第五章 空間、狀態、和時間維度的響應視覺化

系統響應數值計算時，多個維度(Dimension)之間應互相垂直，也就是維度之間並沒有關係互為獨立，維度內包含多個自由度(Degree Of Freedon)。空間維度的自由度，其個數以 m 表示自由度，且 m >= 1 表示至少有 1 個空間自由度，狀態維度的自由度以 r 表示，且 r >= 0 ，若 r = 0 表示微分方程式是零階(Order)，另時間維度上，僅有一個自由度，就是時間自由度(t)，且 t 是連續性。

由空間和狀態維度所構成的平面，該平面共有 m x (r + 1) 個離散節點，而時間維度是垂直於該平面 m x (r + 1) 個平面離散的節點(Node)上，且僅有一個連續的時間自由度(t)。整個系統的響應，就是如上所示，在時間軸上產生的數據。

由上述多階矩陣微分方程式(Higher-Order Matrix Differential Equation)，可以建構系統矩陣 A = A(t)，即整個系統以矩陣 A 表示，可求得齊次解，再加上特別解，就是一般解。使用 SMS 精銳矩陣計算求解器，很容易求出結果，即不同節點、不同時間內的響應數據，如某個時間的變位、速度、和加速度等等的關係，但因爲輸出的資料非常的多，很難看出整體的輸出變化，故需要將這些資料視覺化，使我們能夠瞭解整體系統的變化狀況，但當須要確切的數值時，只有依據時間順序去搜尋輸出確切的結果。SMS 的輸出結果，使用兩種視覺化工具，一是使用 Excel 繪圖工具，二是使用 Python 的 Matplotlib 的套件。

5.1 系統響應與模態的關係 ：

模態 (Mode Shap)其實是系統矩陣 A 的特徵向量(Eigen Vector)，也就是線性代數或是矩陣計算理論中的特徵向量。動態系統的模態不是實數而是複數，尤其系統矩陣不對稱時，一般都是複數不是實數，故不能繪製圖表標示，更不能將實數與虛數分開表示，複數實際上就是一個整體的數，很多論文、期刊常常將虛數部分忽略，這並不是合理的求解方式。

我們在實體模型中作實驗測試分析，若能將實體轉爲數學模擬系統，我們就可以進一步作數值模擬測試分析，包含系統的數學模型的建立，程式撰寫、測試和執行，瞭解系統的響應輸出。若無法建構系統的數學模型，只對輸入和輸出作兩者的關係作分析，或是僅討論輸出模型分析，這種分析法有很多的缺陷待解決，惟有建構系統模型才是重點。

從古至今，複數都是讓人感到最困惑的數值，無論是應用數學的動態系統、機械的控制系統、電機的訊號與系統、土木結構的結構動力學、模態分析、量子

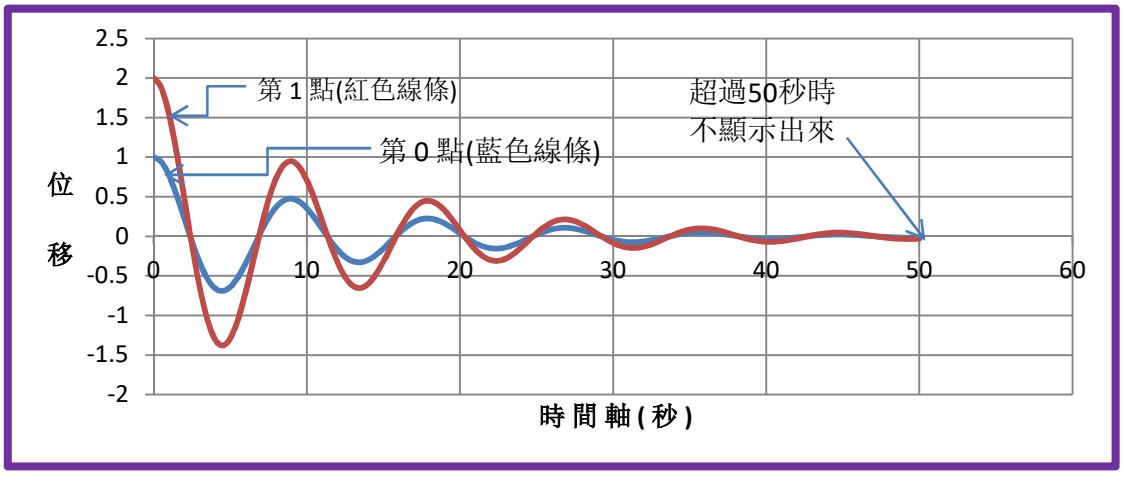
力學等等，都使用到複數。一切的問題在於系統矩陣的對角線化，也就是求取系統矩陣的複數特徵值和複數特徵向量。應用數學、線性代數、和矩陣計算的書籍和論文，都有提到特徵值與特徵向量，但是如何進一步，使用特徵值和特徵向量等系統特徵，求得系統之響應，此部分則完全欠缺，精銳矩陣求解器則能完全補足這方面的不足，茲說明如下。

特徵值和特徵向量是互為共軛(Self-Adjoint)關係，缺一不可，其預設值是複數，複數包含實數和虛數部分，以 C#程式語言的觀點，實數是複數的 subset，兩者之間可以互換，即實數可以隱性(Implicit)轉為複數，複數可以顯性(Explicit)轉為實數，若是無法轉換，則產生錯誤(Exception)，也就是虛數部分應為零，才可轉為實數，即型態轉換(Type Convertion)，在真實世界的數值是實數(Real Number)，實數才可量測，而複數僅是計算的中間過程，最後計算的結果都應是實數，不可能是複數，所有的實例計算中，精銳矩陣求解器都是依據這種計算理論。

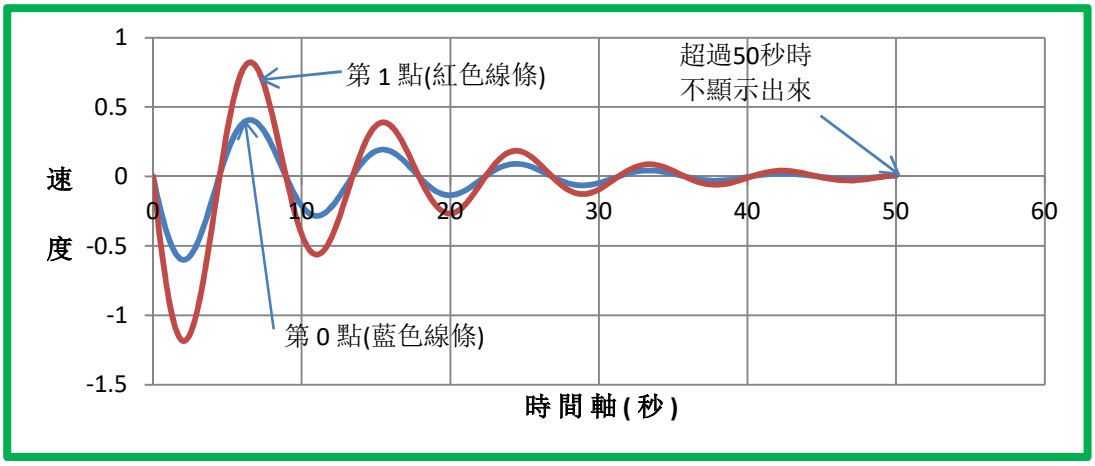
基於以上所述，雖然複數是實數的延伸，但實數就是真實的數值，我們不再對於希爾伯特空間(Hilbert Space)或是量子力學的厄米特運算子(Hermitian)感到迷惑了。

5.2 使用 Excel 繪圖工具 ：

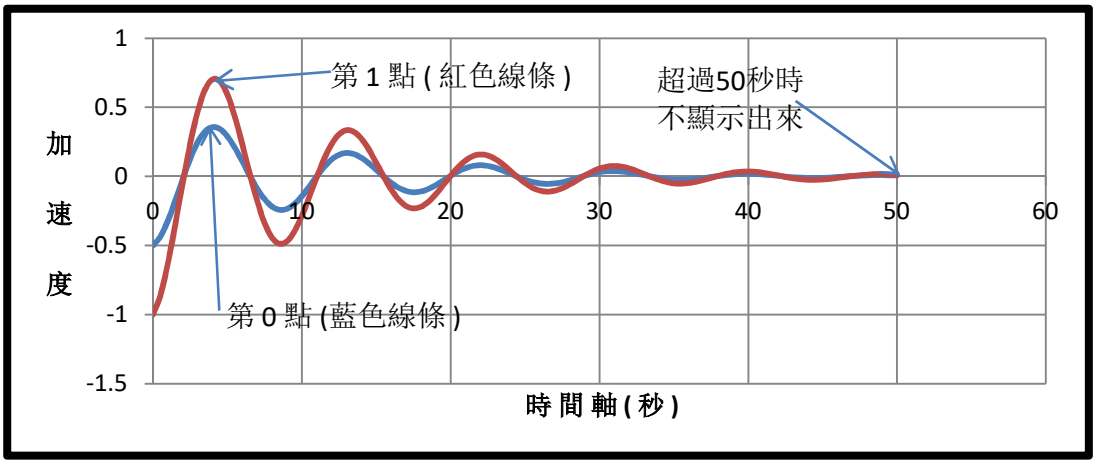
將 ConsoleApp6k 的 C#程式執行的結果，共有 3 個部分，即變位、速度、和加速度的 txt 文字檔，將其結果輸入 Excel 中，畫出如下 ：



位 移 - 時 間 關 係 圖

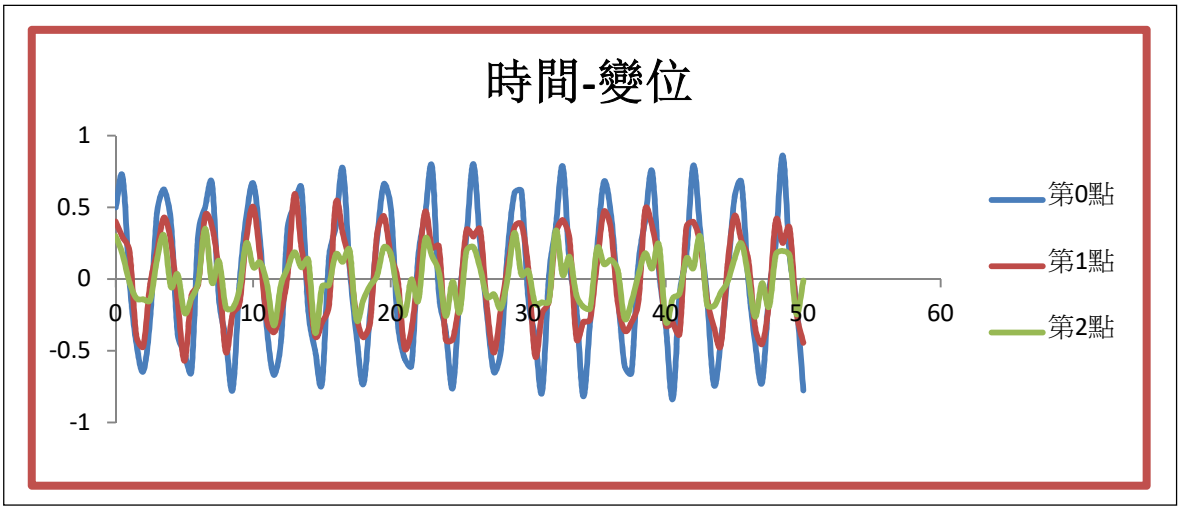


速 度 - 時 間 關 係 圖

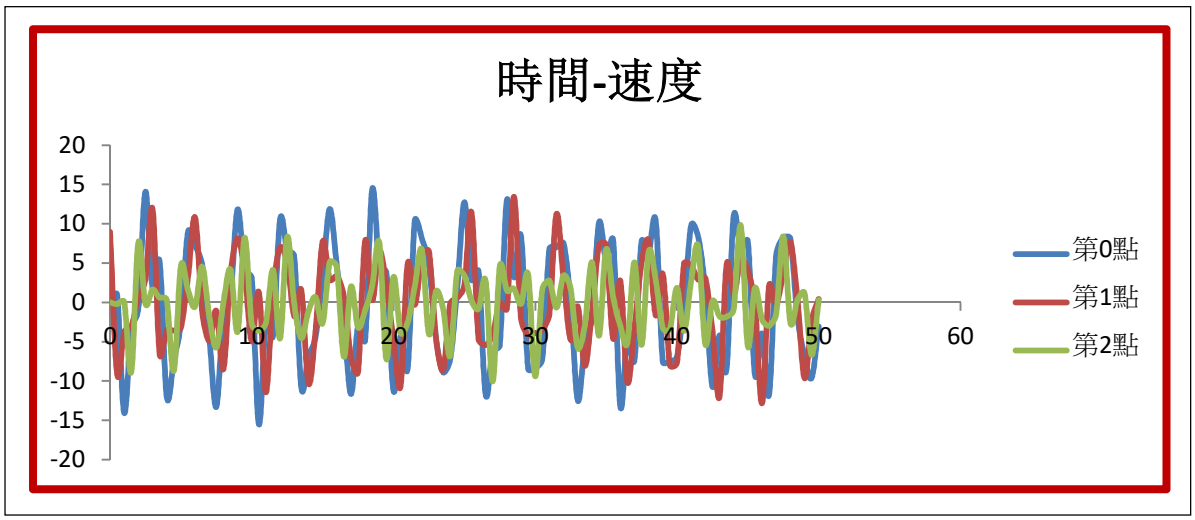


加 速 度 － 時 間 關 係 圖

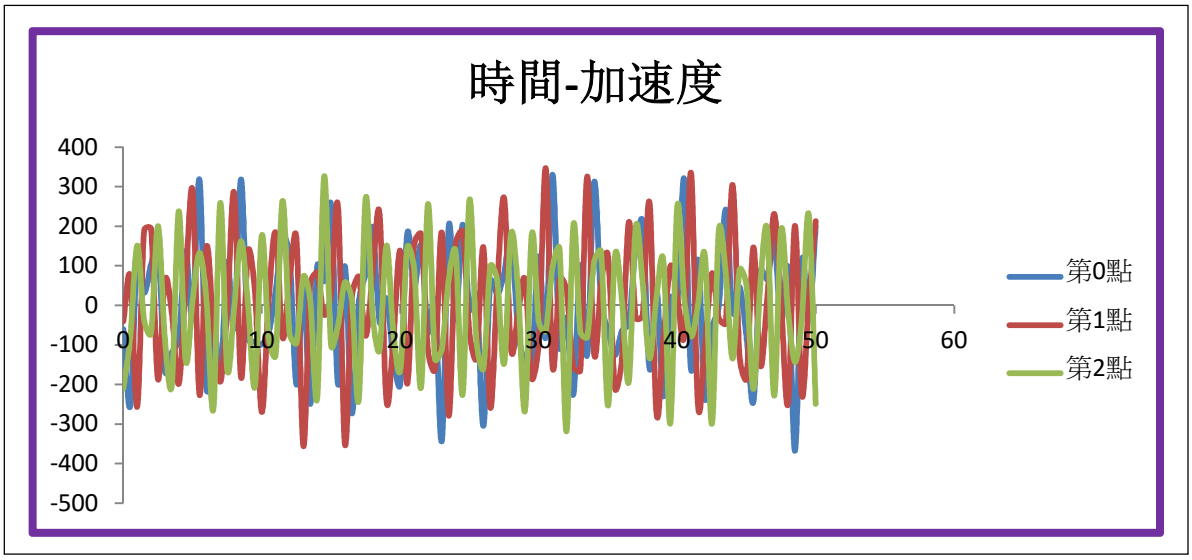
另外再將 ConsoleApp38 的 C#程式碼，執行的結果，共有 3 個文字檔，即變位、速度、和加速度，匯入 Excel 並畫出如下 ：



時間−變位關係圖



時間-速度關係圖



時間-加速度關係圖

5.3 使用 Pyton Matplotlib 繪圖工具套件 :

無論是程式 ConsoleApp6K 或是 ConsoleApp39 的輸出，都是直立式的方式，適合在電腦螢幕上觀看，若要將資料輸入 Python 的 Matpotlib 的套件上繪圖，則必須將資料轉換為橫式，單獨矩陣的輸出。

以下是 Visual Studio(不是 Visual Studio Code)的執行環境下，ConsoleApp38 程式，第 0 點、第 1 點、和第 2 點的變位、速度、和加速度的輸出視覺化，其中 Python 程式碼如下 :

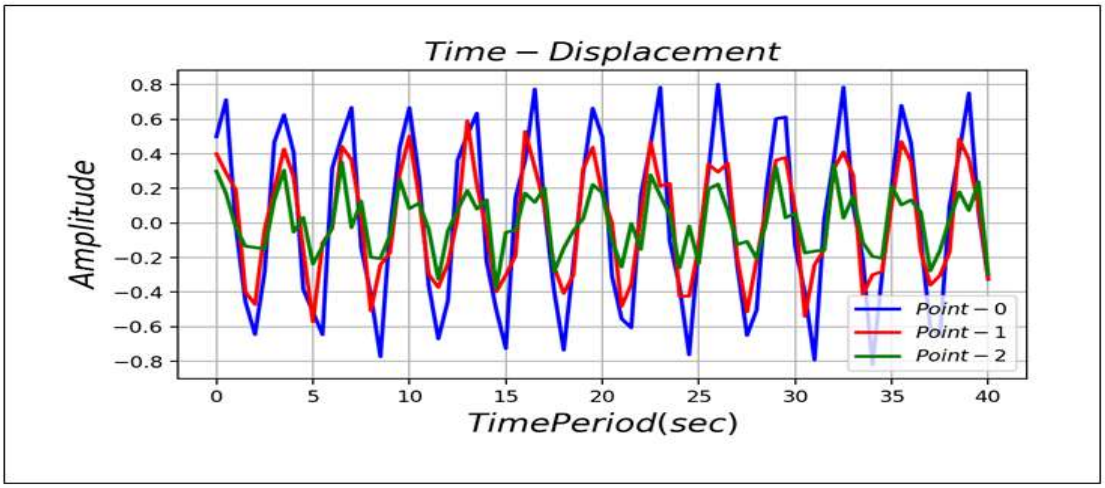
```
import numpy as np
import matplotlib.pyplot as plt

x =  np.arange(0, 40.5, 0.5)  # total 81 Time Steps
y0 = [ 0.5000,  ...  ]  # point 0 Data
y1 = [ 0.4000,  ...  ]  # point 1 Data
y2 = [ 0.3000,  ...  ]  # point 2 Data
plt.figure(figsize = (8, 4))
plt.subplots_adjust(bottom = 0.2, left = 0.2)
```

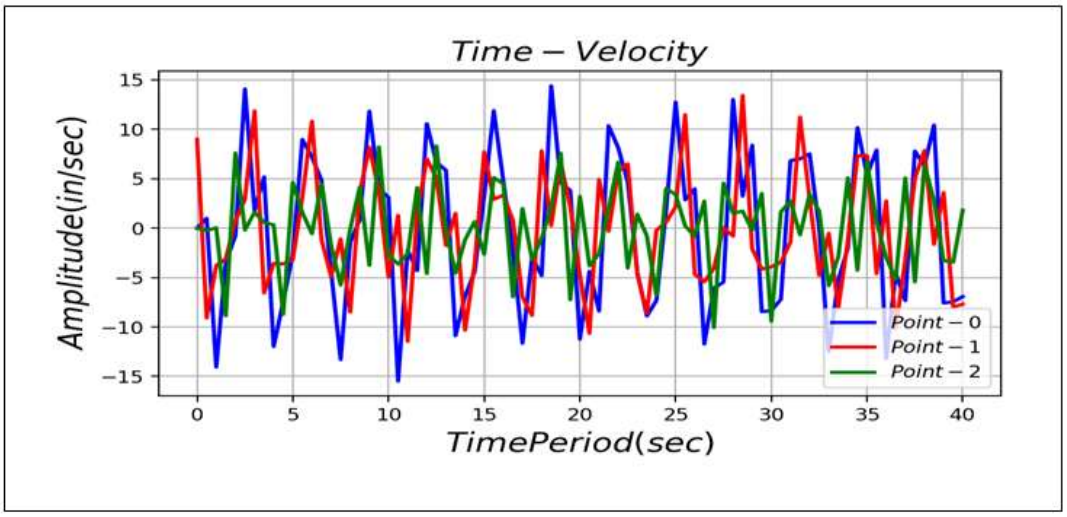
```
plt.plot(x, y0, 'b-', label = r'$Point-0$', lw = 2)
plt.plot(x, y1, 'r-', label = r'$Point-1$', lw = 2)
plt.plot(x, y2, 'g-', label = r'$Point-2$', lw = 2)
plt.xlabel(r'$Time Period(sec)$').set_fontsize(16)
plt.ylabel(r'$Amplitude(in)$').set_fontsize(16)
plt.title(r'$Time - Displacement$').set_fontsize(16)
plt.grid(axis = 'both')
plt.legend(loc = 'best')
plt.savefig('Dwg15.pdf')
plt.show()
```

變位、速度、和加速度執行結果如下所示 ：

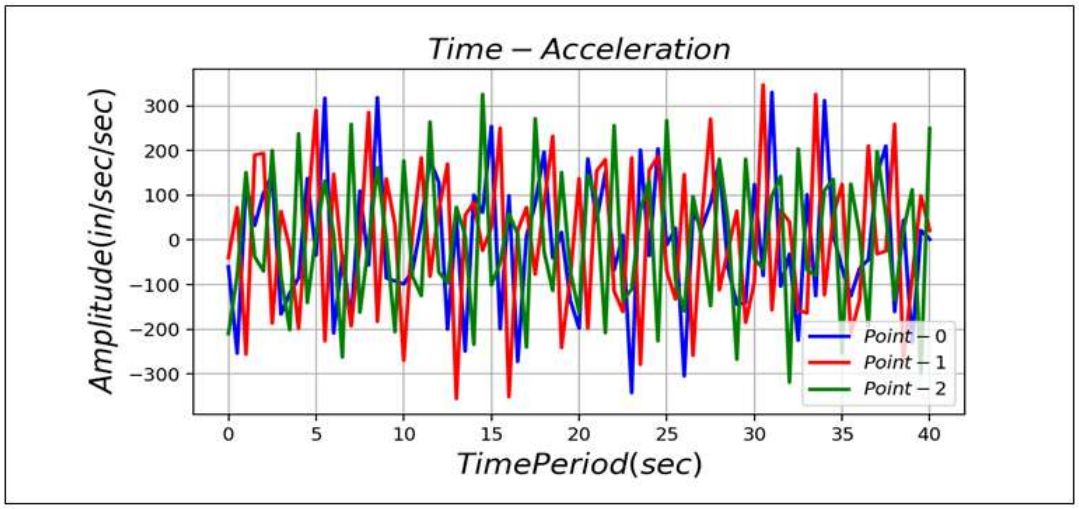
(1) 變位圖 ：



(2) 速度圖 ：


Time – Velocity
Amplitude(in/sec)
TimePeriod(sec)
Point – 0
Point – 1
Point – 2

(3) 加速度圖 ：


Time – Acceleration
Amplitude(in/sec/sec)
TimePeriod(sec)
Point – 0
Point – 1
Point – 2