

Measuring Concentration Risk - A Partial Portfolio Approach

Andrija Djurovic

www.linkedin.com/in/andrija-djurovic

Concentration Risk

- Concentration risk represents a significant source of risk in banking, particularly in emerging and small economies.
- Pillar 1 capital requirements don't cover concentration risk.
- Banks must autonomously estimate and set aside capital buffers to mitigate concentration risk, usually as part of Pillar 2 models.
- Inadequate recognition of concentration risk can lead to insufficient capital levels, even with apparently high capital ratios.
- A partial portfolio approach as a way to address the concentration risk and to complement the existing regulatory capital requirements (details available [here](#)).

A Partial Portfolio Approach (PPA)

Assuming a portfolio of n exposures with a non-granular sub-portfolio of m exposures ($m \leq n$), the PPA can be outlined in the following steps after determining the number of simulations (B):

- 1 Simulate a value for the systemic risk factor z from the standard normal distribution ($N[0, 1]$).
- 2 For each exposure in the non-granular sub-portfolio, simulate the idiosyncratic factor ϵ from the standard normal distribution ($N[0, 1]$).
- 3 Given the simulated systemic, idiosyncratic factor and calculated asset correlation, for each exposure in the non-granular sub-portfolio, simulate the asset return value as:

$$y_i = \sqrt{\rho} * z + \sqrt{1 - \rho} * \epsilon_i$$

where ρ is asset correlation, z and ϵ_i systemic and idiosyncratic factor, respectively.

- 4 For each exposure in the non-granular sub-portfolio, simulate the default indicator as follows:

$$I_i = y_i < N^{-1}(\overline{PD}_i)$$

where y_i is the asset return (step 3), $N^{-1}(\overline{PD}_i)$ is the quantile of the standard normal variable, and \overline{PD}_i is unconditional Probability of Default for exposure i .

A Partial Portfolio Approach (PPA) cont.

- 5 Calculate the loss for the non-granular sub-portfolio as

$$Loss^{non-granular} = \sum_{i=1}^m I_i * LGD_i * EAD_i$$

where I_i is the indicator from the step 4, and LGD_i and EAD_i Loss Given Default and Exposure at Default, respectively.

- 6 Calculate the loss for the granular sub-portfolio as:

$$Loss^{granular} = \sum_{i=1}^{n-m} N \left[\frac{N^{-1}(\overline{PD}_i) - \rho * z}{\sqrt{1 - \rho}} \right] * LGD_i * EAD_i$$

where N and N^{-1} present standard normal cumulative distribution and quantile function.

- 7 Sum up losses from the non-granular and granular portfolio.
- 8 After repeating steps from 1 to 7 selected B times, calculate the 99.9 percentile of the loss distribution and subtract the expected loss to get the required Credit VaR.

Simulation Setup

Simulation dataset available [here](#).

R:

```
#lgd
lgd <- 0.30
#asset correlation
rho <- 0.05
#ead threshold
seq.ft <- c(seq(from = 0,
                to = 1.5,
                length.out = 100),
            10) / 100
#number of simulations
B <- 100000
#confidence level
cl <- 0.999
```

Python:

```
#lgd
lgd = 0.30
#asset correlation
rho = 0.05
#ead threshold
seq_ft = np.concatenate([np.linspace(start = 0,
                                       stop = 1.5,
                                       num = 100),
                          [10]]) / 100
#number of simulations
B = 100000
#confidence level
cl = 0.999
```

R Code Extract

```
...  
  
for (i in 1:B) {  
  #set random seed  
  set.seed(i)  
  #systemic factor  
  z <- rnorm(n = 1)  
  #idiosyncratic factor  
  epsilon <- rnorm(n = nrow(db.ng))  
  #asset return  
  ar <- sqrt(rho)*z + sqrt(1 - rho)*epsilon  
  #default indicator (non-granular portfolio)  
  def.ind <- ifelse(ar < qnorm(p = db.ng$pd), 1, 0)  
  #loss of the non-granular portfolio  
  loss.ng <- sum(def.ind * db.ng$ead * lgd)  
  #loss of the granular portfolio  
  pd.cond <- pnorm(q = (qnorm(p = db.g$pd) - sqrt(rho)*z) / sqrt(1 - rho))  
  loss.g <- sum(db.g$ead * pd.cond * lgd)  
  #portfolio loss  
  res[i] <- loss.ng + loss.g  
}  
  
...
```

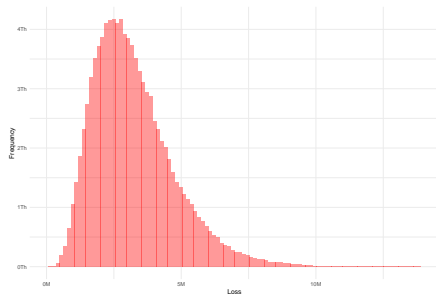
Python Code Extract

```
import numpy as np
from scipy.stats import norm
...

for i in range(B):
    #set random seed
    np.random.seed(i + 1)
    #systemic factor
    z = np.random.normal(size = 1)
    #idiosyncratic factor
    epsilon = np.random.normal(size = db_ng.shape[0])
    #asset return
    ar = np.sqrt(rho) * z + np.sqrt(1 - rho) * epsilon
    #default indicator (non-granular portfolio)
    def_ind = np.where(ar < norm.ppf(db_ng["pd"]), 1, 0)
    #loss of the non-granular portfolio
    loss_ng = np.sum(def_ind * db_ng["ead"] * lgd)
    #loss of the granular portfolio
    pd_cond = norm.cdf((norm.ppf(db_g["pd"]) - np.sqrt(rho) * z) /
                       np.sqrt(1 - rho))
    loss_g = np.sum(db_g["ead"] * pd_cond * lgd)
    #portfolio loss
    res[i] = loss_ng + loss_g
...
```

Simulation Result Extract

Loss Distribution - Full Non-Granular Portfolio:



Loss at 99.9% CL - Varying EAD Threshold:

