財務工程與 Excel VBA 應用

(一)選擇權定價模型

Financial Engineering with Excel VBA (1) Option Pricing Model

KGI 中信證券

風險管理部 協理

董夢雲 博士

目　　　錄

Part 3: Monte-Carlo Simulations

一、迴圈敘述

(一)For … Next

```
For counter = start to end [Step stepval]
    [instructions]
Next [counter]
```

```
Sub MsgLoop()
   Dim counter As Integer

    For counter = 1 to 10
       MsgBox counter
    Next counter
End Sub

Sub SumTen()
   Dim total As Long
   Dim counter As Integer

   total = 0
   For counter = 1 to 10
      total = total + counter
   Next counter
End Sub
```

(二)Do … Loop

```
Do [While condition]
   [instructions]
Loop
```

```
Do
   [instructions]
Loop [While condition]
```

```
Sub DoubleValue()
   Range("A1").Select
   Do While not IsEmpty(ActiveCell)
      ActiveCell.Value = 2 * ActiveCell.Value
      ActiveCell.Offset(1, 0).Select
   Loop
End Sub
```

(三)For Each … Next

```
For Each element In group
   [instructions]
Next[element]
```

```
Sub Macro1()
    Dim MyArray(5) As Double
    For i=0 to 5
      MyArray(i)=Rnd
    Next i
    For Each n In MyArray
      MsgBox n
    Next n
End Sub
```

(四)中斷迴圈的執行

✦ 中斷迴圈的執行，可使用 Exit 敘述

Exit For In For … Next

Exit Do In Do … Loop

```
Dim lngSalary(100) As Long
Dim intCounter As Integer

lngSalary(0)= 50000
lngSalary(1)= 130000
…
lngSalary(99)= 45000

For intCounter = 0 To 99
   If lngSalary(intCounter) > 100000 Then
      MsgBox Str(intCounter)+" Higher than President"
      Exit For
   End If
Next
```

二、陣列宣告與使用

(一)一維陣列

- 一組具有相同資料型態的資料，以矩陣方式來儲存資料，每一筆資料都有獨一的索引可加以辨識。

- 陣列的宣告

```
Dim ArrayName(num) As DataType
Example:   Dim array_1(11) As Float
           Dim array_1!(11)
```

  - 單精準度陣列，12 筆資料
  - array_1(0)，array_1(1)，…..，array_1(11)
  - 可以 Option Base #，改變基底為 #，預設基底為 0

```
Option Base 1
Dim array_2(12) As Float
```

  - 可以明確指定索引的上、下限

```
Dim array_3(10 To 21) As Float
```

✦ 10 位學生，國、英、數三科考試分數資料

擬計算全班國文科之平均數，標準差，及個人之排名

```
Dim St_Name(9) As String
Dim Chn_Scr(9) As Double
Dim Sum As Double
Dim Avg As Double
Dim Std As Double
Dim i As Integer
Dim j As Integer
Dim Temp_Val As Double
Dim Temp_Str As String

For i = 0 to 9
   St_Name(i) = “Student_” + Str(i)
   Chn_Scr(i) = 60 + 40*Rnd()
Next i

Sum = 0
Avg = 0
For i = 0 to 9
   Sum = Sum + Chn_Scr(i)
Next i
Avg = Sum /10
```

```
Sum = 0
Std = 0
For i = 0 to 9
   Sum = Sum + (Chn_Scr(i)-Avg)^2
Next i
Std = Sqr(Sum/9)

For i = 0 to 9
  For j = i+1 to 9
   If Chn_Scr(i) < Chn_Scr(j) then
      Temp_Val = Chn_Scr(i): Temp_Str = St_Name(i)
      Chn_Scr(i) = Chn_Scr(j): St_Name(i) = St_Name(j)
      Chn_Scr(j) = Temp_Val: St_Nmae(j) = Temp_Str
   End If
  Next j
Next i
```

(二)多維陣列

- 多維陣列宣告方式類似

  ```
  Dim array_4(2, 3) As Float
  ```

  - array_4(0, 0)，array_4(0, 1)，……，array_4(2, 2)，array_4(2, 3)，12(3*4)筆資料

```
Dim I, J As Integer
Dim Array_1(10,10) As Double

For I = 1 To 10
  For J = 1 To 10
    Array_1(I, J) = I*J
  Next J
Next J
```

- 10 位學生，國、英、數三科考試分數資料
  - 擬計算各科之平均數、標準差以及每人總分之平均數，及個人之排名

(三)動態陣列

- 事前無法得知陣列大小，暫不指定

```
Dim intStudent() As Integer
```

 待確定後，再以 ReDim 陳述動態改變陣列之大小

```
ReDim intStudent(100)
```

- 二維陣列亦同

```
Dim intTwoDimArray() As Integer
ReDim intTwoDimArray(10, 10)
```

(四)函數陣列參數的傳遞

- 函數程序可以接受陣列當做參數，經過處理後再傳回單一的數值。

```
Function SumArray(List() As Double) As Double
  SumArray = 0
  For Each Item In List
    If Application.IsNumber(Item) Then
      SumArray = SumArray + Item
    End If
  Next Item
End Function

Sub MakeList()
  Dim Nums(1 To 100) As Double
  Dim i As Integer
  For i = 1 To 100
    Nums(i) = Rnd * 1000
  Next i
  MsgBox SumArray(Nums)
End Sub
```

三、亂數函數*

(一)亂數產生概論

- ✦ 利用電腦連續產生準亂數(pseudorandom number)，作為亂數的來源。
  - 準亂數係由確定的方式產生的。
  - 準亂數數列顯現在[0, 1)中間獨立地均等分佈。

➢ 程式語言內附亂數函數

Rnd 函數

傳回一型態為 **Single** 的值，其內容為一亂數值。

✦ 語法

Rnd[**(***number***)**]

選擇性引數 *number* 可以是一型態為 Single 的值，或任何數值運算式。

✦ 傳回值

如果 *number* 的值是	Rnd 傳回的亂數值
小於 0	每次都是使用 *number* 當做亂數種子的相同結果。
大於 0	亂數序列中的下一個亂數值。
等於 0	最近一次產生過的亂數值。
省略	亂數序列中的下一個亂數值。

✦ 注意

Rnd 函數傳回的亂數值介於 0 和 1 之間，可等於 0，但不等於 1。*number* 的值會影響 **Rnd** 傳回亂數值的方法。

給定一個亂數種子後，便會產生一特定的亂數序列，因爲每呼叫一次 **Rnd** 函數，

它就會使用先前呼叫時所產生的亂數值當成新的亂數種子以產生新的亂數值。

在使用 **Rnd** 之前，最好先呼叫 **Randomize** 陳述式，但不要給任何引數，如此便會以作業系統的時間當作亂數種子來起始亂數產生器。

若想產生在某個範圍內(非 0 到 1)的亂數值，可使用下列公式：

```
Int((upperbound - lowerbound + 1) * Rnd + lowerbound)
```

上述公式中，*upperbound* 是亂數範圍的上限(整數)，而 *lowerbound* 則是亂數的下限(整數)。

附註 若想得到重覆的亂數序列，可以在呼叫 **Randomize** 之前先呼叫 **Rnd** 並且傳入一小於 0 的引數值。光是用同樣的亂數種子呼叫 **Randomize** 兩次的話，並不會得到兩次相同的亂數序列。

✦ Rnd 函數範例

本範例使用 Rnd 函數隨機產生一個 1 到 6 的整數。

```
Dim MyValue
MyValue = Int((6 * Rnd) + 1)    ' 產生 1 到 6 之間的亂數值。
```

Randomize 陳述式

初始化亂數產生器。

✦ 語法

`Randomize` `[`*`number`*`]`

選擇性引數 *`number`* 可以是一型態為 `Variant` 的值或任何數值運算式。

✦ 請注意

Randomize 使用 *number* 的值當成新的亂數種子來起始亂數產生器，若要得到一亂數值，則可呼叫 **Rnd** 函數。如果省略 *number*，則會以作業系統現在時間來當做新的亂數種子。

如果沒有呼叫 **Randomize** 來起始亂數產生器，**Rnd** 函數(沒有引數)則使用上次呼叫 **Rnd** 函數所得的亂數值當做新的亂數種子。

附註 若想得到重覆的亂數序列，可以在呼叫 **Randomize** 之前先呼叫 **Rnd** 並且傳入一小於 0 的引數值。光是用同樣的亂數種子呼叫 **Randomize** 兩次的話，並不會得到兩次相同的亂數序列。

✦ Randomize 陳述式範例

本範例使用 **Randomize** 陳述式來對亂數產生器做初始化的動作。範例中呼叫 **Randomize** 時並沒有傳入引數，亂數種子便會自動採用 **Timer** 函數的傳回值。

```
Dim MyValue
Randomize    ' 對亂數產生器做初始化的動作。
MyValue = Int((6 * Rnd) + 1)    ' 產生 1 到 6 之間的亂數值。
```

(二)亂數產生原則

➢ Multiplicative Congruential Method

✦ 取x_0為起始值，稱之為種子，以下式遞迴產生數列

$$x_n = a x_{n-1} \quad \text{modulo} \quad m$$

- a, m 為正整數。
- 產生之數列以 m 標準化，即可得[0, 1)之準亂數數列。

✦ a, m 選取的原則

- 對任何起始種子，產生的數列都需顯現出[0,1)間獨立的均等分配亂數數列的特性。
- 對任何起始種子，產生的數列在出現重複前，已有許多的數目。
- 在電腦系統中，可以有效的計算。
- m應選擇電腦系統可提供的最大質數，以 32 位元，$m = 2^{31} - 1$，$a = 7^5 - 1 = 16{,}870$。

➢ Mixed Congruential Method

✦ 取x_0為起始值，稱之為種子，以下式遞迴產生數列

$$x_n = (a x_{n-1} + c) \quad \text{modulo} \quad m$$

- 選擇 m 等同電腦的 word 長度，以加快計算。

✦ IBM Scientific Subroutine Package 參數選擇如下

- $a = 2^{16} + 3 = 65539, c = 0, m = 2^{31}$ 。

✦ IBM System 360 參數選擇如下

- $a = 7^5 = 16807, c = 0, m = 2^{31} - 1 = 2147483674$ 。

(三) 自行撰寫亂數函數

➢ Program 1

```
Const a As Long = 1229
Const c As Long = 1
Const m As Long = 2048
Dim nextvalue As Long

Function RandI() As Long
  nextvalue = (nextvalue * a) Mod m
  RandI = nextvalue
End Function

Sub SRand(ByVal seed As Long)
  nextvalue = seed
End Sub

Sub RandTest()
  Dim i As Integer
  Dim j As Integer

  SRand(3)
  For j = 1 To 10
    Worksheets("Sheet1").Cells(j, 1).Value = RandI()
  Next j
End Sub
```

<<Random: Sheet1>>

➢ Program 2

```
'Mixed Congruential Method
'Avoid Overflow Algorithm

Const m As Long = 100000000
Const m1 As Long = 10000
Const a As Long = 31415821
Const c As Long = 1
Public x As Long

Function Mult(ByVal p As Long, ByVal q As Long) As Long
  Dim p1, p0, q1, q0 As Long
  P1 = p \ m1
  P0 = p \ m1
  q1 = q \ m1
  q0 = q \ m1
  Mult = ((p0*q1+p1*q0) mod m1)*m1 + p0*q0) mod m
End Function

Function RandII() As Double
  x = (Mult(a, x) + c) mod m
  Random = x / m
End Function
```

<<Random: Sheet7>>

➢ Program 3

```
'A psuedo-random number generator for the system/360.
'IBM System J. 8:136-146; 1969.

Dim RandomX1 As Long

Function Uniform() As Double
  RandomX1 = Int(CDbl(RandomX1) * 16807 + 0.5) - _
    (Int(2147483647+0.5) * Fix(Int(CDbl(RandomX1) * _
    16807 + 0.5) / Int(2147483647 + 0.5)))
  Uniform = RandomX1 * 4.6566128752459E-10
End Function

Sub RandTest01()
  Dim i As Integer
  Dim j As Integer
  RandomX1 = 1
  For j = 1 To 10
    Worksheets("Sheet2").Cells(j, 1).Value=Uniform()
  Next j
End Sub
```

函數	傳回型態	***expression*** 引數範圍
CDbl	Double	負數從 -1.79769313486231E308 至-4.94065645841247E-324；正數從 4.94065645841247E-324 至 1.79769313486232E308。

<<Random: Sheet2>>

(四)亂數函數應用範例：積分的計算

✦ 令 g(x)為一 x 之函數，我們想要計算其積分

$$\theta = \int_0^1 g(x)dx$$

- 若 U 為(0, 1)上之均等分配，我們可將上式表示為

$$\theta = E[g(x)]$$

➢ Code

```
Sub MCIntegrate()
  Dim j As Integer
  Dim Count As Integer
  Dim Sum As Double
  Dim Area As Double

  Count = 1000
  Sum = 0
  For j = 1 To Count
    Sum = Sum + g(Rnd)
  Next j
  Area = Sum / count
End Sub
```

四、常態分配亂數的產生*

(一)Basic Model

- 產生$u_i \sim U[0,1]$, $i = 1,\ldots,12$，令

$$\varepsilon = \sum_{i=1}^{12} u_i - 6$$

  - ε 為平均數零，變異數一之變數。
  - ε 之峰態(kurtosis)為 0.15，常態分配之峰態為 3。
  - 由於$|\varepsilon| \leqq 6$，因此稀少事件被排除了。
  - 易於實作，不適於嚴謹的應用。

➢ Code

```
Function RndNormAvg() As Double
  Dim i As Double
  Dim sum As Double

  sum = 0
  For i = 1 To 12
    sum = sum + Rnd
  Next i
  RndNormAvg = sum - 6
End Function

Sub RandTest()
  Dim j As Integer
  For j = 1 To 10
    Worksheets("Sheet3").Cells(j, 1).Value =
        RndNormAvg()
  Next j
End Sub
```

<<Random: Sheet3>>

(二)Inverse Transform Method

➢ Proposition

Let U be a uniform (0,1) random variable. For any continuous distribution function F, the random variable X defined by

$$X = F^{-1}(U)$$

Has distribution F. [$F^{-1}(u)$ is defined to be the value of x such that F(x)=u.]

➢ Proof

Let F_X denote the distribution function of $X=F^{-1}(U)$. Then

$$F_X(x) = P\{X \le x\} = P\{F^{-1}(U) \le x\}$$

Since F(x) is a monotone increasing function.

$$F_X(x) = P\{F(F^{-1}(U)) \le F(x)\} = P\{U \le F(x)\} = F(x)$$

Since $F(F^{-1}(U))=U$.

$$F_X(x) = P\{U \le F(x)\}$$

Since U is a uniform (0, 1).

$$F_X(x) = P\{U \le F(x)\} = F(x)$$

➢ Excel Normal CDF Inverse Function

NORMSDIST

傳回標準常態累積分配函數。此分配的平均值是 0（零）和標準差 1。利用此函數可代替標準常態分配函數曲線之表格。

✦ 語法

NORMSDIST(z)

Z　是要分配的數值。

✦ 註解

- 如果 z 不是數值，則 NORMSDIST 傳回錯誤值 #VALUE!。
- 標準常態分配密度函數的公式是：

$$f(x) = \frac{1}{\sqrt{2\pi}} e^{-\frac{1}{2}x^2}$$

✦ 範例

NORMSDIST(1.333333) 等於 0.908789

NORMSINV

傳回平均數為 0 且標準差為 1 的標準常態累積分配函數的反函數。

✦ 語法

NORMSINV(probability)

Probability　是對應於常態分配的機率。

✦ 註解

- 如果 probability 不是數值，則 NORMSINV 傳回錯誤值 #VALUE!。
- 如果 probability < 0 或 probability > 1，則 NORMINV 傳回錯誤值 #NUM!。

NORMSINV 使用反覆運算的技巧。從給予的機率值開始，NORMSINV 反覆計算至誤差值在 ±3x10^-7 以內爲止。如果 NORMSINV 在反覆計算 100 次後仍不收斂，則傳回錯誤值 #N/A。

✦ 範例

NORMSINV(0.908789) 等於 1.3333

NORMDIST

根據指定之平均數和標準差，傳回其常態累積分配函數。本函數廣泛應用於包括假設檢定等統計學之應用。

✦ 語法

`NORMDIST(x,mean,standard_dev,cumulative)`

`X` 是要求分配之數值。

`Mean` 是此分配的算術平均數。

`Standard_dev` 是分配的標準差。

`Cumulative` 是決定函數形式的邏輯值。如果累積是 TRUE，則 NORMDIST 傳回累積分配函數；如果是 FALSE，則傳回機率密度函數。

✦ 註解

- 如果平均數或 standard_dev 不是數值，則 NORMDIST 傳回錯誤值 #VALUE!。
- 如果 standard_dev <= 0，則 NORMDIST 傳回錯誤值 #NUM!。
- 如果平均數等於 0 且 standard_dev = 1，則 NORMDIST 傳回標準常態分配 NORMSDIST。

✦ 範例

NORMDIST(42,40,1.5,TRUE) 等於 0.908789

➢ 使用 Excel CDF Inverse Function

```
Const Pi = 3.1415926
Dim RandomX1 As Double

Function Uniform() As Double
  RandomX1 = Int(CDbl(RandomX1) * 16807 + 0.5) -
    (Int(2147483647+0.5)*Fix(Int(CDbl(RandomX1)*
    16807 + 0.5) / Int(2147483647 + 0.5)))
  Uniform = RandomX1 * 4.6566128752459E-10
End Function

Function NormCDFInv() As Double
  Dim G1 As Double
  G1 = Application.WorksheetFunction.NormSInv
       (Uniform())
  NormCDFInv = G1
End Function

Sub RandTest06()
  Dim j As Integer
  RandomX1 = 12345678
  For j = 1 To 100
    Worksheets("Sheet6").Cells(j, 1).Value=
       NormCDFInv()
  Next j
End Sub
```

<<Random: Sheet6>>

(三)Rejection Method

➢ Methodology

Step1: Generate Y having density g.

Step2: Generate a random number U.

Step3: If $U \le \frac{f(Y)}{cg(Y)}$, set X=Y. Otherwise, go to step 1.

Let c be a constant such that $\frac{f(y)}{g(y)} \le c$, for all y.

➢ Theorem

(i) *The random variable generated by the rejection method has the density f.*

(ii) *The number of iterations of the algorithm that are needed is a geometric random variable with mean c.*

➢ Normal Random Generation

$$f(x) = \frac{2}{\sqrt{2\pi}} e^{-\frac{x^2}{2}} \quad 0 < x < \infty$$

$$g(x) = e^{-x} \quad 0 < x < \infty$$

$$\frac{f(x)}{g(x)} = \sqrt{2/\pi} e^{x - x^2/2}$$

$$c = Max \frac{f(x)}{g(x)} = \frac{f(1)}{g(1)} = \sqrt{2e/\pi}$$

$$\frac{f(x)}{cg(x)} = \exp\left\{x - \frac{x^2}{2} - \frac{1}{2}\right\} = \exp\left\{-\frac{(x-1)^2}{2}\right\}$$

Step1: Generate Y, an exponential r.v. with rate 1.

Step2: Generate a random number U.

Step3: If $U \le \exp\left\{-\frac{(Y-1)^2}{2}\right\}$, set X=Y. Otherwise, go to step 1.

In Step3, the value Y is accepted if U≦exp{-(Y-1)^2/2}, which is equivalent to –log(U)≧(Y-1)^2/2.

However, –log(U) is exponential with rate 1.

Step1: Generate independent exponential with rate 1, Y_1 and Y_2.

Step2: If $Y_2 \ge \frac{(Y_1-1)^2}{2}$, set X=$Y_1$. Otherwise, go to step 1.

The amount by which Y_2 exceeds (Y-1)^2/2 is also exponentially distributed with rate 1.

Step1: Generate Y_1, an exponential r.v. with rate 1.

Step2: Generate Y_2, an exponential r.v. with rate 1.

Step3: If $Y_2 - \frac{(Y_1 - 1)^2}{2} \geq 0$, set $Y = Y_2 - (Y_1 - 1)^2/2$ and go to Step4.
Otherwise, go to step 1.

Step4: Generate a r.v. number U and set

$$Z = \begin{cases} Y_1 & \text{if} U \leq \frac{1}{2} \\ -Y_1 & \text{if} U > \frac{1}{2} \end{cases}$$

Z is a normal r.v. with mean 0 and variance 1. Y is a exponential r.v. with rate 1. Z and Y are independent.

➢ Code

```
Dim RandomX1 As Double
Function Uniform() As Double
  RandomX1 = Int(CDbl(RandomX1) * 16807 + 0.5) - _
    (Int(2147483647+0.5) * Fix(Int(CDbl(RandomX1) * _
    16807 + 0.5) / Int(2147483647 + 0.5)))
  Uniform = RandomX1 * 4.6566128752459E-10
End Function

Function StdNorm() As Double
  Dim y1 As Double
  Dim y2 As Double
  Do While y2 <= ((1# - y1) ^ 2) / 2
    y1 = -Log(Uniform)
    y2 = -Log(Uniform)
  Loop
  StdNorm = -y1
  If Uniform < 0.5 Then StdNorm = y1
End Function

Sub RandTest04()
  Dim j As Integer
  RandomX1 = 1
  For j = 1 To 10
    Worksheets("Sheet4").Cells(j, 1).Value=StdNorm()
  Next j
End Sub
```

<<Random: Sheet4>>

(四)Polar Method

➢ Box-Muller Transformation

Let X and Y be independent unit normal random variables and let R and θ denote the polar coordinates of the vector (X,Y). That is

$$R^2 = X^2 + Y^2, \quad \tan\theta = \frac{Y}{X}$$

We can proof that R^2 and θ are independent, with R^2 being exponential with mean 2 and θ being uniform distributed over (0, 2π).

Step1: Generate r.v. numbers U_1 and U_2.

Step2: $R^2=-2\log(U_1)$(and thus R^2 is exponential with mean 2). Set $\theta=2\pi U_2$(and thus θ is uniform between 0 and 2π).

Step3: Now let

$$X = R\cos(\theta) = \sqrt{-2\log U_1}\cos(2\pi U_2)$$

$$Y = R\sin(\theta) = \sqrt{-2\log U_1}\sin(2\pi U_2)$$

This method is known as the Box-Muller transformation. This method is time consuming.

➢ Code

```
Function NormBox() As Double
  Dim U1 As Double
  Dim U2 As Double
  Dim G1 As Double
  Dim G2 As Double

  Do Until U1 > 0
    U1 = Uniform()
  Loop

  U2 = Uniform()
  U1 = Log(U1)
  U2 = Pi * U2
  U1 = Sqr(-U1 - U1)
  U2 = U2 + U2
  G1 = U1 * Cos(U2)
  G2 = U1 * Sin(U2)

  NormBox = G1
End Function
```

➢ Polar Method

Step1: Generate r.v. numbers U_1 and U_2.

Step2: Set $V_1=2U_1$, $V_2=2U_2-1$, $S=V_1^2+V_2^2$.

Step3: If S>1 return to Step1.

Step4: Return the independent unit normals

$$X = \sqrt{\frac{-2\log S}{S}}V_1$$

$$Y = \sqrt{\frac{-2\log S}{S}}V_2$$

五、標的資產價格程序的模擬*

(一)幾何布朗運動程序

➢ Black-Scholes 模型下股票價格變化的程序

✦ 金融資產價格的假設是它遵行著所謂的擴散程序(diffusion process)

$$\Delta S / S = \mu \cdot \Delta t + \sigma \cdot \Delta Z \quad \text{.......................}(5.1.1)$$

- $\dfrac{\Delta S}{S} = \dfrac{S_{t+1} - S_t}{S_t}$ ＝金融資產的報酬率，
- Δt ＝單位時間，
- μ ＝單位時間內預期金融資產的報酬率，
- σ ＝單位時間內預期金融資產的標準差。

✦ Z＝一隨機變數，為平均數為零，變異數為 t 之常態分配，$Z \sim \Phi(0,t)$ 。

- Z 稱之為韋恩程序。
- ΔZ ＝單位時間內，Z 的變動量，為一期望值為零，變異數為 Δt 之常態分配，$\Delta Z \sim \Phi(0,\Delta t)$ 。

- 第一項 $\mu \cdot \Delta t$ 代表趨勢的成份，它指出金融資產價格固定的增加。
  - 這一增加過程為一確定的(deterministic)過程，且固定不變，因此表達了一個趨勢的成份。

$$\frac{S_{t+1} - S_t}{S_t} = \mu \cdot \Delta t$$

$$S_{t+1} = S_t + S_t \times \mu\Delta t = S_t(1 + \mu\Delta t)$$

$$S_{t+2} = S_{t+1} + S_{t+1} \times \mu\Delta t = S_{t+1}(1 + \mu\Delta t) = S_t(1 + \mu\Delta t)^2$$

$$\ldots$$

$$S_{t+n} = S_{t+n-1} + S_{t+n-1} \times \mu\Delta t = S_{t+n-1}(1 + \mu\Delta t) = S_t(1 + \mu\Delta t)^n$$

- 第二項為一干擾的隨機項(stochastic term)。它代表社會行為中不確定的因素，對匯率的隨機干擾。
  - 由於經濟行為為一社會行為，它受到市場上許多突發的因素影響，因此隨機干擾的現象是無法避免的。
  - 我們利用隨機變數 ΔZ 來描述這一現象，而 σ 的用途則是作為刻度的調整。

$$S_{t+1} = S_t(1 + \mu\Delta t + \sigma\Delta Z_t)$$

$$S_{t+2} = S_{t+1}(1 + \mu\Delta t + \sigma\Delta Z_{t+1}) = S_t(1 + \mu\Delta t + \sigma\Delta Z_{t+1})(1 + \mu\Delta t + \sigma\Delta Z_t)$$

(二)資產價格路徑的模擬

➢ 近似作法

- 由擴散程序(diffusion process)可知

$dS_t = \mu \bullet S_t \bullet dt + \sigma \bullet S_t \bullet dZ_t$ ....(5.2.1)

- 以離散之時間間距，Δt ，近似連續時間變動，dt ，

$S(t+\Delta t) - S(t) = \mu \bullet S(t) \bullet \Delta t + \sigma \bullet S(t) \bullet \varepsilon \sqrt{\Delta t}$

- ε 為標準常態分配之隨機亂數。
- Iterative 方式產生整條模擬路徑。

- 離散之時間間距Δt很小時，近似才成立。

➢ Code Example

(三)相關性亂數的產生

- 模擬之標的變數可能不只一個，且變數間有相關性

$$\frac{dS_1}{S_1} = \mu_1 dt + \sigma_1 dZ_1 \text{.....................} (5.3.1)$$

$$\frac{dS_2}{S_2} = \mu_2 dt + \sigma_2 dZ_2 \text{..................} (5.3.2)$$

  - 兩變數報酬率之相關性為 ρ

- Φ_1、Φ_2 為獨立之常態分配隨機變數 $\Phi_i \sim N(0, dt)$

$$dZ_1 = \Phi_1 \text{..} (5.3.3)$$

$$dZ_2 = \sqrt{(1-\rho^2)}\Phi_2 + \rho\Phi_1 \text{.............................} (5.3.4)$$

- 矩陣表示為

$$\begin{bmatrix} dZ_1 \\ dZ_2 \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ \rho & \sqrt{1-\rho^2} \end{bmatrix} \begin{bmatrix} \Phi_1 \\ \Phi_2 \end{bmatrix} \text{.............................} (5.3.5)$$

$$\Lambda = \begin{bmatrix} 1 & 0 \\ \rho & \sqrt{1-\rho^2} \end{bmatrix} \text{，} \Lambda \times \Lambda' = \begin{bmatrix} 1 & \rho \\ \rho & 1 \end{bmatrix}$$

- Code Example

- 考慮 n 種具相關性資產的情況
  - i, j 資產間相關性為 ρ_{ij}
  - Φ_i 為獨立之標準常態分配變數，$1 \le i \le n$，要求

$$dZ_i = \sum_{k=1}^{i} \alpha_{ik} \Phi_k$$

$$\sum_{k=1}^{i} \alpha_{ik}^2 = 1 \text{ ， } 1 \le j \le i$$

$$\sum_{k=1}^{j} \alpha_{ik} \alpha_{jk} = \rho_{ij} \text{ ， } j < i$$

- 上述步驟即為 Cholesky Decomposition。

$$\Lambda = \lfloor \alpha_{ij} \rfloor$$

$$\Lambda \times \Lambda' = \begin{bmatrix} 1 & \rho_{12} & \cdots & \rho_{1n} \\ \rho_{21} & 1 & \cdots & \rho_{2n} \\ \cdots & \cdots & \cdots & \cdots \\ \rho_{n1} & \rho_{n2} & \cdots & 1 \end{bmatrix}$$

➢ Cholesky Decomposition Algorithm

```
'n : Dimension of matrix
'IM: Input Matrix
'OM, OMt: Output Matrix & it's transpose
Function Cholesky(d As Integer, IM() As Double, _
  OM() As Double, OMt() As Double) As Boolean
  Dim n, i, j, k As Integer
  Dim temp1, temp2 As Double
  n = d
  For i = 1 To n
    For j = 1 To n
      OM(i, j) = 0
    Next j
  Next i
  For j = 1 To n
    temp1 = 0
    For k = 1 To j - 1
      temp1 = temp1 + OM(j, k) * OM(j, k)
    Next k
    OM(j, j) = Sqr(IM(j, j) - temp1)
    For i = j + 1 To n
      temp2 = 0
      For k = 1 To i - 1
        temp2 = temp2 + OM(j, k) * OM(i, k)
      Next k
      OM(i, j) = (IM(j, i) - temp2) / OM(j, j)
    Next i
  Next j
```

```
  For i = 1 To n
    For j = 1 To n
      OMt(j, i) = OM(i, j)
    Next j
  Next i

  Cholesky = True
End Function
```