Python Code流程圖

Step 1: Insert Variable

總共有6個變數，分別為履約價（set as x in python）、現貨市場價格（s）、無風險利率（r）、漲價倍數（u）、跌價倍數（d）、期數（n）。

備註：根據 BSM Model 中 u 與 d 應該是由資產價格的波動率來決定，所以正常來說所輸入的變數應該要是 σ 波動率，但因為在 HW3 的範例是直接輸入 u 和 d 因此在此略過此公式。

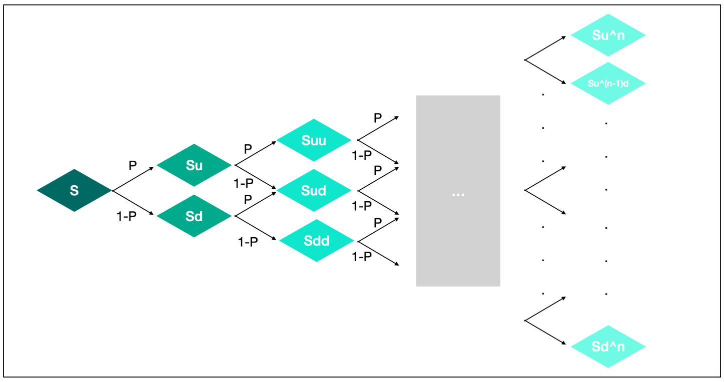
Step 2 : Background

無風險報酬在連續複利的情況下等於 e^r（無風險利率），因此先定義 R=e^r。

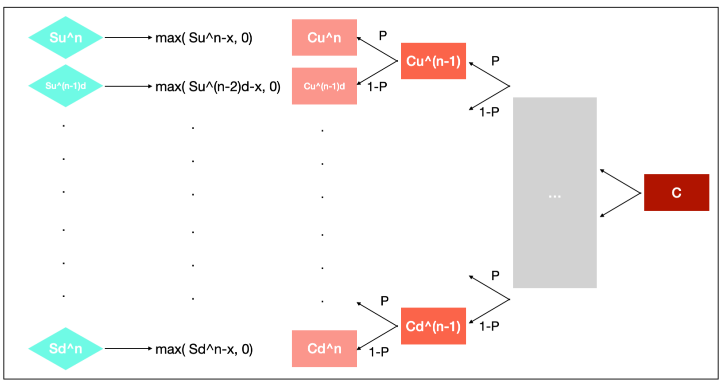
而在 BOPM 中經由推導可知發生 u 的機率（p）是 (R-d)/(u-d) ，d 的機率則是 1-p。

Step 3 : 從現貨價格推導至選擇權價格，選擇權價格再經由巢狀迴圈得現價

根據 Binominal Option Model，假設期數為 n 期，則第n期的現貨價格會有 (n+1) 種可能，其二項樹如下圖，因此利用 python for loop 算出最後一期 (n+1) 個 S。



透過最後一期的S可以直接推導出最後一期的選擇權價格，而倒數第二期的選擇權價格是由倒數一期的選擇權價格乘上機率得出，倒數第三期由倒數第二期推出...以此類推，最後可以得出今期的選擇權價格，不管是買權或是賣權都是同道理，下圖僅列出買權示意圖。



在 python 中，設定的 s list 僅有最後一期 (s_)：

s_	j=0	j=1	…	j=n
i=0	s*u^n	s*u^(n-1)*d	…	s*d^n

該規律為 s*u^(n-i+1)*d^(j-1)，從上述得出 c_ list：

c_	j=0	j=1	…	j=n
i=0	max(s*u^n-x, 0)	max(s*u^(n-1)*d-x, 0)	…	max(s*d^n-x, 0)

c_ list 導出 Binominal process for the call price:

c	j=0	j=1		j=(n-2)	j=(n-1)
i=0	(p*c_[0][0]+(1p)*c_[0][1])/R	(p*c_[0][1]+(1p)*c_[0][2])/R			(p*c_[0][n-1]+(1p)*c_[0][n])/R
i=1	(p*c[0][0]+(1p)*c[0][1])/R	(p*c[0][1]+(1p)*c[0][2])/R		(p*c_[0][n-2]+(1p)*c_[0][n-1])/R	
i=2	(p*c[1][0]+(1p)*c[1][1])/R	(p*c[1][1]+(1p)*c[1][2])/R			
…					
i=(n-1)	(p*c[n-2][0]+(1p)*c[n-2][1])/R				

p等同道理。

最後結果為calls and put 的 Binominal process 和 今期 的買權、賣權價格。