Computational Finance and Risk Management

Assignment 3
Credit Risk Modeling and Simulation

By

Shabari Girish Kodigepalli Venkata Subramanya
1005627644

1. Introduction

Risk management is a systematic approach for minimizing exposure to risk. Credit Risk is a part of financial risk which is the risk that an obligor may default. The risk of monetary loss due to the default, or a change in the perceived likelihood of default, of a counterparty to a contract. It is classified in Non-Business Risk (Regulatory Capital). It more imperative in the commercial banking, investment banking and trading business.

The objective of this assignment is to model a credit risky portfolio of corporate bonds. The structural model is used to infer a counterparty's (CP) one year in future credit state from a continuous random variable known as creditworthiness index (W). The mathematical form of the creditworthiness index of counterparty j is given as:

$$W_j = \beta_j Y_{j(k)} + \sigma_j Z_j$$

Where,

$\beta_j =$ sensitivity of CP to the credit driver

$Y_{j(k)}$= systemic risk credit driver factor correlated credit drivers (correlated std normal random numbers)

$$\sigma_j = \sqrt{1-\beta_j^2}$$

Z_j= idiosyncratic component independent across CPs j (Std normal random numbers)

Using the above-mentioned model, one-year losses were obtained for each corporate bond for 100 counterparties using Monte Carlo Approximation. The three scenarios simulated are as follows:

1. Monte Carlo approximation 1: 5000 in-sample scenarios (N = 1000 * 5 = 5000 (1000 systemic scenarios and 5 idiosyncratic scenarios for each systemic), non-Normal distribution of losses);
2. Monte Carlo approximation 2: 5000 in-sample scenarios (N = 5000 (5000 systemic scenarios and 1 idiosyncratic scenario for each systemic), non-Normal distribution of losses);
3. True distribution: 100000 out-of-sample scenarios (N = 100000 (100000 systemic scenarios and 1 idiosyncratic scenario for each systemic), non-Normal distribution of losses).

Following errors were evaluated resulting from assumptions and sampling:

1. Model Error: Evaluated using two in-sample non-Normal datasets and comparing with the out-of-sample scenario representing true distribution of losses.
2. Sampling Error: Resulting from assuming the that the CP losses are normally distributed.

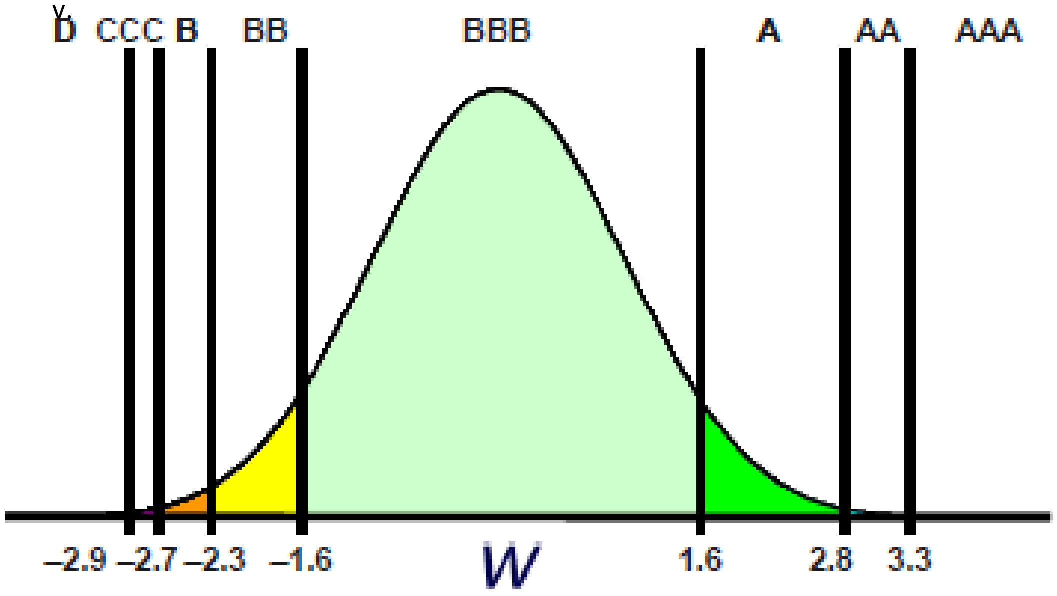
The tail-based risk measure VaR and CVaR were evaluated at quantile levels 99% and 99.9% for the two portfolios:

1. Equal value (dollar amount) is invested in each of 100 bonds
2. One unit invested in each of 100 bonds

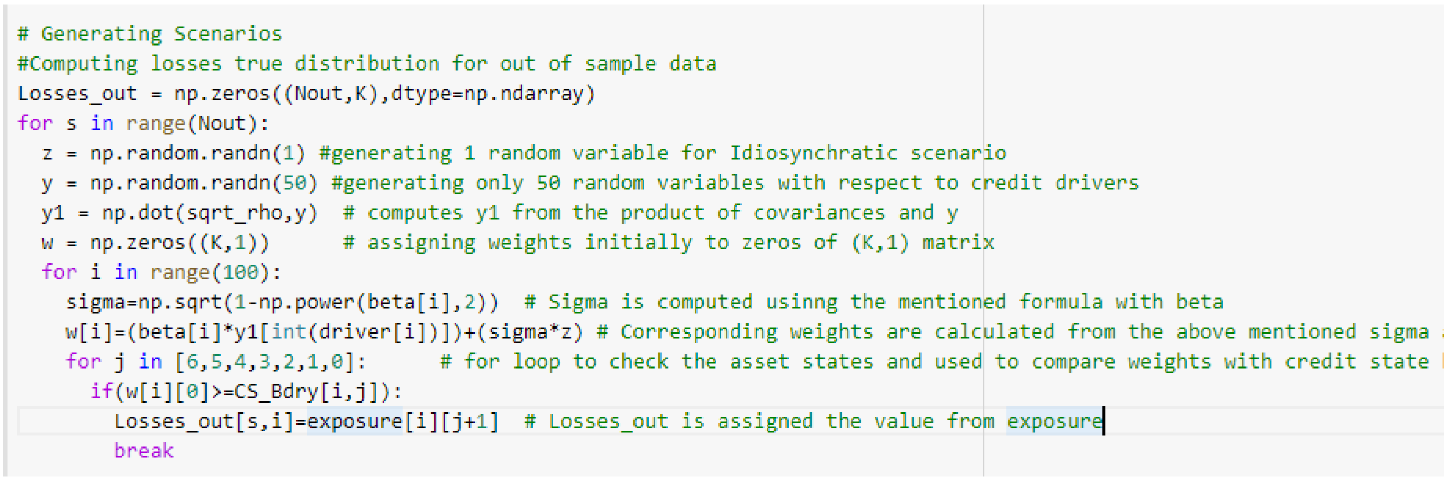
2. Implementation of the structural model for out-of-sample scenario

The out-of-sample scenario has one idiosyncratic scenario for each systemic scenario. The approach used to simulate 100000 out-of-sample scenarios is as follows:

1. Pre-Define size of *Losses_out (Nout, K)* which is 100000 X 100 matrix for (Scenarios, CP)
2. For each scenario (for loop 1 to Nout) compute the following:
   a. Compute Zj idiosyncratic scenario drawn from a standard normal distribution (Mean 0 and Standard Deviation 1)
   b. Compute $Y_{j(k)}$ systemic scenario:
      i. Produce a matrix of 50 X 1 for 50 credit drivers from a standard normal distribution (Mean 0 and Standard Deviation 1)
      ii. Compute the Cholesky Decomposition of the given correlation matrix of the credit drivers
      iii. Find the matrix multiplication of the output from the above two steps to give a systemic scenario for each credit driver(k). Output (50 X 1 matrix)
      iv. The variable driver (100 X 1) identifies the credit driver for a given CP. Thus, obtain the value of $Yj(k)$ (100 X 1) by identifying the driver and assign the value computed in the above step for each CP.
   c. Compute $\sigma_j = \sqrt{1-\beta_j^2}$
   d. Compute creditworthiness index $Wj = \beta jYj(k) + \sigma jZj$ OR w = (beta. *y) +sigma. *z where all the values on RHS are column vectors. The output here is the (100 X 1) i.e. creditworthiness of each CP
   e. Compute Loss for each bond:
      i. The *CS_Bdry* is computed for each CP
      ii. Start with first CP *w (1)* and compare with *CS_Bdry* (1,6).
      iii. If the *w (1)* is greater than the CS_Bdry (1, j) then exposure of (1, j+1) is assigned to the *Losses_out.*
      iv. Repeat this for each CP to find the *Losses_out* for a given scenario and all 100 CP s.



   f. Repeat this for each scenario to result in the matrix of *Losses_out (10000 X 100)*

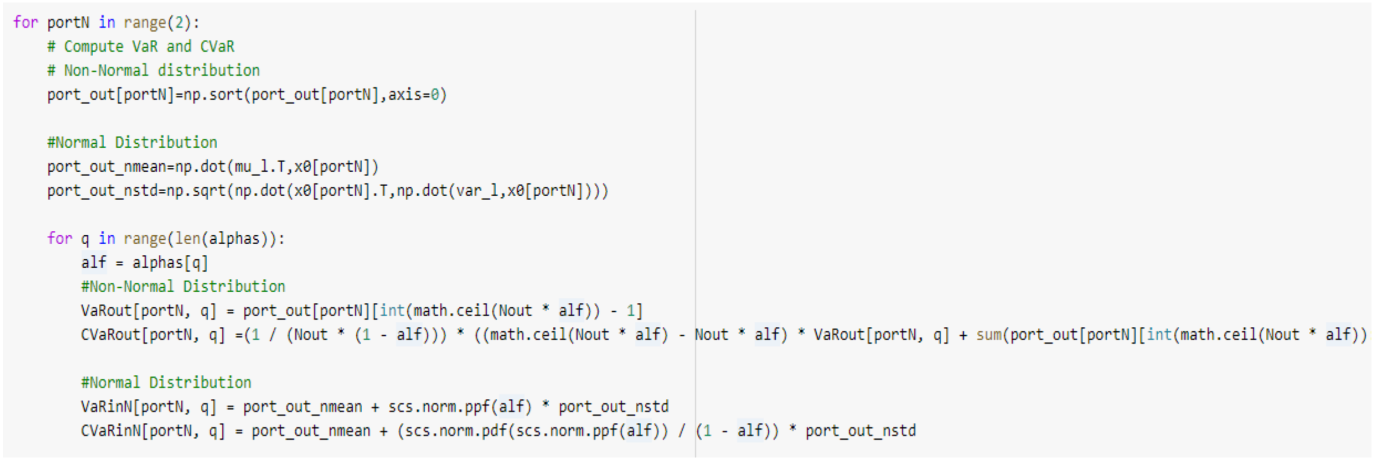


```
# Generating Scenarios
#Computing losses true distribution for out of sample data
Losses_out = np.zeros((Nout,K),dtype=np.ndarray)
for s in range(Nout):
  z = np.random.randn(1) #generating 1 random variable for Idiosynchratic scenario
  y = np.random.randn(50) #generating only 50 random variables with respect to credit drivers
  y1 = np.dot(sqrt_rho,y)  # computes y1 from the product of covariances and y
  w = np.zeros((K,1))      # assigning weights initially to zeros of (K,1) matrix
  for i in range(100):
    sigma=np.sqrt(1-np.power(beta[i],2))  # Sigma is computed usinng the mentioned formula with beta
    w[i]=(beta[i]*y1[int(driver[i])])+(sigma*z) # Corresponding weights are calculated from the above mentioned sigma
    for j in [6,5,4,3,2,1,0]:      # for loop to check the asset states and used to compare weights with credit state
      if(w[i][0]>=CS_Bdry[i,j]):
        Losses_out[s,i]=exposure[i][j+1]  # Losses_out is assigned the value from exposure
        break
```

3. Computing the VaR and CVaR (Non-Normal and Normal) for the out-of-sample scenarios

The steps involved in computing the VaR and CVaR are as follows:

1. Compute the portfolio loss for each portfolio detailed in the section 1, as the product of *Losses_out* and the number of units of a given CP *(X0)* for that portfolio.
2. Sort the losses in ascending order
3. For Normal distribution approximation find the mean and standard deviation of the Portfolio loss.
4. Compute VaR and CVaR for each portfolio and each alpha for normal and non-normal approximation which results in the dimension of the VaR and CVaR as a 2 X 2 matrix for each normal and non-normal approximation.



```
for portN in range(2):
    # Compute VaR and CVaR
    # Non-Normal distribution
    port_out[portN]=np.sort(port_out[portN],axis=0)

    #Normal Distribution
    port_out_nmean=np.dot(mu_l.T,x0[portN])
    port_out_nstd=np.sqrt(np.dot(x0[portN].T,np.dot(var_l,x0[portN])))

    for q in range(len(alphas)):
        alf = alphas[q]
        #Non-Normal Distribution
        VaRout[portN, q] = port_out[portN][int(math.ceil(Nout * alf)) - 1]
        CVaRout[portN, q] =(1 / (Nout * (1 - alf))) * ((math.ceil(Nout * alf) - Nout * alf) * VaRout[portN, q] + sum(port_out[portN][int(math.ceil(Nout * alf))

        #Normal Distribution
        VaRinN[portN, q] = port_out_nmean + scs.norm.ppf(alf) * port_out_nstd
        CVaRinN[portN, q] = port_out_nmean + (scs.norm.pdf(scs.norm.ppf(alf)) / (1 - alf)) * port_out_nstd
```

4. Implementation of the structural model for In-sample scenario (MC1)

To reduce the computational effort the number of samples are reduced from100000 in the out-of-sample scenarios to 5000 in in-sample scenarios (N = 1000 *5 = 5000 (1000 systemic scenarios and 5 idiosyncratic scenarios for each systemic), non-Normal distribution of losses).

Thus, in this case the logic or the approach of obtaining the loss for each bond remains the same as that in out-of-sample scenarios. The only difference being that for one value of systemic scenario there are 5 idiosyncratic scenarios. For a better estimate of the sampling and model error this experiment, or the trails were performed 100 times. This was implemented as follows:

1. Pre-define the size of `Losses_inM1 = [] #Losses in MC1 with 100 trials` to reduce the computational effort.
2. Set a *Scenario Counter* to 0 initially which will be reset to 0 after every trial
3. Obtain systemic scenario as elucidated in section 2
4. For each systemic scenario compute 5 idiosyncratic scenarios
5. For a given idiosyncratic scenario compute the creditworthiness index w and as per the future credit state compute loss for each CP as shown in section 2. Here, the loss of each CP and each trial is stored in Losses_inMC1{trail} (Scenario Counter, CP). Thus, here we have losses as a cell array wherein in each cell represent a trail and has a matrix of 5000 X 100 corresponding to the Scenario X CP.
6. Increase the scenario counter by 1 after generating an idiosyncratic scenario.
7. Reset the scenario counter to 0 after completing a trail.

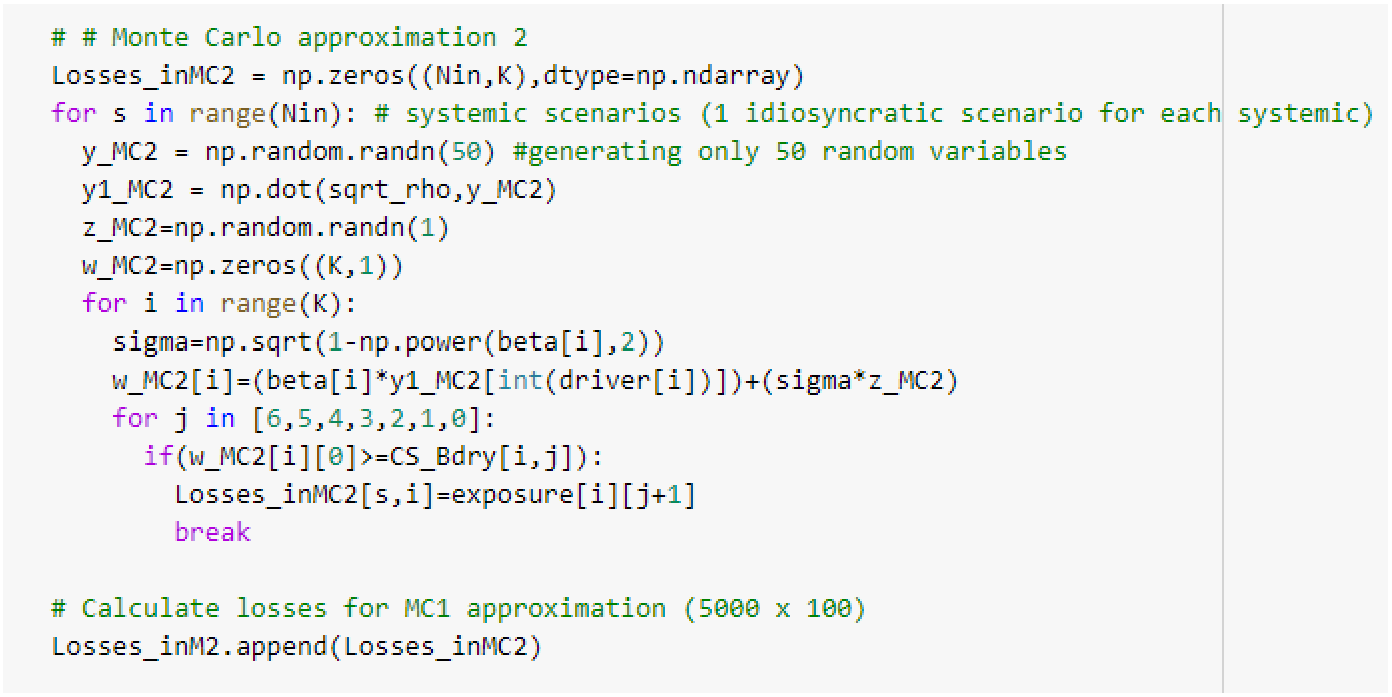
The python code for this process is mentioned below.

```
for tr in range(1,N_trials+1):    # Multiple trials for better approximation fo the model
    # # Monte Carlo approximation 1 (MC1)
    Losses_inMC1 = np.zeros((Nin,K),dtype=np.ndarray)
    sc=0  # scenario counter
    for s in range(1000): # systemic scenarios
        y_MC1 = np.random.randn(50) #generating only 50 random variables
        y1_MC1 = np.dot(sqrt_rho,y_MC1)
        for si in range(Ns): # 5 idiosyncratic scenarios for each  systemic scenario
          z_MC1=np.random.randn(1)
          w_MC1=np.zeros((K,1))
          for i in range(100):
            sigma=np.sqrt(1-np.power(beta[i],2))
            w_MC1[i]=(beta[i]*y1_MC1[int(driver[i])])+(sigma*z_MC1)
            for j in [6,5,4,3,2,1,0]:
              if(w_MC1[i][0]>=CS_Bdry[i,j]):
                Losses_inMC1[sc,i]=exposure[i][j+1]
                break

          sc+=1 # incrementing scenarios
    # Calculate losses for MC1 approximation (5000 x 100)
    Losses_inM1.append(Losses_inMC1)
```

5. **Implementation of the structural model for In-sample scenario (MC2)**

To reduce the computational effort the number of samples are reduced from100000 in the out-of- sample scenarios to 5000 in in-sample scenarios (N = 5000 (1 idiosyncratic scenario for each systemic scenario), non-Normal distribution of losses).

Thus, in this case the logic or the approach of obtaining the loss for each bond remains the same as that in MC1. The only difference being that for one value of systemic scenario there is 1 idiosyncratic scenario. For a better estimate of the sampling and model error this experiment, or the trails were performed 100 times.

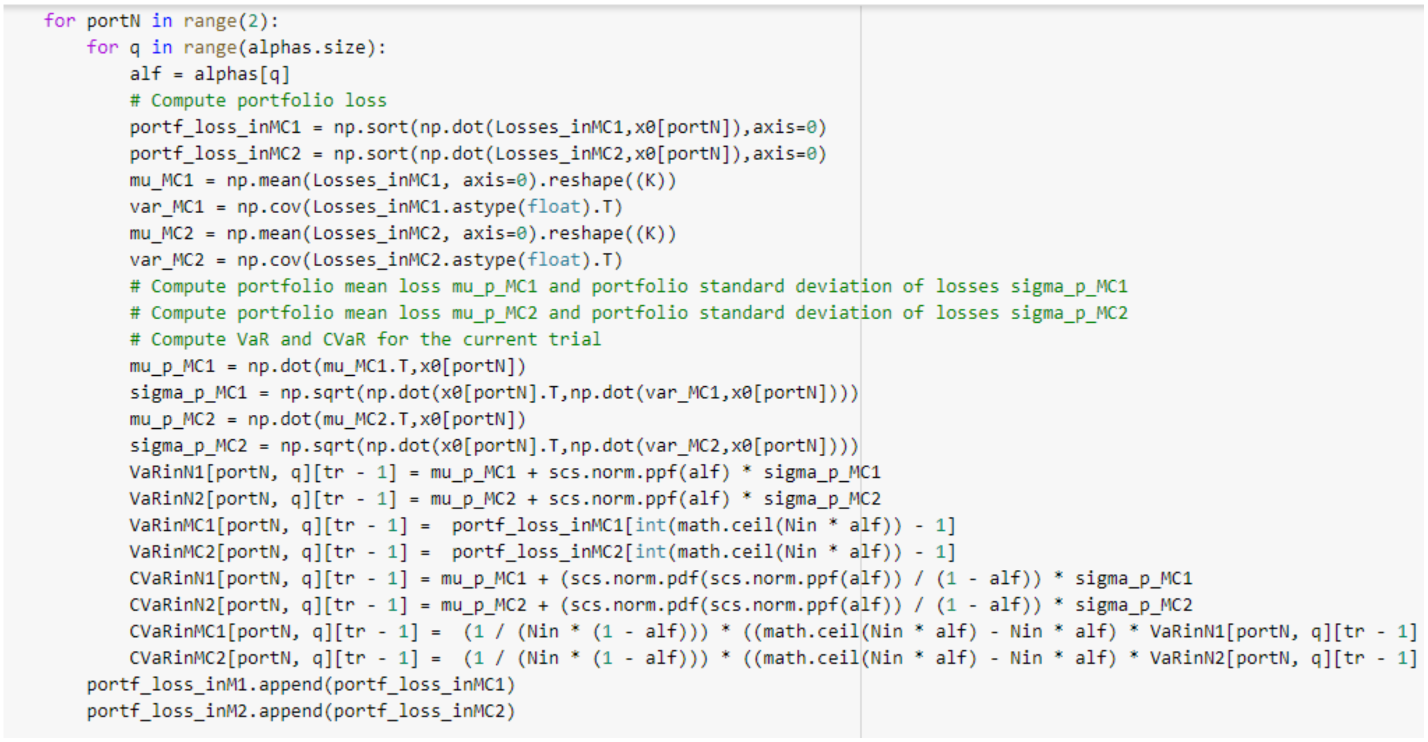


```
# # Monte Carlo approximation 2
Losses_inMC2 = np.zeros((Nin,K),dtype=np.ndarray)
for s in range(Nin): # systemic scenarios (1 idiosyncratic scenario for each systemic)
  y_MC2 = np.random.randn(50) #generating only 50 random variables
  y1_MC2 = np.dot(sqrt_rho,y_MC2)
  z_MC2=np.random.randn(1)
  w_MC2=np.zeros((K,1))
  for i in range(K):
    sigma=np.sqrt(1-np.power(beta[i],2))
    w_MC2[i]=(beta[i]*y1_MC2[int(driver[i])])+(sigma*z_MC2)
    for j in [6,5,4,3,2,1,0]:
      if(w_MC2[i][0]>=CS_Bdry[i,j]):
        Losses_inMC2[s,i]=exposure[i][j+1]
        break

# Calculate losses for MC1 approximation (5000 x 100)
Losses_inM2.append(Losses_inMC2)
```

6. **Computing the VaR and CVaR (Non-Normal and Normal) for the In-sample scenarios (MC1 and MC2)**

The steps involved in computing the VaR and CVaR are as follows:

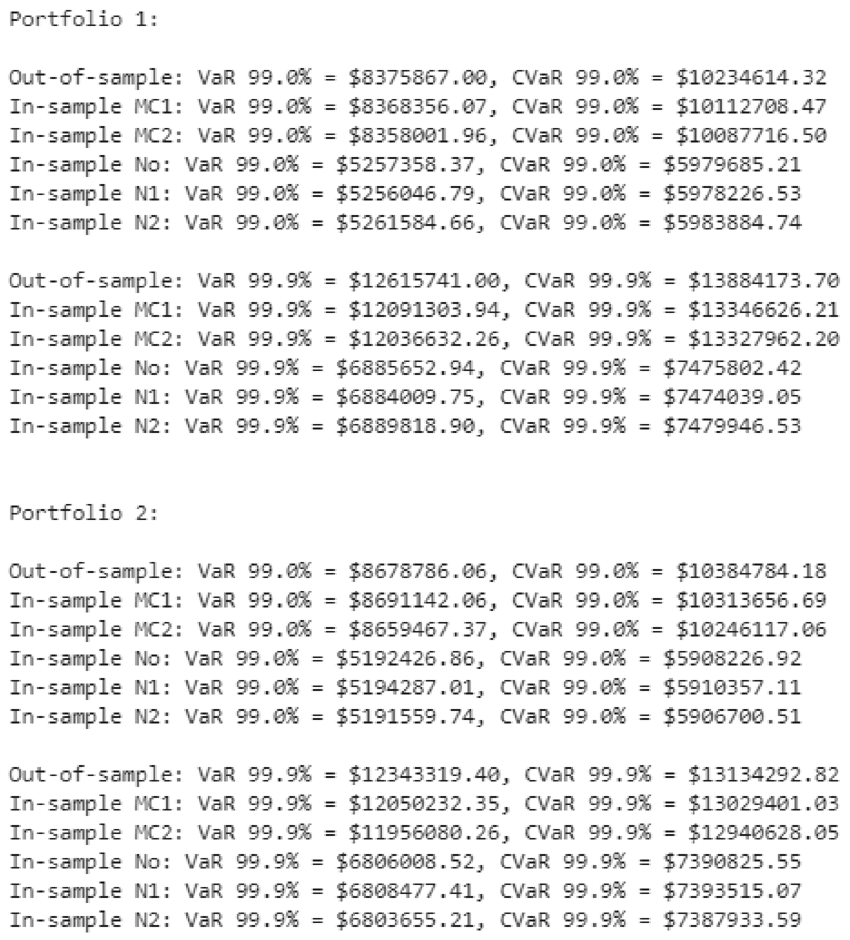
1. Compute the portfolio loss for each portfolio detailed in the section 1, as the product of
2. Losses_inMC1 and the number of units of a given CP (X0) for that portfolio.
3. Sort the losses in ascending order
4. For Normal distribution approximation find the mean and standard deviation of the portfolio loss.
5. Repeat step 1 to 3 for each trail from 1 to 100 and for each portfolio. Thus, the portfolio loss in MC1 and MC2 (portf_loss_inMC1 and portf_loss_inMC1) will be a cell array with 100 cells for 100 trail and each trail consisting of the loss in portfolio 1 and 2 for 5000 scenarios. Thus, the dimensions are portf_loss_inMC1 {trail} (Scenario, Portfolio).
6. Compute VaR and CVaR for each portfolio, each alpha and for each trail for normal and non- normal approximation which results in the cell dimension of the VaR and CVaR as a VaRinMC1{Portfolio, Aplha}(Trail). Thus, each cell holds 100 values of VaR in MC1 for a given portfolio and alpha. This same applies for MC2 and CVaR as well. This is illustrated below:



```
for portN in range(2):
    for q in range(alphas.size):
        alf = alphas[q]
        # Compute portfolio loss
        portf_loss_inMC1 = np.sort(np.dot(Losses_inMC1,x0[portN]),axis=0)
        portf_loss_inMC2 = np.sort(np.dot(Losses_inMC2,x0[portN]),axis=0)
        mu_MC1 = np.mean(Losses_inMC1, axis=0).reshape((K))
        var_MC1 = np.cov(Losses_inMC1.astype(float).T)
        mu_MC2 = np.mean(Losses_inMC2, axis=0).reshape((K))
        var_MC2 = np.cov(Losses_inMC2.astype(float).T)
        # Compute portfolio mean loss mu_p_MC1 and portfolio standard deviation of losses sigma_p_MC1
        # Compute portfolio mean loss mu_p_MC2 and portfolio standard deviation of losses sigma_p_MC2
        # Compute VaR and CVaR for the current trial
        mu_p_MC1 = np.dot(mu_MC1.T,x0[portN])
        sigma_p_MC1 = np.sqrt(np.dot(x0[portN].T,np.dot(var_MC1,x0[portN])))
        mu_p_MC2 = np.dot(mu_MC2.T,x0[portN])
        sigma_p_MC2 = np.sqrt(np.dot(x0[portN].T,np.dot(var_MC2,x0[portN])))
        VaRinN1[portN, q][tr - 1] = mu_p_MC1 + scs.norm.ppf(alf) * sigma_p_MC1
        VaRinN2[portN, q][tr - 1] = mu_p_MC2 + scs.norm.ppf(alf) * sigma_p_MC2
        VaRinMC1[portN, q][tr - 1] =  portf_loss_inMC1[int(math.ceil(Nin * alf)) - 1]
        VaRinMC2[portN, q][tr - 1] =  portf_loss_inMC2[int(math.ceil(Nin * alf)) - 1]
        CVaRinN1[portN, q][tr - 1] = mu_p_MC1 + (scs.norm.pdf(scs.norm.ppf(alf)) / (1 - alf)) * sigma_p_MC1
        CVaRinN2[portN, q][tr - 1] = mu_p_MC2 + (scs.norm.pdf(scs.norm.ppf(alf)) / (1 - alf)) * sigma_p_MC2
        CVaRinMC1[portN, q][tr - 1] =  (1 / (Nin * (1 - alf))) * ((math.ceil(Nin * alf) - Nin * alf) * VaRinN1[portN, q][tr - 1]
        CVaRinMC2[portN, q][tr - 1] =  (1 / (Nin * (1 - alf))) * ((math.ceil(Nin * alf) - Nin * alf) * VaRinN2[portN, q][tr - 1]
    portf_loss_inM1.append(portf_loss_inMC1)
    portf_loss_inM2.append(portf_loss_inMC2)
```

7. Analysis of Results

7.1 Comparing each VaR and CVaR values for different sampling scenarios and alphas

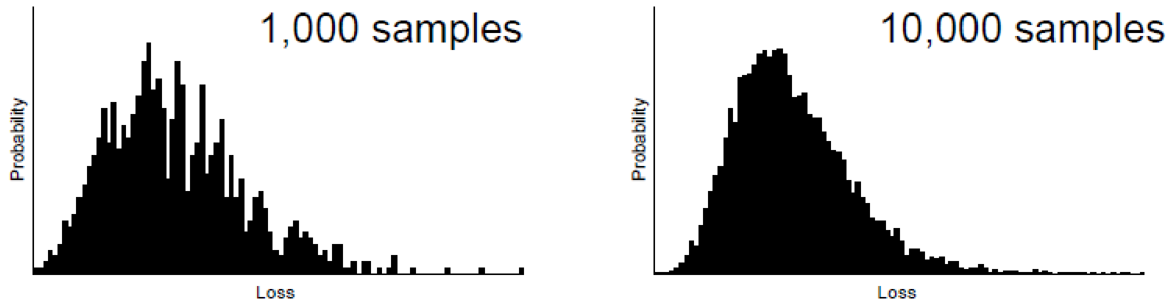


```
Portfolio 1:

Out-of-sample: VaR 99.0% = $8375867.00, CVaR 99.0% = $10234614.32
In-sample MC1: VaR 99.0% = $8368356.07, CVaR 99.0% = $10112708.47
In-sample MC2: VaR 99.0% = $8358001.96, CVaR 99.0% = $10087716.50
In-sample No: VaR 99.0% = $5257358.37, CVaR 99.0% = $5979685.21
In-sample N1: VaR 99.0% = $5256046.79, CVaR 99.0% = $5978226.53
In-sample N2: VaR 99.0% = $5261584.66, CVaR 99.0% = $5983884.74

Out-of-sample: VaR 99.9% = $12615741.00, CVaR 99.9% = $13884173.70
In-sample MC1: VaR 99.9% = $12091303.94, CVaR 99.9% = $13346626.21
In-sample MC2: VaR 99.9% = $12036632.26, CVaR 99.9% = $13327962.20
In-sample No: VaR 99.9% = $6885652.94, CVaR 99.9% = $7475802.42
In-sample N1: VaR 99.9% = $6884009.75, CVaR 99.9% = $7474039.05
In-sample N2: VaR 99.9% = $6889818.90, CVaR 99.9% = $7479946.53


Portfolio 2:

Out-of-sample: VaR 99.0% = $8678786.06, CVaR 99.0% = $10384784.18
In-sample MC1: VaR 99.0% = $8691142.06, CVaR 99.0% = $10313656.69
In-sample MC2: VaR 99.0% = $8659467.37, CVaR 99.0% = $10246117.06
In-sample No: VaR 99.0% = $5192426.86, CVaR 99.0% = $5908226.92
In-sample N1: VaR 99.0% = $5194287.01, CVaR 99.0% = $5910357.11
In-sample N2: VaR 99.0% = $5191559.74, CVaR 99.0% = $5906700.51

Out-of-sample: VaR 99.9% = $12343319.40, CVaR 99.9% = $13134292.82
In-sample MC1: VaR 99.9% = $12050232.35, CVaR 99.9% = $13029401.03
In-sample MC2: VaR 99.9% = $11956080.26, CVaR 99.9% = $12940628.05
In-sample No: VaR 99.9% = $6806008.52, CVaR 99.9% = $7390825.55
In-sample N1: VaR 99.9% = $6808477.41, CVaR 99.9% = $7393515.07
In-sample N2: VaR 99.9% = $6803655.21, CVaR 99.9% = $7387933.59
```

7.2 Comparing each in-sample VaR and CVaR to out-of-sample VaR and CVaR (Analyzing Sampling Error)

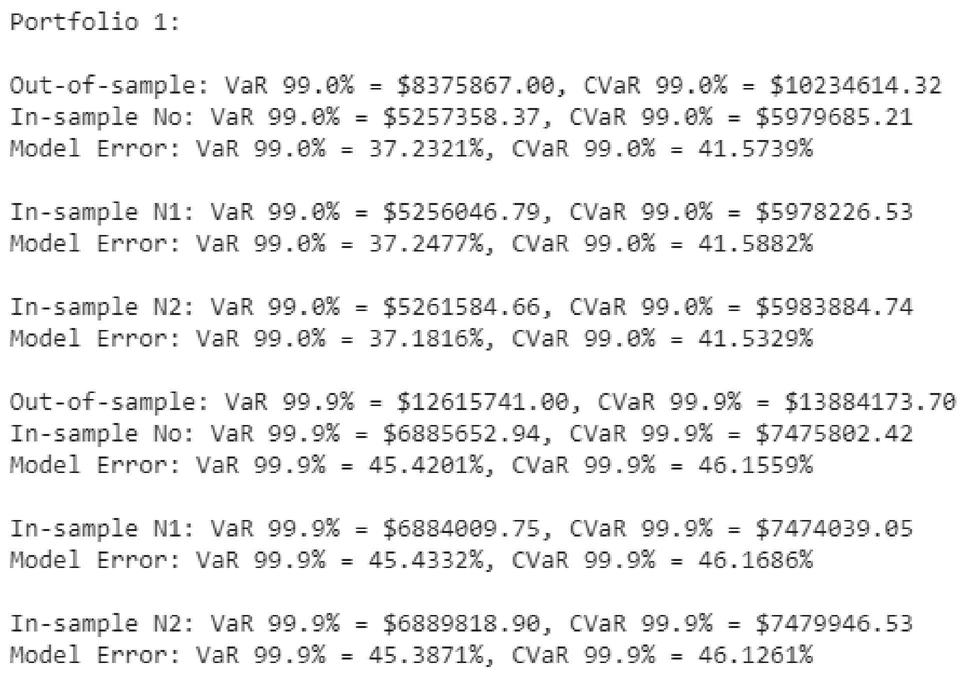
```
Portfolio 1:

Out-of-sample: VaR 99.0% = $8375867.00, CVaR 99.0% = $10234614.32
In-sample MC1: VaR 99.0% = $8368356.07, CVaR 99.0% = $10112708.47
In-sample MC2: VaR 99.0% = $8358001.96, CVaR 99.0% = $10087716.50
Sampling Error MC1: VaR 99.0% = 0.0897%, CVaR 99.0% = 1.1911%
Sampling Error MC2: VaR 99.0% = 0.2133%, CVaR 99.0% = 1.4353%

Out-of-sample: VaR 99.9% = $12615741.00, CVaR 99.9% = $13884173.70
In-sample MC1: VaR 99.9% = $12091303.94, CVaR 99.9% = $13346626.21
In-sample MC2: VaR 99.9% = $12036632.26, CVaR 99.9% = $13327962.20
Sampling Error MC1: VaR 99.9% = 4.1570%, CVaR 99.9% = 3.8717%
Sampling Error MC2: VaR 99.9% = 4.5904%, CVaR 99.9% = 4.0061%


Portfolio 2:

Out-of-sample: VaR 99.0% = $8678786.06, CVaR 99.0% = $10384784.18
In-sample MC1: VaR 99.0% = $8691142.06, CVaR 99.0% = $10313656.69
In-sample MC2: VaR 99.0% = $8659467.37, CVaR 99.0% = $10246117.06
Sampling Error MC1: VaR 99.0% = 0.1424%, CVaR 99.0% = 0.6849%
Sampling Error MC2: VaR 99.0% = 0.2226%, CVaR 99.0% = 1.3353%

Out-of-sample: VaR 99.9% = $12343319.40, CVaR 99.9% = $13134292.82
In-sample MC1: VaR 99.9% = $12050232.35, CVaR 99.9% = $13029401.03
In-sample MC2: VaR 99.9% = $11956080.26, CVaR 99.9% = $12940628.05
Sampling Error MC1: VaR 99.9% = 2.3745%, CVaR 99.9% = 0.7986%
Sampling Error MC2: VaR 99.9% = 3.1372%, CVaR 99.9% = 1.4745%
```

1. For Portfolio 1 as well as Portfolio 2 it can be clearly observed that the In-sample estimates are close to that of the out-of-sample estimates for alpha of 99% as compared to that of 99.9%. This is because the estimates lie at the extreme tail end of the distribution for higher values of alpha which is not refine unless large number of samples are taken. In other words, sampling error will be more for higher value of alpha. This is illustrated below

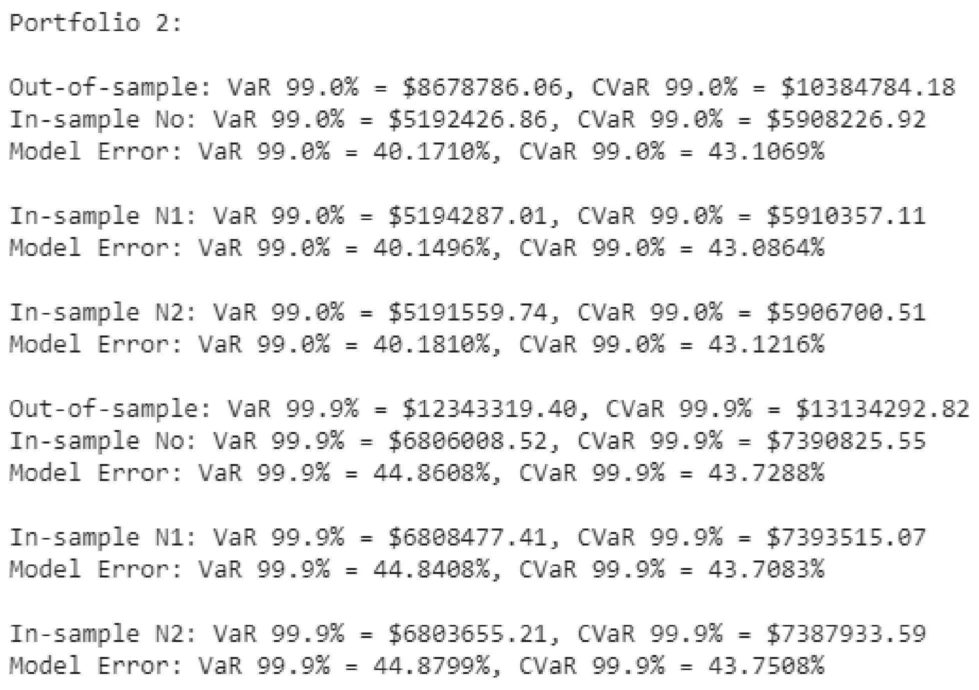


2. Moreover, comparing MC1 and MC2 it can be observed that MC2 gives a closer approximating to the out-of-sample scenario as compared to that of the MC1. This is because in MC2 for each systemic scenario there was one idiosyncratic scenario. On the other hand, MC1 had only one systemic scenario for 5 idiosyncratic scenarios. In other words, sampling error was more in the case of the MC1 as compared to that of MC2. Thus, in MC1 computational effort is reduced, but the compromise of the accuracy.

7.3 Comparing Normal Approximations VaR and CVaR to out-of-sample VaR and CVaR (Analyzing Model Error)



```
Portfolio 1:

Out-of-sample: VaR 99.0% = $8375867.00, CVaR 99.0% = $10234614.32
In-sample No: VaR 99.0% = $5257358.37, CVaR 99.0% = $5979685.21
Model Error: VaR 99.0% = 37.2321%, CVaR 99.0% = 41.5739%

In-sample N1: VaR 99.0% = $5256046.79, CVaR 99.0% = $5978226.53
Model Error: VaR 99.0% = 37.2477%, CVaR 99.0% = 41.5882%

In-sample N2: VaR 99.0% = $5261584.66, CVaR 99.0% = $5983884.74
Model Error: VaR 99.0% = 37.1816%, CVaR 99.0% = 41.5329%

Out-of-sample: VaR 99.9% = $12615741.00, CVaR 99.9% = $13884173.70
In-sample No: VaR 99.9% = $6885652.94, CVaR 99.9% = $7475802.42
Model Error: VaR 99.9% = 45.4201%, CVaR 99.9% = 46.1559%

In-sample N1: VaR 99.9% = $6884009.75, CVaR 99.9% = $7474039.05
Model Error: VaR 99.9% = 45.4332%, CVaR 99.9% = 46.1686%

In-sample N2: VaR 99.9% = $6889818.90, CVaR 99.9% = $7479946.53
Model Error: VaR 99.9% = 45.3871%, CVaR 99.9% = 46.1261%
```



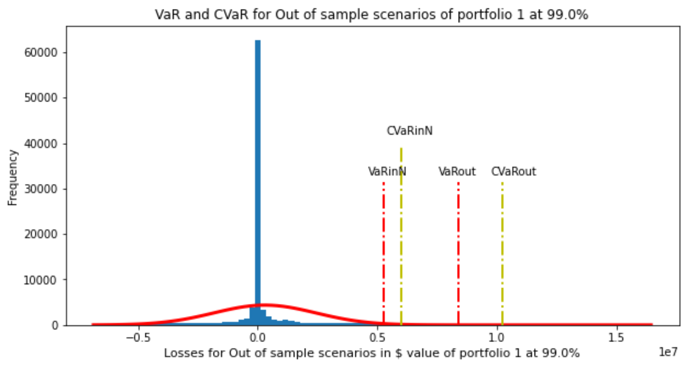
```
Portfolio 2:

Out-of-sample: VaR 99.0% = $8678786.06, CVaR 99.0% = $10384784.18
In-sample No: VaR 99.0% = $5192426.86, CVaR 99.0% = $5908226.92
Model Error: VaR 99.0% = 40.1710%, CVaR 99.0% = 43.1069%

In-sample N1: VaR 99.0% = $5194287.01, CVaR 99.0% = $5910357.11
Model Error: VaR 99.0% = 40.1496%, CVaR 99.0% = 43.0864%

In-sample N2: VaR 99.0% = $5191559.74, CVaR 99.0% = $5906700.51
Model Error: VaR 99.0% = 40.1810%, CVaR 99.0% = 43.1216%

Out-of-sample: VaR 99.9% = $12343319.40, CVaR 99.9% = $13134292.82
In-sample No: VaR 99.9% = $6806008.52, CVaR 99.9% = $7390825.55
Model Error: VaR 99.9% = 44.8608%, CVaR 99.9% = 43.7288%

In-sample N1: VaR 99.9% = $6808477.41, CVaR 99.9% = $7393515.07
Model Error: VaR 99.9% = 44.8408%, CVaR 99.9% = 43.7083%

In-sample N2: VaR 99.9% = $6803655.21, CVaR 99.9% = $7387933.59
Model Error: VaR 99.9% = 44.8799%, CVaR 99.9% = 43.7508%
```

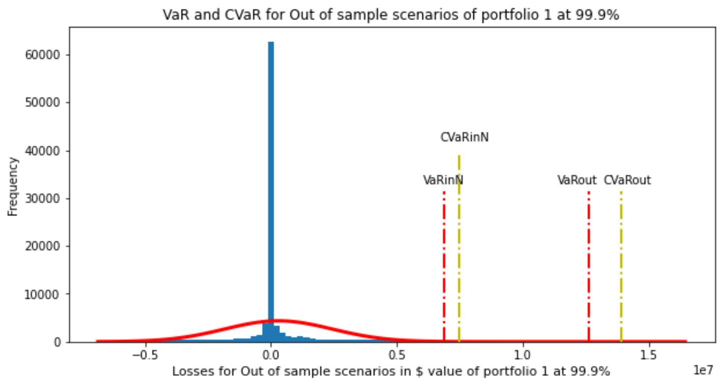
1. For Portfolio 1 as well as Portfolio 2 it can be clearly observed that the normal approximation estimates are close to that of the out-of-sample estimates for alpha of 99% as compared to that of 99.9%. In other words, model error is higher for higher values of alpha.
2. Also, it can be observed that the Normal approximation underestimates risk as compared to the out-of-sample scenario.

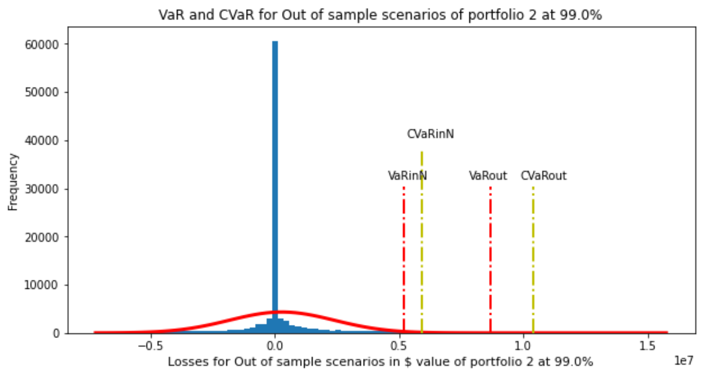
3. Comparing N1 and N2 with the out-of-sample scenario it can be concluded that the N2 provides a closer estimate to the out-of-sample scenario. This is because in MC2 for each systemic scenario there was one idiosyncratic scenario. On the other hand, MC1 had only one systemic scenario for 5 idiosyncratic scenarios. In other words, model error was more in the case of the MC1 as compared to that of MC2.

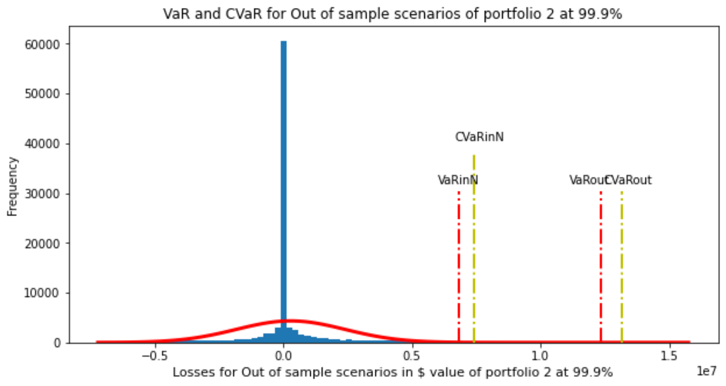
8. <u>Plots</u>

8.1 VaR and CVaR for Out sample Scenario

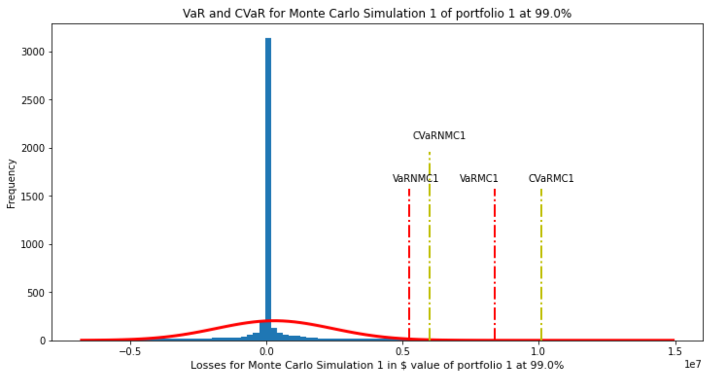


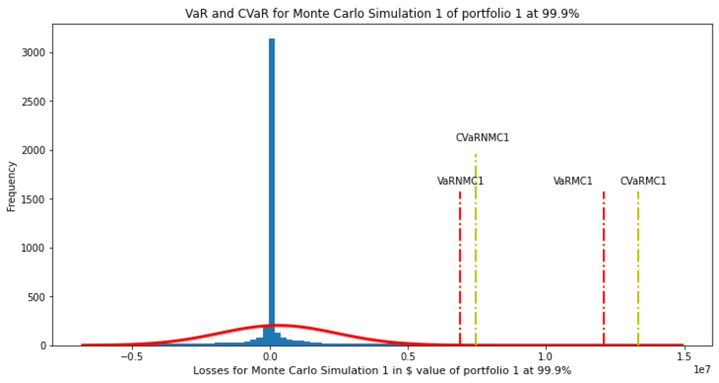


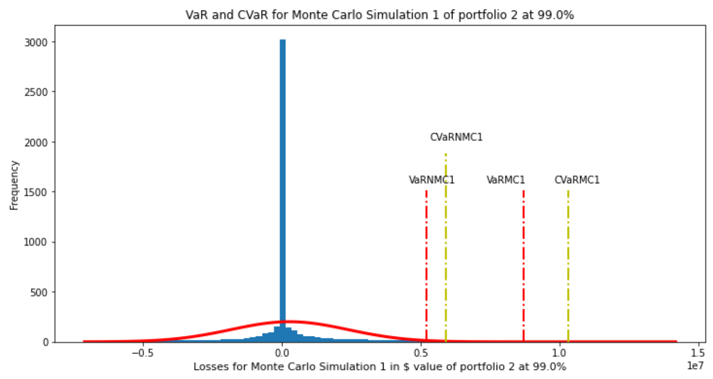


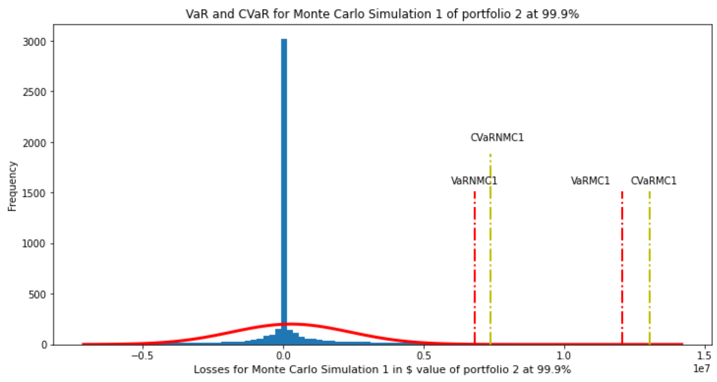


8.2VaR and CVaR for In sample Scenario (Monte Carlo Simulation 1)

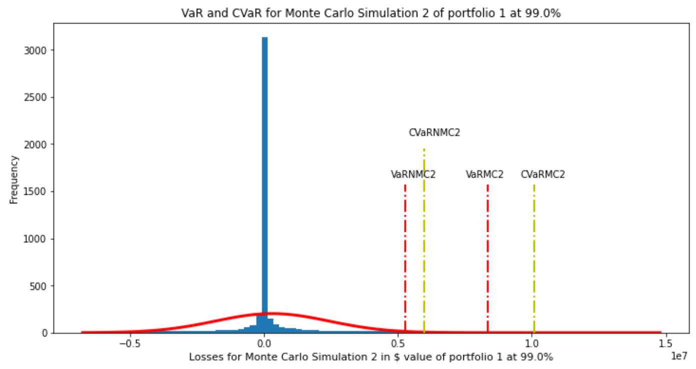


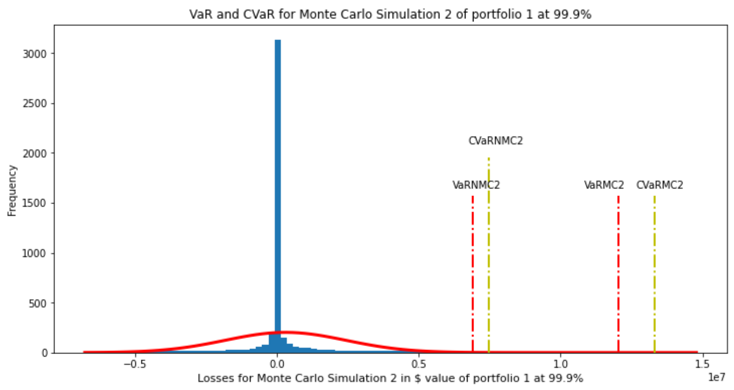


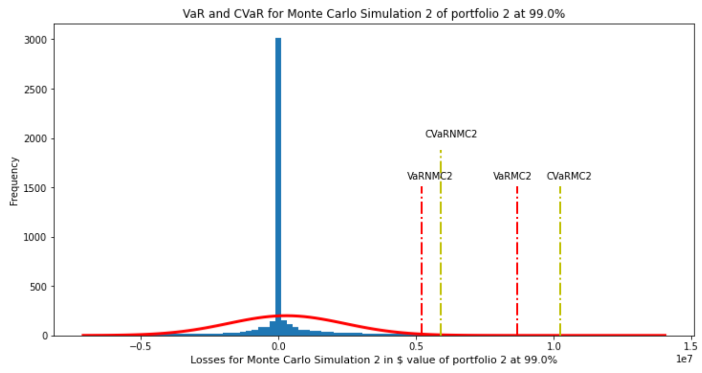


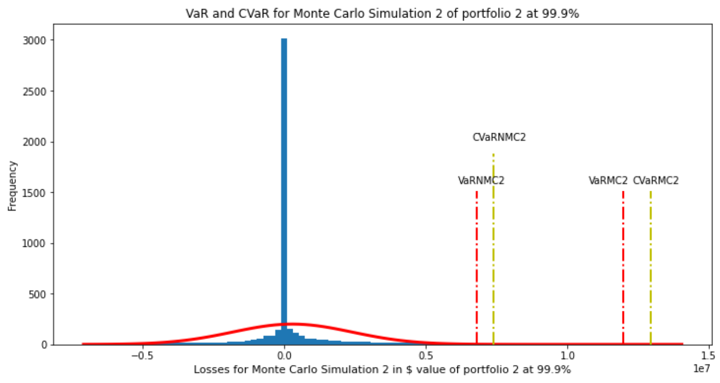


8.3VaR and CVaR for In sample Scenario (Monte Carlo Simulation 2)





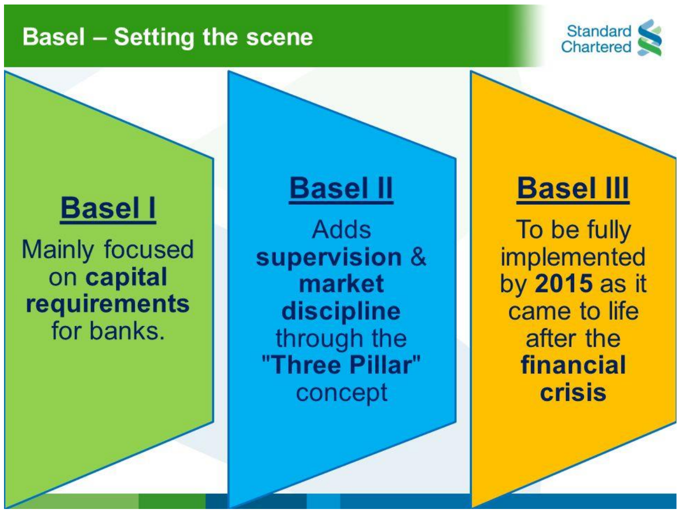




9. Discussion

9.1 Impact of sampling error on the bank capital requirement

Capital Requirement or regulatory capital is the amount of capital a bank or other financial institution must hold as per the regulations to ensure that they do not take excess leverage and become insolvent. Reporting in-sample estimates of VaR and CVaR to the decision makers in the bank will result in the underestimating the value of risk measure especially for higher values of alpha as observed in section 7. This will result in underestimating the bank regulatory capital. This will lead the financial institution taking more risk than allowed by the regulation to increase the return on equity. Also, it was seen from the estimates of the standard deviation of the trail conducted that with the increase in alpha the in- sample scenarios become less certain on the VaR and CVaR due to the increased standard deviation.



Basel – Setting the scene

Basel I
Mainly focused on **capital requirements** for banks.

Basel II
Adds **supervision** & **market discipline** through the **"Three Pillar"** concept

Basel III
To be fully implemented by **2015** as it came to life after the **financial crisis**

Standard Chartered

VaR and CVaR estimation is very crucial for risk management. In no case these estimates should be performed with negligence. If we see the results, we can clearly say that True sample estimates are high (15%) in comparison to 'In sample estimates' if we consider the VaR 99.9% as estimate. So, the capital requirements to set aside increases, this may lead to decrease in Banks profit. But if a bank depends on in sample estimates for its risk management then there is a chance of bankruptcy. This should be avoided so It is highly recommended to stick to Basel Accords.

9.2 Techniques for Minimizing the impact of model and sampling error:

1. Increase number of scenarios to reduce the sampling error and draw a balance between the number of scenarios and computational cost.
2. Avoid using normality assumption of the distribution of the losses to avoid model error
3. Consider accounting for the market as well apart from the credit risk in the model
4. The portfolio loss distribution conditional on ym is the convolution of conditional counterparty loss distributions. Here for simplicity of approximation Monte Carlo and Normal approximations were used.