MSA400: Technical Project 2

Calvin Smith

Chalmers University of Technology/University of Gothenburg

August 25, 2022

Contents

1 Task 1 - Value at Risk

We are considering a static credit portfolio with 35 obligors where the size of each loan is 3 million SEK, the default losses are expected to be 60% and the individual default probability for each obligor is 4%. In this task, we have chosen two homogenous mixed binomial models, and we will derive formulas for obtaining the VaR using the Large Portfolio Approximation (LPA) approach.

The following theorem is central to our calculations:

Theorem 1 *(Large Portfolio Approximation for mixed binomial models). For large portfolios in a mixed binomial model, the distribution of the fractional number of defaults* $\frac{N_m}{m}$ *in the portfolio converges to the distribution of the random variable* $p(Z)$ *as* $m \to \infty$*, that is for any* $x \in [0,1]$ *we have*

$$\mathbb{P}[\frac{N_m}{m} \leq x] \to \mathbb{P}[p(Z) \leq x], \text{ when } m \to \infty. \tag{1}$$

The distribution $\mathbb{P}(p(Z) \leq x)$ *is called the Large Portfolio Approximation (LPA) to the distribution of* $\frac{N_m}{m}$*.*

From this, the following formula for LPA approximation of VaR will be used:

$$VaR_\alpha(L) \approx l \cdot m \cdot F^{-1}(\alpha) \tag{2}$$

where l is the default loss, m is the number of obligors and $F^{-1}(x)$ is the inverse of $F(x) = \mathbb{P}(p(Z) \leq x)$, where $p(Z)$ is the mixing distribution and Z is a random variable.

1.1 Beta Binomial

Letting $p(Z) = Z$ where Z is a beta distribution, $Z \sim Beta(a, b)$ we have that

$$F(x) = \mathbb{P}(Z \leq x) = \frac{1}{B(a,b)} \int_0^x z^{a-1}(1-z)^{b-1} dz. \tag{3}$$

where,

$$B(a,b) = \frac{\Gamma(a)\Gamma(b)}{\Gamma(a+b)}.$$

Here, $\Gamma(y)$ is the gamma function defined as $\Gamma(y) = \int_0^\infty t^{y-1}e^{-t}dt$.

$F(x)$ is by definition the CDF of a beta distribution, since for a continous random variable X its Cumulative Distribution Function $F_X(x)$ can be expressed as

$$F_X(x) = \int_{-\infty}^x f_X(t)dt. \tag{4}$$

Thus, to obtain the value $F^{-1}(\alpha)$ for a given α we do not need to calculate an explicit formula, instead we can simply choose the x that satisfies $F(x) = \alpha$.

1.2 Logit-normal distribution

Letting the mixing distribution $p(Z)$ be a logit normal distribution, we have that

$$p(Z) = \frac{1}{1 + \exp(-(\mu + \sigma Z))} \tag{5}$$

where, $Z \sim N(0, 1)$. Further, the inverse of the logit normal distribution is given by,

$$p^{-1}(x) = \frac{1}{\sigma}(\ln(\frac{x}{1-x}) - \mu), \tag{6}$$

and the mixing distribution $F(x)$ for the logit normal is then

$$F(x) = \mathbb{P}[p(Z) \leq x] = \mathbb{P}[Z \leq p^{-1}(x)] = \Phi(p^{-1}(x)) \tag{7}$$

, where $\Phi(.)$ is the cdf of the standard normal distribution.

Further, we want to find the value of x such that $\Phi(p^{-1}(x)) = \alpha$. Using the definition of quantiles we get that

$$p^{-1}(x) = \Phi^{-1}(\alpha) \tag{8}$$

where $\Phi^{-1}(.)$ is the quantile function of the normal distribution. Thus, using (8) and solving for x we get that

$$x = \frac{\exp(\Phi^{-1}(\alpha)\sigma + \mu)}{1 + \exp(\Phi^{-1}(\alpha)\sigma + \mu)}. \tag{9}$$

To get the estimated $VaR_\alpha(L)$ we multiply (9) with the absolute loss for each obligor and the number of obligors.

2 Task 2 - Choosing two sets of parameters

The second task consists of choosing two sets of parameter for each of the two models from Task 1. The first set should be chosen to provide a high default dependance and the second one low default dependance.

The default dependence of the obligors in a homogenous credit porfolio is measured by the correlation $Corr(X_i, X_j)$ between two pairs i, j in the portfolio. Since the portfolio is assumed to be homogenous the correlation between X_i and X_j is the same for all pairs as long as $i \neq j$. Instead of $Corr(X_i, X_j)$, we will write ρ_X to denote the correlation. Further, the correlation ρ_X in a mixed binomial model is given by the formula:

$$\rho_X = \frac{E[p(Z)^2] - \bar{p}^2}{\bar{p}(1 - \bar{p})} \tag{10}$$

where $\bar{p} = E[p(Z)] = \mathbb{P}[X_i = 1]$, is the default probability of each obligor.

2.1 Beta-Binomial

In the mixed binomial beta distribution we have that $p(Z) = Z$ where $Z \sim Beta(a, b)$. The expected value of a beta distributed random variable is

$$E[Z] = \frac{a}{a+b} \tag{11}$$

and the second moment is given by

$$E[Z^2] = \frac{(a+1)a}{(a+b+1)(a+b)}. \tag{12}$$

Since, the individual default probability for each obligor is 4% we have that

$$E[Z] = \frac{a}{a+b} = 0.04 = \bar{p}, \tag{13}$$

and thus we can rewrite the second moment as

$$E[Z^2] = \frac{(a+1)\bar{p}}{a+b+1}, \tag{14}$$

so inserting (8) and (9) in the formula for ρ_X we get

$$\rho_X = \frac{\frac{(a+1)\bar{p}}{a+b+1} - \bar{p}^2}{\bar{p}(1-\bar{p})} \tag{15}$$

which after some simplifications results in

$$\rho_X = \frac{1}{a+b+1}. \tag{16}$$

Thus, we can obtain a specified correlation ρ_X by choosing a and b that solves the system of equations:

$$\frac{a}{a+b} = 0.04 \tag{17}$$

$$\frac{1}{a+b+1} = \rho_X \tag{18}$$

From this, we can choose two sets of parameters $\{a, b\}$ for the Beta distribution, which results in two different models with the same mean $a/(a+b) = 0.04$, but with different correlations. The first set is $\{a = 0.01, b = 0.24\}$ which leads to a high correlation of $\rho_X = 0.8$. For the second set we chose $\{a = 0.36, b = 8.36\}$ which leads to a correlation $\rho_X = 0.1$.

	a	b	μ	ρ_X
Beta Model 1	0.36	8.64	0.04	0.10
Beta Model 2	0.01	0.24	0.04	0.80

Table 1: Two mixed beta binomial models with the same mean, but different correlations.

2.2 Logit-normal distribution

In the logit-normal case, it is difficult to find closed form expressions of the expectation $E[p(Z)]$ and second moment $E[p(Z)^2]$, so we will use another approach. Using the fact that we know that $E[p(Z)] = 0.04$ and the formula

$$\rho_X = \frac{E[p(Z)^2] - \bar{p}^2}{\bar{p}(1-\bar{p})}. \tag{19}$$

Note that the numerator in the above formula is the variance of $p(Z)$ i.e $Var(p(Z))$. So setting the correlation ρ_X to some specified value we can obtain the variance of our logit normal distribution by

$$Var(p(Z)) = \bar{p}(1-\bar{p})\rho_X. \tag{20}$$

Thus, for a specified correlation, we know the mean and variance of $p(Z)$ and from this we can find the μ and σ that produces the distribution. This has been done numerically in R by doing a grid search over different values of μ and σ and then choosing the pair μ and σ that produces a mean of approximately 0.04 and a variance of approximately $\bar{p}(1-\bar{p})\rho_X$. As in the beta binomial case, we have choosen a high correlation of 0.8 and a low correlation of 0.1. However, since the implementation has been done numerically in R the resulting correlations are not exactly equal to 0.8 and 0.1 respectively. The results are displayed in Table 2.

	mu	sigma	mean	var	ρ_X
Logit 1	-17.58	9.90	0.040	0.03	0.78
Logit 2	-3.94	1.31	0.039	0.0033	0.09

Table 2: Two logit normal distributions with the same mean but different correlations.

3 Task 3 - Comparing models

The value at risk estimates have been calculated using the formula (2) from Task 1. Further, the expected shortfall has been calculated using the following formula:

$$ES_\alpha(L) \approx \frac{l*m}{1-\alpha}\int_\alpha^1 F^{-1}(u)du \tag{21}$$

where $F^{-1}(x)$ is the quantile function of our loss distribution.

3.1 Beta-Binomial

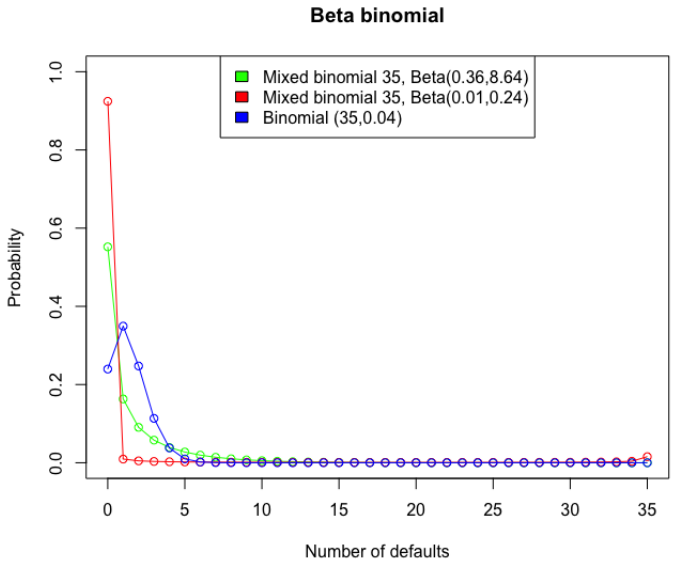

Figure 1: Beta binomial probability density function and a binomial function as reference.

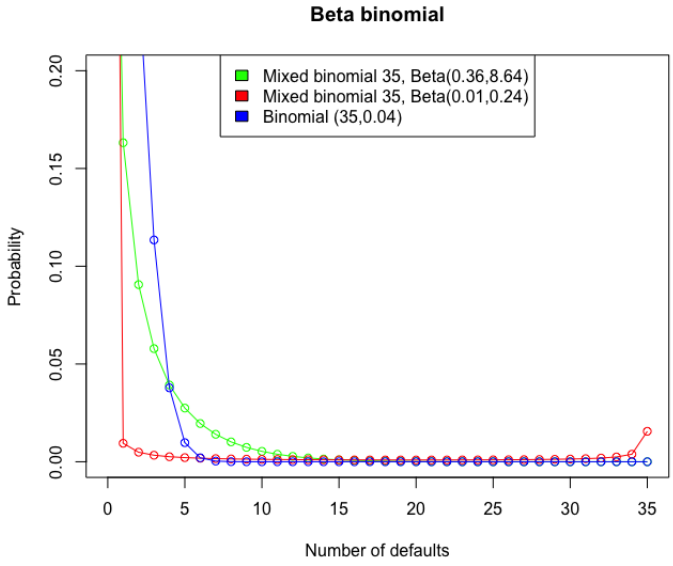

Figure 2: Beta binomial probability function and a binomial function as reference. Zoomed in to capture the difference in the tails.

Figure 1 and 2 shows the probability density functions of our two mixed binomial beta models and a standard binomial probability density function as a reference. As we can see, the most noticable difference is how the model with a high dependance between obligors acts around the tails, meaning that the number of defaults gets pushed to the extremes. Since the probability of an individual default is fairly small ($p = 0.04$) but dependance between obligors in this model is very high, we have a scenario where there is a high probability of all obligors defaulting and all obligors not defaulting, while the probability of some defaulting and some not is extremely low. The low dependancy model is on the other hand more centered around the mean. For example, the probability of only 1 default in the high dependency model is 0.009 and in the low dependency model it is 0.16. Conversely, the probability of 35 defaults in the high dependancy model is 0.015, while in the low dependency model it is 0.000000000285. This also corresponds well to the results from table 3 and 4 where we see a much higher value at risk and expected shortfall for the high dependency model compared to the low dependancy model.

α	95%	99%	99.9
VaR	10.70	18.31	27.91
ES	15.36	22.53	31.35

Table 3: VaR and ES estimates for Beta Model 1.

α	95%	99%	99.9
VaR	13.27	62.81	63.00
ES	51.26	62.96	63.00

Table 4: VaR and ES estimates for Beta Model 2.

3.2 Logit-normal

Figure 3 shows the loss distributions obtained from the logit normal models. The green curve represents the model with a high default dependance (Logit Model 1) and the red represents the model with a low default dependance (Logit Model 2). In comparison to the beta binomial case, the differences between the high and low correlation models are larger. As we can see, low default dependance gives us a distribution centered around the mean ($0.04 * 35 = 1.4$). High default dependance instead gives us a distribution that is "pushed" downwards and spread out over an increasing number of defaults. In this case we can see that for example the probability of having a single default is not much higher than having ten defaults. As in the beta binomial case, this result also corresponds well to the estimated VaR and ES in Tables 5 and 6. The high correlation model gives much higher estimates compared to the low correlation model.

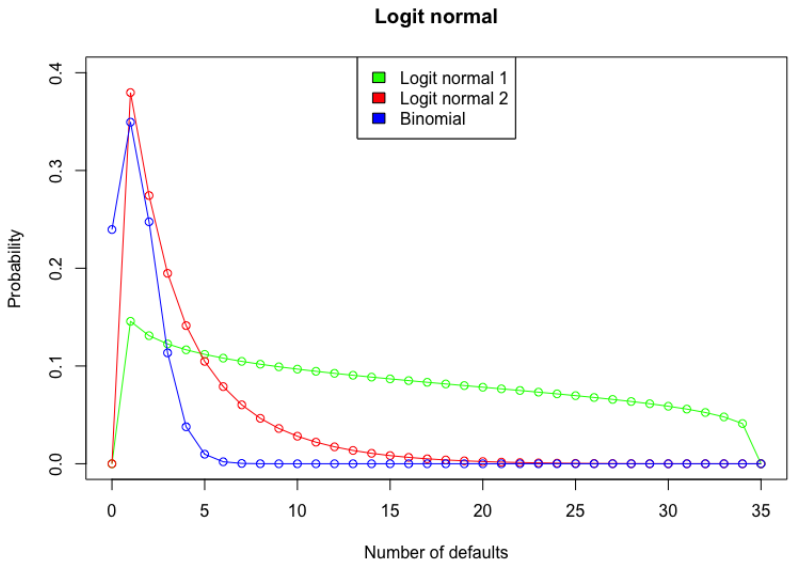

Figure 3: Mixed binomial logit normal probability density function and a binomial function as reference.

α	95%	99%	99.9%
VaR	13.56	62.73	63.00
ES	47.06	62.94	63.00

Table 5: VaR and ES estimates for Logit Model 1.

α	95%	99%	99.9%
VaR	9.10	18.41	33.36
ES	14.85	24.82	38.70

Table 6: VaR and ES estimated for Logit Model 2

4 Task 4 - Monte Carlo Simulation

4.1 Method (Beta-Binomial)

As a comparison to the LPA approximations we want to use Monte Carlo simulations to generate an empirical loss distribution in order to calculate VaR and ES. The outline of the Monte Carlo algorithm works as follows:

1. Let n be the number of simulations.
2. for $j = 1, 2, ..., n$:

   - Generate a sample z_j from the random variable $Z \sim Beta(a, b)$.

   - Generate m samples $u_1, u_2, ..., u_m$ from a uniformly distributed random variable $U \sim Uniform(0, 1)$, where m is the number of obligors.
3. for $i = 1, 2, ..., m$:

   We create a vector of defaults X_i by

$$X_i = \begin{cases} 1 & \text{if } u_i \leq z_j \\ 0 & \text{otherwise} \end{cases} \tag{22}$$

4. Compute $L_m^{(j)} = l \sum_{i=1}^{m} X_i$, where $L_m^{(j)}$ is the total loss from simulation j and l is the default loss, constant for all obligors.

From the simulated empirical loss distribution $\left\{L_m^{(j)}\right\}_{j=1}^{n}$ we can calculate the empirical VaR by finding the α-quantile of our distribution, which is

$$\widehat{VaR_\alpha(L)} = q_\alpha(\left\{L_m^{(j)}\right\}_{j=1}^{n}) = \left\{L_m^{(j)}\right\}_{j=n*\alpha} \tag{23}$$

and

$$\widehat{ES_\alpha(L)} = \frac{\sum_{k=n*\alpha}^{n} \left\{ L_m^{(j)} \right\}_k}{n(1-\alpha)}. \tag{24}$$

In the implementation of the Monte Carlo simulation we have used $n = 1000000$.

4.2 Results (Beta Binomial)

The results of the VaR and ES from the Monte Carlo simulations are presented in tables 7 and 8. Comparing with the LPA approximations from table 3 and 4 we can see that the results are quite similar for both models, concluding that in this case LPA approximation works fairly well.

α	95%	99%	99.9%
VaR	12.60	19.80	30.60
ES	16.98	23.76	33.61

Table 7: Results from the MC simulation with Beta Model 1.

α	95%	99%	99.9%
VaR	12.60	63.00	63.00
ES	46.92	63.00	63.00

Table 8: Results from MC simulation with Beta Model 2.

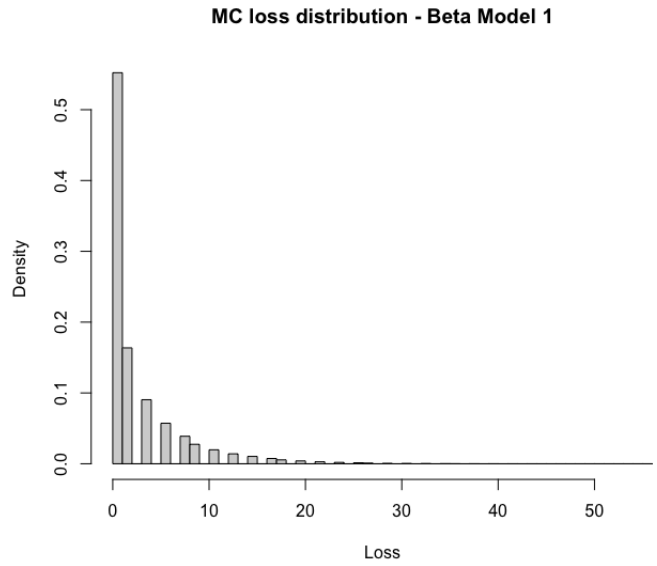

Figure 4: Histogram of the MC loss distribution - Beta Model 1.

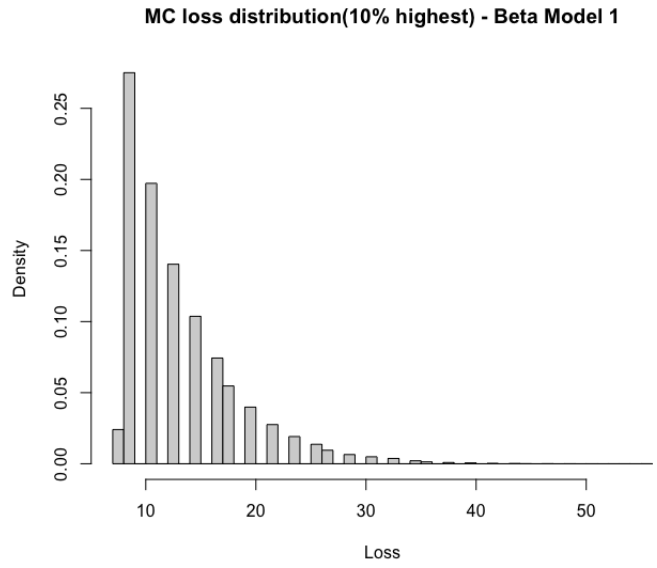

Figure 5: Histogram of the MC loss distribution(highest 10%) - Beta Model 1

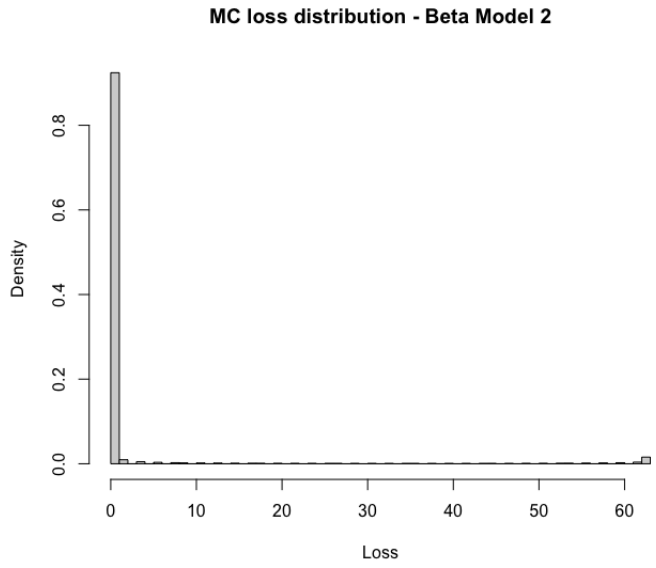

Figure 6: Histogram of the MC loss distribution - Beta Model 2.

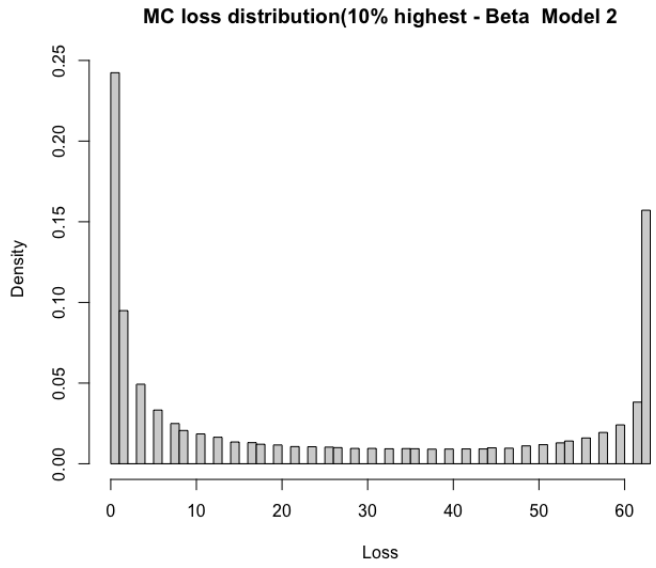

Figure 7: Histogram of the MC loss distribution(highest 10%) - Beta Model 2

5 Task 5 - Estimation of parameters

5.1 Beta-Binomial

In the case of the Beta-Binomial framework, the easiest (and most straightforward) way of estimating the parameters of our model is probably using the method of moments, since the Bayesian framework of using a Binomial likelihood and a Beta conjugate prior is well defined and gives us a nice formula to compute the first two moments of our Beta-Binomial distribution. The first two moments of the Beta-Binomial distribution are given by

$$\mu_1 = \frac{ma}{a+b} \tag{25}$$

$$\mu_2 = \frac{ma(m(1+a)+b)}{(a+b)(1+a+b)} \tag{26}$$

where m is the number of obligors and a, b are the parameters to be estimated. So, assuming that we have some sample data of defaults $\boldsymbol{X} = (X_1, X_2, ..., X_n)$ where each X_i represents the number of defaults (for a given number of obligors m) over a set time period T, we can use the sample moments

$$\hat{\mu}_1 = \frac{1}{n}\sum_{i=1}^{n} X_i = k_1$$

$$\hat{\mu}_2 = \frac{1}{n}\sum_{i=1}^{n} X_i^2 = k_2$$

and setting them equal to (25) and (26), we solve for a and b and get the estimates:

$$\hat{a} = \frac{nk_1 - k_2}{n(k_2/k_1 - k_1 - 1) + k_1} \tag{27}$$

$$\hat{b} = \frac{(n-k_1)(n-k_2/k_1)}{n(k_2/k_1 - k_1 - 1) + k_1}. \tag{28}$$

Another approach is of course to use Maximum Likelihood to estimate the parameters of our model. However, finding closed form expressions for the MLE-estimates in a Beta-Binomial model is not obvious (at least not for me) and one should use a built in package in some programming language (for example the VGAM package in R) that can solve the problem numerically. From reading about the **betabinomial**-function in VGAM, it uses the Fisher Scoring algorithm to solve the Maximum Likelihood equations.

Appendix

R code - Beta Binomial implementation

```
##############################################
################## TASK 1 ####################
##############################################

OBLIGORS <- 35
LOAN <- 3
DEFAULT_LOSSES <- 0.6
DEFAULT_PROBABILITY <- 0.04



a <- 0.2
b <- 4.59

VaR_beta <- LOAN*DEFAULT_LOSSES*OBLIGORS*qbeta(alpha,a,b)

##############################################
############# TASK 2 and 3 ###################
##############################################
library("extraDistr")


DEFAULTS <- 0:35

# low correlation (corr = 0.1)
a1 <- 0.36
b1 <- 8.64

mod1 <- c(a1,b1,a1/(a1+b1),0.1)

# high correlation (corr = 0.8)
a2 <- 0.01
b2 <- 0.24

mod2 <- c(a2,b2,a2/(a2+b2),0.8)

summary <- rbind(mod1,mod2)
rownames(summary) <- c("Model 1","Model 2")
colnames(summary) <- c("a","b","p","rho")

# Fitting density functions
```

```
betamodel_1 <- dbbinom(DEFAULTS,OBLIGORS,alpha = a1,beta = b1)

betamodel_2 <- dbbinom(DEFAULTS,OBLIGORS,alpha = a2,beta = b2)

binomial <- dbinom(DEFAULTS,OBLIGORS,0.04)

plot(DEFAULTS,betamodel_1,"o",col = "green",bg = "green",lwd = 1,
ylim = c(0,0.2),main = "Beta binomial" ,xlab = "Number of defaults",ylab = "Probability")
lines(DEFAULTS,betamodel_2,"o",col = "red",bg = "red",lwd = 1)
lines(DEFAULTS,binomial,"o",col = "blue",bg="blue",lwd = 1)

color.names <- c("green","red","blue")
legend("top",c("Mixed binomial 35, Beta(0.36,8.64)",
"Mixed binomial 35, Beta(0.01,0.24)","Binomial (35,0.04)"),fill = color.names)

# Value at Risk
quantiles <- c(0.95,0.99,0.999)

VaR_beta1 <- LOAN*DEFAULT_LOSSES*OBLIGORS*qbeta(quantiles,a1,b1)

VaR_beta2 <- LOAN*DEFAULT_LOSSES*OBLIGORS*qbeta(quantiles,a2,b2)

# Expected Shortfall
ES_model1 <- numeric(3)
ES_model2<- numeric(3)

for (i in 1:length(quantiles)){
  ES_model1[i] <-  ((LOAN*OBLIGORS*DEFAULT_LOSSES)/(1-quantiles[i]))*integrate(qbeta,
  lower = quantiles[i],upper = 1,shape1 = a1,shape2=b1)$value
  ES_model2[i] <-  ((LOAN*OBLIGORS*DEFAULT_LOSSES)/(1-quantiles[i]))*integrate(qbeta,
  lower = quantiles[i],upper = 1,shape1 = a2,shape2=b2)$value

}

# Making tables for LaTex
model1_var_es <- rbind(VaR_beta1,ES_model1)
rownames(model1_var_es) <- c("VaR","ES")
colnames(model1_var_es) <- c("95%","99%","99.9")

model2_var_es <- rbind(VaR_beta2,ES_model2)
rownames(model2_var_es) <- c("VaR","ES")
colnames(model2_var_es) <- c("95%","99%","99.9")

library(xtable)

xtable(model1_var_es)
```

```
xtable(model2_var_es)


######################################################################
################## TASK 4: MONTE CARLO SIMULATION ####################
######################################################################

# function for Monte Carlo simulation
monte_carlo <- function(nsim,a,b,obligors,loan,default_losses){
  L <- numeric(nsim)
  for (j in 1:nsim){
    Z <- rbeta(1,shape1 = a,shape2 = b)
    U <- runif(obligors,0,1)
    X <- numeric(obligors)
    for (i in 1:obligors){

      if (U[i] <= Z){
        X[i] <- 1
      }
      else{
        X[i] <- 0
      }
    }
    L[j] <- sum(X)*loan*default_losses
  }
  return(L)
}

n <- 1000000

# Beta Binomial model 1, VaR
L_model1 <- monte_carlo(n,a1,b1,OBLIGORS,LOAN,DEFAULT_LOSSES)
MC_VaR_model1 <- quantile(L_model1,quantiles)

# Beta Binomial model 2, VaR
L_model2 <- monte_carlo(n,a2,b2,OBLIGORS,LOAN,DEFAULT_LOSSES)
MC_VaR_model2 <- quantile(L_model2,quantiles)


# function for calculating Expected Shortfall

ES <- function(vaR,losses){

  expectedShortfall <- numeric(length(vaR))

  for (i in 1:length(vaR)){
```

```
    expectedShortfall[i] <- sum(losses[which(losses >= vaR[i])])/
    (length(which(losses >= vaR[i])))
  }
  return(expectedShortfall)
}


# Beta Binomial Model 1, ES
MC_ES_model1 <- ES(MC_VaR_model1,L_model1)

# Beta Binomial Model 1, ES
MC_ES_model2 <- ES(MC_VaR_model2,L_model2)

# Creating tables for LaTex
summaryMC_1 <- rbind(MC_VaR_model1,MC_ES_model1)
rownames(summaryMC_1) <- c("VaR","ES")

summaryMC_2 <- rbind(MC_VaR_model2,MC_ES_model2)
rownames(summaryMC_1) <- c("VaR","ES")

# plots of MC distributions

hist(L_model1,freq = FALSE, breaks = 50,
main = "MC loss distribution - Beta Model 1", xlab = "Loss")

hist(L_model2,freq = FALSE, breaks = 50,
main = "MC loss distribution - Beta Model 2", xlab = "Loss")

hist(sort(L_model1)[900000:1000000],freq = FALSE,breaks = 50,
main ="MC loss distribution(10% highest) - Beta Model 1", xlab = "Loss")

hist(sort(L_model2)[900000:1000000],freq = FALSE, breaks = 50,
main = "MC loss distribution(10% highest - Beta  Model 2", xlab = "Loss")
```

R Code - Logit Normal implementation

```
library(logitnorm)
library(tidyverse)
library(xtable)


#### TASK 2 ####

#function to calculate correlation
correlation <- function(expectation,variance) {
```

```
  variance/(expectation*(1-expectation))
}

mu <- seq(-20,10,length.out = 100)
sigma <- seq(0,10,length.out = 100)

#matrix for storing mu, sigma , E(p(Z)) and Var(p(Z))
param <- matrix(0,nrow = length(mu)*length(sigma),ncol = 4)
colnames(param) <- c("mu","sigma","mean","var")
count = 0;

#grid search over mu and sigma to find the
# parameters of our models
for(i in 1:length(mu)){
  for(j in 1:length(sigma)){
    count = count + 1;
    param[count,3] <- momentsLogitnorm(mu[i],sigma[j])[1]
    param[count,4] <- momentsLogitnorm(mu[i],sigma[j])[2]
    param[count,1] <- mu[i]
    param[count,2] <- sigma[j]
  }
}

tol = 0.001

param <- as.data.frame(param)

#obtaining the combinations of mu and sigma that gives a mean of approximately 0.04
#and a variance depending on which correlation is desired
param_new <- param %>% filter(abs(mean-0.04) <= 0.001)

## correlation = 0.8
high_set <- param_new[2,]
correlation(high_set$mean,high_set$var)


## correlation = 0.1
low_set <- param_new[46,]
correlation(low_set$mean,low_set$var)

##### TASK 3 #####

DEFAULTS <- 0:35
obligors <- 35
loan <- 3
DEFAULT_LOSSES <- 0.6
```

```
loss <- obligors*DEFAULT_LOSSES*loan


x <- DEFAULTS/obligors

#inverse of the mixing distribution function
p_inverse <- function(mu,sigma,x){
  (1/sigma)*(log(x/(1-x))-mu)
}

# plots of loss distributions

# los distribution (LPA) N(p^-1(x)
loss_dist1 <- dnorm(p_inverse(high_set$mu,high_set$sigma,x),0,1)
loss_dist2 <- dnorm(p_inverse(low_set$mu,low_set$sigma,x),0,1)

plot(DEFAULTS,loss_dist1,"l",ylim = c(0,0.4))
lines(DEFAULTS,loss_dist2,"l")

plot(DEFAULTS,loss_dist1,"o",col = "green",bg = "green",lwd = 1,ylim = c(0,0.4),
main = "Logit normal" ,xlab = "Number of defaults",ylab = "Probability")
lines(DEFAULTS,loss_dist2,"o",col = "red",bg = "red",lwd = 1)
lines(DEFAULTS,binomial,"o",col = "blue",bg="blue",lwd = 1)

color.names <- c("green","red","blue")
legend("top",c("Logit normal 1","Logit normal 2","Binomial"),fill = color.names)

## value at risk and expected shortfall

quantiles <- c(0.95,0.99,0.999)

# function for calculating Value At Risk
ValueAr <- function(quantiles,mu,sigma,loss)
{
  valueAtRisk <- (exp(qnorm(quantiles)*sigma + mu)/(1+exp(qnorm(quantiles)*sigma + mu)))
}

# value at risk
VaR_model1 <- loss*ValueAr(quantiles,high_set$mu,high_set$sigma,loss)
VaR_model2 <- loss*ValueAr(quantiles,low_set$mu,low_set$sigma,loss)

#expected shortfall
ES_model1 <- numeric(3)
ES_model2 <- numeric(3)

  for(i in 1:length(quantiles)){
```

```
  ES_model1[i] <- (loss/(1-quantiles[i]))*integrate(ValueAr,lower = quantiles[i],
  upper = 1,high_set$mu,high_set$sigma,loss)$value
  ES_model2[i] <- (loss/(1-quantiles[i]))*integrate(ValueAr,lower = quantiles[i],
  upper = 1,low_set$mu,low_set$sigma,loss)$value
}
```