Chapter 2. MathScript Node and Formula Node

2.1 What is MathScript

MathScript is math-oriented, text-based computing language to address tasks mathematic calculation:

- Most suitable for Mathematic calculation.
- Matrix based data type system.
- Can create a custom function.
- Same Syntax with Matlab.

2.2 Accessing the MathScript Window

Getting Start – Tools -- MathScript Window
2.3 MathScript Windows:

- Command window: for command line input
- Output Window: shows the resulting output
- Script Editor window: for loading, saving, editing, and running
- Variables window: shows variables, dimensions, and type
- Command History window: shows historical commands and scripts.

Example 1

Input \( a=1+2 \) in the Script Editor window, and run this code. Check each window.
2.4 View Data in a Variety Formats

2.4.1 View in Variable window

Input:
\[ t = [0: 0.1: 10]; \]
\[ y = \sin(t); \]

View \( y \) in the Graph or XY Graph format.
View \( y \) or \( t \) in the Numeric format
2.4.2 You can also use MathScript “Plot”

Input:
\[ t=[0: 0.1: 10]; \]
\[ y=\sin(t); \]
\[ \text{Plot}(t, y); \]

Input:
\[ t=[0: 0.1: 10]; \]
\[ X=\cos(t); \]
\[ y=\sin(t); \]
\[ \text{Plot}(x, y); \]
2.5 MathScript Help

In the Command Window input:

- help classes: provide a list of all MathScript classes of function.
- help function: provide help for a particular MathScript function.
A Summary of MathScript Functions

- Plot (2D and 3D)
- Digital Signal Processing (DSP)
- Curve Fitting & Interpolation
- Ordinary Differential Equation (ODE) Solves
- Polynomial Operations
- Linear Algebra
- Matrix Operations
- Vector Operations
- Probability and Statistics
- Optimization
- Basic Functions
- Advanced Functions
- Trigonometric Functions
- Boolean and Bit operations
- Dada Acquisition
- Others
2.6 Syntax

1. Scalar Operation

```>> 16+3
>> x=16+3
>> x=16+3;
>> display(x);

>> 16-3
>>16/3
>>16*3```
2. Creating Matrices and Vectors

Vector:
>> A=[1;2;3]

>> B=[1 -2 7]

>> B=[1,-2,7]

Matrix:
>> C=[-1 2 0; 4 10 -2; 1 0 6]

>> C=[-1,2,0;4,10,0;1,0,6]
3. Creating Vector

```
>> t=1:10;
>> t=1:0.5:10
```

4. Access individual elements of a vector or matrix

```
>> C=[-1 2 0;4 10 -2;1 0 6]
>> C(2,3)
>> F=C(2,3)
>> C(2,:)
>> C(:,3)
>> C([2 3], [1 2])
```
5. Calling Functions

>> help linspace

>>G=linspace(1,10,13)

>>Help abs
>>A=-3
>>B=abs(A)
6. Assigning data types to variables

```python
>>a=sin(3*pi/2)
>>a='temperature'
```

7. Using complex numbers

```python
>>a=2+3j
>>a=2+3i
>>b=3+4i
>>c=a+b
```
8. Matrix operations

```matlab
>> K = [-1 2 0; 4 10 -2; 1 0 6]
>> L = [1 0 0; 0 1 0; 0 0 1]
>> K + L
>> K * L
>> L .* K
```

```matlab
>> a = [1 2; 3 4]
>> b = a^-1
>> c = a * b
```
9. Logical Express

```matlab
>>a=2
>>b=3
>>a==b
>>a~_=b
>>a=2
>>b=2
>>a==b
>>a~_=b
```

10. Adding comments

```matlab
>>% In this MathScript, the inputs are x and y
>>% and the output is Z
>>z=x+y  % z is the addition of x and y
```
11. Control flow structure (For Script)

Case-Switch Syntax:
switch expression
case expression
statement, ... , statement
...
otherwise
statement, ... , statement
end

Script Example:
color = 'green';
switch color
case 'green'
disp('color is green');
case 'red'
disp('color is red');
otherwise
disp('color is neither green nor red')
end
For Loop Syntax:
for variable = expression
    statement1,
    ....
    ....
    statement,
end

Script Example:
A = 1
for X = 1:1:10
    A = A+1
end
If-Else Syntax:
for variable = expression
    statement1,
    ....
    ....
    statementn,
end

Script Example
A = 1
for X = 1:1:10
    A = A+1
end
If-Else Syntax:
if expression
    statement, ... , statement
elseif expression
    statement, ... , statement
else
    statement, ... , statement
end

Script Example:
b = 10;
if b == 1
    c = 3
else
    c = 4
end
While Loop Syntax:
while expression
  statement1
  ....
  ....
  statementn
end

Script Example:
A = 1
X = 1
while X<10,
  A = A+1,
  X = X+1,
end
2.7 MathScript Node (Formula Node)

MathScript Node is scripts node that can be used in VI as a node function. It is a text-base code.

MathScript can be found at:
Programming>>Structure palette
Or Mathematics>>Scripts & Formulas.

Figure 1. A MathScript Node.
Procedure to create a MathScript Node:

1) Place a MathScript Node on the block diagram
2) Right click the board to add input or output
3) Add the names of the variables.
Assignment 1

Write the following MathScript Node to generate, plot, and analyze a sequence of a random number.

```plaintext
x=rand(n,1);
y=mean(x);
figure(1);
plot(x)
title('Random number plot')
```
Assignment 2

For \( A = \begin{bmatrix} 1 & 2 & 3 \\ 4 & 2 & 5 \\ 1 & 2 & 7 \end{bmatrix}, \quad B = \begin{bmatrix} -1 & 0 & 2 \\ 3 & 1 & 4 \\ 2 & 1 & 3 \end{bmatrix} \)

Write a Script code to calculate Matrix C:
\[ C = A \cdot B, \quad E = A \cdot C, \quad \text{if } d = 1; \]
\[ C = A^{-1}, \quad E = A \cdot C, \quad \text{if } d \text{ is not equal to 1}; \]

d is the node input; and C is the node output.
Assignment 3 (MathScript)

Open a new VI and place a MathScript Node on the block diagram. Create a numeric input on the MathScript Node frame and name it w. Generate the sine wave \( y = \sin(w \cdot t) \) within the node, where \( t \) starting at \( t=0 \) and ending at \( t=10 \). Replace the frequency with the variable \( w \) on the input controlled from a front panel knob. Create 2 output on the MathScript Node frame name t and y, where t is the time history and y is the sine function associated with t. Plot the sine wave using an XY Graph.
Assignment 4 (Formula Node)

Open a new VI and place a Formula Node on the block diagram. Create a numeric input on the Formula Node frame and name it w. Generate the sine wave $y = \sin(w \cdot t)$ within the node, where $t$ starting at $t=0$ and ending at $t=10$. Replace the frequency with the variable $w$ on the input controlled from a front panel knob. Create 2 output on the Formula Node frame name t and y, where t is the time history and y is the sine function associated with t. Plot the sine wave using an XY Graph.
Assignment 5

Construct a VI that solves the quadratic formula to find the real roots of the equation:

\[ ax^2 + bx + c = 0 \]

Where \( x \) is the variable. \( a, b \) and \( c \) are constant. The real solution of the equation is given by

\[
x_1 = (-b + \sqrt{b^2 - 4ac}) / (2a) \quad \text{and} \quad x_1 = (-b - \sqrt{b^2 - 4ac}) / (2a)
\]

Construct a Mathscript Formula Node to compute the above roots. There are should 2 outputs for the 2 roots, and 3 inputs for the constant \( a, b \) and \( c \). Once you finish the code, using this Mathscript Formula Node to create a MathScript.