ESCAPE TIME DISTRIBUTION FOR STOCHASTIC FLOWS

by

Han Liu, BS

A Thesis

In

Mathematics and Statistics

Submitted to the Graduate Faculty
of Texas Tech University in
Partial Fulfillment of
the Requirements for the Degree of

Master of Science

Approved

Clyde F. Martin

Chair of Committee

Lance Drager

Peggy Gordon Miller

Dean of the Graduate School

August, 2011

ACKNOWLEDGEMENTS

I wish to express my deep gratitude to my advisor, Dr. Martin for his continuous help and support. Within the past two years' studying in Texas Tech University, I learned a lot from Dr. Martin's patience, motivation, enthusiasm, and immense knowledge. From choosing the subject to final completion of this thesis, Dr. Martin always gives careful guidance and unremitting support. I could not have imagined having a better advisor and mentor for my Master study.

I'm grateful to my committee member Dr. Lance Drager, for his kind help and insightful comments.

Many sincere and warm thanks goes to my boyfriend Cameron, who's supportive, encouraging and provided the best company during my thesis writing period.

I owe my loving thanks to my parents who bore me, raised me, supported me and loved me. My special gratitude is due to my brother for his loving support. They let me own a happy family in China.

TABLE OF CONTENTS

ABSTRACT

The model is based on models developed at the Federal Reserve Board of Governors by Robert Martin, PhD. His models were used to model data arising from subprime mortgages. They are very simple but capture data very well. In this thesis we used his model and derived the partial differential equations describing the time history of the corresponding distributions. In the case of Brownian motion this reduced to just the Fokker-Planck equation and in the case of the jump process we followed the derivation in the notes by Roger Brockett. In doing this, a deep understanding of how to use and manipulate the Itô formula and other aspects of stochastic differential equations is gained.

We assume x, as a weighted variable, to evaluate the borrower's ability to continue making payments, refinance, default or pay off. It is scaled so that 0 represents default and 1 represents paid. For each treatment we assume the approximation difference equation $x_{n+1} = (1 + r)x_n - s\epsilon_n$ as the model where the parameters r and s are two positive constants to be determined. r stands for the growth rate which is a positive real number in $(0, 1)$. The $s\epsilon_n$ term, as the bad accidents such as divorce, job loss, career moves, etc., can dramatically affect the ability to pay. After 10,000 treatments, we will find the histograms which are obtained by recording the frequency of those jump time points. We will then analyze and explain our results of simulation based on the histograms of the escape time distributions.

LIST OF FIGURES

CHAPTER 1
INTRODUCTION

Subprime mortgage [1] means making loans to people who may have difficulty maintaining the repayment schedule. Subprime borrowers typically have weakened credit histories and reduced repayment capacity. Subprime loans have a higher risk of default than loans to prime borrowers.

If a borrower is delinquent in making timely mortgage payments to the loan servicer (a bank or other financial firm), the lender may take possession of the property in a process called foreclosure. On the servicer's side, they actually expect the borrowers will fail to pay because in this way they can either obtain all the previous payments or take possession of the property. However, they undertake a major risk. Uncontrollable default rates necessarily contribute to the subprime mortgage crisis. The most recent one is triggered by the bursting of the United States housing bubble which peaked in approximately 2005-2006. Defaults and foreclosure activity increased dramatically in late 2006 in the U.S. and continue to be a factor in the global economy. Due to the big impact for not only the economic market in U.S. but also in global economy, it is necessary to do some analysis about the subprime mortgage.

In this paper, we develop a very simple simulation model for such analysis. The model is based on models developed at the Federal Reserve Board of Governors by Robert Martin, PhD [2]. His models were used to model data arising from subprime mortgages. They are very simple but capture data very well. In this paper we used his model and derived the partial differential equations describing the time history of the corresponding distributions. In the case of Brownian motion this reduced to just the Fokker-Planck equation and in the case of the jump process we followed the derivation in the notes by Roger Brockett. In doing this, a deep understanding of how to use and manipulate the Itô formula and other aspects of stochastic differential equations is gained.

We assume x, as a weighted variable, to evaluate the borrower's ability to continue making payments, refinance, default or pay off. It is scaled so that 0 represents defaulted and 1 represents paid off. For each treatment we assume the approximation difference equation $x_{n+1} = (1 + r)x_n - s\epsilon_n$ as the model where the parameters r and s are positive constants to be determined. The variable r stands for the growth rate which is a positive real number in $(0, 1)$. The $s\epsilon_n$ term, as the bad accidents such as divorce, job loss, career moves, etc., can dramatically affect the ability to pay. After 10,000 treatments, we will find the histograms which are obtained by recording the frequency of those jump time points. We will then analyze and explain our results of simulation based on the following histograms:

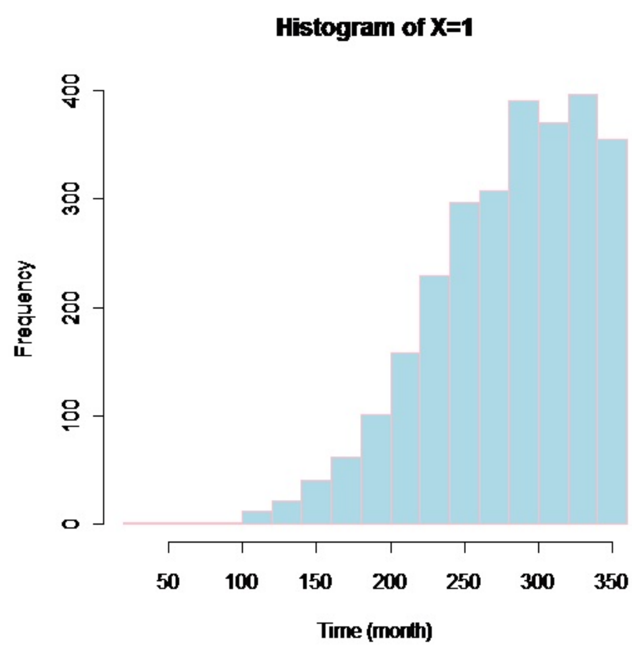

Figure 1.1. Introduction a

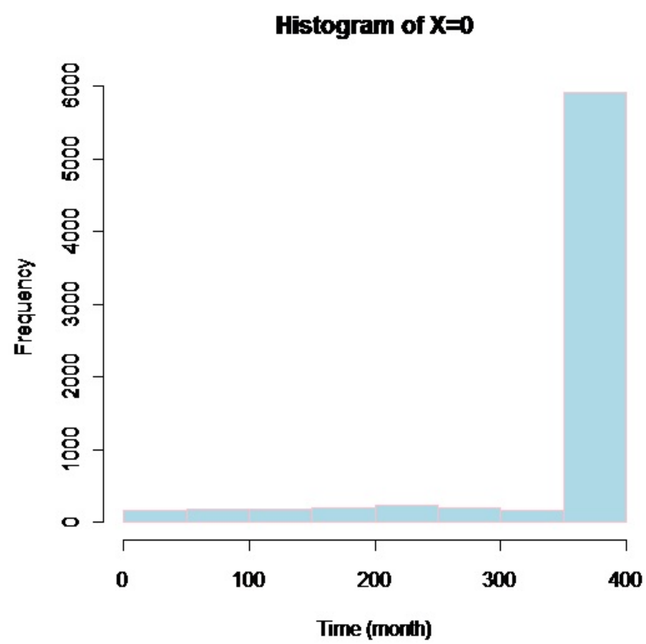

Figure 1.2. Introduction b

CHAPTER 2
DERIVATION OF PARTIAL DIFFERENTIAL EQUATIONS

2.1 Derivation of PDE with jump process

We are interested in the Poisson counter driven stochastic differential equation (PCSDE) of this form

$$dx(t) = f(x(t))dt + g(x(t))dN_\lambda(t), \tag{2.1}$$

where $x(t)$ is a stochastic process described by the stochastic differential equation, $N_\lambda(t)$ is Poisson counter that drives x(t), and f(x), g(x) are real-valued functions. We briefly review the properties of the Poisson counter [3] for the purpose of this thesis. Consider a Poisson process N with rate λ. We have

$$dN = \begin{cases} 1, & \text{at Poisson arrival;} \\ 0, & \text{elsewhere.} \end{cases} \tag{2.2}$$

$$\mathcal{E}(dN) = \lambda dt. \tag{2.3}$$

In this part, we turn to the problem of finding an evolution equation for the probability law associated with the state

$$dx = rdt + sdN_\lambda, \tag{2.4}$$

where r and s are small constants. We compute the time rate of change of the expected value of a "test" function in two different ways. We set the two expected values equal to each other and then after some routine integrations by parts we find an expression of the following form

$$\int_R \psi(x)(f(x))dx = 0, \tag{2.5}$$

and this is true for all such test functions.

Therefore, we must have $f(x) = 0$ which is the derivation of the equation for the density. Let $\mathcal{A}$ denote the set of all subsets of $\mathbb{R}$ which are intervals. Suppose that there exists a differentiable function ρ such that for all sets $A \subset \mathcal{A}$

$$P(x(t) \in A) = \int_A \rho(t,x)dx, P(\mathbb{R}) = 1. \tag{2.6}$$

In this case it is of interest to find an equation for the evolution of ρ. First of all introduce a smooth "test function" $\psi(t)$, a map of R into R. We assume that for each ψ, there exists an interval $[a,b]$

$$x \notin [a,b] \Rightarrow \psi(x) = 0. \tag{2.7}$$

The composition $\psi(x(t))$ is a stochastic process. Itô 's lemma [4] is used in Itô stochastic calculus to find the differential of a function of a particular type of stochastic process. It is named after its discoverer, Kiyoshi Itô. It is the stochastic calculus counterpart of the chain rule in ordinary calculus and is best constructed using the Taylor series expansion and retaining the second order term related to the stochastic component change. Using the Itô 's lemma, we could find the solution of an Itô equation for jump process [3], we see that

$$d\psi = \frac{\partial \psi}{\partial x} r dt + (\psi(x+s) - \psi(x))dN_\lambda. \tag{2.8}$$

Next, we compute the expected value of ψ. The general rule for computing the expectations [3] is to replace the dN by λdt, then divide by dt and take expectations. This gives

$$\frac{d}{dt}\mathcal{E}\psi(x(t)) = \mathcal{E}\frac{\partial \psi}{\partial x} r + \mathcal{E}(\psi(x(t)+s) - \psi(x(t)))\lambda. \tag{2.9}$$

Now notice that if ρ exists then we can also compute the expectation of ψ by integrating against ρ. That is, $\mathcal{E}\psi(x) = \int \psi(x)\rho(t,x)dx$. Differentiating this expression with respect to time gives

$$\frac{d}{dt}\mathcal{E}\psi(x(t)) = \frac{d}{dt}\int \psi(x)\rho(t,x)dx. \tag{2.10}$$

Then we equate the two functions above to get,

$$\begin{aligned} \frac{d}{dt}\int \psi(x)\rho(t,x)dx &= \mathcal{E}\frac{\partial \psi}{\partial x}r + \mathcal{E}(\psi(x(t)+s)-\psi(x(t)))\lambda \quad (2.11)\\ &= \int \left[\frac{\partial \psi(x)}{\partial x}r + (\psi(x+s)-\psi(x))\lambda\right]\rho(t,x)dx. \quad (2.12)\end{aligned}$$

Integrable in the sense of Riemann, if $\rho(t,x)$ is smooth function of x, we may change the order of integration and differentiation. Then we use integration-by-parts to get

$$\begin{aligned} \int \psi(x)\frac{\partial \rho(t,x)}{\partial t}dx &= \int \frac{\partial \psi(x)}{\partial x}r\rho - \int \lambda\rho\psi dx + \int \lambda\psi(x+s)\rho dx \quad (2.13)\\ &= r\rho\psi(x)| - \int \psi\frac{\partial(r\rho)}{\partial x} - \lambda\psi\rho dx \quad (2.14)\\ &+ \int \lambda\psi(x+s)\rho(t,x)dx \quad (2.15)\\ &= \int -\psi\frac{\partial(r\rho)}{\partial x} - \lambda\psi\rho dx + \int \lambda\psi(x+s)\rho(t,x)dx. \quad (2.16)\end{aligned}$$

The term $r\rho\psi(x)| = 0$ provided $\psi(x) = 0$ for $|x|$ sufficiently large. In order to go further, we define a function by

$$g(x) = x + s. \quad (2.17)$$

The variable x takes on values in R and g defines a map of R onto R which is one to one. Letting $z = g(x)$ we have

$$dz = dx. \quad (2.18)$$

This allows us to make a change of variables to get

$$\int \psi(x+s)\rho(t,x)dx = \int \psi(z)\rho(t,g(z)^{-1})dz = \int \psi(x)\rho(t,x-s)dx. \quad (2.19)$$

Now we can replace $\psi(x+s)$ with $\psi(x)$ in equation (2.16)to get

$$\begin{aligned} \int \psi(x)\frac{\partial \rho(t,x)}{\partial t}dx &= \int -\psi\frac{\partial(r\rho)}{\partial x} - \lambda\psi\rho dx + \int \lambda\psi(x)\rho(t,x-s)dx \quad (2.20)\\ &= \int \psi(x)(-\frac{\partial(r\rho)}{\partial x} - \lambda\rho + \lambda\rho(t,x-s))dx. \quad (2.21)\end{aligned}$$

Finally,

$$0 = \int \psi(x) \left[\frac{\partial \rho(t,x)}{\partial t} + \frac{\partial r\rho(t,x)}{\partial x} + \lambda\rho(t,x) - \lambda\rho(t,x-s) \right] dx. \tag{2.22}$$

In this case we can argue that because ψ is arbitrary this integral equation can be replaced by the differential difference equation.

$$\frac{\partial \rho(t,x)}{\partial t} = -\frac{\partial [r\rho(t,x)]}{\partial x} + \lambda\rho(t,x-s) - \lambda\rho(t,x). \tag{2.23}$$

This is the evolution equation for the density, provided a smooth density exists. If we have $dx = rxdt - sdN_\lambda$ then we can apply similar calculation to get the evolution equation for the density

$$\frac{\partial \rho(t,x)}{\partial t} = -r\frac{\partial [x\rho(t,x)]}{\partial x} + \lambda\rho(t,x+s) - \lambda\rho(t,x). \tag{2.24}$$

However, given this partial differential equation, it is still extremely difficult, or nearly impossible, to find out the explicit solution for the density.

2.2 Derivation of PDE with Brownian motion

Secondly, our goal is to find out the evolution equation for the density associated with the state

$$dx = -xdt + dw, \tag{2.25}$$

where dw is the limit as λ goes to infinity of $(dN_\lambda - dN_{-\lambda})/\sqrt{\lambda}$, and has the properties of Brownian Motion.

Now, let's take a quick review of the Brownian Motion. Brownian motion [5] is the random movement of microscopic particles suspended in a liquid or gas, caused by collisions between these particles and the molecules of the liquid or gas. This movement is named for its identifier, Scottish botanist Robert Brown (1773-1858) who became interested in stochastic processes after looking under a microscope at grains of pollen suspended in water. After he published influential papers on this subject, Norbert Wiener developed the mathematical study of stochastic processes in the 1920's. We chose one way of approaching the mathematics of this subject that is related to the Poisson counter we discussed.

The next three facts are basis of Brownian motion [3].

- $x_\lambda = 0$.
- $x_\lambda(t) - x_\lambda(\tau)$ is a random variable whose distribution depends only on $|t-\tau|$ and if $[t,\tau]\bigcup[s,\sigma] = \emptyset$ then the random variables $x_\lambda(t) - x_\lambda(\tau)$ is independent of $x_\lambda(s) - x_\lambda(\sigma)$. This is because N on $[t,\tau]$ is independent of N on $[s,\sigma]$.
- The limit as $\lambda \to \infty$ of

$$\mathcal{E}(x_\lambda(t) - x_\lambda(\tau))^2 = \mathcal{E}(x_\lambda(t))^2 - 2\mathcal{E}x_\lambda(t)x_\lambda(\tau) + \mathcal{E}(x_\lambda(\tau))^2 \tag{2.26}$$

  exists and is just $|t-\tau|$.

Our work begins by the derivation

$$dw = \lim_{\lambda\to\infty} \frac{dN_\lambda - dN_{-\lambda}}{\sqrt{\lambda}}. \tag{2.27}$$

As before, let ψ be a twice differentiable smooth function having compact support. Consider the evaluation for

$$dx = -xdt + (dN_\lambda - dN_{-\lambda})/\sqrt{\lambda}, \tag{2.28}$$

where N_λ and $N_{-\lambda}$ are independent Poisson counters of rate $\lambda/2$. Using $It\hat{o}$ rule we get

$$\begin{aligned} d\psi &= -\frac{\partial\psi}{\partial x}xdt + \left[\psi(x+\frac{1}{\sqrt{\lambda}}) - \psi(x)\right]dN_\lambda + \\ &\left[\psi(x-\frac{1}{\sqrt{\lambda}}) - \psi(x)\right]dN_{-\lambda}. \end{aligned}$$

In order to explore the limit as λ goes to infinity, we expand ψ in a Taylor series about x.

The result is

$$d\psi = -\frac{\partial\psi}{\partial x}xdt + \frac{\partial\psi}{\partial x}(dN_\lambda - dN_{-\lambda})/\sqrt{\lambda} + \frac{1}{2}\frac{\partial^2\psi}{\partial x^2}(dN_\lambda + dN_{-\lambda})/\lambda + O(1/\lambda^{3/2})dN_i. \tag{2.29}$$

Consider now the case

$$dz_\lambda = (dN_\lambda + dN_{-\lambda})/\lambda, \tag{2.30}$$

where N_λ and $N_{-\lambda}$ are independent Poisson counters of rate $\lambda/2$. Using the formula for expectation, we see that

$$\mathcal{E}z_\lambda(t) = t + \mathcal{E}z(0). \tag{2.31}$$

Let $m_\lambda = z_\lambda^2(t)$. Then from the *Itô* rule for jump processes we see that

$$\begin{aligned} dm_\lambda &= \left(\left(z_\lambda + \frac{1}{\lambda}\right)^2 - z_\lambda^2\right)(dN_\lambda + dN_{-\lambda}) & (2.32)\\ &= \frac{2z_\lambda}{\lambda}(dN_\lambda + dN_{-\lambda}) + \frac{1}{\lambda^2}(dN_\lambda + dN_{-\lambda}). & (2.33)\end{aligned}$$

A short calculation, using $\mathcal{E}dN_i = (\lambda/2)dt$, yields

$$\frac{d}{dt}\mathcal{E}m_\lambda(t) = \mathcal{E}(2z\lambda) + 1/\lambda. \tag{2.34}$$

Thus if $\mathcal{E}z(0) = 0$ we have

$$\mathcal{E}m_\lambda(t) = t^2 + t/\lambda. \tag{2.35}$$

It is remarkable that the variance of the process defined by z, namely

$$\sigma(z(t)) = \mathcal{E}(z_\lambda(t) - \mathcal{E}z_\lambda(t))^2, \tag{2.36}$$

which goes along with the initial condition $z(0) = 0$ is just

$$\begin{aligned} \sigma(z(t)) &= t^2 + t/\lambda - 2t^2 + t^2 & (2.37)\\ &= t/\lambda & (2.38)\end{aligned}$$

and goes to zero as λ goes to infinity for every fixed t. This says that the uncertainty associated with z decreases with increasing λ and hence the process tends to the deterministic process defined by $z(t) = t$. As a result, in the limit we may make the substitution, as λ goes to infinity, $(dN_\lambda + dN_{-\lambda})/\lambda = dt$.

This finishes the analysis of $(dN_\lambda + dN_{-\lambda})/\lambda$. Thus for

$$dx = -xdt + dw \tag{2.39}$$

we have

$$d\psi = -\frac{\partial\psi}{\partial x}xdt + \frac{\partial\psi}{\partial x}dw + \frac{1}{2}\frac{\partial^2\psi}{\partial x^2}dt. \tag{2.40}$$

If we are interested in $\frac{d}{dt}\mathcal{E}\psi$, we have on one hand

$$\frac{d}{dt}\mathcal{E}\psi = \frac{d}{dt}\int_R \psi(x)\rho(t,x)dx = \int_R \psi(x)\frac{\partial\rho(t,x)}{\partial t}dx. \tag{2.41}$$

On the other hand,

$$\frac{d}{dt}\mathcal{E}\psi = \mathcal{E}(-\frac{\partial\psi}{\partial x}x) + \frac{1}{2}\mathcal{E}(\frac{\partial^2\psi}{\partial x^2}) \tag{2.42}$$

$$= \int_R (-\frac{\partial\psi}{\partial x}x + \frac{1}{2}\frac{\partial^2\psi}{\partial x^2})\rho(t,x)dx. \tag{2.43}$$

Integrating this by parts we get

$$\frac{d}{dt}\mathcal{E}\psi = \int_R -\psi\frac{\partial(-x\rho)}{\partial x} + \psi\frac{1}{2}\frac{\partial^2}{\partial x^2}\rho(t,x)dx. \tag{2.44}$$

Putting these two formulas for $\frac{d}{dt}\mathcal{E}\psi$ together we have

$$0 = \int_R \psi\left(\frac{\partial\rho}{\partial t} + \frac{\partial(-x\rho)}{\partial x} - \frac{1}{2}\frac{\partial^2}{\partial x^2}\rho(t,x)\right)dx. \tag{2.45}$$

The only way this can be hold true for all ψ is if

$$\frac{\partial\rho(t,x)}{\partial t} = \frac{\partial[x\rho(t,x)]}{\partial x} + \frac{1}{2}\frac{\partial^2}{\partial x^2}\rho(t,x). \tag{2.46}$$

This is, then, the evolution equation for the Itô equation $dx = -xdt + dw$.

2.3 PDE with both jump process and Brownian motion

From the previous work, we have seen that when we derive for the evolution equations for the density, what we actually derived is only the terms with dN or dw.

The derivation of the term $f(x)dt$ is independent from the other term. Based on this fact, now it's very easy for us to find out the evolution equation for the following linear equation

$$dx = \tau x dt + a dw + b dN \tag{2.47}$$

where dw is the limit as λ goes to infinity of $(dN_\lambda - dN_{-\lambda})/\sqrt{\lambda}$.

We can simply combine the result to get

$$\frac{\partial \rho(t,x)}{\partial t} = \frac{\partial[-\tau x \rho(t,x)]}{\partial x} + \frac{a}{2}\frac{\partial^2}{\partial x^2}\rho(t,x) + \lambda\rho(t,x-b) - \lambda\rho(t,x). \tag{2.48}$$

This is the evolution stochastic differential equation with both poisson counter and Brownian motion. It is more likely the practical model which could perfectly approach the real situation. However, it is extremely difficult, even impossible, to find out the explicit solution so that we could not do further analysis.

CHAPTER 3
SIMULATION AND ANALYSIS

As an application of the Poisson counter driven stochastic differential equation of this form $dx = rxdt + sdN_\lambda$. We focus on applying the model to analyze Subprime mortgage problem. When the households must finance their purchase, they probably will choose to pay off in several decades, usually 10, 20 or 30 years. We develop a very simple simulation model for such analysis.

We assume x , as a combinatory variable, to evaluate the borrower's ability to continue making payments, refinance, default or pay off. It is scaled so that 0 represents defaulted and 1 represents paid off. This variable is actually determined by complicated factors including the household's underlying credit quality, their revenue, annual bonus, etc. The household's problem can be considered as an optimal stopping problem, which is proposed by Dr. Robert Martin [2]. The upper boundary (where x=0) is defined as the region of the state-space in which the household optimally defaults. The lower boundary (where x=1) is the region where the household pays off the mortgage. Our goal is to respectively find out the distribution of time points when the household pays off or default the loan. We will record the numerical simulation solution for the model and draw the histogram based on those data for further analysis.

Given the equation $dx = rxdt - sdN_\lambda$, for each treatment we assume the following approximation difference equation as the model

$$x_{n+1} = (1 + r)x_n - s\epsilon_n, \tag{3.1}$$

where x is the variable which describes the mortgage borrower's repayment capability, and the initial value x_0 for x is distributed in accordance with normal so that $x_0 \sim N(0.5, 0.01)$. Ideally, we suppose the household's paying ability will increase slowly with time, hence r stands for the growth rate which is a positive real number in $(0, 1)$. During the mortgage period, we apply the $s\epsilon_n$ term as the bad accidents such as divorce, job loss, career moves, etc. which will dramatically decrease the paying ability. We make assumptions for ϵ_n as follows:

$$\epsilon_n = \begin{cases} 1, & \text{at jump timepoints (accidents happened at these timepoint)} \\ 0, & \text{elsewhere.} \end{cases} \tag{3.2}$$

To determine the jump time points, we apply the algorithm as follows:

- The jump time points are determined by $P(t) = \int_0^t 0.005e^{-0.005s}ds$.
- Draw a random number as the value of P.
- Calculate the expression for t, $t = ln(1-P)/(-0.005)$. Given P, we could compute the jump time points t_i^j, where i stands for the number of treatment, j is the number of jump in one treatment.
- $n_{ij} = \lfloor t_i^j \rfloor$, where n_{ij} represents the jump months.
- When $\sum_{j=1} t_i^j > 360$, stop and return n_{ij}.

The constant s is the value which could determine the effect of the jump term. We are going to use computational simulation to find out the escape time distribution. We assume the mortgage year is 30 years. During this 360 month, the jump time is randomly chosen. Through the computational simulation, we return the month in which either the household paid off (x=1) or defaulted (x=0). In the model, the two constant parameters r and s are mutually checked and supervised. We would like to switch the values of them in an effort to best approach the practical situation. After 10,000 treatments, we may find the histograms which are obtained by recording the frequency of those jump time points (in months).

Finally, we determined the following as the ideal values for parameters: $r = 0.2, s = 0.4$. Figure 3.1 and Figure 3.2 show us the escape time distribution for stochastic flows.

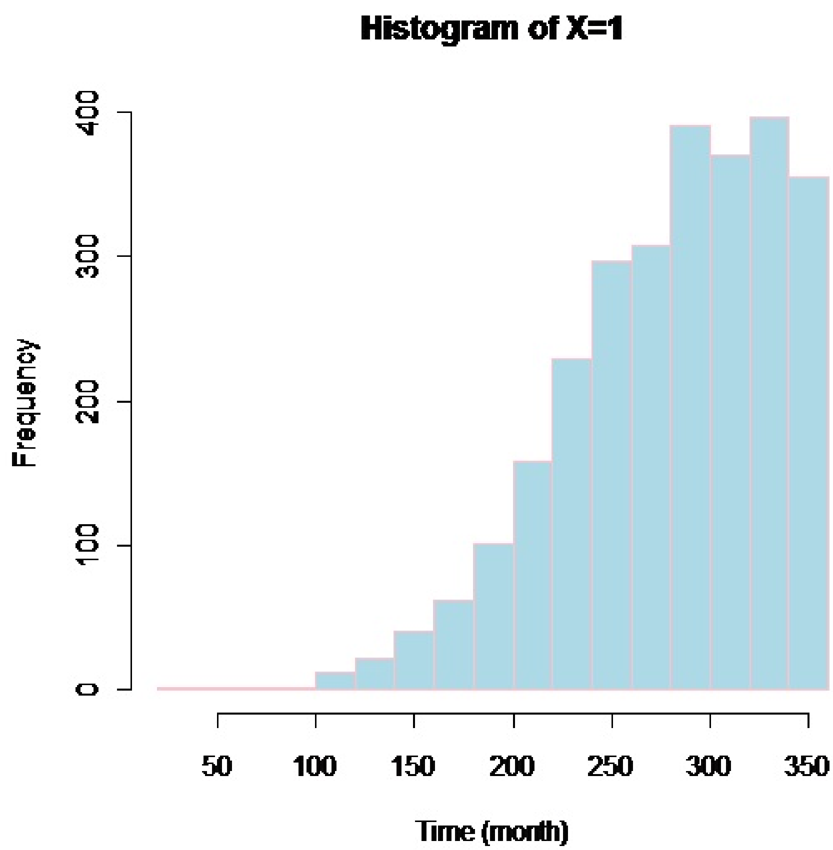

Figure 3.1. Escape time distribution for mortgage paid off

Figure 3.1 represents the escape time distribution when the households pay off the mortgage. We can see that a negatively skewed curve is shown in this figure. It is skewed to the left. Its greatest frequency occurs at values between 250 to 350 at the right of the graph. The results above show that it is possible for the household to pay off the mortgage in the third decades.

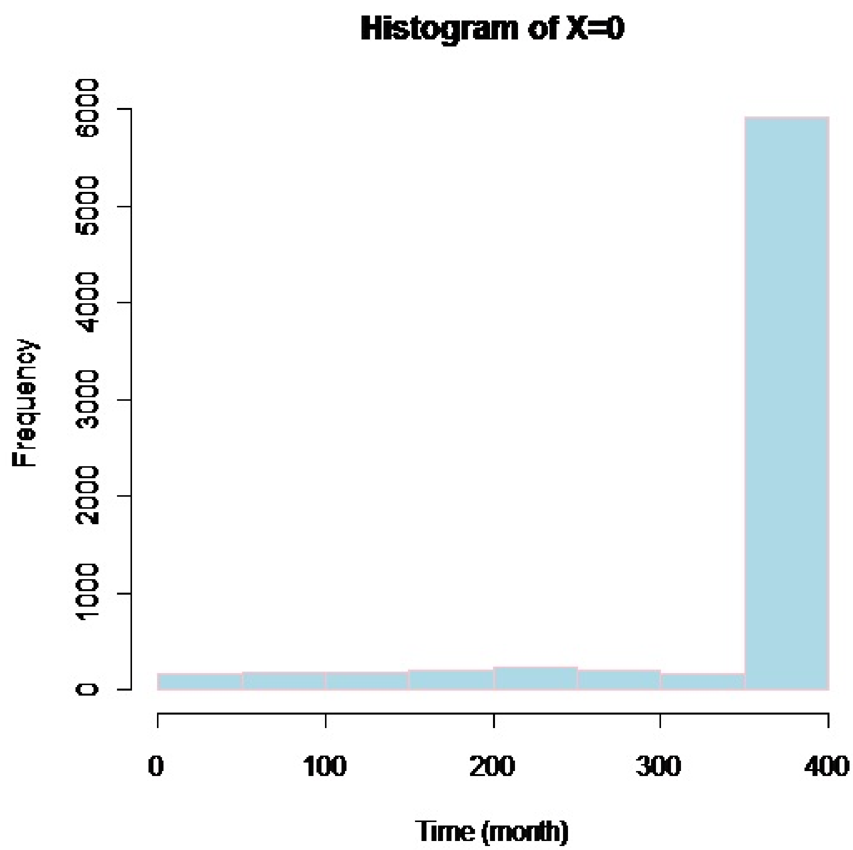

Figure 3.2. Escape time distribution for mortgage default

Figure 3.2 shows out the escape time distribution when the households default the mortgage. Evidently, the frequency of the last 10 months is significantly higher than any previous years. Based on our simulation model, there are two mainly sources for the failures to pay at the end of the paying period. One case is that t0o many jump process were randomly created which resulted the negative factors added to x, so that x is deducted to 0 finally. The other case is that after 360 month, x is neither 0 nor 1. Under this situation, we consider it fail to pay. x is defaulted to be 0. According to the data, the latter case is the main reason why there are so many defaults happened in the last 10 months.

BIBLIOGRAPHY

[1] B. S. Bernanke, *The Subprime Mortgage Market*, Conference Speech, Chicago, Illinois, May 2007.

[2] R. F. Martin, C. F. Martin, N. Kojucharov and L. Xu, *The Subprime Mortgage Crisis: Irrational Exuberance or Rational Error?*, Working Paper, Federal Reserve Board of Governors, 2008.

[3] R. Brockett, *Stochastic control*, Lecture Notes, Harvard University, 2009.

[4] K. Ito, *On stochastic differential equations*, Memoirs, American Mathematical Society 4(1951): 1-51.

[5] R. Brown, *A brief account of microscopical observations made in the months of June, July and August, 1827, on the particles contained in the pollen of plants; and on the general existence of active molecules in organic and inorganic bodies*, Phil. Mag. 4(1828): 161-173.

A

C++ CODE

```
#include <iostream> #include <fstream> #include <cstring> #
    include
<string> #include "randomc.h" #include <vector> #include <map
    >
#include <time.h> #include <math.h>

TRandomMersenne gRandGen( ( unsigned long )time( NULL ) );
    using
namespace std;

double round(double a) {
    double intpart;
    double result;
    if (modf(a,&intpart)>=0.5)  result=ceil(a);
    else result=floor(a);
    return(result);
}

vector<double> detertimepoints() {
    int i;
    int complete = 0;
    int duplicate = 0;
    double temp;
    vector<double> timepoints;
    vector<double> result;
    double random;
    double totaltime = 0;
    while(complete!=1)
```

```
    {
        duplicate = 0;
        random = (double)(gRandGen.IRandom(0,9)*100000000+
            gRandGen.IRandom(0,9)*10000000+gRandGen.IRandom
            (0,9)*1000000+gRandGen.IRandom(0,9)*100000+
            gRandGen.IRandom(0,9)*10000+gRandGen.IRandom(0,9)
            *1000+gRandGen.IRandom(0,9)*100+gRandGen.IRandom
            (0,9)*10+gRandGen.IRandom(0,9)+1)/1000000000;
        temp = log(1-random)/(-0.002);

        if (timepoints.size()<1) timepoints.push_back(temp);
        else timepoints.push_back(timepoints[timepoints.size
            ()-1]+temp);
        if (timepoints[timepoints.size()-1] > 360) complete =
             1;

        i = 0;
        while(i<result.size())
        {
            if (timepoints[timepoints.size()-1]>= result[i]) i
                ++;
            else break;
        }
        if (complete != 1) result.insert(result.begin()+i,
            floor(timepoints[timepoints.size()-1]));
    }
    return (result);
}

double generatenormal() {
    double random1 = (double)(gRandGen.IRandom(0,9)
        *100000000+gRandGen.IRandom(0,9)*10000000+gRandGen.
```

```
        IRandom(0,9)*1000000+gRandGen.IRandom(0,9)*100000+
        gRandGen.IRandom(0,9)*10000+gRandGen.IRandom(0,9)
        *1000+gRandGen.IRandom(0,9)*100+gRandGen.IRandom(0,9)
        *10+gRandGen.IRandom(0,9)+1)/1000000000;
    double random2 = (double)(gRandGen.IRandom(0,9)
        *100000000+gRandGen.IRandom(0,9)*10000000+gRandGen.
        IRandom(0,9)*1000000+gRandGen.IRandom(0,9)*100000+
        gRandGen.IRandom(0,9)*10000+gRandGen.IRandom(0,9)
        *1000+gRandGen.IRandom(0,9)*100+gRandGen.IRandom(0,9)
        *10+gRandGen.IRandom(0,9)+1)/1000000000;
    double normal;
    double pi = 3.1415926535897932;
    normal = sqrt(-2*log(random1))*cos(2*pi*random2)/10+0.5;
    return(normal);
}

vector<int> singlerun(double scalar) {
    vector<double> timepoints;
    vector<double> x;
    double temp;
    int i = 1;
    int j;
    int jump;
    vector<int> result;
    timepoints = detertimepoints();
    temp = generatenormal();
    x.push_back(temp);
    while(i<=360)
    {
        jump = 0;
        for(j=0; j<timepoints.size(); j++)
        {
```

```
            if(i == timepoints[j]) jump = 1;
        }
        temp = (pow(2.0,double(1)/360)*temp)-scalar*jump;
        x.push_back(temp);
        if (temp <=0)
        {
            result.push_back(0);
            result.push_back(i);
            break;
        }
        if (temp >=1)
        {
            result.push_back(1);
            result.push_back(i);
            break;
        }
        if (i==360 && temp>0 && temp<1)
        {
            result.push_back(0);
            result.push_back(360);
        }
        i++;
    }

    for(j=0; j<timepoints.size(); j++) result.push_back(
       timepoints[j]);

    return(result);
}

void main() {
```

```
int i;
int j;
double fre=0;
double scalar=0.4;
double total0=0,total1=0;
vector<int> temp;
FILE *f;
FILE *o;
FILE *z;
f=fopen("Result_new.txt", "w");
o=fopen("Result_one.txt", "w");
z=fopen("Result_zero.txt", "w");
fprintf(f,"Scalar=0.4\tStandard Deviation=0.1\n");

for (i=0; i<10000; i++)
{
    temp = singlerun(scalar);
    if (temp[0] == 0)
    {
        total0+=temp[1];
        fre++;
        fprintf(z,"%d\n",temp[1]);
    }
    if (temp[0] == 1)
    {
        total1+=temp[1];
        fprintf(o,"%d\n",temp[1]);
    }

    for (j = 0; j<temp.size(); j++)
    {
        if (j<2)
```

```
            {
                fprintf(f,"%d\t",temp[j]);
                //printf("1\n");
                //getchar();
            }
            else if (j==2)
            {
                fprintf(f,"Jump Time Points:\t%d\t",temp[2]);
                //printf("2\n");
                //getchar();
            }
            else
            {
                fprintf(f,"%d\t",temp[j]);
                //printf("3\n");
                //getchar();
            }
        }

        fprintf(f,"\n");
    }
    fprintf(f, "Avg. 0 =%f\t Avg. 1 =%f\n",total0/fre,total1
       /(10000-fre));
    fprintf(f, "Proportion of 0 =%f\t Proportion of 1 =%f%\n"
       ,fre/100,(1-(fre/10000))*100);
    fclose(f);
    fclose(o);
    fclose(z);
    printf("Complete");
    getchar();

}
```

B

R CODE

```
x <- scan("hist_zero.txt") hist(x, breaks = 12, xlab = "XXX",
    ylab
= "XXX", main = "XXX", col="lightblue", border="pink")
```