FIM 548-001 Research Project

Raiden Han

April 15, 2022

Perface

This project uses Python programming language for Monte-Carlo simulation. I imported the NumPy and the CuPy package for calculation and GPU acceleration to implement the method.

Baseline Scenario

Basic Setting

Using the parameters defined by the authors, we have

- Unissued Stock Options' Ratio: $o = 5\%$
- Volatility: $\sigma = 0.9$
- Exit rate: $\lambda = 0.25$
- Risk-free rate: $r_f = 2.5\%$

Also, if we denote all values in the unit of one million dollars, we have $P_A = 450$, $P_B = 1,000$, $I_A = 50$, and $I_B = 100$. And the post-money valuations without unissued stock options are

$$
\begin{aligned}
P'_A &= (1-o)P_A = (1-5\%) \times 450 = 427.5, \\
P'_B &= (1-o)P_B = (1-5\%) \times 1,000 = 950.
\end{aligned}
$$

Therefore, the holding ratios for all investors are

$$
\begin{aligned}
R_B &= \frac{I_B}{P'_B} = \frac{100}{950} \approx 10.53\%, \\
R_A &= \frac{I_A}{P'_A}(1-r_B) = \frac{50}{427.5} \times (1-10.53\%) \approx 10.46\%, \\
R_C &= 1 - r_B - r_A \approx 1 - 10.53\% - 10.46\% = 79.01\%.
\end{aligned}
$$

Algorithm

Define the fair valuation at Series B as X_0. For each simulation,

- Generate the exit time $T \sim Exp(\lambda)$
- Generate a standard normal random variable $Z \sim N(0,1)$
- Calculate the exit value $X_T = X_0 \exp\left[\left(r_f - \sigma^2/2\right)T + \sigma\sqrt{T}Z\right]$
- Calculate the IPO probability

$$p = \begin{cases} 0, & X_T \le 32, \\ 0.65 \times \frac{\log(X_T/32)}{\log(1000/32)}, & 32 < X_T \le 1,000, \\ 0.65 + 0.2 \times \frac{\log(X_T/1000)}{\log(100)}, & 1,000 < X_T \le 100,000, \\ 1, & X_T \ge 100,000 \end{cases}$$

- Calculate the outcomes to all investors under different situations using the pari passu seniority assumption

$$\begin{aligned} f_A^{IPO}(X_T) &= X_T \times R_A, \\ f_A^{M\&A}(X_T) &= \max\left\{\min\left\{\frac{I_A}{I_A+I_B}X_T, I_A\right\}, X_T \times R_A\right\}, \\ f_B^{IPO}(X_T) &= X_T \times R_B, \\ f_B^{M\&A}(X_T) &= \max\left\{\min\left\{\frac{I_B}{I_A+I_B}X_T, I_B\right\}, X_T \times R_B\right\}, \\ f_C^{IPO}(X_T) &= X_T - f_A^{IPO}(X_T) - f_B^{IPO}(X_T), \\ f_C^{M\&A}(X_T) &= X_T - f_A^{M\&A}(X_T) - f_B^{M\&A}(X_T) \end{aligned}$$

- Derive the fair investment for Series B investor

$$e^{-rT}\left[pf_B^{IPO}(X_T) + (1-p)f_B^{M\&A}(X_T)\right]$$

  and the fair value of the common shares

$$\frac{e^{-rT}\left[pf_C^{IPO}(X_T) + (1-p)f_C^{M\&A}(X_T)\right]}{(1-o)R_C P_B/1}$$

Result

By running the above simulation $N = 10^{10}$ times and calculating the means of all fair investments and fair values, I validated that when $X_0 = 771$, the fair investment is $I_B = 100$. The overvaluation ratio of the company is

$$\Delta_V = \frac{P_B - X_0}{X_0} = \frac{1,000 - 771}{771} \approx 30\%.$$

Also, the fair value of the common shares is $FV_C = \$0.78$, and the percentage by which the share price overstates the value of common shares is

$$\Delta_C = \frac{1 - FV_C}{FV_C} = \frac{1 - 0.78}{0.78} \approx 28\%.$$

Automatic Conversion Veto at 0.75X Scenario

All the calculation is the same as the baseline scenario, except that the simulated outcomes are

$$\begin{aligned}
f_A^{M\&A}(X_T) &= \max\left\{\min\left\{\frac{I_A}{I_A+I_B}X_T, I_A\right\}, X_T \times R_A\right\}, \\
f_A^{IPO}(X_T) &= \begin{cases} f_A^{M\&A}(X_T), & \text{if } f_A^{M\&A}(X_T) > 0.75 X_T \times R_A, \\ X_T \times R_A, & \text{otherwise,} \end{cases} \\
f_B^{M\&A}(X_T) &= \max\left\{\min\left\{\frac{I_B}{I_A+I_B}X_T, I_B\right\}, X_T \times R_B\right\}, \\
f_B^{IPO}(X_T) &= \begin{cases} f_B^{M\&A}(X_T), & \text{if } f_B^{M\&A}(X_T) > 0.75 X_T \times R_B, \\ X_T \times R_B, & \text{otherwise,} \end{cases} \\
f_C^{IPO}(X_T) &= X_T - f_A^{IPO}(X_T) - f_B^{IPO}(X_T), \\
f_C^{M\&A}(X_T) &= X_T - f_A^{M\&A}(X_T) - f_B^{M\&A}(X_T).
\end{aligned}$$

Again, I ran 10^{10} simulations and proved that when $X_0 = 651$, the fair investment is $I_B = 100$. The overvaluation ratio of the company is

$$\Delta_V = \frac{P_B - X_0}{X_0} = \frac{1,000 - 651}{651} \approx 54\%.$$

Also, the fair value of the common shares is $FV_C = \$0.63$, and the percentage by which the share price overstates the value of common shares is

$$\Delta_C = \frac{1 - FV_C}{FV_C} = \frac{1 - 0.63}{0.63} \approx 58\%.$$

Appendix - Python Code

April 22, 2022

[1]:
```
import numpy as np
import cupy as cp

cp.random.seed(415)
```

[2]:
```
def reinforce_function(func, n, *args):
    """ Repeat a function with certain parameters several times and return the
    mean result

    Parameters
    ----------
    func : function
        The function to be repeated
    n : int
        The repeating time
    args : tuple
        The parameters for the function

    Returns
    -------
    mean_value : float
        The mean value of n times running
    """

    n = int(n)
    value = []
    for i in range(n):
        value.append(func(*args))
    value = cp.array(value)
    mean_value = cp.mean(value, axis=0)

    return mean_value
```

1 Parameters

[3]:
```
unis_stock = 0.05  # Unissued stock options' ratio
sigma = 0.9  # Volatility
lam = 0.25  # Exit rate
rf = 0.025  # Risk-free rate
```

2 Baseline Scenario

[4]:
```
def baseline_valuation(n, x0, unis_stock, sigma, lam, rf):
    # Convert the simulation size into an integer
    n = int(n)
    # Initialize parameters of the asset
    pa, pb, ia, ib = 450 * (1 - unis_stock), 1000 * (1 - unis_stock), 50, 100
    # Calculate the holding ratios for all investors
    rb = ib / pb  # Series B Investor
    ra = (1 - rb) * ia / pa  # Series A Investor
    nc = pb - ib - ib / rb * ra  # Number of Common shares
    # Generate random variables
    t = cp.random.exponential(1 / lam, size=n)
    z = cp.random.normal(size=n)
    # Calculate the exit value
    xt = x0 * cp.exp((rf - sigma ** 2 / 2) * t + sigma * cp.sqrt(t) * z)
    # Calculate the payout to the new investor
    fb_ipo = xt * rb
    fb_mna = cp.maximum(cp.minimum(ib * xt / (ia + ib), ib), xt * rb)
    # Calculate the payout to common shareholders
    fc_ipo = xt * (1 - ra - rb)
    fa_mna = cp.maximum(cp.minimum(ia * xt / (ia + ib), ia), xt * ra)
    fc_mna = xt - fb_mna - fa_mna
    # Calculate the proability of an IPO
    p_ipo = cp.ones(n)
    p_ipo[xt <= 32] = 0
    p_ipo[(xt > 32) & (xt <= 1000)] = 0.65 * (
        cp.log(xt[(xt > 32) & (xt <= 1000)] / 32)) / (cp.log(1000 / 32))
    p_ipo[(xt > 1000) & (xt <= 100000)] = 0.65 + 0.2 * (
        cp.log(xt[(xt > 1000) & (xt <= 100000)] / 1000)) / (cp.log(100))
    # Generate the outcomes
    outcome_b = p_ipo * fb_ipo + (1 - p_ipo) * fb_mna
    outcome_c = p_ipo * fc_ipo + (1 - p_ipo) * fc_mna
    eob = cp.mean(cp.exp(-rf * t) * outcome_b)
    eoec = cp.mean(cp.exp(-rf * t) * outcome_c) / nc

    return eob, eoec
```

[5]:
```
# Output the results
x0 = 771
print("The fair investment for Series B investor: ${:.0f}m\n"
      "The fair value of common shares: ${:.2f}".format(
      *reinforce_function(baseline_valuation,
                          1e3, 1e7, x0, unis_stock, sigma, lam, rf)))
```

```
The fair investment for Series B investor: $100m
The fair value of common shares: $0.78
```

3 Automatic Conversion Veto Scenario

[6]:
```
def auto_conv_veto_valuation(n, x0, unis_stock, sigma, lam, rf, acv_rate):
    # Convert the simulation size into an integer
    n = int(n)
    # Initialize parameters of the asset
    pa, pb, ia, ib = 450 * (1 - unis_stock), 1000 * (1 - unis_stock), 50, 100
    # Calculate the holding ratios for all investors
    rb = ib / pb  # Series B Investor
    ra = (1 - rb) * ia / pa  # Series A Investor
    nc = pb - ib - ib / rb * ra  # Number of Common shares
    # Generate random variables
    t = cp.random.exponential(1 / lam, size=n)
    z = cp.random.normal(size=n)
    # Calculate the exit value
    xt = x0 * cp.exp((rf - sigma ** 2 / 2) * t + sigma * cp.sqrt(t) * z)
    # Calculate the payout to the new investor
    fb_mna = cp.maximum(cp.minimum(ib * xt / (ia + ib), ib), xt * rb)
    fb_ipo = cp.where(fb_mna > acv_rate * xt * rb, fb_mna, xt * rb)
    # Calculate the payout to common shareholders
    fa_mna = cp.maximum(cp.minimum(ia * xt / (ia + ib), ia), xt * ra)
    fa_ipo = cp.where(fa_mna > acv_rate * xt * ra, fa_mna, xt * ra)
    fc_mna = xt - fb_mna - fa_mna
    fc_ipo = xt - fb_ipo - fa_ipo
    # Calculate the proability of an IPO
    p_ipo = cp.ones(n)
    p_ipo[xt <= 32] = 0
    p_ipo[(xt > 32) & (xt <= 1000)] = 0.65 * (
        cp.log(xt[(xt > 32) & (xt <= 1000)] / 32)) / (cp.log(1000 / 32))
    p_ipo[(xt > 1000) & (xt <= 100000)] = 0.65 + 0.2 * (
        cp.log(xt[(xt > 1000) & (xt <= 100000)] / 1000)) / (cp.log(100))
    # Generate the outcomes
    outcome_b = p_ipo * fb_ipo + (1 - p_ipo) * fb_mna
    outcome_c = p_ipo * fc_ipo + (1 - p_ipo) * fc_mna
    eob = cp.mean(cp.exp(-rf * t) * outcome_b)
    eoec = cp.mean(cp.exp(-rf * t) * outcome_c) / nc
```

```
    return eob, eoec
```

[7]:
```
# Output the results
x0 = 651
acv_rate = 0.75
print("The fair investment for Series B investor: ${:.0f}m\n"
      "The fair value of common shares: ${:.2f}".format(
      *reinforce_function(auto_conv_veto_valuation,
                          1e3, 1e7, x0, unis_stock, sigma, lam, rf, acv_rate)))
```

```
The fair investment for Series B investor: $100m
The fair value of common shares: $0.63
```