with decreasing of ϕ, and 0.0145 till 0.2478 declining ϕ

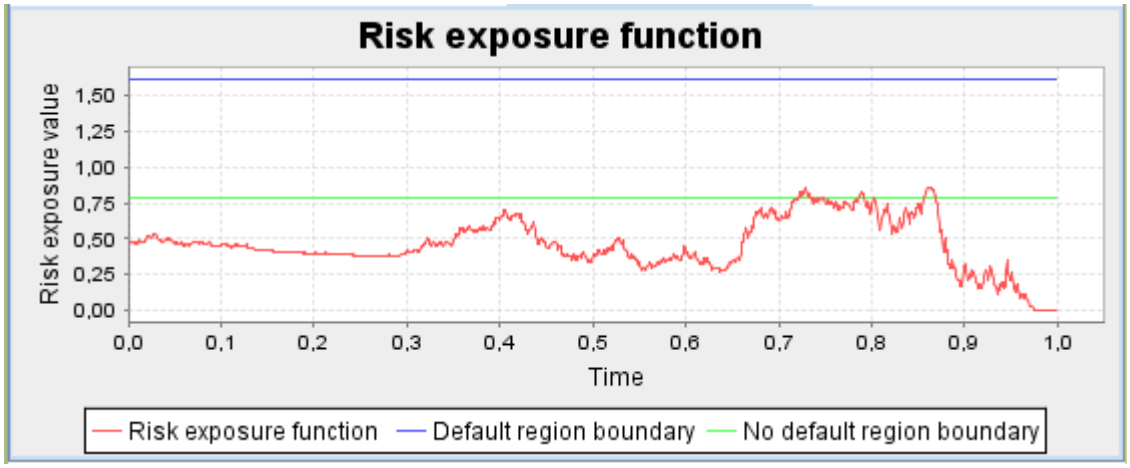


Figure 26 for ϕ = 0.015

Default-wealth region ...	0.83184	Default region boundary	1.24261
No default-wealth regi...	1	No default region boun...	0.86138
Optimal wealth	1.14477	Probability of default	0.4765
Wealth mean	1.14437	Default mean	0.34673
Wealth variance	0.1444	Default variance	0.05995
Wealth quantile 10%	0.76466	Default quantile 10%	0.088
Wealth quantile 25%	1	Default quantile 25%	0.148
Wealth quantile 50%	1	Default quantile 50%	0.28
Wealth quantile 75%	1.276	Default quantile 75%	0.495
Wealth quantile 90%	1.63065	Default quantile 90%	0.741

Table 7 for $\phi = 0.015$

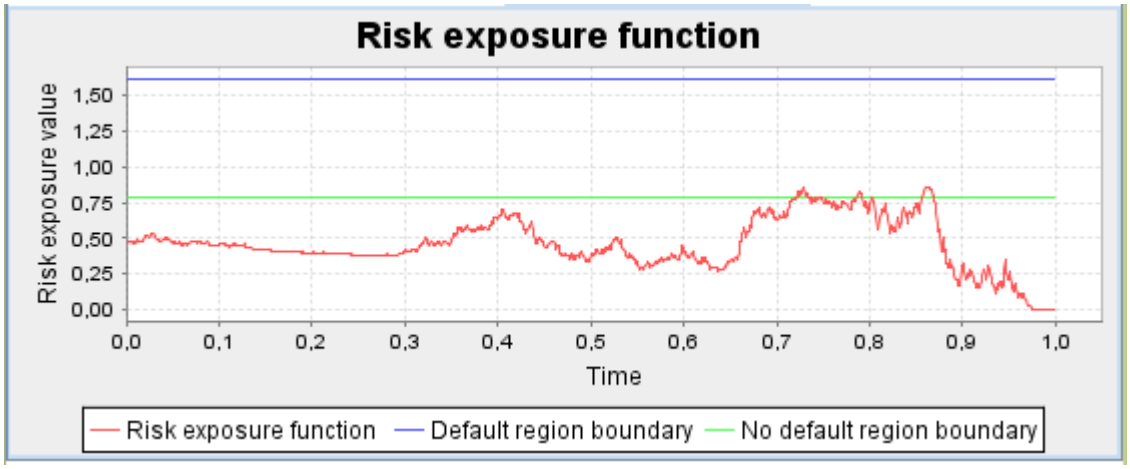


Figure 27 for $\lambda = 0.01$

Default-wealth region ...	0.58752	Default region boundary	1.44672
No default-wealth regi...	1	No default region boun...	0.84156
Optimal wealth	1.13676	Probability of default	0.2478
Wealth mean	1.13645	Default mean	0.50494
Wealth variance	0.13199	Default variance	0.0593
Wealth quantile 10%	0.68301	Default quantile 10%	0.201
Wealth quantile 25%	1	Default quantile 25%	0.301
Wealth quantile 50%	1	Default quantile 50%	0.478
Wealth quantile 75%	1.2547	Default quantile 75%	0.699
Wealth quantile 90%	1.59282	Default quantile 90%	0.864

Table 8 for $\lambda = 0.01$

4. Java Applet for Basak-Shapiro model

4.1. Introduction

Java programming language enable user's to write an applet program that can be comprised in an HTML page. Similarly to the way an image is including in a page. When one use a Java technology-enabled browser to view a page that has an applet's. The browsers Java Virtual Machine (JVM) transfer and execute the applets cod's to your system [7].

Applet program is a special type of Java technology possibly downloads and run from internet. This program entrenched inside a web-page and operates in background of the browser. An applet is a subclass of the *Java.applet.Applet* class, supplied with standard interface between the applet and the browser context.

The Swing package was the code name of the project that developed the new components. It is used to refer to the new component and related API that is apart of the Java Foundation Classes (JFC) in the Java platform.
These classes contain a group of characteristics to aid user's build GUI; Swing provides a special subclass of Applet, named *Javax.swing.JApplet.* That is used in all applets programming using Swing components from buttons to split panes and tables to construct their GUIs.

First step start by compiler for Java, which translate a source code into in-between language named *Java byte codes.* Second step is that the Java byte code interpreted by the interpreter on Java platform and run on the computer.

There are four main methods in the Applet class (init, start, stop, destroy) on which any applet is built.

Init methods is projected and designed for all types of initialization requested for Applet.
Start methods called after init method automatically, and at any time user returns to the page including the applet after opening other pages.

Stop methods is called automatically at any time the user left the page consisting Applet:

Destroy method, specifically, called when the browser closed normally.

Basak-Shapiro Applet, as extension of Applet class, uses different types of objects. These interact and communicate with each others. Such as Panels, Chart Panel, buttons, radio buttons, tapped Panes, labels, and text fields among other objects. The objects are displayed in a border grid Layouts.

4.2. Applet: User Guide and Parameters

BasakShaoiro2 class is extended from applet class. The objects are created from this class which help as also to perform the necessary control actions needed in this program. Generally, border layout used in input, output and control panels and main panel have been utilized, grid layout. These layouts give high flexibility for placement of objects in different location

4.2.1. J Pane

Input, control and output panels created for java applet Basak-Shapiro, Which is used in calculation the credit risk and others variables, will be placed in North, Centre and South location. The applet viewer shown in the Figure 28 below is obtained building, compiling and running the codes as it mentioned in Appendix for the BasakShapiro2 model.

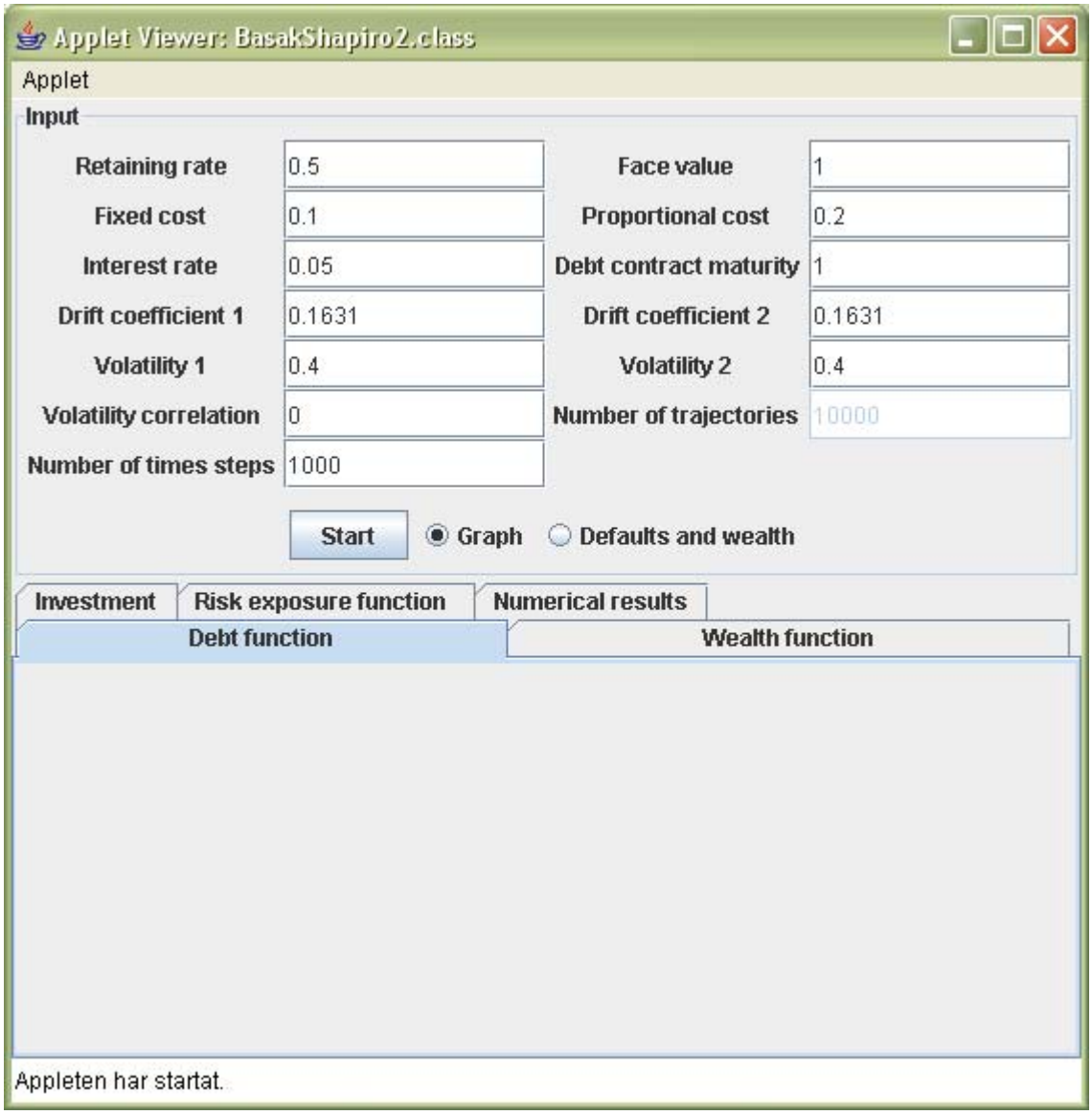


Figure 28 Java Applet Viewer: BasakShapiro2 credit risk model

4.2.1.1. Input panel

At the north of the applet viewer BasakShabiro2.class as displayed in Figure 28 located the input panel. This panel contains the input data values of the model parameters which set by default and initially appear when the user build and run Applet.
However, user can insert the suitable values in each relevant text filed revealed in input data panel, for example, retaining rate, face value, fixed cost, proportional cost and excreta .
In case a user entered negative number in one of data text fields, or chose to enter numbers more than ± 1 in a volatility correlation text field, and press the start button the following error message is obtained. In order to correct the problem clicks the OK button to set that value to default. Figure 29 display this situation.

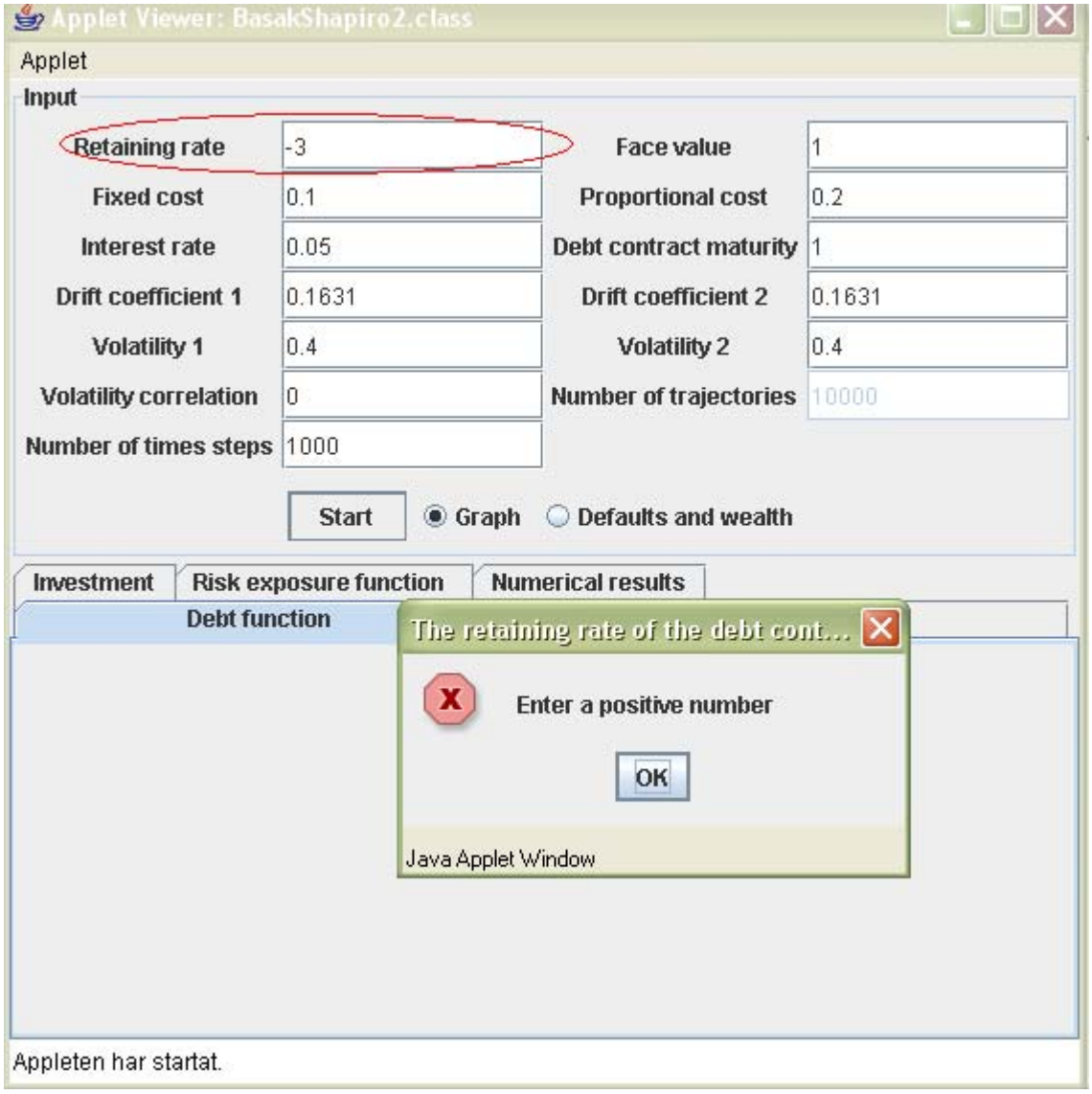


Figure 29 error messages for retaining rate value

4.2.1.2. Control panel

The control panel has two alternatives, Graph and defaults and wealth's radio buttons. When the user chose the graph button, number of trajectories text filed is not active. But By choosing the default and wealth's radio button the number of trajectories text field will activate. In addition the progress bar will appear. Figure 30 below illustrates this.

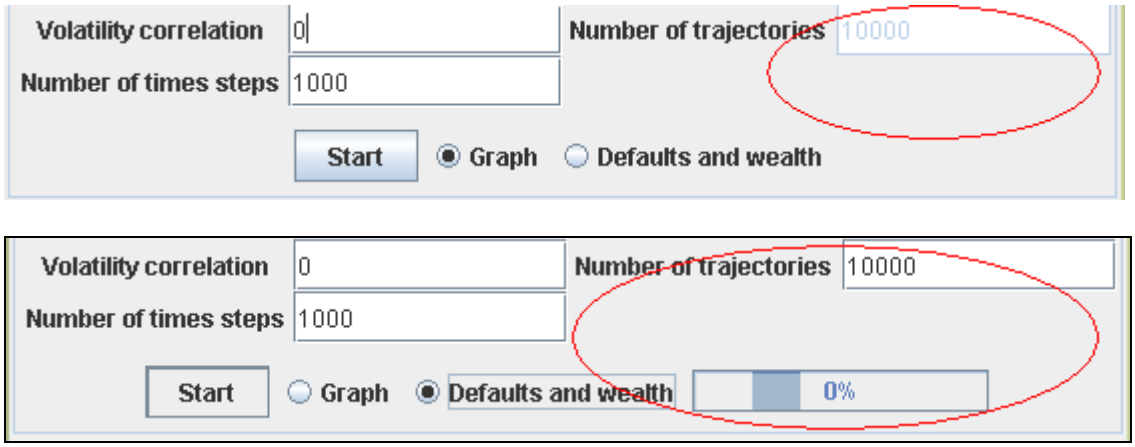


Figure 30 control panel

4.2.1.3. Output panel

The output panel displays both the numerical results and graphs of calculations. After choosing graph radio button and click the start button the output panel will be consist of debt function label that has been already activated, wealth function, investment, risk exposure function, numerical results labels. This is shown in Figure 31

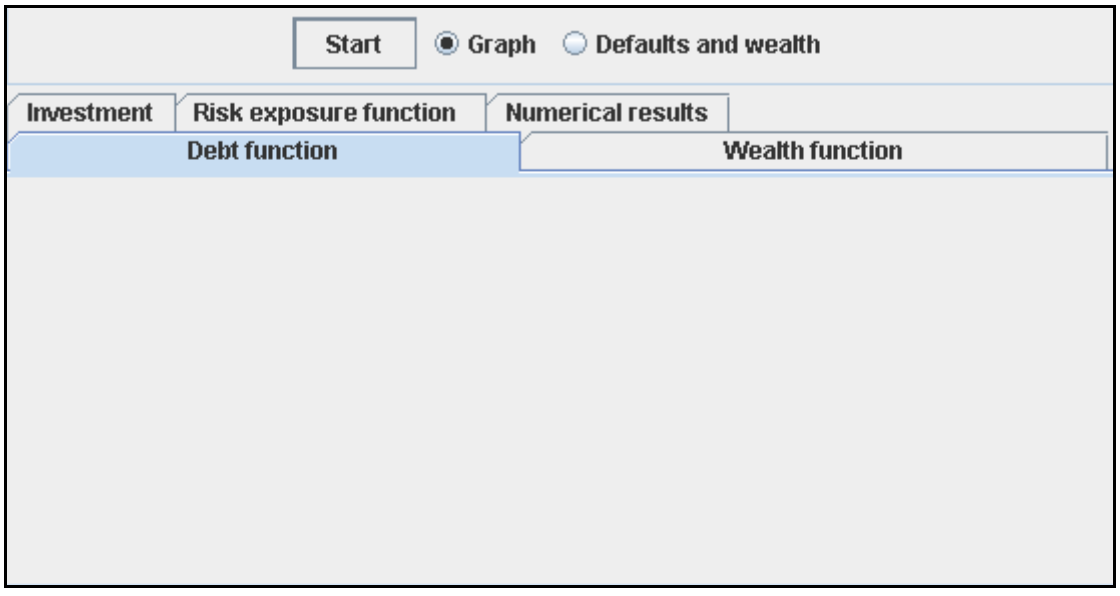


Figure 31 output panels

In case the user choose default and wealth's button, the output panel will be made of default label in activate appearance, wealth, joint distribution, risk exposure function, numerical results labels. This is shown in Figure 32

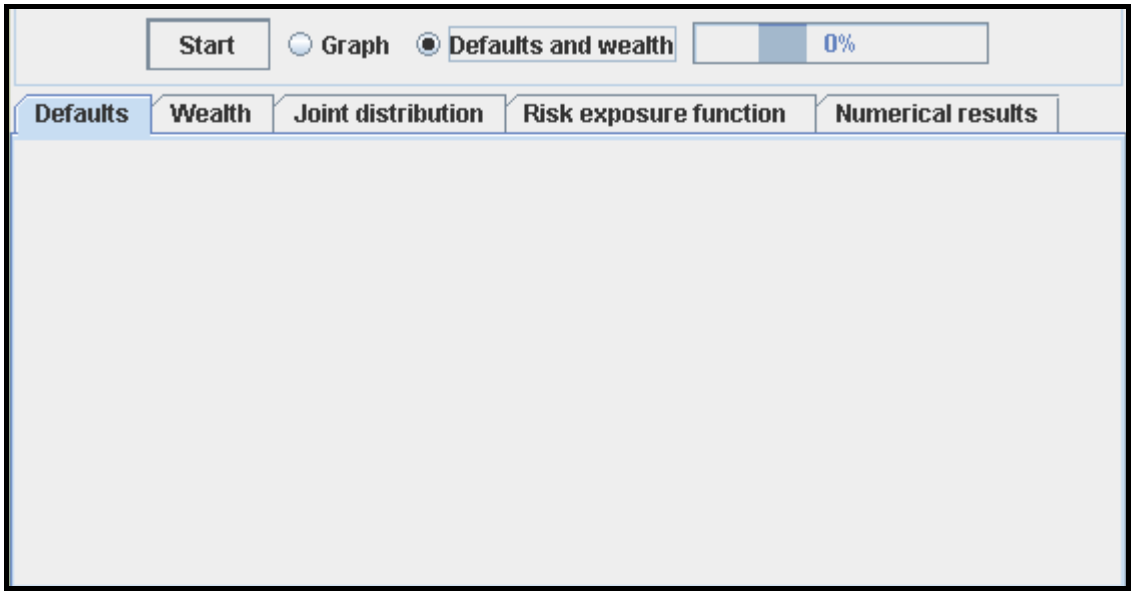


Figure 32 output panels

4.2.2. Debt function trajectory

Display the graphical lines representation of debt value function, default region boundary and no default boundary for one random trajectory calculated by the program.
X axis represent the time periods interval's and y axis is debt value. Figure 33 below shown this

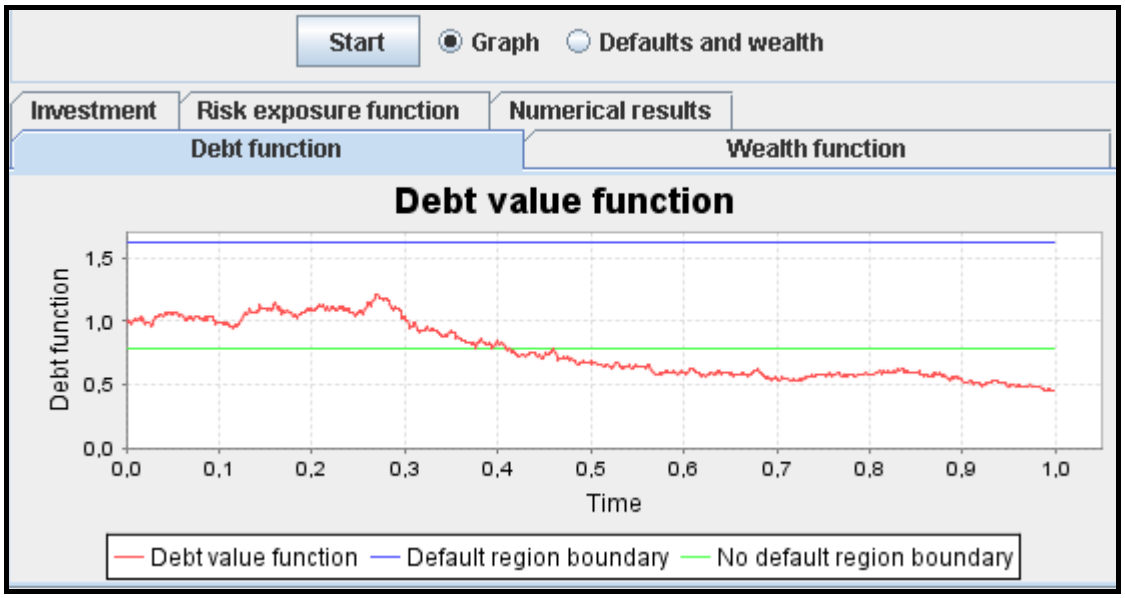


Figure 33 debt value functions

4.2.3. Wealth function

When the user click on wealth function label he or she will get Figure 34 below that display the graphical lines representation of wealth function, no default-wealth region boundary and default wealth region boundary for one random trajectory calculated by the program.
X axis represent the time periods interval's and y axis is wealth value.

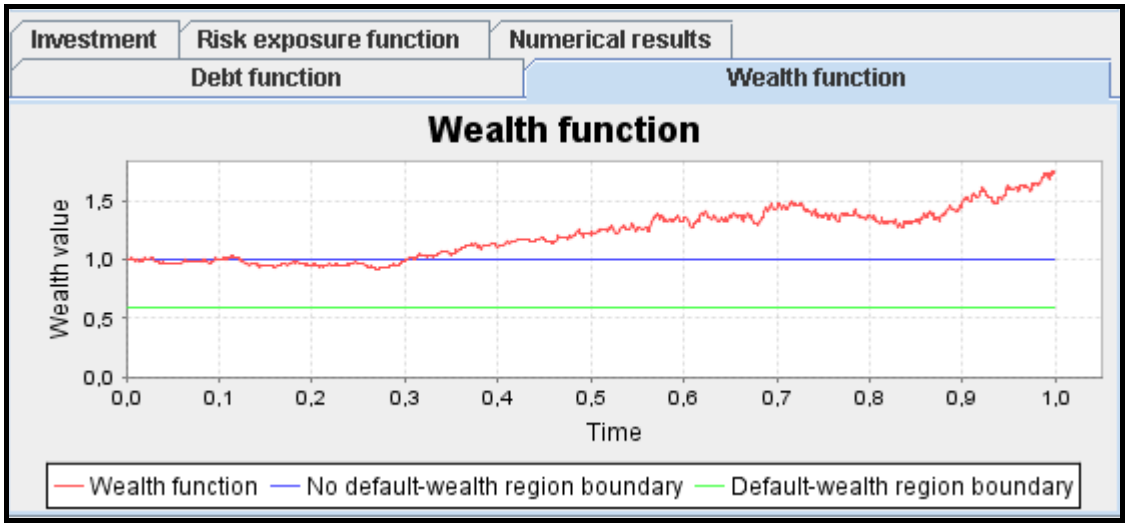


Figure 34 wealth function

4.2.4. Risk investment function

By clicking investment labels in output panel, the user will have Figure 35 that illustrate the investment opportunities policies graphs, where lines represent risk investment functions for first and second opportunity. X axis represent the time periods interval's and y axis is investment values for each opportunity.

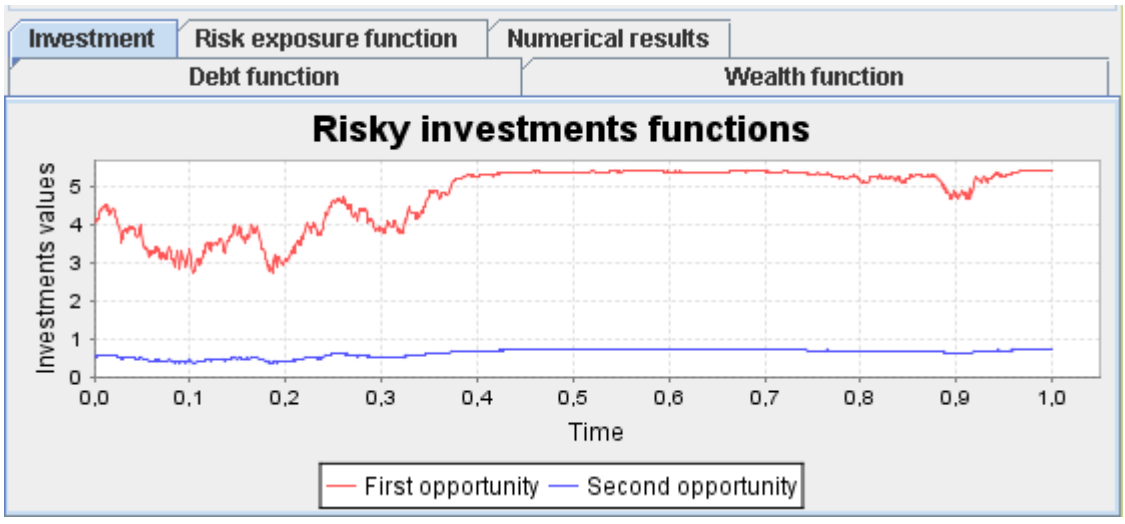


Figure 35 risk investment policy

4.2.5. Numerical results

This panel demonstrates the numerical values calculated by the program. These values concern the default-wealth region boundary and no default-wealth region boundary which was used in figure 16 for wealth function. And the values of default region boundary and no default region boundary which have been used in Figure 33 for debt value functions.
Figure 36 display this panel

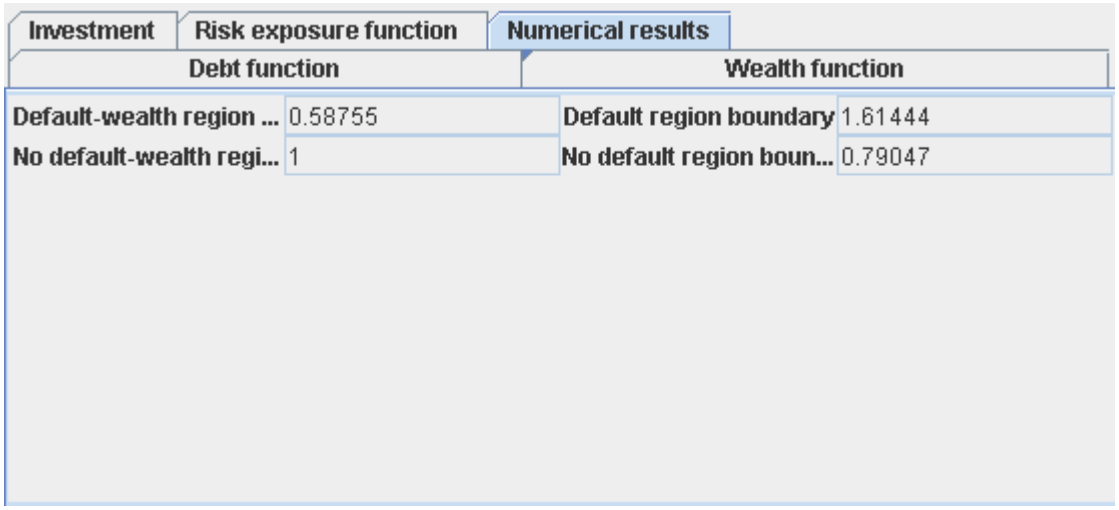


Figure 36 numerical results

If the user chooses default and wealth radio button and then click on start button in control panel he or she shall get the following:

4.2.6. Default Relative Frequency Histogram

This histogram represents the default trajectories simulated by the programs among the total numbers of trajectories chooses in input panel.
The code of calculating the numbers of histogram columns, as following:

```
int nbins1 = (int) Math.pow (defaultCounter, 0.333);
```

X axis represent the time periods interval's and y axis is number of default trajectories in an interval. Figure 37 below illustrated this.

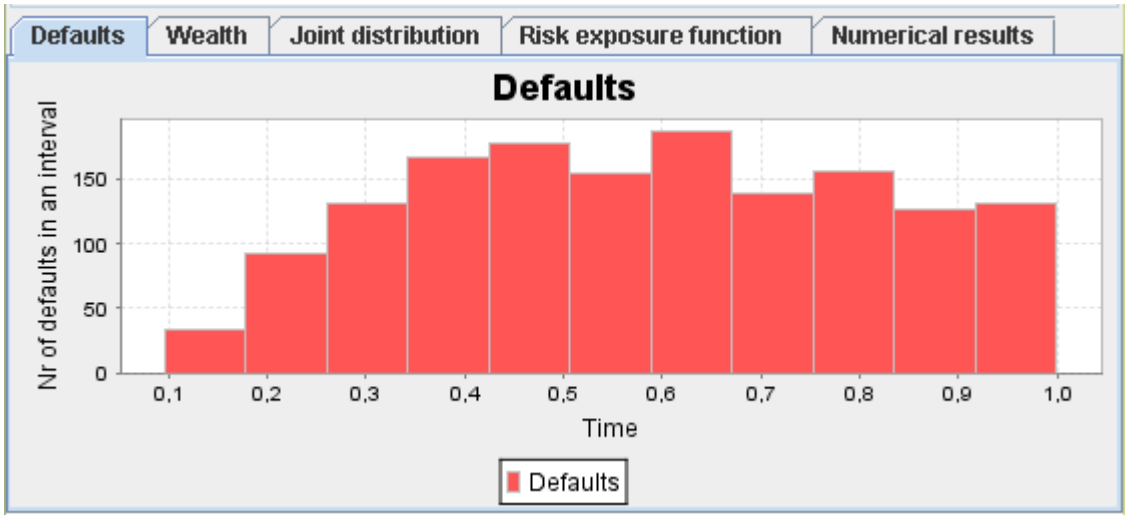


Figure 37 default relative frequency histogram

4.2.7. Wealth Relative Frequency Histogram

When the user click on wealth label in output panel she or he will get the wealth histogram as mentioned before.
The code of calculating the numbers of histogram columns, as following:

```
int nbins2 = (int) Math.pow (N, 0.333);
```

X axis represent wealth values intervals at maturity, time T, and y axis is number of wealth baths in an interval. Figure 38 shows that

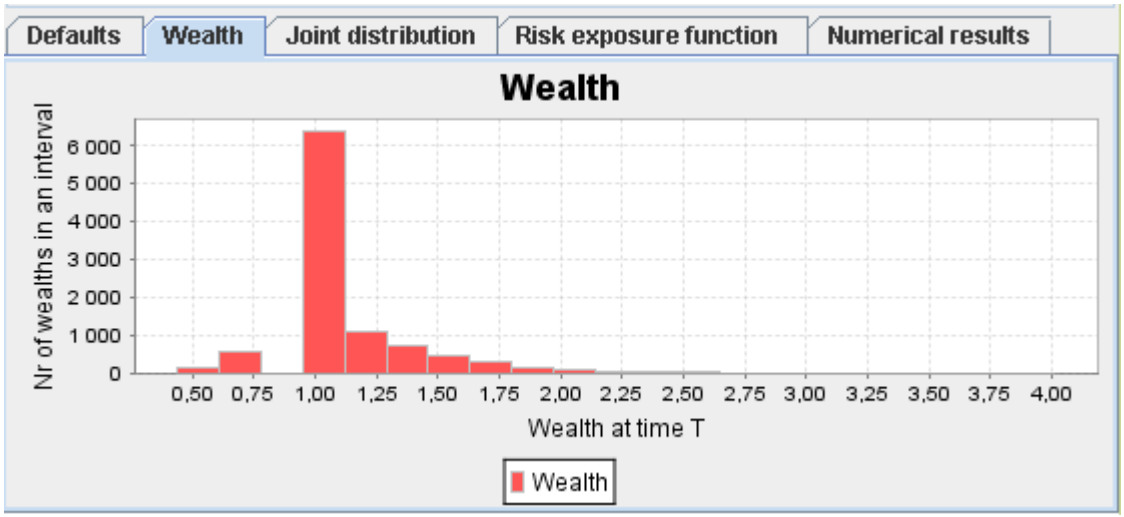


Figure 38 wealth relative frequency histograms

4.2.8. Joint Histogram

Joint histogram represents the bivariate density of the borrower's time –T optimal wealth, $W^*(T)$ when debt maturity coincides with the borrower's planning horizon $(T = T')$ with relation to the default wealth trajectories. Figure 39 shows that

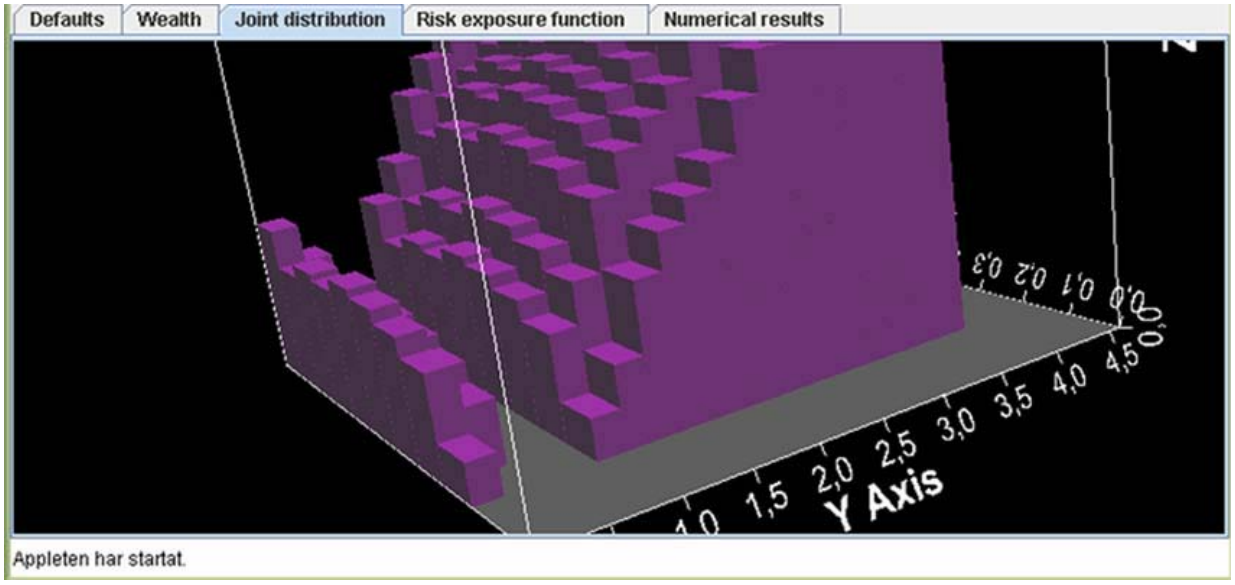
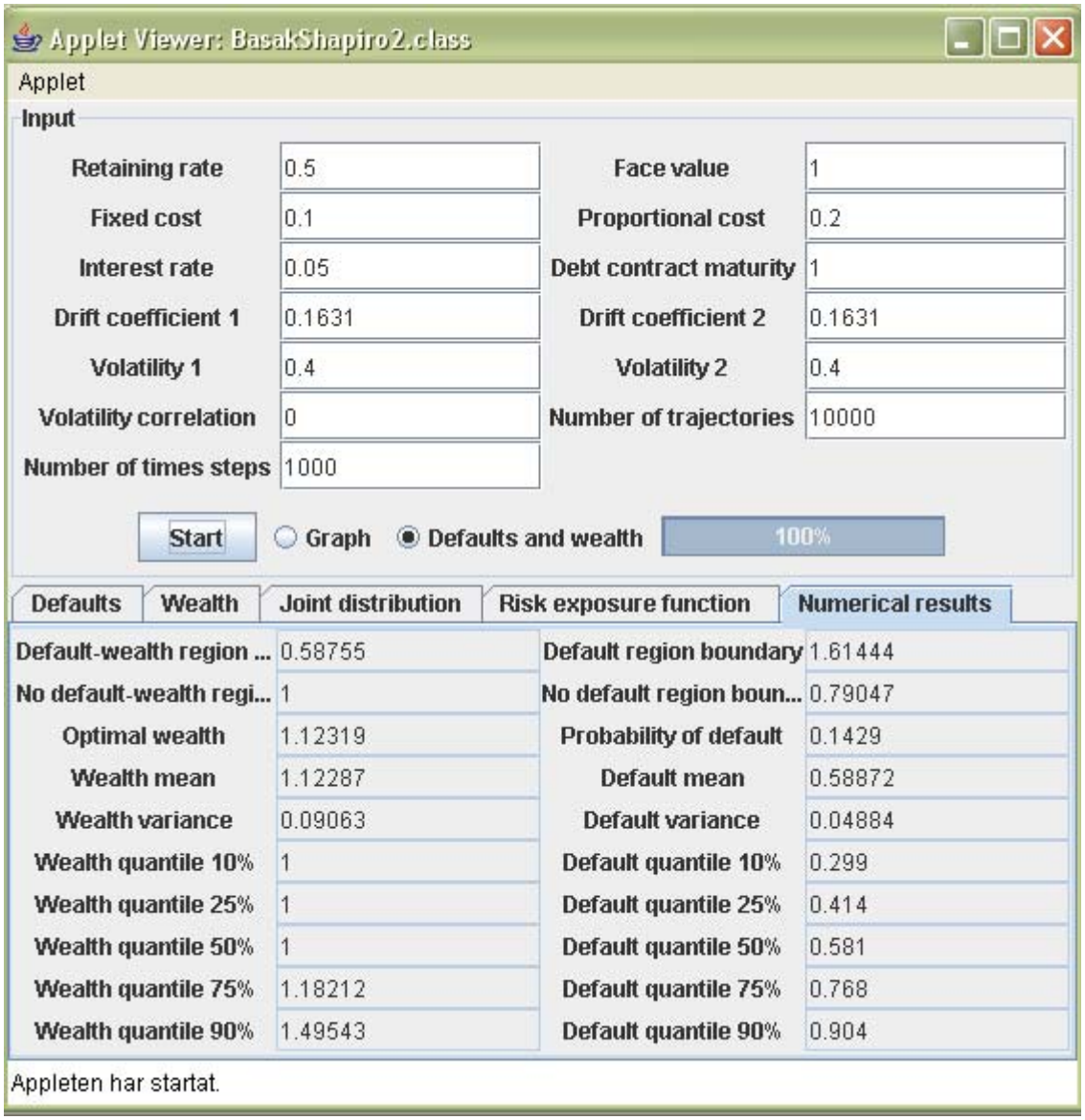


Figure 39 joint histogram

4.2.9. Final Numerical Results

Here user can get almost all the necessary output of BasakShapiro2 models.
These outputs are useful in decisions making or problem analysis. The discretion of this output shown in figure 41 below



4.2.10. Risk Exposure function

To get the risk exposure function the just click on risk exposure function label's in out panel where X axis represent time intervals and y axis is risk exposure value which calculated according to formula 14. As shown in Figure 40

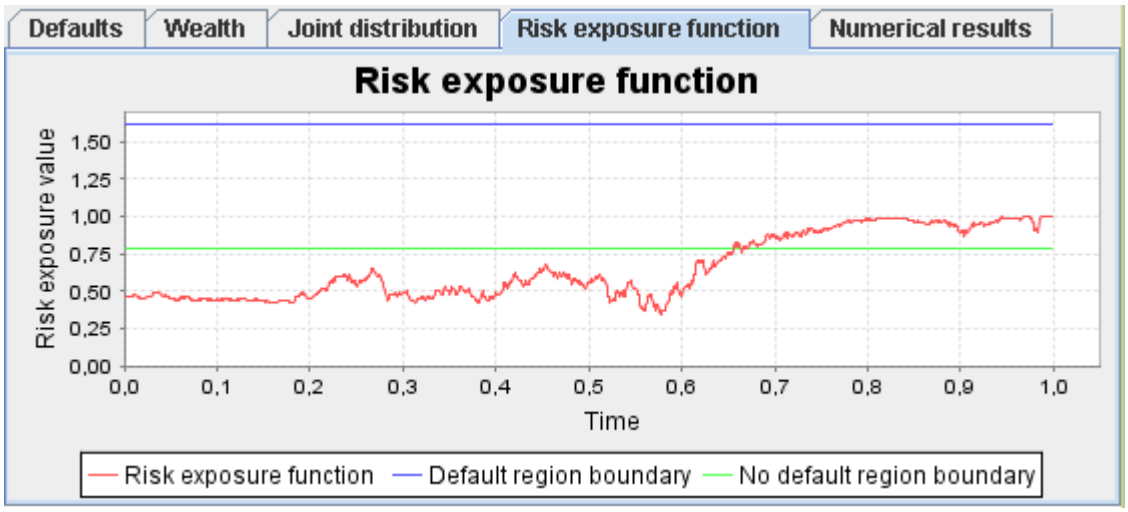


Figure 40 risk exposure function

5. Conclusion

Our study is based on Basak-Shapiro model that aim to verify if the default costs, when default occurs, have an effect on the optimal decisions and polices of borrowers (firm or household) and analyzed the associated investment policies and implications for market dynamics.

In part 2 the study viewing Basak-Shapiro model approaches in modeling the debt contract and default costs when debt maturity coincides with planning horizon . In part 3 we introduce the study approaches for deriving the necessaries formulas for model outputs and others variables.

The study design Java program to re-calculate, generally, with new approaches the various Outputs and variables needed for the model. Accordingly, we find that our Java program has the same results as Basak-Shapiro model calculations. In other words, it also shows that our program has a high flexibility and advantages that make it appropriate for credit risk model.

The state price density process (ζ) figure out with the times periods, as a new methods of estimation that adapted in this study, and redefined as a debt value function describing a change that it taking place between debt value and time periods. The graph of the debt function supplied with the calculation of non default region boundary (ζ_*) and default region boundary (ζ^*) with almost compatible values as in Basak-Shapiro models.

The borrower's time-T, plus time-t, optimal wealth introduced in this study proportional with times periods and depicts in one compact graph and redefined as wealth function $W^*(t)$,this in turn provided with no default-wealth region boundary $W^*(\zeta_*) = \frac{\beta F}{(1-\beta)}$ and default-wealth region boundary $W^*(\zeta^*) = I\left\{\frac{\beta y \zeta^*}{[\beta + \lambda(1-\beta)]}\right\}$. This technique differs than what have been used in Bask-Shapiro models. This is allowing end user to follow the graph of the functions, typically its values, through out times periods of debt contract until the maturity date of debt contract T. The same approach was used also in depicting risk investment functions, as representing the investment opportunities, and risk exposure function $m^*(t)$.

As end user can easily change the inputs values of the Java program so he or she is able to see the effects of changing model parameters F, β, λ, ϕ on borrower's optimal wealth across the defaults regions and analysis the reasons of the probability of default at time state of the economy.

Defaults probabilities were demonstrated with default histogram. Java program presents histograms of default of the model simulated trajectories.
Probability of default is estimated through the program, in addition to the mean and variance of the default. The program calculates also quartiles (10%, 25%, 50%, 75% and 90%) that aid end user to carry out the necessary statistic inferences needed for this model.

Java Program arranges wealth histograms of credit risk model. Borrower's optimal value and wealth mean takes approximately same values when the numbers of simulated trajectories increases. Java program compute wealth variance and quartiles (10%, 25%, 50%, 75% and 90%)

those all indeed offer a helpful tool to make statistical inferences concern probabilistic interpretation of borrower's optimal wealth at maturity date of debt contact.

Java program figure out risky investment opportunities functions with time period till maturity. The study concluded as a time headed for debt maturity threaten of default is very likely so the borrowers prefer risk less investments to the default and fulfill his obligations.

Then, we analyze the sensitivity of risk exposure function and exam the effects of changing the debt-contract parameters (F, β) default-costs parameters $(F\phi, \lambda)$. These effects displayed graphically and numerically. We highlighted the different influence of variety types of cost and explained analytically how a borrower's risk function may be seen.

We also pointed out how the probability of default vanishes or reduced with the extension of resistance region by changing default boundaries. These changes improve the borrower's financial positions by reducing the risk and likewise default on financial obligations.

Finally, we remember that our main objective was to build an applet for Basak-Shapiro model for estimating credit risk. After identifying our aims, we in advance to inspect the mathematical backdrop for model computation. This guide us to the conception of layout of our applet; we then utilize of Java library to build out the essential codes appropriate
For create the applet.

We then uphold that the objective of our study has been attained, and the user guide described in part 4 will assist the user recognize how the UI works. Up till now, we have never test the optimization when prehorizon default is allowed in this study. We recommended the others researchers to follow and complete this goal.

6. References

[1] Dempstar.M.A.H, *Risk management: Value at risk and beyond*, Cambridge University, Cambridge, 2002

[2] Jorion philippe, Value at risk: The *new benchmark for managing market risk*. 2nd ed. New York, McGraw-Hill, 2000

[3] Suleyman Basak and Shapiro Alexander, *a Model of credit risk, optimal policies and asset prices*, journal of business, University of Chicago, Vol.**78**.no **4**(2005), 1215-1266

[4] Jarrow. Robert A, Turnbull. S, *Derivative securities*, 2nd ed, South-Western College Publishing, Ohio, 2000

[5] Hull.J.C, Option future and other derivatives.,6th ed. Upper Saddle River, NJ: Prentice Hall, 2006

[6] Dennis D. Wackerly, William M III, Richard L. Scheaffer, Mathematical statistics applications., 6th ed, Duxbury and Book, 2002

[7] Campione, M., Walrath, K. and Huml, A. The *Java TM tutorial: a short course on the basic*, 3rd ed, The JavaTM Series, Addison Wesley, Upper Saddle River, NL, 2006

[8] Dimitrii, S., H., Jönsson and A. Malyarenko, *How to write seminar reports and thesis*, 2006

[9]Mathew. John H, Fink. Kurtis D, *Numerical methods using MAT LAB.*, 4th ed , Person Prentice Hall, 2004

[10]Howard Anton, Chris Robres, Elementary linear algebra, 8th ed, John Wiley & Sons.Inc, 2000

[11] Stewart James, Calculus: early transcendental, 5th ed, Thomson Learning. Inc, 2003

7. Appendix

Java Code

```
/**
* @(#) BasakShapiro.java  1.0 06/09/21
*
* Copyright (c) 2006 Mälardalen University
* Högskoleplan Box 883, 721 23 Västerås, Sweden.
* All Rights Reserved.
*
* The copyright to the computer program(s) herein
* is the property of Mälardalen University.
* The program(s) may be used and/or copied only with
* the written permission of Mälardalen University
* or in accordance with the terms and conditions
* stipulated in the agreement/contract under which
* the program(s) have been supplied.
*
* Description: Monte Carlo simulation of the state price
* density process in Basak-Shapiro model
*
* Glasserman, P.
* Monte Carlo methods in financial engineering
* Springer, Berlin, 2003
*
* @version 1.0 21 Sep 2006
* @author Basil Wakid Hassan
* Mail:bwd0300@student.mdh.se
*/

import java.awt.*;
import java.awt.event.*;
import javax.swing.*;
import javax.swing.border.*;
import java.text.*;
import java.util.*;

import org.jfree.chart.*;
import org.jfree.chart.axis.*;
import org.jfree.chart.plot.*;
import org.jfree.data.xy.*;
import org.jfree.data.statistics.*;

import org.freehep.j3d.plot.*;

public class BasakShapiro2 extends JApplet
                           implements FocusListener,
                                      ActionListener {
   // class variables
   // panels
   private JPanel mainPanel = null;
   private JPanel inputPanel = null;
   private JPanel dataPanel = null;
   private JPanel controlPanel = null;
```

```
private ChartPanel debtPanel = null;
private ChartPanel wealthPanel = null;
private JPanel riskyPanel = null;
private ChartPanel investmentPanel = null;
private JPanel numericalPanel = null;
private ChartPanel riskExposurePanel = null;


//Labels
private JLabel retainingRateLabel = null;
private JLabel faceValueLabel = null;
private JLabel fixedCostLabel = null;
private JLabel proportionalCostLabel = null;
private JLabel interestRateLabel = null;
private JLabel debtContractMaturityLabel = null;
private JLabel driftCoefficient1Label = null;
private JLabel driftCoefficient2Label = null;
private JLabel volatility1Label = null;
private JLabel volatility2Label = null;
private JLabel volatilityCorrelationLabel = null;
private JLabel numberOfTrajectoriesLabel = null;
private JLabel numberOfTimesStepsLabel = null;

// labels for numerical panel

private JLabel defaultRigonBoundaryLabel = null;//1
private JLabel noDefaultRigonBoundaryLabel = null;//3
private JLabel defaultValueLabel = null;//5
private JLabel probabilityOfDefaultLabel = null;//7
private JLabel defaultMeanLabel = null;//9
private JLabel defaultVarianceLabel = null;//11
private JLabel defaultTenLabel = null;//13
private JLabel defaultTwentyfiveLabel = null;//15
private JLabel defaultFiftyLabel = null;//17
private JLabel defaultSeventyFiveLabel = null;//19
private JLabel defaultNinetyLabel = null;//21
private JLabel wealthPriorDebtMaturityLabel = null;//2
private JLabel optimalWealthLabel = null;//4
private JLabel probabilityOfOptimalWealthLabel = null;//6
private JLabel wealthMeanLabel = null;//8
private JLabel wealthVarianceLabel = null;//10
private JLabel wealthTenLabel = null;//12
private JLabel wealthTwentyfiveLabel = null;//14
private JLabel wealthFiftyLabel = null;//16
private JLabel wealthSeventyFiveLabel = null;//18
private JLabel wealthNinetyLabel = null; //20

//Text fields
private JTextField retainingRateField = null;
private JTextField faceValueField = null;
private JTextField fixedCostField = null;
private JTextField proportionalCostField = null;
private JTextField interestRateField = null;
private JTextField debtContractMaturityField = null;
private JTextField driftCoefficient1Field = null;
private JTextField driftCoefficient2Field = null;
private JTextField volatility1Field = null;
private JTextField volatility2Field = null;
private JTextField volatilityCorrelationField = null;
private JTextField numberOfTrajectoriesField = null;
private JTextField numberOfTimeStepsField = null;
```

```
    // text fields for numerical panel

    private JTextField defaultRigonBoundaryField = null;//1
    private JTextField noDefaultRigonBoundaryField = null;//3
    private JTextField defaultValueField = null;//5
    private JTextField probabilityOfDefaultField = null;//7
    private JTextField defaultMeanField = null;//9
    private JTextField defaultVarianceField = null;//11
    private JTextField defaultTenField = null;//13
    private JTextField defaultTwentyfiveField = null;//15
    private JTextField defaultFiftyField = null;//17
    private JTextField defaultSeventyFiveField = null;//19
    private JTextField defaultNinetyField = null;//21
    private JTextField defaultWealthRegionBoundaryField = null;//2
    private JTextField noDefaultWealthRegionBoundaryField = null;//4
    private JTextField optimalWealthField = null;//6
    private JTextField wealthMeanField = null;//8
    private JTextField wealthVarianceField = null;//10
    private JTextField wealthTenField = null;//12
    private JTextField wealthTwentyfiveField = null;//14
    private JTextField wealthFiftyField = null;//16
    private JTextField wealthSeventyFiveField = null;//18
    private JTextField wealthNinetyField = null;//20

    // buttons
    private JButton startButton = null;

    // radio button group
    private ButtonGroup group = null;

    // radio buttons
    private JRadioButton graphButton = null;
    private JRadioButton defaultsButton = null;

    // progress bar
    private JProgressBar progressBar = null;

    // tabbed panes
    private JTabbedPane outputTabbedPane = null;

    // the corresponding boolean variable
    private boolean bGraph = true;

    // Lego plot
    private LegoPlot plot = null;

    // String constants
    private final String INPUT_LABEL = "Input";
    private final String RETAININGRATE_LABEL = "Retaining rate";
    private final String FACEVALUE_LABEL = "Face value";
    private final String FIXEDCOST_LABEL = "Fixed cost";
    private final String PROPORTIONALCOST_LABEL = "Proportional cost";
    private final String INTERESTRATE_LABEL = "Interest rate";
    private final String DEBTCONTRACTMATURITY_LABEL = "Debt contract matur-
ity";
    private final String DRIFTCOEFFICIENT1_LABEL = "Drift coefficient 1";
    private final String DRIFTCOEFFICIENT2_LABEL = "Drift coefficient 2";
    private final String VOLATILITY1_LABEL = "Volatility 1";
    private final String VOLATILITY2_LABEL = "Volatility 2";
```

```
   private final String VOLATILITYCORRELATION_LABEL = "Volatility correla-
tion";
   private final String NUMBEROFTRAJECTORIES_LABEL = "Number of trajecto-
ries";
   private final String NUMBEROFTIMESTEPS_LABEL = "Number of times steps";
   private final String START = "Start";
   private final String GRAPH = "Graph";
   private final String DEFAULTS_AND_WEALTH = "Defaults and wealth";
   private final String DEFAULT = "Defaults";
   private final String DEBT_VALUE_FUNCTION = "Debt value function";
   private final String DEFAULT_REGION_BOUNDARY = "Default region boundary";
   private final String NO_DEFAULT_REGION_BOUNDARY = "No default region
boundary";
   private final String TIME = "Time";
   private final String DEBT_FUNCTION = "Debt function";
   private final String WEALTH_FUNCTION = "Wealth function";
   private final String WEALTH_VALUE = "Wealth value";
   private final String DEFAULT_NUMBER = "Nr of defaults in an interval";
   private final String WEALTH = "Wealth";
   private final String WEALTH_NUMBER = "Nr of wealths in an interval";
   private final String WEALTH_AT_TIME_T = "Wealth at time T";
   private final String INVESTMENT = "Investment";
   private final String JOINT = "Joint distribution";
   private final String OPPORTUNITY0 = "First opportunity";
   private final String OPPORTUNITY1 = "Second opportunity";
   private final String RISKY_INVESTMENTS = "Risky investments functions";
   private final String INVESTMENTS_VALUES = "Investments values";
   private final String NUMERICAL_RESULTS = "Numerical results";
   private final String DEFAULT_WEALTH_REGION_BOUNDARY = "Default-wealth re-
gion boundary";
   private final String NO_DEFAULT_WEALTH_REGION_BOUNDARY = "No default-
wealth region boundary";
   private final String RISK_EXPOSURE_FUNCTION = "Risk exposure function ";
   private final String RISK_EXPOSURE_VALUE = "Risk exposure value";


   // text constants for labels on numerical panel
   private final String DEFAULTRIGONBOUNDARY_LABEL = "Default region bound-
ary";
   private final String NODEFAULTRIGONBOUNDRY_LABEL = "No default region
boundry";
   private final String DEFAULTVALUE_LABEL = "Default value";
   private final String PROBABILITYOFDEFAULT_LABEL = "Probability of de-
fault";
   private final String DEFAULTMEAN_LABEL = "Default mean";
   private final String DEFAULTVARIANCE_LABEL = "Default variance";
   private final String DEFAULTTEN_LABEL= "Default quantile 10%";
   private final String DEFAULTTWENTYFIVE_LABEL= "Default quantile 25%";
   private final String DEFAULTFIFTY_LABEL= "Default quantile 50%";
   private final String DEFAULTSEVENTYFIVE_LABEL= "Default quantile 75%";
   private final String DEFAULTNINETY_LABEL= "Default quantile 90%";
   private final String DEFAULTWEALTHREGIONBOUNDARY_LABEL = "Default-wealth
region boundary";
   private final String NODEFAULTWEALTHREGIONBOUNDARY_LABEL = "No default-
wealth region boundary";
   private final String OPTIMALWEALTH_LABEL = "Optimal wealth";
   private final String WEALTHMEAN_LABEL = "Wealth mean";
   private final String WEALTHVARIANCE_LABEL = "Wealth variance";
   private final String WEALTHTEN_LABEL= "Wealth quantile 10%";
   private final String WEALTHTWENTYFIVE_LABEL= "Wealth quantile 25%";
   private final String WEALTHFIFTY_LABEL= "Wealth quantile 50%";
```

```
    private final String WEALTHSEVENTYFIVE_LABEL= "Wealth quantile 75%";
    private final String WEALTHNINETY_LABEL= "Wealth quantile 90%";

    // tooltip texts
    private final String RETAINING_RATE_TOOLTIP = "The retaining rate of the
debt contract";
    private final String FACE_VALUE_TOOLTIP = "The face value of a zero-
coupon debt contract";
    private final String FIXED_COST_TOOLTIP = "The fixed cost of the de-
fault";
    private final String PROPORTIONAL_COST_TOOLTIP = "The proportional cost
of the default";
    private final String INTEREST_RATE_TOOLTIP = "The interest rate of the
investement opportunities";
    private final String DEBT_CONTRACT_MATURITY_TOOLTIP = "The maturity date
of the debt contract";
    private final String DRIFT_COEFFICIENT1_TOOLTIP = "The drift coefficient
of the first investement";
    private final String DRIFT_COEFFICIENT2_TOOLTIP = "The drift coefficient
of the second investement";
    private final String VOLATILITY1_TOOLTIP = "The volatility of the first
investement";
    private final String VOLATILITY2_TOOLTIP = "The volatility of the second
investement";
    private final String VOLATILITY_CORRELATION_TOOLTIP = "The volatility
correlation";
    private final String NUMBER_OF_TRAJECTORIES_TOOLTIP = "The number of
paths used in simulation";
    private final String NUMBER_OF_TIME_STEPS_TOOLTIP = "The number of times
intervals used in simulation";
    private final String GRAPH_TOOLTIP = "Investigation of the debt value";
    private final String DEFAULT_TOOLTIP = "Investigation to time default";

    // Error messages
    private final String NOT_A_NUMBER = " Enter a number";
    private final String NON_POSITIVE = " Enter a positive number";
    private final String NOT_A_CORRELATION = " Enter a number between -1 and
1";
    private final String NOT_INTEGER = " Enter an integer number";

    // Numerical constants
    private final double RETAINING_RATE = 0.5;
    private final double FACE_VALUE = 1;
    private final double FIXED_COST = 0.1;
    private final double PROPORTIONAL_COST = 0.2;
    private final double INTEREST_RATE = 0.05;
    private final double DEBT_CONTRACT_MATURITY = 1;
    private final double DRIFT_COEFFICIENT1 = 0.1631;
    private final double DRIFT_COEFFICIENT2 = 0.1631;
    private final double VOLATILITY1 = 0.4;
    private final double VOLATILITY2 = 0.4;
    private final double VOLATILITY_CORRELATION = 0;
    private final int NUMBER_OF_TRAJECTORIES = 10000;
    private final int NUMBER_OF_TIME_STEPS = 1000;
    private final double MULTIPLIER = 1.0/Math.sqrt(2*Math.PI);

    // numerical variables
    private double beta = RETAINING_RATE;
    private double F = FACE_VALUE;
    private double phi = FIXED_COST;
    private double lambda = PROPORTIONAL_COST;
```

```
private double r = INTEREST_RATE;
private double T = DEBT_CONTRACT_MATURITY;
private double mu1 = DRIFT_COEFFICIENT1;
private double mu2 = DRIFT_COEFFICIENT2;
private double sigma1 = VOLATILITY1;
private double sigma2 = VOLATILITY2;
private double rho = VOLATILITY_CORRELATION;
private int N = NUMBER_OF_TRAJECTORIES;
private int M = NUMBER_OF_TIME_STEPS;

// Number formatters
private DecimalFormat myFormatter = null;

// Random number generator
private Random generator = null;


// class methods
public void init() {

// Initialise formatter
DecimalFormatSymbols symbols = new DecimalFormatSymbols();
symbols.setDecimalSeparator('.');
myFormatter = new DecimalFormat("###.#####",symbols);


// get content pane
Container contentPane = getContentPane();

// create main panel
mainPanel = new JPanel();
mainPanel.setLayout(new GridLayout(0,1));
// add it to the content pane
contentPane.add(mainPanel);

// create input panel
inputPanel = new JPanel(new BorderLayout());
inputPanel.setBorder(new TitledBorder(INPUT_LABEL));

// add it to main panel
mainPanel.add(inputPanel);

// create output tabbed pane
outputTabbedPane = new JTabbedPane();
// add it to main panel
mainPanel.add(outputTabbedPane);

// create debt panel
debtPanel = new ChartPanel(null);
// add it to output tabbed pane
outputTabbedPane.add(debtPanel,DEBT_FUNCTION);

// create wealth panel
wealthPanel = new ChartPanel(null);
// add it to output tabbed pane
outputTabbedPane.add(wealthPanel,WEALTH_FUNCTION);

// create risky panel
riskyPanel = new JPanel(new BorderLayout());
// add it to output tabbed pane
outputTabbedPane.add(riskyPanel,INVESTMENT);
```

```
   // create investment panel
   investmentPanel = new ChartPanel(null);
   // add it to risky panel
   riskyPanel.add(investmentPanel);


   // create riskExposure  panel
   riskExposurePanel = new ChartPanel(null);
   // add it to output tabbed pane
   outputTabbedPane.add(riskExposurePanel,RISK_EXPOSURE_FUNCTION);


   // create numerical panel
   numericalPanel = new JPanel(new GridLayout(0,4));
   // add it to output tabbed pane
   outputTabbedPane.add(numericalPanel,NUMERICAL_RESULTS);

   // Add labels and text fields to numerical panel here

   // create data panel
   dataPanel = new JPanel(new GridLayout(0,4));
   //dataPanel.setBorder(new TitledBorder(DATA_LABEL));

   // add data panel to imput panel
   inputPanel.add(dataPanel,BorderLayout.CENTER);

   // create control panel
   controlPanel = new JPanel();
   // add it to input panel
   inputPanel.add(controlPanel,BorderLayout.SOUTH);

   // Create Labels.....

   // create retainingRate label.....add,JLabel.CENTER
   retainingRateLabel = new JLabel(RETAININGRATE_LABEL, JLabel.CENTER);

   // create  faceValue label
   faceValueLabel = new JLabel(FACEVALUE_LABEL, JLabel.CENTER);

   // create  fixedCost label
   fixedCostLabel = new JLabel(FIXEDCOST_LABEL, JLabel.CENTER);

   // create  proportionalCost label
   proportionalCostLabel = new JLabel(PROPORTIONALCOST_LABEL, JLa-
bel.CENTER);

   // create  interestRate label
   interestRateLabel = new JLabel(INTERESTRATE_LABEL, JLabel.CENTER);

   // create  debtContractMaturity label
   debtContractMaturityLabel = new JLabel(DEBTCONTRACTMATURITY_LABEL, JLa-
bel.CENTER);

   // create  driftCoefficient1 label
   driftCoefficient1Label = new JLabel(DRIFTCOEFFICIENT1_LABEL, JLa-
bel.CENTER);


   // create  driftCoefficient2 label
```

```
   driftCoefficient2Label = new JLabel(DRIFTCOEFFICIENT2_LABEL, JLa-
bel.CENTER);


   // create  volatility1 label
   volatility1Label = new JLabel(VOLATILITY1_LABEL, JLabel.CENTER);

   // create  volatility2 label
   volatility2Label = new JLabel(VOLATILITY2_LABEL, JLabel.CENTER);


   // create  volatilityCorrelation label
   volatilityCorrelationLabel = new JLabel(VOLATILITYCORRELATION_LABEL, JLa-
bel.CENTER);

   // create  numberOfTrajectories label
   numberOfTrajectoriesLabel = new JLabel(NUMBEROFTRAJECTORIES_LABEL, JLa-
bel.CENTER);

   // create numberOfTimesSteps label
   numberOfTimesStepsLabel = new JLabel(NUMBEROFTIMESTEPS_LABEL, JLa-
bel.CENTER);


   // create numerical panel labels&&&&&

   // create defaultRigonBoundary label.....add,JLabel.CENTER
   defaultRigonBoundaryLabel = new JLabel(DEFAULTRIGONBOUNDARY_LABEL, JLa-
bel.CENTER);

   // create  noDefault Rigon Boundary  label
   noDefaultRigonBoundaryLabel = new JLabel(NODEFAULTRIGONBOUNDRY_LABEL,
JLabel.CENTER);

   // create  defaultValueLabel
   defaultValueLabel = new JLabel(DEFAULTVALUE_LABEL, JLabel.CENTER);

   // create  probabilityOfDefault label
   probabilityOfDefaultLabel = new JLabel(PROBABILITYOFDEFAULT_LABEL, JLa-
bel.CENTER);

   // create  defaultMean label
   defaultMeanLabel = new JLabel(DEFAULTMEAN_LABEL, JLabel.CENTER);

   // create  defaultVariance Label
   defaultVarianceLabel = new JLabel(DEFAULTVARIANCE_LABEL, JLabel.CENTER);

   // create  defaultTen Label
   defaultTenLabel = new JLabel(DEFAULTTEN_LABEL, JLabel.CENTER);

   // create  defaultTwentyfiveDefault Label
   defaultTwentyfiveLabel = new JLabel(DEFAULTTWENTYFIVE_LABEL, JLa-
bel.CENTER);

   // create  defaultFifty Label
   defaultFiftyLabel = new JLabel(DEFAULTFIFTY_LABEL, JLabel.CENTER);

   // create  defaultSeventy five Label
   defaultSeventyFiveLabel = new JLabel(DEFAULTSEVENTYFIVE_LABEL, JLa-
bel.CENTER);
```

```
   // create  defaultNinety Label
   defaultNinetyLabel = new JLabel(DEFAULTNINETY_LABEL, JLabel.CENTER);

   // create  wealthPriorDebtMaturity Label
   wealthPriorDebtMaturityLabel = new JLa-
bel(DEFAULTWEALTHREGIONBOUNDARY_LABEL, JLabel.CENTER);

   // create  optimalWealth Label
   optimalWealthLabel = new JLabel(NODEFAULTWEALTHREGIONBOUNDARY_LABEL, JLa-
bel.CENTER);

   // create  probabilityOfOptimalWealth Label
   probabilityOfOptimalWealthLabel = new JLabel(OPTIMALWEALTH_LABEL, JLa-
bel.CENTER);

   // create  wealthMean Label
   wealthMeanLabel = new JLabel(WEALTHMEAN_LABEL, JLabel.CENTER);

   // create  wealthVariance Label
   wealthVarianceLabel = new JLabel(WEALTHVARIANCE_LABEL, JLabel.CENTER);

   // create  wealthTen Label
   wealthTenLabel = new JLabel(WEALTHTEN_LABEL, JLabel.CENTER);

   // create  wealthTwentyfive Label
   wealthTwentyfiveLabel = new JLabel(WEALTHTWENTYFIVE_LABEL, JLa-
bel.CENTER);

   // create  wealthFifty Label
   wealthFiftyLabel = new JLabel(WEALTHFIFTY_LABEL, JLabel.CENTER);

   // create  wealthSeventy Label
   wealthSeventyFiveLabel = new JLabel(WEALTHSEVENTYFIVE_LABEL, JLa-
bel.CENTER);

   // create  wealthNinety Label
   wealthNinetyLabel = new JLabel(WEALTHNINETY_LABEL, JLabel.CENTER);



   // Create Textfields

   //create  RetainingRate Textfield
   retainingRateField = new JTextField();
   // add number beta
   retainingRateField.setText(myFormatter.format(beta));
   // add tooltip
   retainingRateField.setToolTipText(RETAINING_RATE_TOOLTIP);
   // add focus listener
   retainingRateField.addFocusListener(this);


   //create  FaceValue Textfield
   faceValueField = new JTextField();
   // add number F
   faceValueField.setText(myFormatter.format(F));
   // add tooltip
   faceValueField.setToolTipText(FACE_VALUE_TOOLTIP);
   // Add Focus listener
   faceValueField.addFocusListener(this);
```

```
//create  fixedCost Textfield
fixedCostField = new JTextField();
// add number phi
fixedCostField.setText(myFormatter.format(phi));
// add tooltip
 fixedCostField.setToolTipText(FIXED_COST_TOOLTIP);
// Add Focus listener
fixedCostField.addFocusListener(this);


//create  proportionalCost Textfield
proportionalCostField = new JTextField();
// add number lamda
proportionalCostField.setText(myFormatter.format(lambda));
// add tooltip
proportionalCostField.setToolTipText(PROPORTIONAL_COST_TOOLTIP);
// Add Focus listener
proportionalCostField.addFocusListener(this);


//create  interestRate Textfield
interestRateField = new JTextField();
// add number r
interestRateField.setText(myFormatter.format(r));
// add tooltip
interestRateField.setToolTipText(INTEREST_RATE_TOOLTIP);
// Add Focus listener
interestRateField.addFocusListener(this);


//create  debtContractMaturity Textfield
debtContractMaturityField = new JTextField();
// add number T
debtContractMaturityField.setText(myFormatter.format(T));
// add tooltip
debtContractMaturityField.setToolTipText(DEBT_CONTRACT_MATURITY_TOOLTIP);
// Add Focus listener
debtContractMaturityField.addFocusListener(this);


//create  driftCoefficient1 Textfield
driftCoefficient1Field = new JTextField();
// add number mu1
driftCoefficient1Field.setText(myFormatter.format(mu1));
// add tooltip
driftCoefficient1Field.setToolTipText(DRIFT_COEFFICIENT1_TOOLTIP);
// Add Focus listener
driftCoefficient1Field.addFocusListener(this);


//create  driftCoefficient2 Textfield
driftCoefficient2Field = new JTextField();
// add number mu2
driftCoefficient2Field.setText(myFormatter.format(mu2));
// add tooltip
driftCoefficient2Field.setToolTipText(DRIFT_COEFFICIENT2_TOOLTIP);
// Add Focus listener
driftCoefficient2Field.addFocusListener(this);
```

```
    //create  volatility1 Textfield
    volatility1Field = new JTextField();
    // add number sigma1
    volatility1Field.setText(myFormatter.format(sigma1));
    // add tooltip
    volatility1Field.setToolTipText(VOLATILITY1_TOOLTIP);
    // Add Focus listener
    volatility1Field.addFocusListener(this);


    //create  volatility2 Textfield
    volatility2Field = new JTextField();
    // add number sigma2
    volatility2Field.setText(myFormatter.format(sigma2));
    // add tooltip
    volatility2Field.setToolTipText(VOLATILITY2_TOOLTIP);
    // Add Focus listener
    volatility2Field.addFocusListener(this);


    //create  volatilityCorrelation Textfield
    volatilityCorrelationField = new JTextField();
    // add number rho
    volatilityCorrelationField.setText(myFormatter.format(rho));
    // add tooltip
    volatilityCorrelation-
Field.setToolTipText(VOLATILITY_CORRELATION_TOOLTIP);
    // Add Focus listener
    volatilityCorrelationField.addFocusListener(this);


    //create  numberOfTrajectories Textfield.....
    numberOfTrajectoriesField = new JTextField();
    // add number N
    numberOfTrajectoriesField.setText(myFormatter.format(N));
    // add tooltip
    numberOfTrajectoriesField.setToolTipText(NUMBER_OF_TRAJECTORIES_TOOLTIP);
    // Add Focus listener
    numberOfTrajectoriesField.addFocusListener(this);
    // make it inactive
    numberOfTrajectoriesField.setEnabled(false);


    //create  numberOfTimesSteps Textfield
    numberOfTimeStepsField = new JTextField();
    // add number M
    numberOfTimeStepsField.setText(myFormatter.format(M));
    // add tooltip
    numberOfTimeStepsField.setToolTipText(NUMBER_OF_TIME_STEPS_TOOLTIP);
    // Add Focus listener
    numberOfTimeStepsField.addFocusListener(this);


    // Create texet field for numerical panel


     //create text defaultRigonBoundary Field
     defaultRigonBoundaryField = new JTextField();
     defaultRigonBoundaryField.setText("");
     defaultRigonBoundaryField.setEditable(false);
```

```
//create text noDefaultRigonBoundary Field
noDefaultRigonBoundaryField = new JTextField();
noDefaultRigonBoundaryField.setText("");
noDefaultRigonBoundaryField.setEditable(false);


//create text defaultValue Field
defaultValueField = new JTextField();
defaultValueField.setText("");
defaultValueField.setEditable(false);

//create text probabilityOfDefault Field
probabilityOfDefaultField = new JTextField();
probabilityOfDefaultField.setText("");
probabilityOfDefaultField.setEditable(false);

//create text defaultMean Field
defaultMeanField = new JTextField();
defaultMeanField.setText("");
defaultMeanField.setEditable(false);

//create text defaultVariance Field
defaultVarianceField = new JTextField();
defaultVarianceField.setText("");
defaultVarianceField.setEditable(false);


//create text defaultTen Field
defaultTenField = new JTextField();
defaultTenField.setText("");
defaultTenField.setEditable(false);

//create text defaultTwenty fiveField
defaultTwentyfiveField = new JTextField();
defaultTwentyfiveField.setText("");
defaultTwentyfiveField.setEditable(false);

//create text defaultFifty Field
defaultFiftyField = new JTextField();
defaultFiftyField.setText("");
defaultFiftyField.setEditable(false);

//create text defaultSeventyFive Field
defaultSeventyFiveField = new JTextField();
defaultSeventyFiveField.setText("");
defaultSeventyFiveField.setEditable(false);

//create text defaultNinety Field
defaultNinetyField = new JTextField();
defaultNinetyField.setText("");
defaultNinetyField.setEditable(false);

//create text wealthPriorDebtMaturity Field
defaultWealthRegionBoundaryField = new JTextField();
defaultWealthRegionBoundaryField.setText("");
defaultWealthRegionBoundaryField.setEditable(false);

//create text optimalWealthField Field
noDefaultWealthRegionBoundaryField = new JTextField();
noDefaultWealthRegionBoundaryField.setText("");
noDefaultWealthRegionBoundaryField.setEditable(false);
```

```
//create text probabilityOfOptimalWealt Field
optimalWealthField = new JTextField();
optimalWealthField.setText("");
optimalWealthField.setEditable(false);

//create text wealthMean Field
wealthMeanField = new JTextField();
wealthMeanField.setText("");
wealthMeanField.setEditable(false);

//create text wealthVariance Field
wealthVarianceField = new JTextField();
wealthVarianceField.setText("");
wealthVarianceField.setEditable(false);

//create text wealthTen Field
wealthTenField = new JTextField();
wealthTenField.setText("");
wealthTenField.setEditable(false);

//create text wealthTwentyfive Field
wealthTwentyfiveField = new JTextField();
wealthTwentyfiveField.setText("");
wealthTwentyfiveField.setEditable(false);

//create text wealthFifty Field
wealthFiftyField = new JTextField();
wealthFiftyField.setText("");
wealthFiftyField.setEditable(false);

//create text wealthSeventyFive Field
wealthSeventyFiveField = new JTextField();
wealthSeventyFiveField.setText("");
wealthSeventyFiveField.setEditable(false);

//create text wealthNinet Field
wealthNinetyField = new JTextField();
wealthNinetyField.setText("");
wealthNinetyField.setEditable(false);


 // add retainingRateLabel to the data panel
    dataPanel.add(retainingRateLabel);

 // add retaingRate Textfield to data Panel
    dataPanel.add(retainingRateField);

 // add faceValueLabel to the data panel
    dataPanel.add(faceValueLabel);

 // add FaceValue Textfield to data Panel
    dataPanel.add(faceValueField);

 // add fixedCostLabel to the data panel
    dataPanel.add(fixedCostLabel);

 // add fixedCost Textfield to data Panel
    dataPanel.add(fixedCostField);

 // add proportionalCostLabel to the data panel
```

```
        dataPanel.add(proportionalCostLabel);

    // add proportionalCost Textfield to data Panel
        dataPanel.add(proportionalCostField);

    // add interestRateLabel to the data panel
        dataPanel.add(interestRateLabel);

    // add interestRate Textfield to data Panel
        dataPanel.add(interestRateField);

    // add debtContractMaturityLabel to the data panel
        dataPanel.add(debtContractMaturityLabel);

    // add debtContractMaturity Textfield to data Panel
        dataPanel.add(debtContractMaturityField);

    // add driftCoefficient1 Label to the data panel
        dataPanel.add(driftCoefficient1Label);

    // add driftCoefficient1 Textfield to data Panel
        dataPanel.add(driftCoefficient1Field);

    // add driftCoefficient2 Label to the data panel
        dataPanel.add(driftCoefficient2Label);

    // add driftCoefficient2 Textfield to data Panel
        dataPanel.add(driftCoefficient2Field);

    // add volatility1 Label to the data panel
        dataPanel.add(volatility1Label);

    // add volatility1 Textfield to data Panel
        dataPanel.add(volatility1Field);

     // add volatility2 Label to the data panel
        dataPanel.add(volatility2Label);

     // add volatility2 Textfield to data Panel
        dataPanel.add(volatility2Field);

     // add volatilityCorrelation Label to the data panel
        dataPanel.add(volatilityCorrelationLabel);

     // add volatilityCorrelation Textfield to data Panel
        dataPanel.add(volatilityCorrelationField);


     // add numberOfTrajectories Label to the data panel
        dataPanel.add(numberOfTrajectoriesLabel);

     // add numberOfTrajectories TextField to data Panel
        dataPanel.add(numberOfTrajectoriesField);

     // add numberOfTimesSteps Label to the data panel
        dataPanel.add(numberOfTimesStepsLabel);

     // add numberOfTimesSteps TextField to data Panel
        dataPanel.add(numberOfTimeStepsField);
```

```
 // Add labels and text fields to numerical panel here


 // Add wealthPriorDebtMaturity Label to numerical panel
    numericalPanel.add(wealthPriorDebtMaturityLabel);

 // Add wealthPriorDebtMaturity text field to numerical panel
    numericalPanel.add(defaultWealthRegionBoundaryField);

 // Add defaultRigonBoundary Label to numerical panel
    numericalPanel.add(defaultRigonBoundaryLabel);

 // Add defaultRigonBoundary texet fiel to numerical panel
    numericalPanel.add(defaultRigonBoundaryField);

// Add optimalWealth Label to numerical panel
    numericalPanel.add(optimalWealthLabel);

// Add optimalWealth Label to numerical panel
   numericalPanel.add(noDefaultWealthRegionBoundaryField);

// Add noDefaultRigonBoundary Label to numerical panel
   numericalPanel.add(noDefaultRigonBoundaryLabel);

// Add noDefaultRigonBoundary text field to numerical panel
   numericalPanel.add(noDefaultRigonBoundaryField);

// Add probabilityOfOptimalWealth texet field to numerical panel
   numericalPanel.add(probabilityOfOptimalWealthLabel);

// Add probabilityOfOptimalWealth texet field to numerical panel
   numericalPanel.add(optimalWealthField);

// Add probabilityOfDefault Label to numerical panel
   numericalPanel.add(probabilityOfDefaultLabel);

// Add probabilityOfOptimalWealth texet field to numerical panel
   numericalPanel.add(probabilityOfDefaultField);

// Add wealthMean Label to numerical panel
   numericalPanel.add(wealthMeanLabel);

// Add wealthMean texet field to numerical panel
   numericalPanel.add(wealthMeanField);

// Add defaultMean Label to numerical panel
   numericalPanel.add(defaultMeanLabel);

// Add defaultMean texet field to numerical panel
   numericalPanel.add(defaultMeanField);

// Add wealthVariance Label to numerical panel
   numericalPanel.add(wealthVarianceLabel);

// Add wealthVariancetexet field to numerical panel
   numericalPanel.add(wealthVarianceField);

// Add defaultVariance Label to numerical panel
   numericalPanel.add(defaultVarianceLabel);

// Add defaultVariance texet field to numerical panel
```

```
        numericalPanel.add(defaultVarianceField);

     // Add wealthTen Label to numerical panel
        numericalPanel.add(wealthTenLabel);

     // Add wealthTen texet field to numerical panel
        numericalPanel.add(wealthTenField);

     // Add defaultTen Label to numerical panel
        numericalPanel.add(defaultTenLabel);

     // Add wealthTen texet field to numerical panel
        numericalPanel.add(defaultTenField);

     // Add wealthTwentyfive Label to numerical panel
        numericalPanel.add(wealthTwentyfiveLabel);

     // Add wealthTwentyfive texet field to numerical panel
        numericalPanel.add(wealthTwentyfiveField);

     // Add defaultTwentyfiveDefault Labelto numerical panel
        numericalPanel.add(defaultTwentyfiveLabel);

     // Add defaultTwentyfiveDefaul texet field to numerical panel
        numericalPanel.add(defaultTwentyfiveField);

     // Add wealthFifty texet field to numerical panel
        numericalPanel.add(wealthFiftyLabel);

    // Add wealthFifty texet field to numerical panel
       numericalPanel.add(wealthFiftyField);

    // Add defaultFifty Label to numerical panel
       numericalPanel.add(defaultFiftyLabel);

    // Add defaultFifty texet field to numerical panel
       numericalPanel.add(defaultFiftyField);

    // Add wealthSeventyFive Labelto numerical panel
       numericalPanel.add(wealthSeventyFiveLabel);

    // Add wealthSeventyFive texet field to numerical panel
       numericalPanel.add(wealthSeventyFiveField);

    // Add defaultSeventyFive Label to numerical panel
       numericalPanel.add(defaultSeventyFiveLabel);

    // Add defaultSeventyFive texet field to numerical panel
       numericalPanel.add(defaultSeventyFiveField);

    // Add wealthNinetyLabel to numerical panel
        numericalPanel.add(wealthNinetyLabel);

    // Add wealthNinetyField texet field to numerical panel
       numericalPanel.add(wealthNinetyField);

    // Add defaultNinety Label to numerical panel
       numericalPanel.add(defaultNinetyLabel);

    // Add defaultNinety texet field to numerical panel
       numericalPanel.add(defaultNinetyField);
```

```
  // create start button
       startButton = new JButton(START);
  // add action listener
       startButton.addActionListener(this);
  // add it to control panel
      controlPanel.add(startButton);


// create button group
   group = new ButtonGroup();


// create graph radio button
   graphButton = new JRadioButton(GRAPH,true);
// add it to button group
   group.add(graphButton);
// add tooltip text
   graphButton.setToolTipText(GRAPH_TOOLTIP);
// add action listener
   graphButton.addActionListener(this);
// add it to control panel
   controlPanel.add(graphButton);


// create default radio button
   defaultsButton = new JRadioButton(DEFAULTS_AND_WEALTH);
// add it to button group
   group.add(defaultsButton);
// add tooltip
   defaultsButton.setToolTipText(DEFAULT_TOOLTIP);
// add action listener
   defaultsButton.addActionListener(this);
// add it to control panel
   controlPanel.add(defaultsButton);

// create progress bar
   progressBar = new JProgressBar();
   progressBar.setStringPainted(true);
   progressBar.setValue(0);
   progressBar.setIndeterminate(true);
// add it to control panel
   controlPanel.add(progressBar);
   progressBar.setVisible(false);
// Start random number generation
   generator = new Random();
// Click graph button
   graphButton.doClick();
}


// focus listener implementation
   public void focusGained(FocusEvent e) {
}
   public void focusLost(FocusEvent e) {

// Obtain source
   Object source = e.getSource();

// if retaining rate field
   if (source == retainingRateField) {
```

```
   beta = readNumber(retainingRateField,
                     beta,
                     retainingRateField.getToolTipText());
   return;
}
// insert if operators for remaining 9 text fields

// if face value field
   if (source == faceValueField) {
   F = readNumber(faceValueField,
                     F,
                     faceValueField.getToolTipText());
   return;
}

// if fixed cost field
   if (source == fixedCostField) {
   phi = readNumber(fixedCostField,
                     phi,
                     fixedCostField.getToolTipText());
   return;
 }

// if proportional cost field
   if (source == proportionalCostField) {
   lambda = readNumber(proportionalCostField,
                     lambda,
                     proportionalCostField.getToolTipText());
   return;
 }

// if interest rate field
   if (source == interestRateField) {
   r = readNumber(interestRateField,
                     r,
                     interestRateField.getToolTipText());
   return;
 }

// if debt contract maturity field
   if (source == debtContractMaturityField) {
   T = readNumber(debtContractMaturityField,
                     T,
                     debtContractMaturityField.getToolTipText());
   return;
 }

// if drift coefficient1 field
   if (source == driftCoefficient1Field) {
   mu1 = readNumber(driftCoefficient1Field,
                     mu1,
                     driftCoefficient1Field.getToolTipText());
   return;
 }

// if drift coefficient2 field
   if (source == driftCoefficient2Field) {
   mu2 = readNumber(driftCoefficient2Field,
                     mu2,
                     driftCoefficient2Field.getToolTipText());
   return;
```

```
    }

   // if volatility1 field
      if (source == volatility1Field) {
      sigma1 = readNumber(volatility1Field,
                          sigma1,
                          volatility1Field.getToolTipText());
      return;
    }

   // correlation field
      if (source == volatilityCorrelationField) {
      rho = readCorrelation(volatilityCorrelationField,
                            rho,
                            volatilityCorrelationField.getToolTipText());
      return;
    }
    //if volatility2 field
      if (source == volatility2Field) {
      sigma2 = readNumber(volatility2Field,
                          sigma2,
                          volatility2Field.getToolTipText());
      return;
    }
    //if number of trajectories field
      if (source == numberOfTrajectoriesField) {
      N = readInt(numberOfTrajectoriesField,
                  N,
                  numberOfTrajectoriesField.getToolTipText());
      return;
    }
    //if number of time steps field
      if (source == numberOfTimeStepsField) {
      M = readInt(numberOfTimeStepsField,
                  M,
                  numberOfTimeStepsField.getToolTipText());
      return;
    }

}

   // Reads double numbers
      private double readNumber(JTextField field,
                            double oldValue,
                            String title) {
      boolean isOK = true;
      double newValue = 1;
      try {   // test input
      newValue = Double.parseDouble(field.getText());
 }
    catch (NumberFormatException e) { // ERROR message
      JOptionPane.showMessageDialog(null,
                                    NOT_A_NUMBER,
                                    title,
                                    JOptionPane.ERROR_MESSAGE);
      isOK = false;
    }
    if (newValue <= 0) {  // ERROR message
      JOptionPane.showMessageDialog(null,
                                    NON_POSITIVE,
                                    title,
```

```
                                    JOptionPane.ERROR_MESSAGE);
     isOK = false;
    }
    if (isOK) {
     return newValue;
    }
    else {
     field.setText(Double.toString(oldValue));
     return oldValue;
    }
}

// Reads integer numbers
 private int readInt(JTextField field,
                     int oldValue,
                     String title) {
 boolean isOK = true;
 int newValue = 100;
 try { // test input
    newValue = Integer.parseInt(field.getText());
 }
 catch (NumberFormatException e) { // ERROR message
    JOptionPane.showMessageDialog(null,
                                    NOT_INTEGER,
                                    title,
                                    JOptionPane.ERROR_MESSAGE);
      isOK = false;
    }
    if (newValue <= 0) {  // ERROR message
     JOptionPane.showMessageDialog(null,
                                    NON_POSITIVE,
                                    title,
                                    JOptionPane.ERROR_MESSAGE);
     isOK = false;
    }
    if (isOK) {
     return newValue;
    }
    else {
     field.setText(Integer.toString(oldValue));
     return oldValue;
    }
}

// Reads correlations
    private double readCorrelation(JTextField field,
                                   double oldValue,
                                   String title) {
 boolean isOK = true;
 double newValue = 1;
 try { // test input
    newValue = Double.parseDouble(field.getText());
 }
 catch (NumberFormatException e) { // ERROR message
    JOptionPane.showMessageDialog(null,
                                    NOT_A_NUMBER,
                                    title,
                                    JOptionPane.ERROR_MESSAGE);
      isOK = false;
    }
    if (Math.abs(newValue) > 1) {  // ERROR message
```

```
    JOptionPane.showMessageDialog(null,
                                  NOT_A_CORRELATION,
                                  title,
                                  JOptionPane.ERROR_MESSAGE);
    isOK = false;
   }
   if (isOK) {
    return newValue;
   }
   else {
    field.setText(Double.toString(oldValue));
    return oldValue;
   }
}
 // calculate d2 ( according to the method bellow:)
 private double d2(double x, double norm, double t, double xi) {
 double result = Math.log(x/xi);
 result += (r-norm*norm/2)*(T-t);
 result /= norm;
 result /= Math.sqrt(T-t);
 return result;
}
 // calculate standard normal distribution function
 private double normal(double x) {
 boolean flag = false;
 double z = x;
 if (x<0) {
    flag = true;
    z = -x;
 }
 double d = 1.0/(1.0+0.3316419*z);
 final double[] A = { 0,
                      0.31938153,
                      -0.356563782,
                      1.781477937,
                      -1.821255978,
                      1.330274429
                      };
   // Gorner's scheme
   double p = A[5];
   for (int k=4; k >=0; k--) {
    p = p*d+A[k];
   }
   double y = 1 - Math.exp(-z*z/2)*p/Math.sqrt(2*Math.PI);
   if (flag) {
    y = 1 - y;
   }
   return y;

}


 // calculate wStar(0)
 private double wStar0(double xiStar,double z, double norm) {
 // first term
 double result = xiStar/z;
 double argument = xiStar*(1.0-beta)/(beta*z*F);
 double d = d2(argument, norm, 0, 1);
 // second term
 result += beta*F/(1.0-beta)*Math.exp(-r*T)*normal(-d);
 double d1 = d+norm*Math.sqrt(T);
```

```
    //third term
    result -= xiStar*normal(-d1)/z;
    double term1 = d2(xiStar, norm, 0, 1);
    double term = normal(-term1);
    term *= Math.exp(-r*T);
    term *= (beta/(beta+lambda*(1.0-beta)));
    term *= (beta*F/(1.0-beta)-phi);
    // fourth term
    result -= term;
    term = term1+norm*Math.sqrt(T);
      term = normal(-term);
      term*=xiStar/z;
      // fifth term
      result += term;



    return result;

  }

    // calculate wStar(t)
    private double wStar(double xiStar,double z, double norm, double t, doub-
le xi) {
    // first term
    double result = xiStar/(z*xi);
    double argument = xiStar*(1.0-beta)/(beta*z*F);
    double d = d2(argument, norm, t, xi);
    // second term
    result += beta*F/(1.0-beta)*Math.exp(-r*(T-t))*normal(-d);
    double d1 = d+norm*Math.sqrt(T-t);
    // third term
    result -= xiStar*normal(-d1)/(z*xi);
    double term1 = d2(xiStar, norm, t, xi);
    double term = normal(-term1);
    term *= Math.exp(-r*(T-t));
    term *= (beta/(beta+lambda*(1.0-beta)));
    term *= (beta*F/(1.0-beta)-phi);
    // fourth term
    result -= term;
    term = term1+norm*Math.sqrt(T-t);
      term = normal(-term);
      term*=xiStar/(z*xi);
      // fifth term
      result += term;
    return result;
  }
    //Sorts na array ra[1..n]into ascending numerical order using the Heap-
sort algorithm.
    //n is input ;ra is replaced on the output
     public void hpSort(double[] ra) {
     int n = ra.length-1;
     double rra;
    //unsigned long i,ir,j,1;float rra;
     if (n<2) {
       return;
    }
    int l = (n >> 1)+1;
    int ir = n;
    //The index 1 will be decrement from its initial value down to 1 during
the "hiring"
```

```
   //(heap creation) phase.Once it reaches1,the index ir will be decrement
from its initial
   //value down to 1 during the "retirement-and-promotion"heap section pha-
se.
    for (;;) {
      if (l>1) {
   //still in hiring phase
         rra = ra[--l];
      }
      else {
   //In retirement-and- promotion phase.Clear a space at the end of the ar-
ray.
   //Retire at the top of the heap into it.Done with the last promotion.
   //The least competent worker of all
         rra = ra[ir];
         ra[ir] = ra[1];
         if (--ir == 1) {
            ra[1] = rra;
            break;
         }
      }
   //Whether in the hiring phase or promotion phase,
   //We here set up shift up to shift down element rra to its proper level.
      int i = l;
      int j = l+1;
      while (j <= ir) {
         if (j<ir && ra[j]<ra[j+1]) {
            j++;
   //Compare to the better underling.
   //Demote rra.
         }
         if (rra < ra[j]) {

   //Demote rra.
         ra[i] = ra[j];
            i = j;
            j <<= 1;
         }
   //Found rra's level. Terminate the shift-down.

         else {
            break;
         }
      }
   //Put rra into its slot.
     ra[i] = rra;
   }
  }
    double riskExposure(double t,
                        double wStar,
                        double xiStarDown,
                        double xiStarUp,
                        double z,
                        double norm,
                        double xi) {
     double result = 1;
     double x = beta*F/(1.0-beta);
     double y = x;
     double u = x;
     x *= normal(-d2(xiStarDown,norm,t,xi));
     y -= phi;
```

```
    y *= (beta/(beta+lambda*(1.0-beta)));
    double d = d2(xiStarUp,norm,t,xi);
    y *= normal(-d);
    x -= y;
    y = beta/(beta+lambda*(1.0-beta));
    u -= phi;
    u -= (beta+lambda*(1.0-beta))/(beta*z);
    y *= u;
    y *= MULTIPLIER*Math.exp(-d*d/2.0);
    y /= (norm*Math.sqrt(T-t));
    x -=y;
    result -= x*Math.exp(-r*(T-t))/wStar;

    return result;

}


 // action performed
    public void actionPerformed(ActionEvent e) {
 // Obtain source
    Object source = e.getSource();
 // if start button
    if (source == startButton) {
 // go to the first output panel
    outputTabbedPane.setSelectedIndex(0);
 // erase old data
      defaultRigonBoundaryField.setText("");
      noDefaultRigonBoundaryField.setText("");
      defaultValueField.setText("");
      probabilityOfDefaultField.setText("");
      defaultMeanField.setText("");
      defaultVarianceField.setText("");
      defaultTenField.setText("");
      defaultTwentyfiveField.setText("");
      defaultFiftyField.setText("");
      defaultSeventyFiveField.setText("");
      defaultNinetyField.setText("");
      defaultWealthRegionBoundaryField.setText("");
      noDefaultWealthRegionBoundaryField.setText("");
      optimalWealthField.setText("");
      wealthMeanField.setText("");
      wealthVarianceField.setText("");
      wealthTenField.setText("");
      wealthTwentyfiveField.setText("");
      wealthFiftyField.setText("");
      wealthSeventyFiveField.setText("");
      wealthNinetyField.setText("");
    // calculate a and b
      double b = beta*beta*F;
      b/=(1.0-beta);
      b/=(beta+lambda*(1.0-beta));
      b = Math.log(b);
      b+=1.0;
      double a = beta*F-phi*(1.0-beta);
      a*=beta;
      a/=(1.0-beta);
      a/=(beta+lambda*(1.0-beta));
    // find limits for z
       double zMin = 1.0/a;
       double zMax = zMin + 1.0;
```

```
            while ((a*zMax-Math.log(zMax))<b) {
            zMax+=1.0;
        }
        // find z by bisection
            double z=0;
            while((zMax-zMin)>0.0001) {
            z = (zMin+zMax)/2;
            if ((a*z-Math.log(z))<b) {
                zMin = z;
            }
            else {
                zMax = z;
            }
        }

        // Calculate the value of z
        //System.out.println("Value of z is: "+z);
        //System.out.println("w1Star at 0 is: "+w1Star(z,0));

        // create matrix sigma:

        double[][] sigma = {{0,0},{0,0}};  // create matrix sigma by initiali-
zer{}

        // Assigne values to matrix sigma
        sigma[0][0] = sigma1;
        sigma[1][1] = sigma2;
        sigma[0][1] = rho*Math.sqrt(sigma1*sigma2);
        sigma[1][0] = sigma[0][1];

        // calculate inverse:
        double determinant = sigma[0][0]*sigma[1][1]-sigma[0][1]*sigma[1][0];
//create determinant

        // create matrix sigmaInverse by initializer{}
        double[][] sigmaInverse = {{0,0},{0,0}};

        // Assigne values to matrix sigmaInverse to calculate it
        sigmaInverse[0][0] = sigma2/determinant;
        sigmaInverse[1][1] = sigma1/determinant;
        sigmaInverse[0][1] = -sigma[0][1]/determinant;
        sigmaInverse[1][0] = sigmaInverse[0][1];

          // calculate kappa

          //create matrix driftCoefficint
          double[] driftCoefficient = {mu1,mu2};
          // subtract r
          driftCoefficient[0] -=r;
          driftCoefficient[1] -=r;

          // Evalue kappa
          double[] kappa = {0,0};
          // matrix multiplication
          kappa[0] = sigmaInverse[0][0]*driftCoefficient[0]
                    +sigmaInverse[0][1]*driftCoefficient[1];
          kappa[1] = sigmaInverse[1][0]*driftCoefficient[0]
                    +sigmaInverse[1][1]*driftCoefficient[1];
          double norm = Math.sqrt(kappa[0]*kappa[0]+kappa[1]*kappa[1]);
          // System.out.println("norm of cappa = "+norm);
          //System.out.println(wStar(1.68,z,norm));
```

```
        // calculate xiStarUp by bisection

        // find limits for xiStarUp
        double xiStarUpMin = 0;
        double xiStarUpMax = 1;
        while (wStar0(xiStarUpMax,z,norm)<1) {
         xiStarUpMax += 1.0;
        }
        // find xiStarUp by bisection
         double xiStarUp = 0;
         while((xiStarUpMax-xiStarUpMin)>0.0001) {
         xiStarUp = (xiStarUpMin+xiStarUpMax)/2;
         if (wStar0(xiStarUp,z,norm)<1.0) {
            xiStarUpMin = xiStarUp;
         }
         else {
            xiStarUpMax = xiStarUp;
         }
      }
        // calculate xiStarDown
        double xiStarDown = xiStarUp*(1.0-beta)/(beta*z*F);

        // calculate (sigma^T)-1
        double[][] sigmaTInverse = {{0,0},{0,0}};
      // Assigne values to matrix sigmaTInverse
        sigmaTInverse[0][0] = sigmaInverse[0][0];
        sigmaTInverse[1][1] = sigmaInverse[1][1];
        sigmaTInverse[0][1] = sigmaInverse[1][0];
        sigmaTInverse[1][0] = sigmaInverse[0][1];

        // calculate vector thetaB
        double[] thetaB = {0,0};
        thetaB[0] = sigmaTInverse[0][0]*kappa[0]
                   +sigmaTInverse[0][1]*kappa[1];
        thetaB[1] = sigmaTInverse[1][0]*kappa[0]
                   +sigmaTInverse[1][1]*kappa[1];
       // System.out.println("thetaB[0]= "+thetaB[0]);
       // System.out.println("thetaB[1]= "+thetaB[1]);

       // Calaulate Minimum wealth value that avoid default crrespond to va-
lue of default boundary
            double wStarXiDown = beta*F/(1.0-beta);
            double wStarXiUp = (beta+lambda*(1.0-beta))/(beta*z);

       // System.out.println("No default-wealth region boundary is: "
+wStarXiDown );
       // System.out.println("Default-wealth region boundary: " +wStarXiUp
);

       // simulate xi
         double[] xi = new double[M+1];
         double[] W = new double[M+1];

       // create mStar
         double[] mStar = new double[M+1];
         mStar[0]= 1.0;

         xi[0]=1.0;
         W[0] = 1.0;
```

```
         mStar[0] = riskExposure(0,W[0],xiStarDown,xiStarUp,z,norm,xi[0]);
         double deltaT = T/M;
         double sqrtDeltaT = Math.sqrt(deltaT);

         System.out.println(" mStar[0]: " + mStar[0] );


        // graph
        if (bGraph) {

        // generate one trajectory
         for (int j=0; j<M; j++) {

            // generator.nextGaussian() = standard normal random variable
            double x = 1.0-r*deltaT
                    -kappa[0]*sqrtDeltaT*generator.nextGaussian()
                    -kappa[1]*sqrtDeltaT*generator.nextGaussian();
            xi[j+1] = xi[j]*x;
            W[j+1] = wStar(xiStarUp, z, norm, (j+1)*deltaT, xi[j+1]);
            mStar[j+1] = riskExposu-
re((j+1)*deltaT,W[j+1],xiStarDown,xiStarUp,z,norm,xi[j+1]);
           }

           // graph debt
           XYSeriesCollection debtDataset = new XYSeriesCollection();
           XYSeries xiSeries = new XYSeries(DEBT_VALUE_FUNCTION);

           // make this strings constants
           XYSeries xiStarUpSeries = new XYSeries(DEFAULT_REGION_BOUNDARY);
           XYSeries xiStarDownSeries = new XYSeri-
es(NO_DEFAULT_REGION_BOUNDARY);

           for (int j=0; j<M; j++) {
            xiSeries.add(j*deltaT, xi[j]);
            xiStarUpSeries.add(j*deltaT, xiStarUp);
            xiStarDownSeries.add(j*deltaT, xiStarDown);
           }
           debtDataset.addSeries(xiSeries);
           debtDataset.addSeries(xiStarUpSeries);
           debtDataset.addSeries(xiStarDownSeries);

           JFreeChart debtChart = ChartFactory.createXYLineChart(
            DEBT_VALUE_FUNCTION,  // chart title
            TIME,     // x axis label
            DEBT_FUNCTION, // y axis label
            debtDataset, // data
            PlotOrientation.VERTICAL,
            true, // include legend
            true, // tooltips
            false // urls
             );
            debtPanel.setChart(debtChart);
            debtPanel.setVisible(true);


            // graph wealth
            XYSeriesCollection wealthDataset = new XYSeriesCollection();
            XYSeries wSeries = new XYSeries(WEALTH_FUNCTION);

            // make this strings constants
```

```
            XYSeries wStarXiDownSeries = new XYSeri-
es(NO_DEFAULT_WEALTH_REGION_BOUNDARY);
            XYSeries wStarXiUpSeries = new XYSeri-
es(DEFAULT_WEALTH_REGION_BOUNDARY);

            for (int j=0; j<M; j++) {
            wSeries.add(j*deltaT, W[j]);
            wStarXiDownSeries.add(j*deltaT, wStarXiDown);
            wStarXiUpSeries.add(j*deltaT, wStarXiUp);


          }
            wealthDataset.addSeries(wSeries);
            wealthDataset.addSeries(wStarXiDownSeries);
            wealthDataset.addSeries(wStarXiUpSeries);



            JFreeChart wealthChart = ChartFactory.createXYLineChart(
               WEALTH_FUNCTION,  // chart title
            TIME,    // x axis label
            WEALTH_VALUE, // y axis label
            wealthDataset, // data
            PlotOrientation.VERTICAL,
            true, // include legend
            true, // tooltips
            false // urls
             );
            wealthPanel.setChart(wealthChart);
            wealthPanel.setVisible(true);

            // graph investments
            XYSeriesCollection investmentDataset = new XYSeriesCollection();
            XYSeries theta0Series = new XYSeries(OPPORTUNITY0);
            XYSeries theta1Series = new XYSeries(OPPORTUNITY1);

            for (int j=0; j<M; j++) {
            theta0Series.add(j*deltaT, thetaB[0]*mStar[j]);
            theta1Series.add(j*deltaT, thetaB[1]*mStar[j]);
          }
            investmentDataset.addSeries(theta0Series);
            investmentDataset.addSeries(theta1Series);
            JFreeChart investmentChart = ChartFactory.createXYLineChart(
            RISKY_INVESTMENTS,  // chart title
            TIME,    // x axis label
            INVESTMENTS_VALUES, // y axis label
            investmentDataset, // data
            PlotOrientation.VERTICAL,
            true, // include legend
            true, // tooltips
            false // urls
             );
            investmentPanel.setChart(investmentChart);
            investmentPanel.setVisible(true);


            // graph debt
          XYSeriesCollection mStarDataset = new XYSeriesCollection();
          XYSeries mStarSeries = new XYSeries(RISK_EXPOSURE_FUNCTION);

          // make this strings constants
```

```
    for (int j=0; j<M; j++) {
     mStarSeries.add(j*deltaT, mStar[j]);
     xiStarUpSeries.add(j*deltaT, xiStarUp);
     xiStarDownSeries.add(j*deltaT, xiStarDown);
    }
     mStarDataset.addSeries(mStarSeries);
     mStarDataset.addSeries(xiStarUpSeries);
     mStarDataset.addSeries(xiStarDownSeries);

     JFreeChart mStarChart = ChartFactory.createXYLineChart(
     RISK_EXPOSURE_FUNCTION,  // chart title
     TIME,    // x axis label
     RISK_EXPOSURE_VALUE, // y axis label
     mStarDataset, // data
     PlotOrientation.VERTICAL,
     true, // include legend
     true, // tooltips
     false // urls
      );
     riskExposurePanel.setChart(mStarChart);
     riskExposurePanel.setVisible(true);


}
// defaults
else {
// create defaults and wealth array
 double[] defaults = new double[N];
 double[] wealth = new double[N];
 int defaultCounter = 0;
 // Prepare progress bar
   progressBar.setMaximum(N);
   progressBar.setValue(0);
   progressBar.setIndeterminate(false);

 // simulate trajectories
   for (int n=0; n<N; n++) {
    progressBar.setValue(n+1);
       progressBar.paint(progressBar.getGraphics());
    xi[0] = 1.0;
    boolean bDefault = false;
    for (int j=0; j<M; j++) {
 // generator.nextGaussian() = standard normal random variable
       double x = 1.0-r*deltaT
            -kappa[0]*sqrtDeltaT*generator.nextGaussian()
            -kappa[1]*sqrtDeltaT*generator.nextGaussian();
       xi[j+1] = xi[j]*x;
       // check for default
       if (!bDefault && xi[j+1]>=xiStarUp) {
          // save default time
          defaults[defaultCounter] = deltaT*(j+1);
          // increase counter
          defaultCounter++;
          // set default flag
          bDefault = true;
       }
    }
    // calculate wealth
    wealth[n] = wStar(xiStarUp, z, norm, T, xi[M-1]);
    // calculate and show default characteristics
```

```
  }
  // calculate optimal wealth (expected value)
  double optimalWealth = 0;
  for (int n=0; n<N; n++) {
   optimalWealth += wealth[n];
  }
  optimalWealth /= N;
 // System.out.println("The optimal wealth is: "+optimalWealth);

 //DEfault histogram
  HistogramDataset defaultDataset = new HistogramDataset();
  double[] results = new double[defaultCounter];
  System.arraycopy(defaults,0,results,0,defaultCounter);
  int nbins1 = (int) Math.pow(defaultCounter, 0.333);
  defaultDataset.addSeries(DEFAULT,results,nbins1);
  JFreeChart defaultHistogram = ChartFactory.createHistogram (
                  DEFAULT,
                  TIME,
                  DEFAULT_NUMBER,
                  defaultDataset,
                  PlotOrientation.VERTICAL,
                  true,
                  true,
                  false);
  debtPanel.setChart(defaultHistogram);
  debtPanel.setVisible(true);
  // wealth histogram
  HistogramDataset wealthDataset = new HistogramDataset();
  int nbins2 = (int) Math.pow(N, 0.333);
  wealthDataset.addSeries(WEALTH,wealth,nbins2);
  JFreeChart wealthHistogram = ChartFactory.createHistogram (
                  WEALTH,
                  WEALTH_AT_TIME_T,
                  WEALTH_NUMBER,
                  wealthDataset,
                  PlotOrientation.VERTICAL,
                  true,
                  true,
                  false);
  wealthPanel.setChart(wealthHistogram);
  wealthPanel.setVisible(true);
  // make joint histogram
   double xMin = 0;
   double yMin = 0;
   double xMax = results[0];
   double yMax = wealth[0];
   for (int i=1; i<results.length; i++) {
   if (results[i]>xMax) {
      xMax = results[i];
   }
}
   for (int i=1; i<wealth.length; i++) {
   if (wealth[i]>yMax) {
      yMax = wealth[i];
   }
}
   double xStep = xMax / nbins1;
   double yStep = yMax / nbins2;
   double[][] data = new double[nbins1][nbins2];
   for (int i=0; i<nbins1; i++) {
   for (int j=0; j<nbins2; j++) {
```

```
                data[i][j] = 0;
            }
         }
            for (int i=0; i<results.length; i++) {
            int k = 0;
            while (k*xStep < results[i]) {
               k++;
            }
            while (k>=nbins1) {
               k--;
            }
            for (int j=0; j<wealth.length; j++) {
               int l = 0;
               while (l*yStep < wealth[j]) {
                  l++;
               }
               while (l>=nbins2) {
                  l--;
               }
               data[k][l]+=1;
            }
         }
            if (plot == null) {
            plot = new LegoPlot();
            plot.setLogZscaling(true);
          }
            else {
            riskyPanel.remove(plot);
            }
            plot.setData(new PriceData(data,xMax,yMax));
            riskyPanel.add(plot,BorderLayout.CENTER);
            riskyPanel.revalidate();
            riskyPanel.repaint();
          // write results in defaults and wealth mode
            double probability = ((double)(defaultCounter))/N;
            probabilityOfDefaultFi-
eld.setText(myFormatter.format(probability));
            hpSort(results);
            double[] statistics = getData(results);
            defaultMeanField.setText(myFormatter.format(statistics[0]));
            defaultVarianceField.setText(myFormatter.format(statistics[1]));
            defaultTenField.setText(myFormatter.format(statistics[2]));
            defaultTwentyfiveFi-
eld.setText(myFormatter.format(statistics[3]));
            defaultFiftyField.setText(myFormatter.format(statistics[4]));
            defaultSeventyFiveFi-
eld.setText(myFormatter.format(statistics[5]));
            defaultNinetyField.setText(myFormatter.format(statistics[6]));

            hpSort(wealth);
            statistics = getData(wealth);
            wealthMeanField.setText(myFormatter.format(statistics[0]));
            wealthVarianceField.setText(myFormatter.format(statistics[1]));
            wealthTenField.setText(myFormatter.format(statistics[2]));
            wealthTwentyfiveField.setText(myFormatter.format(statistics[3]));
            wealthFiftyField.setText(myFormatter.format(statistics[4]));
            wealthSeventyFiveFi-
eld.setText(myFormatter.format(statistics[5]));
            wealthNinetyField.setText(myFormatter.format(statistics[6]));
            optimalWealthField.setText(myFormatter.format(optimalWealth));
         }
```

```
        // write results independent of mode
           defaultRigonBoundaryField.setText(myFormatter.format(xiStarUp));
           noDefaultRigonBoundaryFi-
eld.setText(myFormatter.format(xiStarDown));
           defaultWealthRegionBoundaryFi-
eld.setText(myFormatter.format(wStarXiUp));
           noDefaultWealthRegionBoundaryFi-
eld.setText(myFormatter.format(wStarXiDown));
           return;
   }
           if (source == graphButton) {
           bGraph = true;
           numberOfTrajectoriesField.setEnabled(false);
           outputTabbedPane.setTitleAt(0,DEBT_FUNCTION);
           outputTabbedPane.setTitleAt(1,WEALTH_FUNCTION);
           outputTabbedPane.setTitleAt(2,INVESTMENT);
           debtPanel.setChart(null);
           debtPanel.setVisible(false);
           progressBar.setVisible(false);

      // At the numerical panel to se the label and text field invisible
      probabilityOfDefaultLabel.setVisible(false);
      probabilityOfDefaultField.setVisible(false);
      probabilityOfOptimalWealthLabel.setVisible(false);
      optimalWealthField.setVisible(false);
      wealthMeanLabel.setVisible(false);
      wealthMeanField.setVisible(false);
      defaultMeanLabel.setVisible(false);
      defaultMeanField.setVisible(false);
      wealthVarianceLabel.setVisible(false);
      wealthVarianceField.setVisible(false);
      defaultVarianceLabel.setVisible(false);
      defaultVarianceField.setVisible(false);
      wealthTenLabel.setVisible(false);
      wealthTenField.setVisible(false);
      defaultTenLabel.setVisible(false);
      defaultTenField.setVisible(false);
      wealthTwentyfiveLabel.setVisible(false);
      wealthTwentyfiveField.setVisible(false);
      defaultTwentyfiveLabel.setVisible(false);
      defaultTwentyfiveField.setVisible(false);
      wealthFiftyLabel.setVisible(false);
      wealthFiftyField.setVisible(false);
      defaultFiftyLabel.setVisible(false);
      defaultFiftyField.setVisible(false);
      wealthSeventyFiveLabel.setVisible(false);
      wealthSeventyFiveField.setVisible(false);
      defaultSeventyFiveLabel.setVisible(false);
      defaultSeventyFiveField.setVisible(false);
      wealthNinetyLabel.setVisible(false);
      wealthNinetyField.setVisible(false);
      defaultNinetyLabel.setVisible(false);
      defaultNinetyField.setVisible(false);
      return;
   }
   if (source == defaultsButton) {
      bGraph = false;
      numberOfTrajectoriesField.setEnabled(true);
      outputTabbedPane.setTitleAt(0,DEFAULT);
      outputTabbedPane.setTitleAt(1,WEALTH);
      outputTabbedPane.setTitleAt(2,JOINT);
```

```
    debtPanel.setChart(null);
    debtPanel.setVisible(false);
    progressBar.setVisible(true);

    // At the numerical panel to set the label and text visible
    probabilityOfDefaultLabel.setVisible(true);
    probabilityOfDefaultField.setVisible(true);
    probabilityOfOptimalWealthLabel.setVisible(true);
    optimalWealthField.setVisible(true);
    wealthMeanLabel.setVisible(true);
    wealthMeanField.setVisible(true);
    defaultMeanLabel.setVisible(true);
    defaultMeanField.setVisible(true);
    wealthVarianceLabel.setVisible(true);
    wealthVarianceField.setVisible(true);
    defaultVarianceLabel.setVisible(true);
    defaultVarianceField.setVisible(true);
    wealthTenLabel.setVisible(true);
    wealthTenField.setVisible(true);
    defaultTenLabel.setVisible(true);
    defaultTenField.setVisible(true);
    wealthTwentyfiveLabel.setVisible(true);
    wealthTwentyfiveField.setVisible(true);
    defaultTwentyfiveLabel.setVisible(true);
    defaultTwentyfiveField.setVisible(true);
    wealthFiftyLabel.setVisible(true);
    wealthFiftyField.setVisible(true);
    defaultFiftyLabel.setVisible(true);
    defaultFiftyField.setVisible(true);
    wealthSeventyFiveLabel.setVisible(true);
    wealthSeventyFiveField.setVisible(true);
    defaultSeventyFiveLabel.setVisible(true);
    defaultSeventyFiveField.setVisible(true);
    wealthNinetyLabel.setVisible(true);
    wealthNinetyField.setVisible(true);
    defaultNinetyLabel.setVisible(true);
    defaultNinetyField.setVisible(true);

    return;
 }
}
// calculate mean, variance, and quantiles
   double[] getData(double[] data) {
   int n = data.length-1;
   double[] result = new double[7];
 // mean value
   result[0] = 0;
   for (int i=0; i<n; i++) {
   result[0] += data[i];
 }
   result[0] /= n;
 // variance
   result[1] = 0;
   for (int i=0; i<n; i++) {
   result[1] += (data[i]-result[0])*(data[i]-result[0]);
 }
   result[1] /= (n-1);
  //Quantiles
   final double[] QUANTILES = {0.1, 0.25, 0.5, 0.75, 0.9};
   for (int i=0; i<5; i++) {
    result[i+2] = data[(int)(n*QUANTILES[i])];
```

```
      }
    return result;
  }

}
```