模擬與統計計算 HW4

N26120870 電機所 林耕澤

HOMEWORK (POISSON RANDOM NUMBER GENERATOR)

- Method 1 (from Chap 4.2, p. 54)
- Method 2 (from Chap 5.1, p. 71)
- Compare these two methods.
  - Check if they really generate the same distribution based on Poisson distribution?
  - Why?

- ## Method 1:

程式碼:

$$p_i = P\{X = i\} = e^{-\lambda}\frac{\lambda^i}{i!} \quad i = 0, 1, \ldots$$

```
def Method_1(Lambda):
    U = random.random()
    i = 0
    baseline = np.exp(-Lambda) # i == 0
    while U >= baseline:
        i += 1
        baseline += math.exp(-Lambda) * Lambda ** i / math.factorial(i)
    return i
```

Figure 1, 寫法一

$$p_{i+1} = \frac{\lambda}{i+1}p_i, \quad i \geq 0$$

```
def Method_1(Lambda):
    U = random.random()
    i = 0
    baseline = np.exp(-Lambda) # i == 0
    p = baseline
    while U >= baseline:
        p = Lambda * p / (i + 1)
        i += 1
        baseline += p
    return i
```

Figure 2, 寫法二

- # Method 2:

程式碼:

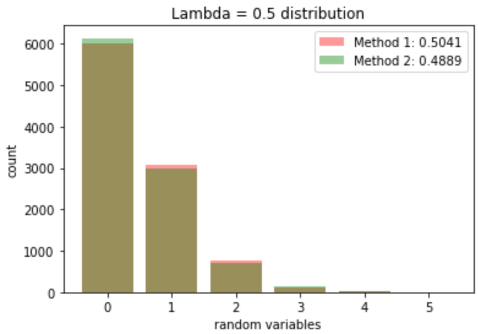
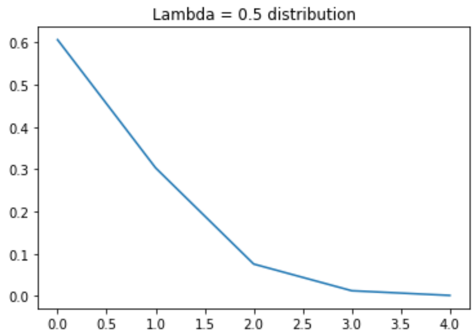
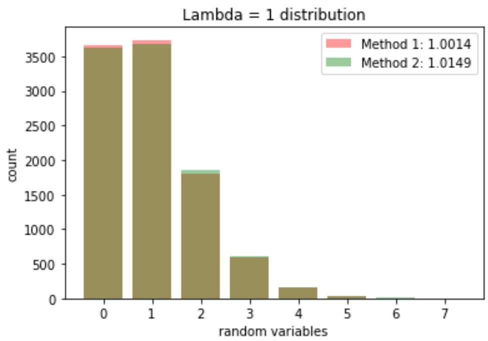
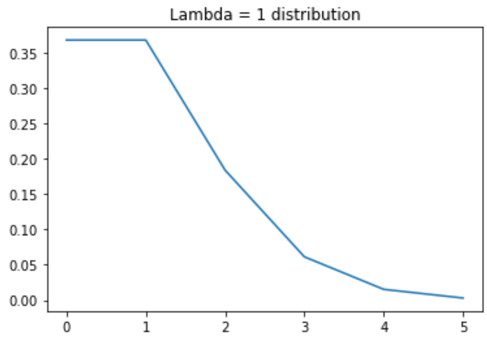
$$N = \mathrm{Max}\left\{n: \sum_{i=1}^{n} -\frac{1}{\lambda}\log U_i \leqslant 1\right\}$$

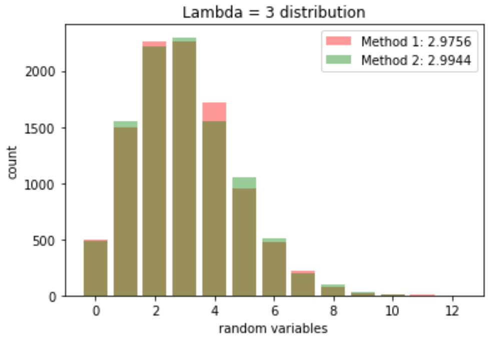
```python
def Method_2(Lambda):
    i = 0
    total = 0
    while total <= 1:
        U = random.random()
        total += -1 / Lambda * math.log(U)
        i += 1

    return i - 1
```

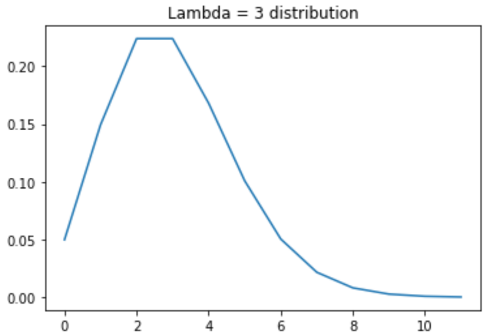
Figure 3, 方法二程式碼

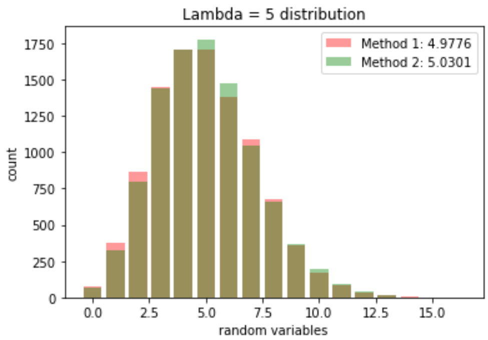
- # 比較兩種帕松(Poisson)隨機變數生成器:

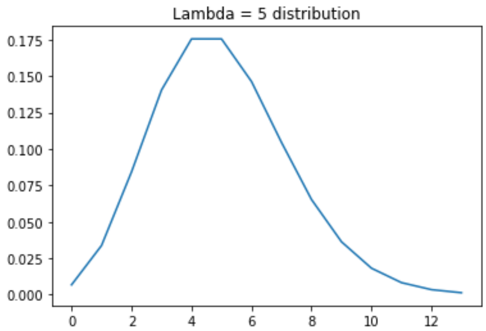
1. 產生的隨機變數分布:

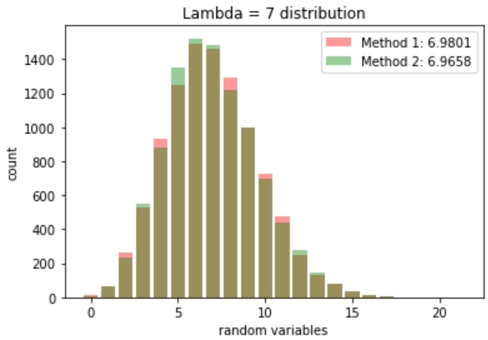
Simulation	Poisson Distribution
	
	

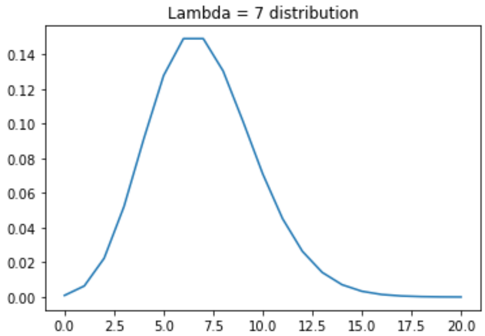


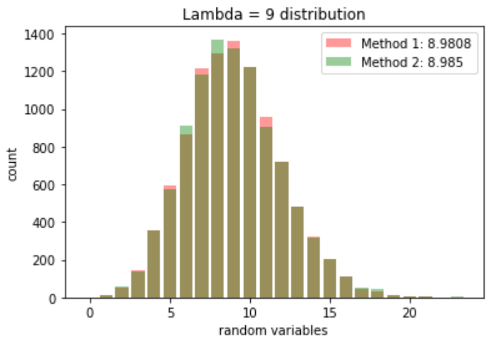


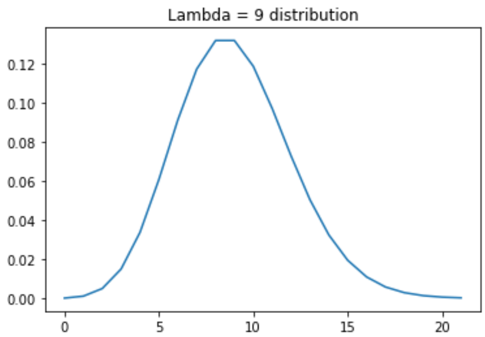












討論:

經過上面表格中圖片可以很明顯得知不論是使用 Method 1 或是 Method 2 所得到的隨機變數分布是差不多的(不過還是會有些微差距)。另外還要判斷使否達到預期，判斷使否達到預期主要觀察兩個點:

- 模擬結果的隨機變數的平均值是否與 Lambda 相近。

- 模擬的分布是否與理想分布相近。

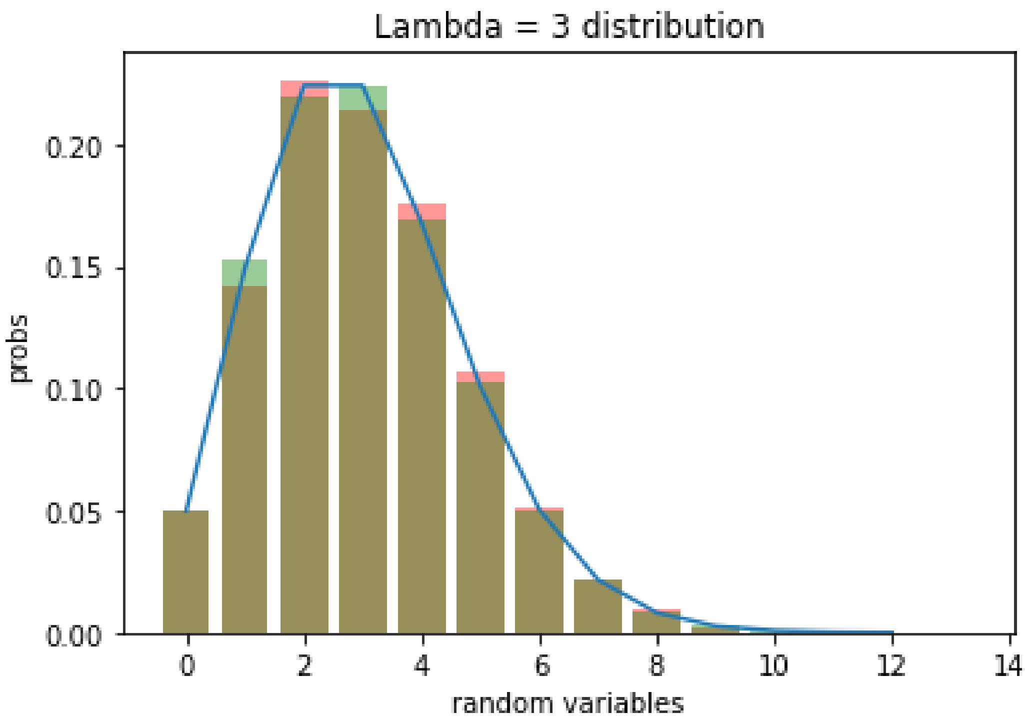
本次模擬結果都有符合上述條件，因此有達到預期。

兩種方法為何都能產生帕松隨機變數:

Method 1 使用逆變換法(inverse transform method)，根據帕松分佈的 CDF 找到其反函數並代入[0,1]之間的隨機值，很直接的就能得到帕松的隨機變數。而 Method 2 利用到帕松過程的 Lambda 為兩指數分布的獨立事件的時間區間的 Lambda 的特性，將並且專注於一個時間區間 N(1)，就變成了一個固定時間區間發生多少次事件，也因此能得到帕松的隨機變數。

實際例子應證:

假設一個交叉路口，每個小時發生車禍的平均次數為 3 次。那麼它的 Lambda 就會等於 3，由此帶入帕松分布所得到的機率分布以藍色折線表示，另外我以 10000 小時作為採樣時間丟入 Method 1、Method 2 算出平均每小時發生的車禍次數的機率分布(Method 1 為紅色直方圖、Method 2 為綠色、泥巴色為重疊之處)，發現結果都非常接近帕松分布。



2. 執行時間:

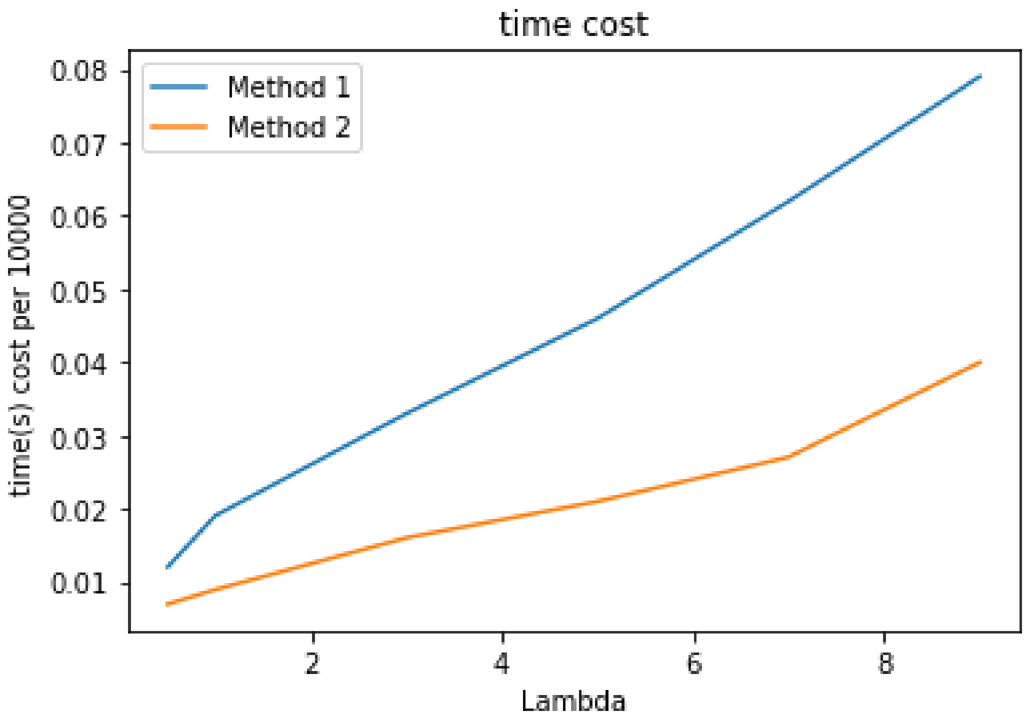


Figure 4,時間花費

討論:

由此圖可以得知不論是 Method 1 或是 Method 2 都會隨著 Lambda 上升而導致執行時間增加，在 Method 1 中是使用逆變換法(Inverse Transform Method)生成帕松分布的隨機數。生成的過程中，需要計算並比較隨機數（U）和 baseline。一開始的 baseline 是帕松分布中事件數為 0 的機率，也就是 i == 0 的機率。隨著 Lambda 的增加，exp(-Lambda) 的值變得非常接近於 0，一開始的 baseline 也迅速趨近於 0。因此隨機數需要更多次的迭代才能不滿足 U >= baseline 的條件，因此執行時間增加。而在 Method 2 中是使用另一種方法生成帕松分布的隨機數，具體來說是使用了指數分佈的特性。在每一次循環中，計算 total 的值，它是指數分佈的累積分佈函數（CDF）在[0, 1]區間內的值。當 total 小於等於 1 時，循環停止。隨著 Lambda 的增加，計算 total 需要更多次的迭代，因為 total 的值在每次迭代中都只增加一小部分。當 Lambda 增加時，total 達到 1 的機率減小，需要更多次的迭代來滿足 total <= 1 的條件，因此執行時間增加。除此之外，從此圖也可得知 Method 2 比 Method 1 的執行速度快，我推斷主要原因應該與計算的複雜度有關。