RMS 4001 Tutorial 5

1. Introduction to Matrices in Excel

(i) Transposing a Matrix

\[
\begin{bmatrix}
  1 & 2 \\
  2 & 4 \\
\end{bmatrix}
\]

\[x=[2 \ 4]'\]

Example: We want to take the transpose of the 2 by 1 vector x in cells A1:A2 and store it in the cells B1:C1

Steps:  
1. Select the cell range B1:C1
2. Key in the formula: =TRANSPOSE(A1:A2)
3. Ctrl+Shift+Enter

(ii) Multiplying Matrix

\[
\begin{bmatrix}
  A & B & C & D & E & F & G & H & I & J \\
  1 & 12 & 4 & 16 & 19 & -2 \\
  2 & 3 & 13 & 5 & 12 & 14 \\
\end{bmatrix}
\]

\[
C = \begin{bmatrix}
12 & 4 \\
3 & 13 \\
\end{bmatrix} \quad D = \begin{bmatrix}
16 & 19 & -2 \\
5 & 12 & 14 \\
\end{bmatrix}
\]

Example: We want to get the product of CD and store the matrix in the cells H1:J2

Steps:  
1. Select the cell range H1:J2
2. Key in the formula: =MMULT(A1:B2,D1:F2)
3. Ctrl+Shift+Enter

(iii) Matrix Inversion

\[
\begin{bmatrix}
  A & B & C & D & E & F & G \\
  1 & -3 & 2 & 7 \\
  2 & 2 & 20 & 19 \\
  3 & 7 & 9 & 21 \\
\end{bmatrix}
\]
Example: We want to get the inverse of A and store it in the cells E1:G3

Steps:  
(1) Select the cell range E1:G3  
(2) Key in the formula: MINVERSE(A1:C3)  
(3) Ctrl+Shift+Enter

2. Simulation of Stock Price Paths in VBA:

Simulation of stock price under the risk neutral world, assume the stock price follow the following dynamic \( dS_t = rS_t dt + \sigma S_t dW_t \)

<table>
<thead>
<tr>
<th>Sub samplepath()</th>
</tr>
</thead>
<tbody>
<tr>
<td>Dim deltat As Double  'length of each time step</td>
</tr>
<tr>
<td>Dim T As Double      'the time interval of the whole path</td>
</tr>
<tr>
<td>Dim n As Integer     'number of time steps</td>
</tr>
<tr>
<td>Dim S0 As Double     'initial stock price</td>
</tr>
<tr>
<td>Dim S() As Double    'the array store up the generated stock price</td>
</tr>
<tr>
<td>Dim r As Double      'drift of the process</td>
</tr>
<tr>
<td>Dim sigma As Double  'volatility of the process</td>
</tr>
<tr>
<td>Dim i As Integer</td>
</tr>
<tr>
<td>Dim timeline() As Double</td>
</tr>
<tr>
<td>Dim path As Integer</td>
</tr>
<tr>
<td>'Input parameters from worksheet</td>
</tr>
<tr>
<td>deltat = Cells(2, 2).Value</td>
</tr>
<tr>
<td>T = Cells(3, 2).Value</td>
</tr>
<tr>
<td>n = Round(T / deltat)</td>
</tr>
<tr>
<td>S0 = Cells(4, 2).Value</td>
</tr>
<tr>
<td>r = Cells(5, 2).Value</td>
</tr>
<tr>
<td>sigma = Cells(6, 2).Value</td>
</tr>
<tr>
<td>path = Cells(7, 2).Value</td>
</tr>
<tr>
<td>'clear the previous contents</td>
</tr>
<tr>
<td>Rows(&quot;8:65536&quot;).Select</td>
</tr>
<tr>
<td>Selection.ClearContents</td>
</tr>
<tr>
<td>ReDim S(0 To n, 1 To path)  'the dynamic array</td>
</tr>
<tr>
<td>ReDim timeline(0 To n)       'the dynamic array</td>
</tr>
<tr>
<td>For j = 1 To path</td>
</tr>
<tr>
<td>S(0, j) = S0</td>
</tr>
<tr>
<td>Cells(11, j + 1) = S(0, j)</td>
</tr>
<tr>
<td>For i = 1 To n</td>
</tr>
<tr>
<td>S(i, j) = S(i - 1, j) + S(i - 1, j) * r * deltat + S(i - 1, j) * sigma * Sqr(deltat) * _</td>
</tr>
<tr>
<td>Application.WorksheetFunction.NormSInv(Rnd())</td>
</tr>
<tr>
<td>Cells((11 + i), (1 + j)) = S(i, j)</td>
</tr>
<tr>
<td>Next i</td>
</tr>
<tr>
<td>Next j</td>
</tr>
<tr>
<td>timeline(0) = 0</td>
</tr>
<tr>
<td>Cells(11, 1) = timeline(0)</td>
</tr>
<tr>
<td>For i = 1 To n</td>
</tr>
<tr>
<td>timeline(i) = timeline(i - 1) + deltat</td>
</tr>
<tr>
<td>Cells((11 + i), 1) = timeline(i)</td>
</tr>
<tr>
<td>Next i</td>
</tr>
<tr>
<td>End Sub</td>
</tr>
</tbody>
</table>
3. Pricing of European Call Option in VBA by using MC Simulation

Assume the same stock price dynamic \( dS_t = rS_t dt + \sigma S_t dW_t \)

Call Option Price = \( e^{-rT} \frac{1}{N} \sum_{i=1}^{N} \max(S_{T_i} - K, 0) \)

```vba
Sub samplepath()
    Dim deltat As Double 'length of each time step
    Dim T As Double 'the time interval of the whole path
    Dim n As Integer 'number of time steps
    Dim S0 As Double 'initial stock price
    Dim S() As Double 'the array store up the generated stock price
    Dim r As Double 'drift of the process
    Dim sigma As Double 'volatility of the process
    Dim i As Integer
    Dim path As Integer
    Dim K As Double
    Dim payoff() As Double
    Dim calloption As Double

    'Input parameters from worksheet
    deltat = Cells(2, 2).Value
    T = Cells(3, 2).Value
    n = Round(T / deltat)
    S0 = Cells(4, 2).Value
    r = Cells(5, 2).Value
    sigma = Cells(6, 2).Value
    path = Cells(7, 2).Value
    K = Cells(8, 2).Value

    'clear the previous contents
    Rows("9:65530").Select
    Selection.ClearContents

    ReDim S(0 To n, 1 To path) 'the dynamic array
    ReDim payoff(0 To path)
    payoff(0) = 0

    For j = 1 To path
        S(0, j) = S0
        For i = 1 To n
            S(i, j) = S(i - 1, j) + S(i - 1, j) * r * deltat + S(i - 1, j) * sigma * Sqr(deltat) * _
            Application.WorksheetFunction.NormSInv(Rnd())
        Next i
        payoff(j) = payoff(j - 1) + Application.WorksheetFunction.Max((S(n, j) - K), 0)
    Next j

    calloption = Exp(-r * T) * payoff(path) / path
    Cells(9, 2) = calloption
End Sub
```
4. Appendix

Loops:

(i) **Do While Statement** (top-checking loop family)

\[
\text{Do While (condition)} \quad \text{‘if condition is true run the expression and loop}
\text{\{} \text{expression}\} \quad \text{‘until the condition is false}
\text{Loop}
\]

(ii) **Do …… Loop While Statement** (bottom-checking loop family)

\[
\text{Do} \quad \text{‘do the expression}
\text{(expression)} \quad \text{‘loop the expression again if condition is true}
\text{Loop While (condition)}
\]

(iii) **Do Until Statement** (top-checking loop family)

\[
\text{Do Until(condition)} \quad \text{‘if condition is false run the expression and loop}
\text{(expression)} \quad \text{‘until the condition is true}
\text{Loop}
\]

(iv) **Do Loop Until Statement** (bottom-checking loop family)

\[
\text{Do} \quad \text{‘do and loop the expression}
\text{(expression)} \quad \text{‘until the condition is false}
\text{Loop Until (condition)}
\]

(v) **The While Statement** (top-checking loop family)

\[
\text{While (condition)} \quad \text{‘if condition is true, execute the expression}
\text{(expression)} \quad \text{‘before Wend, then check back the condition.}
\text{Wend} \quad \text{‘if the condition is true, repeated again}
\text{‘if false, run the command after Wend}
\]

(vi) **The For loop**

\[
\text{For i=1 To N Step (numeric)} \quad \text{‘run the expression (i) from 1 to N}
\text{expression(i)} \quad \text{‘with step that you can input by}
\text{Next} \quad \text{‘numerical value(e.g. 2 or –1).}
\]

Conditional Execution:

(i) **Select Case**

\[
\text{Select Case Parameter}
\text{Case 1 (expression 1)}
\text{Case 2 (expression 2)}
\text{………}
\text{Case N (expression N)}
\text{Case Else}
\text{(expression N+1)}
\text{End Select}
\]