Simplified CreditMetrics™ Model Using Monte Carlo Simulation

By James, Chan Kwok Cheung

June 2023

About

CreditMetrics™ is a widely used framework developed by J.P. Morgan & Co. in 1997 for assessing potential credit risk in a bond portfolio due to obligor's failure to repay their debts. Due to the model's complex mathematical derivations, it is not sensible to find a closed form formula for a well-diversified bond's portfolio. Thus, this project aims to approximate the Value-at-Risk (VaR) and the Expected Shortfall (ES) of a bond portfolio using Monte Carlo simulation, in order to provide a quantified assessment of a bond portfolio's risk profile.

Project GitHub Link: https://github.com/jamesckcc/CreditMetrics

Theoretical Background

One Bond Case:

The model is based on the analysis of migration in the credit rating transition matrix. Let's say there are seven rating categories (AAA, AA, A, BBB, B, CCC). For a single BBB rated bond, in one-years' time, it's credit rating could either upgrade to AAA, AA, A, retain its rating (BBB), downgrade to BBB, B, CCC, or Default. Assuming that the One-year transition matrix is known (See *Table 1.1*).

Table 1.1

One-year transition matrix (%)

Initial rating	Rating at year-end (%)							
	AAA	AA	A	BBB	BB	B	CCC	Default
AAA	90.81	8.33	0.68	0.06	0.12	0	0	0
AA	0.70	90.65	7.79	0.64	0.06	0.14	0.02	0
A	0.09	2.27	91.05	5.52	0.74	0.26	0.01	0.06
BBB	0.02	0.33	5.95	86.93	5.30	1.17	0.12	0.18
BB	0.03	0.14	0.67	7.73	80.53	8.84	1.00	1.06
B	0	0.11	0.24	0.43	6.48	83.46	4.07	5.20
CCC	0.22	0	0.22	1.30	2.38	11.24	64.86	19.79

Source: Standard & Poor's CreditWeek (15 April 96)

Then the credit rating migration probability for the BBB Rated Bond could be obtained, as shown in *Table 1.2.*

Table 1.2

Probability of credit rating migrations in one year for a BBB

Year-end rating	Probability (%)
AAA	0.02
AA	0.33
A	5.95
BBB	86.93
BB	5.30
B	1.17
CCC	0.12
Default	0.18

Since the value of a bond is directly related to the counterparty's credibility, the bond's value in one-years' time would be different if it falls to different rating categories. Consider we have also obtained the one-year forward zero curves by credit rating category (*Table 1.3)*, then we could find the value of the bond given what category it falls to after one-year.

Table 1.3

Example one-year forward zero curves by credit rating category (%)

Category	Year 1	Year 2	Year 3	Year 4
AAA	3.60	4.17	4.73	5.12
AA	3.65	4.22	4.78	5.17
A	3.72	4.32	4.93	5.32
BBB	4.10	4.67	5.25	5.63
BB	5.55	6.02	6.78	7.27
B	6.05	7.02	8.03	8.52
CCC	15.05	15.02	14.03	13.52

Let's say the BBB-rated bond have the following attributes:

- Face Value = 100
- Maturity = 5 years
- Coupon Rate = 6%
- Senior Secured Class

If the bond upgrades to A, the value in one year could be calculated using discounted cashflow as follow:

$$V_{BBB} = 6 + \frac{6}{1.0372} + \frac{6}{(1.0432)^2} + \frac{6}{(1.0493)^3} + \frac{106}{(1.0532)^2} = 108.66$$

By using this method, we can then find the value of the bond if it falls into other rating categories. Further assume we obtained the recovery rates given default by seniority class in *Table 1.4*, then we would have the distribution of bond value in one year's time, *Table 1.5*.

Table 1.4

Recovery rates by seniority class (% of face value, i.e., "par")

Seniority Class	Mean (%)	Standard Deviation (%)
Senior Secured	53.80	26.86
Senior Unsecured	51.13	25.45
Senior Subordinated	38.52	23.81
Subordinated	32.74	20.18
Junior Subordinated	17.09	10.90

Source: Carty & Lieberman [96a] —Moody's Investors Service

Table 1.5

Distribution of value of a BBB par bond in one year

Year-end rating	Value ($)	Probability (%)
AAA	109.37	0.02
AA	109.19	0.33
A	108.66	5.95
BBB	107.55	86.93
BB	102.02	5.30
B	98.10	1.17
CCC	83.64	0.12
Default	51.13	0.18

Then the mean and the standard deviation of the portfolio would be defined as following,

$$\mu = \sum p_i * \mu_i \ = 107.09$$

$$\sigma = \sqrt{\sum p_i * \left(\mu_i^2 + \sigma_i^2\right) - \mu^2} = 3.18$$

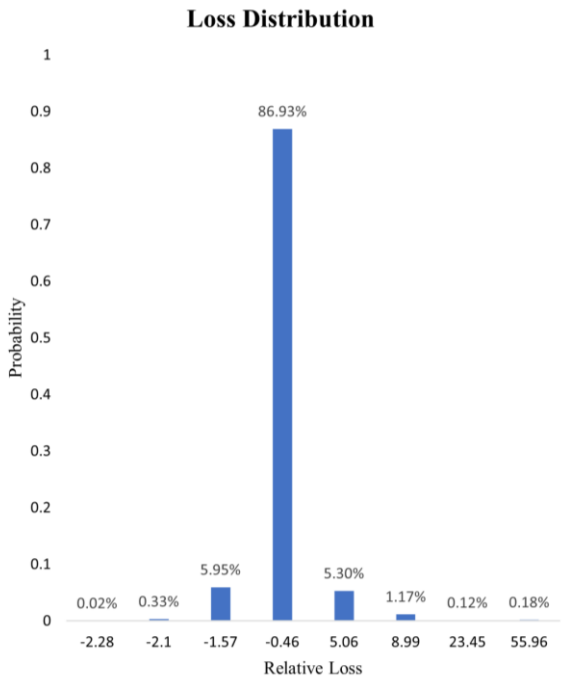
Note that the standard deviation added a component σ_i^2 to include the uncertainty in the bond value. In this case, it is only applicable to $i = 8$, the defaulted state, as this is the only situation where the value is uncertain. This result is proven in the original technical document.

By setting up the relative loss distribution as

$$\begin{aligned} Relative\ Loss &= Expected\ Portfolio\ Value - Portfolio\ Value \\ &= 107.09 - Portfolio\ Value \end{aligned}$$

We obtained *Chart 1.1:*

Chart 1.1



The 95% relative Value-at-Risk (VaR) of the portfolio would be found to be $5.06.

The Expected Shortfall (ES) is:

$$E[L|L > 95\%\ rel.VaR] = 8.25$$

Two Bonds Case:

In order to construct the portoflio's value distribution, the joint migration probability is to be considered. Since it is impossible to obtain the real joint probaility density function of the two bonds credit migration, CreditMetrics™ proposed a clever way to estimate the distribution, by using normal approximation, that could also capture the correlation between bonds.

To make an example, assume the portfolio consist of the following two bonds with the same weight, and have a correlation $\rho = 0.3$ between them.

Bond 1

- BBB-rated
- Face Value = 100
- Maturity = 5 years
- Coupon Rate = 6%
- Senior Secured Class

Bond 2

- A-rated
- Face Value = 100
- Maturity = 5 years
- Coupon Rate = 6%
- Senior Secured Class

Bond 1, the BBB-rated bond, by using normal approximation, would have this marginal value distribution:

Graph 1.1

Distribution of Bond Credit Rating in one-year

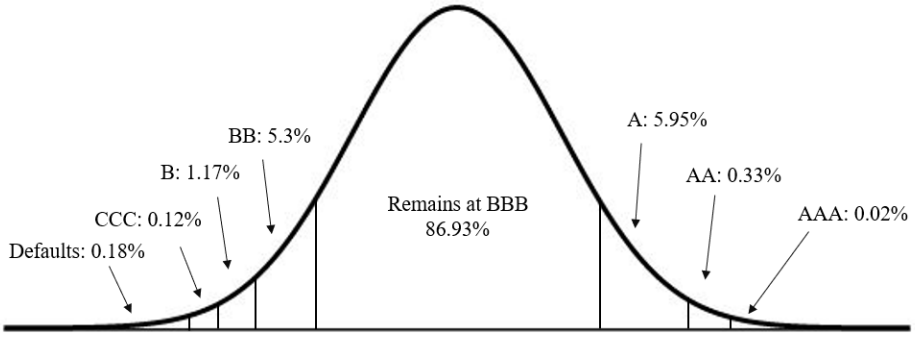


Table 1.6

Rating	Probability	Z-value range	
		Min	Max
AAA	0.02%	3.54	inf
AA	0.33%	2.70	3.54
A	5.95%	1.53	2.70
BBB	86.93%	-1.49	1.53
BB	5.3%	-2.18	-1.49
B	1.17%	-2.74	-2.18
CCC	0.12%	-2.91	-2.75
Default	0.18%	-inf	-2.91

We could validate the result mathematically, take the bond remaning at BBB as an example:

$$Pr(\text{Remains at BBB}) = \int_{-1.49}^{1.53} \frac{1}{\sqrt{2\pi}} exp(-\frac{x^2}{2}) dx = 86.93\%$$

Also consider Bond 2, an A-rated bond:

Table 1.7

Rating	Probability	Z-value range	
		Min	Max
AAA	0.09%	3.12	inf
AA	2.27%	1.98	3.12
A	91.05%	-1.51	1.98
BBB	5.52%	-2.30	-1.51
BB	0.74%	-2.72	-2.30
B	0.26%	-3.19	-2.72
CCC	0.01%	-3.24	-3.19
Default	0.06%	-inf	-3.24

Assuming joint normality, we have the joint pdf:

$$f(x_1, x_2; \rho) = \frac{1}{2\pi\sqrt{1-\rho^2}} exp\left\{-\frac{1}{2(1-\rho^2)}[x_1^2 - 2\rho x_1 x_2 + x_2^2]\right\}$$

In this portfolio, it would be:

$$f(x_1, x_2; 0.3) = \frac{1}{2\pi\sqrt{1-0.3^2}} exp\left\{-\frac{1}{2(1-0.3^2)}[x_1^2 - 2 * 0.3 x_1 x_2 + x_2^2]\right\}$$

Then for example, the probability that both bonds retains their rating is

$$\int_{-1.51}^{1.98}\int_{-1.49}^{1.53}\frac{1}{2\pi\sqrt{1-0.3^2}}exp\left\{-\frac{1}{2(1-0.3^2)}[x_1^2-2*0.3x_1x_2+x_2^2]\right\}dx_1dx_2=79.69\%$$

By doing so for the remaning 63 possibilities, we would obtain the joint migration probabilites matrix, as shown in *Table 1.8.*

Table 1.8

Joint migration probabilities with 0.30 asset correlation (%)

Obligor #1 (BBB)		Obligor #2 (single-A)							
		AAA	AA	A	BBB	BB	B	CCC	Default
		0.09	2.27	91.05	5.52	0.74	0.26	0.01	0.06
AAA	0.02	0.00	0.00	0.02	0.00	0.00	0.00	0.00	0.00
AA	0.33	0.00	0.04	0.29	0.00	0.00	0.00	0.00	0.00
A	5.95	0.02	0.39	5.44	0.08	0.01	0.00	0.00	0.00
BBB	86.93	0.07	1.81	79.69	4.55	0.57	0.19	0.01	0.04
BB	5.30	0.00	0.02	4.47	0.64	0.11	0.04	0.00	0.01
B	1.17	0.00	0.00	0.92	0.18	0.04	0.02	0.00	0.00
CCC	0.12	0.00	0.00	0.09	0.02	0.00	0.00	0.00	0.00
Default	0.18	0.00	0.00	0.13	0.04	0.01	0.00	0.00	0.00

By further finding the portfolio value given any of the 64 situations, we could find the portfolio expected one-year value and it's standard deviation and further risk profile:

Table 1.9

All possible 64 year-end values for a two-bond portfolio ($)

Obligor #1 (BBB)		Obligor #2 (single-A)							
		AAA	AA	A	BBB	BB	B	CCC	Default
		106.59	106.49	106.30	105.64	103.15	101.39	88.71	51.13
AAA	109.37	215.96	215.86	215.67	215.01	212.52	210.76	198.08	160.50
AA	109.19	215.78	215.68	215.49	214.83	212.34	210.58	197.90	160.32
A	108.66	215.25	215.15	214.96	214.30	211.81	210.05	197.37	159.79
BBB	107.55	214.14	214.04	213.85	213.19	210.70	208.94	196.26	158.68
BB	102.02	208.61	208.51	208.33	207.66	205.17	203.41	190.73	153.15
B	98.10	204.69	204.59	204.40	203.74	201.25	199.49	186.81	149.23
CCC	83.64	190.23	190.13	189.94	189.28	186.79	185.03	172.35	134.77
Default	51.13	157.72	157.62	157.43	156.77	154.28	152.52	139.84	102.26

$$\mu=\sum_{i=1}^{64}p_i*\mu_i=213.63$$

$$\sigma=\sqrt{\sum_{i=1}^{64}p_i*(\mu_i^2+\sigma_i^2)-\mu^2}\approx 3.69$$

$$95\%\ Rel.VaR\approx 4.905$$

$$ES\approx 12.20$$

Three or more bonds case:

For three or more bonds, the calculations procedures started to be complex. We will have to find out all of the possible rating combination's price and probability within the bonds.

By normal approximation, the joint pdf of the *n* bonds would be:

$$f(\mathbf{x};\mathbf{\Sigma}) = \left[(2\pi)^{n/2}\sqrt{|\mathbf{\Sigma}|}\right]^{-1} exp\left[-\frac{1}{2}\mathbf{x}^T\mathbf{\Sigma}\mathbf{x}\right]$$

The derivation of the probability of a particular state would be:

$$Pr(\text{Bond 1 rated XYZ} \cap \text{ Bond 2 rated XYZ} \cap \cdots \cap \text{ Bond n rated XYZ})$$
$$= \int_{b_1}^{a_1}\int_{b_2}^{a_2}\cdots\int_{b_n}^{a_n}\left[(2\pi)^{n/2}\sqrt{|\mathbf{\Sigma}|}\right]^{-1} exp\left[-\frac{1}{2}\mathbf{x}^T\mathbf{\Sigma}\mathbf{x}\right] dx_1 dx_2 \ldots dx_n$$

Since for each bond there are 8 possible states in our scenario (AAA, AA, A, BBB, BB, B, CCC, Default), for a *n* bonds portfolio, we have to compute 8^n combination's probability and price. For example, the fixed income fund by BlackRock, *iShares iBoxx $ Investment Grade Corporate Bond ETF(LQD)*, consist of over 1,000 investment grades corporate bonds in a single fund. It is absolutely impossible to evaluate all 8^{1000} combinations. Not even for computer programs, since this computation have a time complexity of $O(8^n)$, it is not sensible to derive the risk profile of the portfolio algebarically. Using Monte Carlo simulation to assess the risk profile would serve a better option, which will be discussed next.

Model Implementation

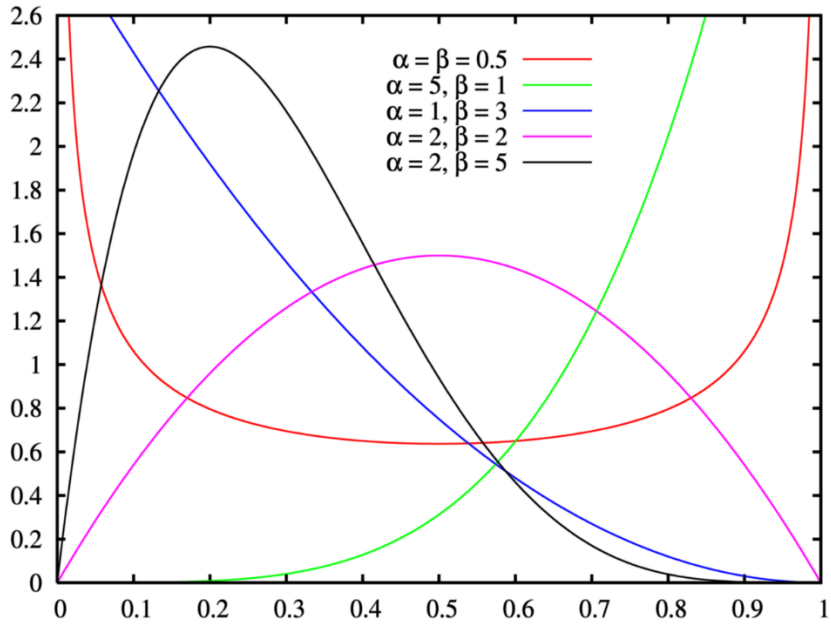
Modelling Recovery Rates

Before delving into the exact implementation of the Monte Carlo simulation in python, we will discuss the way the program models the recovery rates first. The CreditMetrics™ framework did not explicitly mentioned how to model the recovery rate since this is not its main intention. The recovery rate's induced volatility is incorporated in the equation to calculate the portfolio's standard deviation, recall the formula:

$$\sigma = \sqrt{\sum p_i * \left(\mu_i^2 + \sigma_i^2\right) - \mu^2}$$

The σ_i is only non zero iff state i consist at least one of the bonds defaulted. Since for non-defaulted bonds, the exact value of a bond could be found using discounted cash flow method as mentioned earlier. But when a bond defaulted, the Recovery Rate is a random variable with aforementioned mean and standard deviation, without specifying its actual distribution, see *Table 1.4.* As our program intends to also find the ES, the recovery rate distribution when default happened is important for us to assess tail risk. A common probability distribution used to simulate recovery rates is Beta distribution, since it is a flexible, continuous distribution in range (0,1).

Graph 2.1: Different Beta distribution's pdf



The common Beta distribution pdf is defined as follow:

$$f(x;\alpha,\beta) = \begin{cases} \frac{\Gamma(\alpha+\beta)}{\Gamma(\alpha)\Gamma(\beta)} x^{\alpha-1}(1-x)^{\beta-1} & \text{for } 0 < x < 1, \\ 0 & \text{otherwise.} \end{cases}$$

Where $\Gamma(\cdot)$ is the gamma function.

We will have the following attributes after solving it algebarically, which will not be proved entirely here:

For $X \sim Beta(\alpha, \beta)$

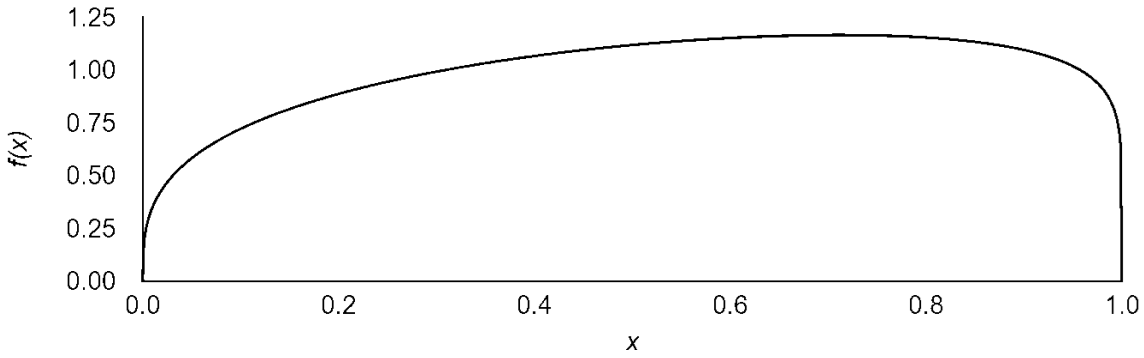
$$E(X) = \frac{\alpha}{\alpha+\beta} \qquad Var(X) = \frac{\alpha\beta}{(\alpha+\beta)^2(\alpha+\beta+1)}$$

If we have the mean and the variance of a Beta Distributed random variable, the maximum likelihood estimator (MLE) of the parameters would be:

$$\alpha = \left(\frac{1-\mu}{\sigma^2} - \frac{1}{\mu}\right)\mu^2 \qquad \beta = \alpha\left(\frac{1}{\mu} - 1\right)$$

For example, the seniority class of Senior Secured have a recovery rate of mean 53.8% and standard deviation of 26.86%, the MLE estimate of the parameters would be $\alpha = 1.3155$, $\beta = 1.1297$.

Graph 2.2: MLE beta distribution of Senior Secured Class's Recovery Rate



The recovery rate modelling methodology will be used throughout the simulation process for all classes of seniority.

Portfolio Example

Since there are too many possible combinations of credit ratings of each bond in a bond portfolio, it isn't sensible to find out each combination's information algebarically. As we mentioned, calculating each possible combination's probability and value would have a time complexity of $O(8^n)$. While Monte Carlo simulation will drastically reduce the time complexity.

```
class Bond:
    def __init__(self, Rating, SeniorityClass, M, Coupon, FV):
        self.Rating = Rating
        self.SeniorityClass = SeniorityClass
        self.M = M
        self.Coupon = Coupon
        self.FV = FV

    def bondinfo(self, df1, df2, df3):
        RatingInfo = []
        Ratings = ["AAA","AA","A","BBB","BB", "B","CCC"]
        cum = 100
        for i in Ratings:
            bondval = bond1y(df2, i, self.M, self.Coupon, self.FV)
            Prob = transprob(df1, self.Rating, i)
            Threshold = stats.norm.ppf(cum/100)
            cum -= Prob
            RatingInfo.append([i,bondval,Threshold, Prob])

        DThreshold = stats.norm.ppf(cum/100)
        RatingInfo.append(["Default", 0, DThreshold])

        RR = df3[self.SeniorityClass]

        Info = [RatingInfo, RR, self.FV]

        return Info
```

In a simplified CreditMetrics model, and also this project's simulation, users will have to input these bonds attributes, as in the above defined *Bond* class:

- Initial Rating
- Seniority Class
- Maturity
- Coupon Rate
- Face Value

```
#Assume There are only ratings AAA, AA, A, BBB, BB, B, CCC, Default
OneYearTransitiondict = {
    "AAA":[90.81,8.33,0.68,0.06,0.12,0,0,0],
    "AA":[0.70,90.65,7.79,0.64,0.06,0.14,0.02,0],
    "A":[0.09,2.27,91.05,5.52,0.74,0.26,0.01,0.06],
    "BBB":[0.02,0.33,5.95,86.93,5.30,1.17,0.12,0.18],
    "BB":[0.03,0.14,0.67,7.73,80.53,8.84,1,1.06],
    "B":[0,0.11,0.24,0.43,6.48,83.46,4.07,5.2],
    "CCC":[0.22,0,0.22,1.3,2.38,11.24,64.86,19.79]
    }

OneYearTransition = pd.DataFrame(OneYearTransitiondict, index = ["AAA", "AA", "A", "BBB", "BB", "B", "CCC", "Default"])
OneYearTransition = OneYearTransition.T



OneYearForwardZeroCurvesdict = {
    "AAA":[3.6,4.17,4.73,5.12,5.51,5.9],
    "AA":[3.65,4.22,4.78,5.17,5.56,5.95],
    "A":[3.72,4.32,4.93,5.32,5.71,6.10],
    "BBB":[4.1,4.67,5.25,5.63,6.01,6.39],
    "BB":[5.55,6.02,6.78,7.27,7.76,8.25],
    "B":[6.05,7.02,8.03,8.52,9.01,9.5],
    "CCC":[15.05,15.02,14.03,13.52,13.01,12.50]
    }

OneYearForwardZeroCurves = pd.DataFrame(OneYearForwardZeroCurvesdict)
OneYearForwardZeroCurves = OneYearForwardZeroCurves.T


Seniority = {
    "Senior Secured": [53.8,26.86],
    "Senior Unsecured": [51.13, 25.45],
    "Senior Subordinated": [38.52,23.81],
    "Subordinated": [32.74,20.18],
    "Junior Subordinated": [17.09, 10.9]
    }

SeniorityPara = create_alpha_beta_dict(Seniority)


df1 = OneYearTransition
df2 = OneYearForwardZeroCurves
df3 = SeniorityPara
```

df1, *df2* and *df3* in *Bond.bondinfo* are predefined DataFrame as shown above. They are one-year transition matrix (*Table 1.1*), one-year forward zero curves by credit rating category (*Table 1.3)* and recovery rates given default by seniority class (*Table 1.4*) respectively. The *create_alpha_beta_dict* function is essentially turning *Table 1.4*'s mean and standard deviation values into Beta distribution's parameters.

In our simulation, the portfolio consist of this 10 bonds:

```
BondA = Bond("A", "Senior Secured", 5,6,3000000)
BondB = Bond("BB", "Senior Unsecured", 4,7,1000000)
BondC = Bond("AA", "Senior Secured", 3,5,4200000)
BondD = Bond("B", "Senior Unsecured", 6,2,1000000)
BondE = Bond("CCC", "Subordinated", 2,8,1100000)
BondF = Bond("BBB", "Senior Unsecured", 4,3,3000000)
BondG = Bond("AAA", "Senior Secured", 3,4,2000000)
BondH = Bond("A", "Senior Secured", 3,5,1900000)
BondI = Bond("CCC", "Junior Subordinated", 2,4,1000000)
```

We will further define the correlation matrix between the bonds, as in our simulation:

```
correlation = [
[1.00,  0.82,  0.45,  0.71,  0.89,  0.65,  0.23,  0.79,  0.76,  0.68],
[0.82,  1.00,  0.51,  0.78,  0.81,  0.61,  0.29,  0.75,  0.73,  0.70],
[0.45,  0.51,  1.00,  0.53,  0.47,  0.56,  0.05,  0.49,  0.52,  0.59],
[0.71,  0.78,  0.53,  1.00,  0.69,  0.72,  0.14,  0.67,  0.64,  0.66],
[0.89,  0.81,  0.47,  0.69,  1.00,  0.63,  0.21,  0.77,  0.74,  0.60],
[0.65,  0.61,  0.56,  0.72,  0.63,  1.00,  0.10,  0.62,  0.59,  0.57],
[0.23,  0.29,  0.05,  0.14,  0.21,  0.10,  1.00,  0.25,  0.22,  0.19],
[0.79,  0.75,  0.49,  0.67,  0.77,  0.62,  0.25,  1.00,  0.72,  0.64],
[0.76,  0.73,  0.52,  0.64,  0.74,  0.59,  0.22,  0.72,  1.00,  0.61],
[0.68,  0.70,  0.59,  0.66,  0.60,  0.57,  0.19,  0.64,  0.61,  1.00]
]
```

The program will then generate 100,000 random sample list, all of them follows $N_{10}(0, \mathbf{\Sigma})$.

(Note that $\mathbf{\Sigma}$ is the above defined correlation matrix)

```
n = 100000

mean = [0,0,0,0,0,0,0,0,0,0]
samples = np.random.multivariate_normal(mean, correlation, size = n)
samplesdf = pd.DataFrame(samples)
```

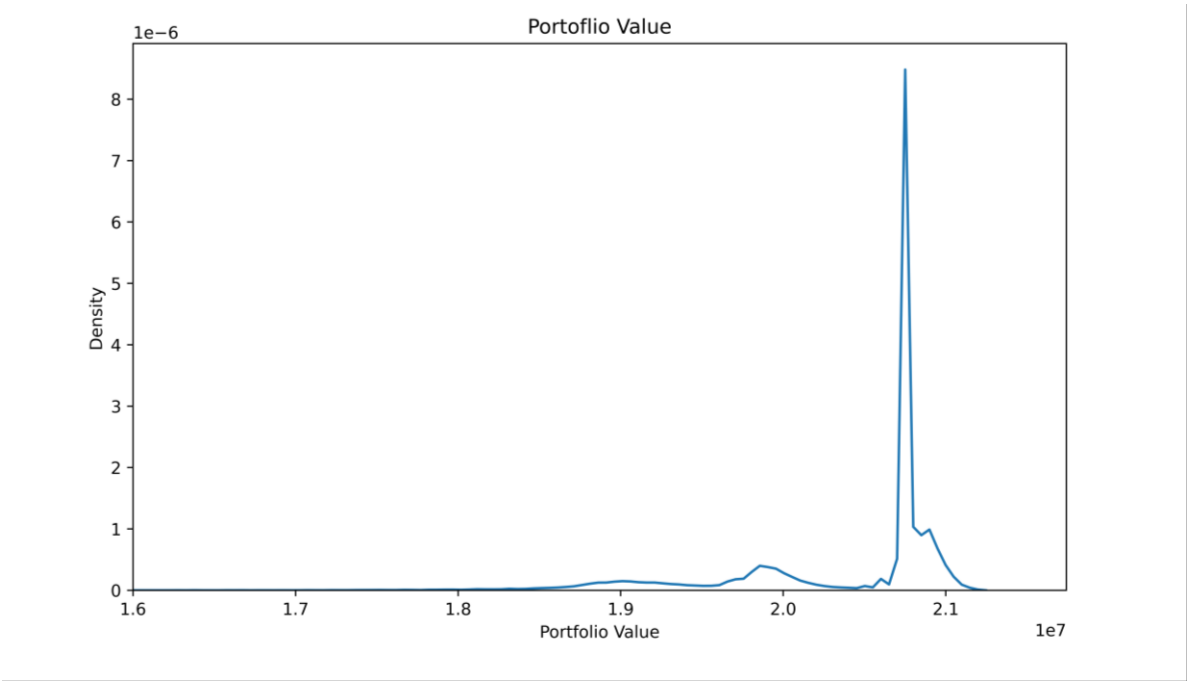
The above code takes in the zero list (vector), the correlation matrix, and will then output 100,000 scenarios of Z-values, using *random.multivariate_normal* in *numpy*.

```
def simulatedprice(z, bondinfo):
    price = []
    for i in z:
        for j in range(len(bondinfo[0])):
            if i >= bondinfo[0][j][2]:
                price.append(bondinfo[0][j-1][1])
                break
        else:
            price.append(bondinfo[2] * np.random.beta(bondinfo[1][0], bondinfo[1][1]))
    return price
```

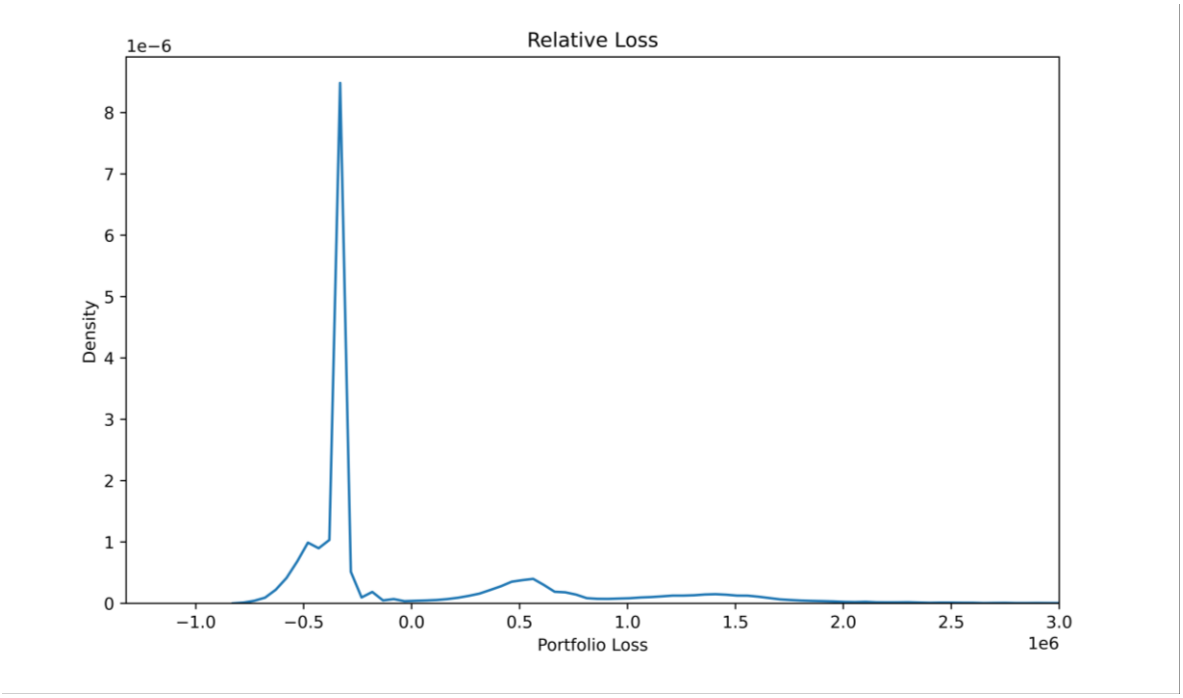
In the *simulatedprice* function, it takes in all the lists of random Z-value and turn it to the corresponding prices in to the list *price*, as per the z-value range, for example *Table 1.6* and *Table 1.7*. If the event happens to have defaulted, it will append the random recovered portion to *price*.

By adding up each bond's simulated value in each different portfolio, we will have 100,000 scenarios of portfolio prices after one year. By plotting out the one-year portfolio price, we have:

Graph 2.3



Graph 2.4



Note: These plots are plotted using seaborn's kdeplot function, which soothes the distribution a bit.

In these 100,000 scenarios:

Portfolio statistics

- Mean Portfolio Value = $20,414,460
- S.D. of Portfolio Value = $690,156
- Lower 5% percentile Portfolio Value = $18,914,182
- Upper 5% percentile Portfolio Value = $20,949,845

Relative Loss statistics

- Upper 5% percentile (95% Relative VaR) = $1,500,277
- Expected Shortfall = $1,984,652

We observed that, in most cases, the bonds retained their original ratings; specifically, in 35,171 scenarios (35.171% of the time), the bonds maintained their initial ratings. Additionally, the bond portfolio loss exhibited a heavily right-skewed, bimodal distribution. The first peak, found at approximately -$300,000, represents a relative gain of $300,000, while the second peak, located near $500,000, signifies a relative loss of $500,000. Interestingly, the actual portfolio value rarely occurs around the mean of the portfolio value.

REFERENCE

- J.P. Morgan & Co. (1997). *CreditMetrics – Technical Decument.*