第五章 空間相依狀態視覺化

多維度(Degree of Freedom)和多階(Order)動態系統微分方程式，維度是離散式，但時間是連續性，很難以數學式來表示該動態方程式，即使使用複雜的數學式表示，由上述兩個維度兩階的微分方程式中，也很難清楚表示速度和位移。若使用精銳矩陣計算求解器，很容易求得輸出結果，即不同位置於時間和位移、速度、和加速度的關係，但因爲輸出的資料非常的多，故需要將這些資料視覺化，使我們能夠瞭解位置、時間、和狀態的變化。如果須要確切的數值，只有由這些資料去尋找。作者將這些資料予以視覺化，使用兩種方式。一是使用 Excel 繪圖工具，二是使用 Python 的 Matplotlib 的套件。

5.1 模態計算與空間狀態輸出 ：

模態 (Mode Shap)其實是系統矩陣的特徵向量(Eigen Vector)，也就是線性代數或是矩陣計算理論中的特徵向量。但模態不全是實數，一般是複數，尤其系統矩陣不對稱時，往往(但並不一定全都)是複數，故模態其實不能完全繪製與表示。

另作者使用空間狀態(Space-state)而不是狀態空間(state-space)，是因為空間狀態表示是多個自由度(Degree of Freedom)的空間且多個狀態(state)，而狀態空間是指一階但多個狀態變數(One order with multiple state variables)，請參考電機訊號與系統的書籍或 Wikipedia 網路。

在實體模型中，我們可以作實驗測試分析，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，瞭解系統的狀態反應輸出。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，即為輸出模型分析，這種分析法有很多的缺陷待解決。

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。一切一切的問題就是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。另很多應用數學、線性代數、和矩陣計算的書籍和專業論文，都有求取特徵值與特徵向量的方法，但是如何進一步使用特徵值和特徵向量則完全欠缺。精銳矩陣求解器則能完全補足這方面的不足，茲說明如下。

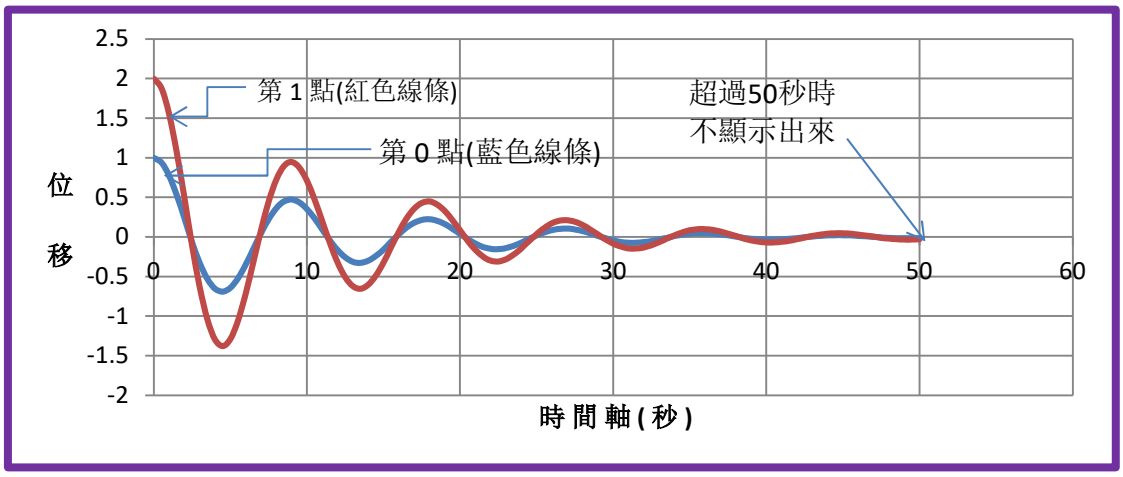
特徵值和特徵向量是互為共軛(Self-Adjoint)關係，缺一不可，其預設值是複數，複數包含實數，也就是實數是複數的 subset，但兩者之間可以互換，實數可以隱性(Implicit)轉為複數，複數可以顯性(Explicit)轉為實數，若是無法轉換，則產生錯誤(Exception)，也就是虛數部分應為零，才可轉為實數。即在 C#程式

語言的型態轉換(Type Convertion)。真實的世界(Real World)是實數，實數才可量測，複數僅是計算的中間過程，最後計算的結果都是實數，不可為複數。在所有的實例計算中，精銳矩陣求解器都是這種計算結果。

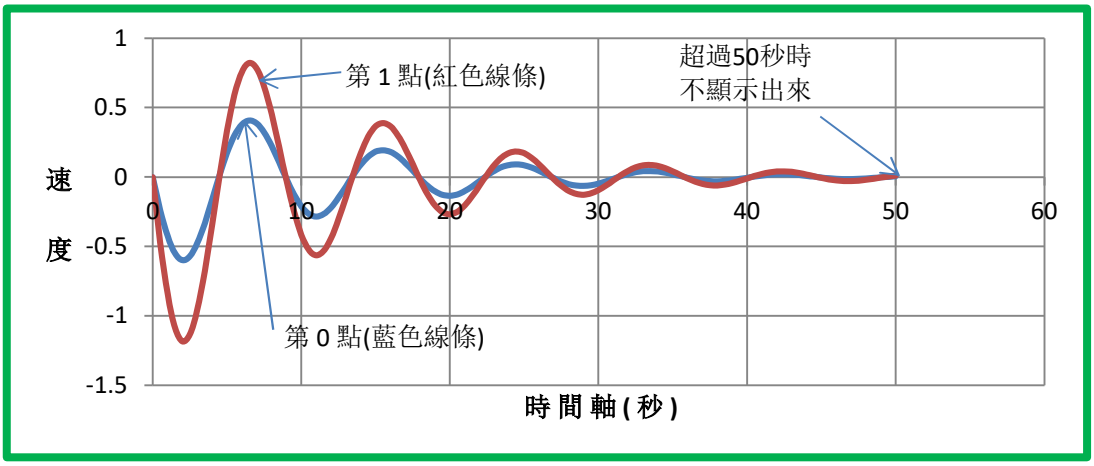
基於以上所述，複數是實數的延伸，再加上 C#程式語言具有的特性，不再對於希爾伯特空間(Hilbert Space)感到迷惑，不論是實數或是複數的兩個向量 a 和 b，其內積(Inner Product)是，在線性代數使用<a，b>方式表示，在精銳矩陣求解器中，使用矩陣運算子"^"表示，即 a ^ b 等於 b^h 乘 a，b^h 是 b 向量的厄米特運算子(Hermitian)表示，因此無須再對於複數感到困惑了。

5.2 使用 Excel 繪圖工具 ：

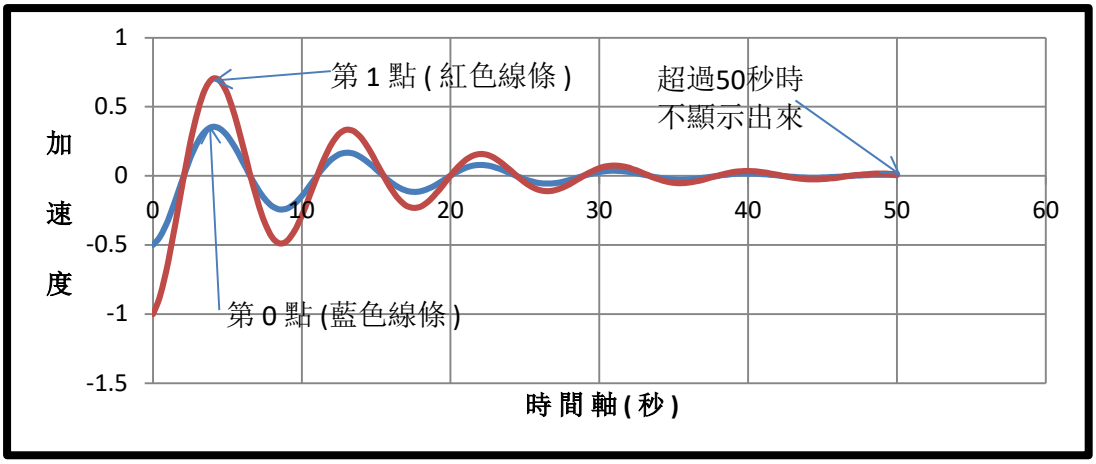
將 ConsoleApp6k 的 C#程式執行的結果，共有 3 個部分，即變位、速度、和加速度的 txt 文字檔，將其結果輸入 Excel 中，畫出如下 ：



位 移 - 時 間 關 係 圖

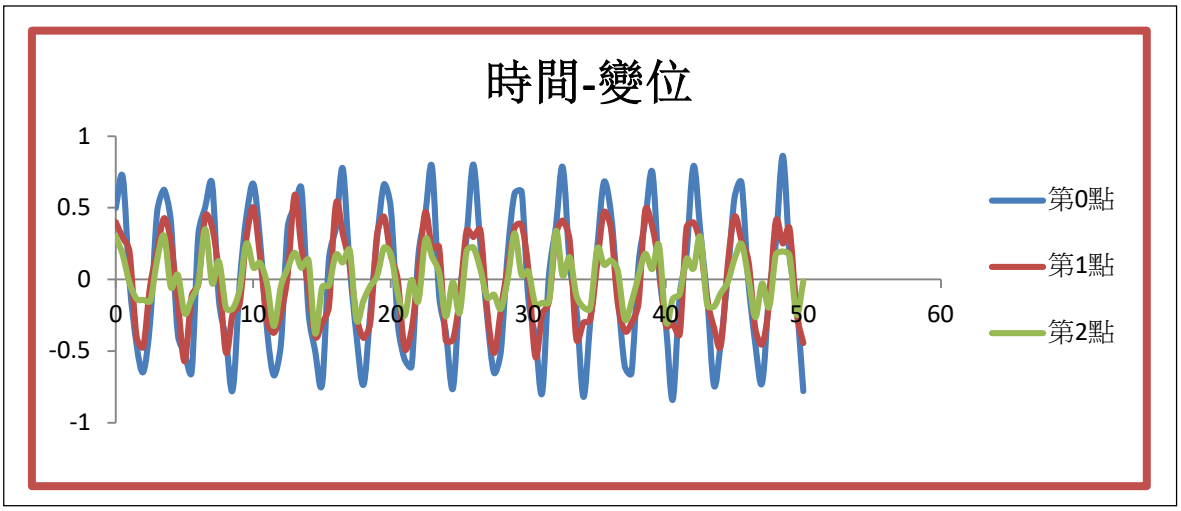


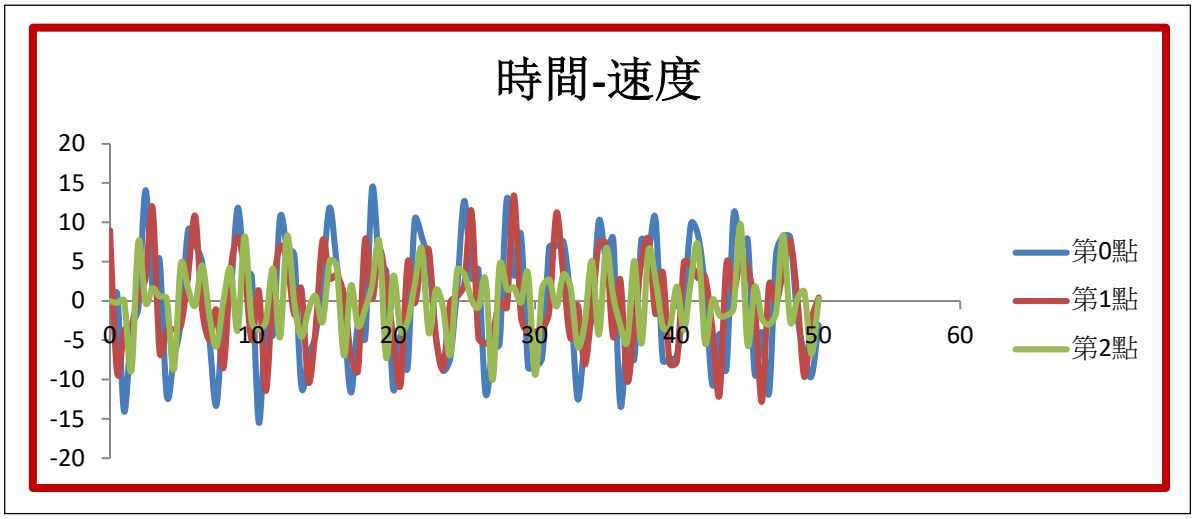
速 度 - 時 間 關 係 圖

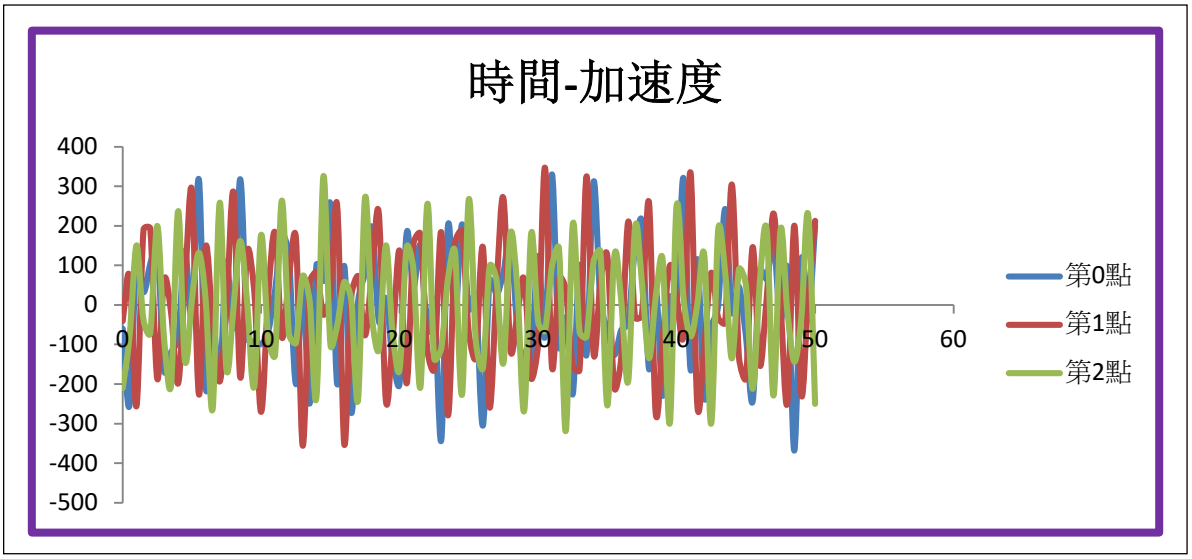


加 速 度 - 時 間 關 係 圖

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
1
0.5
0
-0.5
-1
0
10
20
30
40
50
60
第0點
第1點
第2點


時間-速度
20
15
10
5
0
-5
-10
-15
-20
0
10
20
30
40
50
60
第0點
第1點
第2點



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：

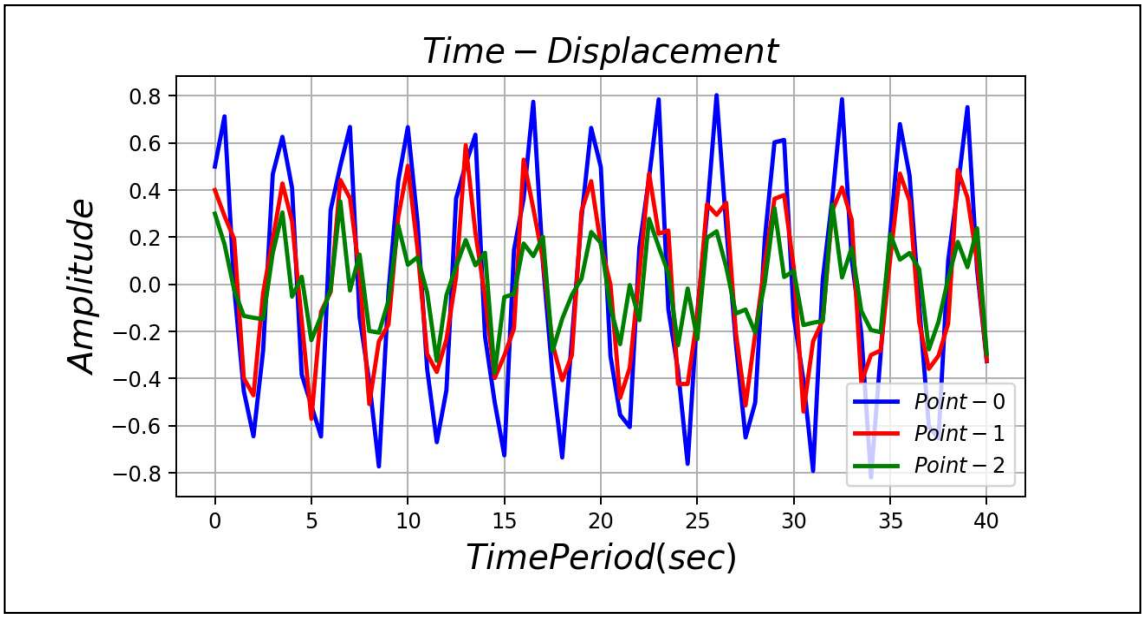
```
import numpy as np
import matplotlib.pyplot as plt

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)
```

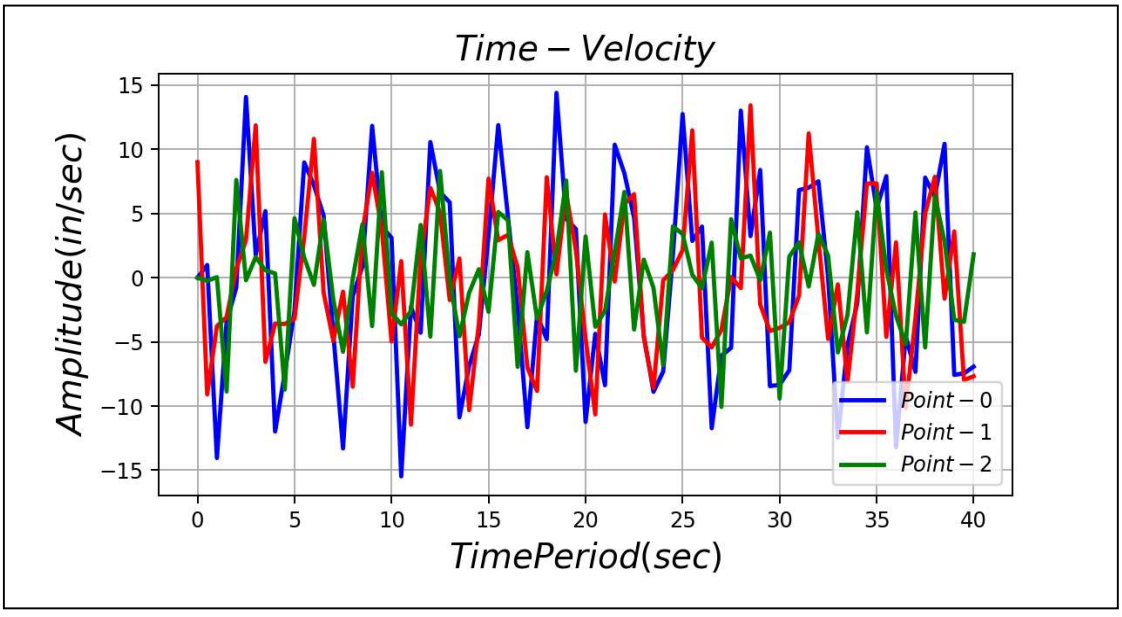
```
plt.plot(x, y1, 'r-', label = r'$Point-1$', lw = 2)
plt.plot(x, y2, 'g-', label = r'$Point-2$', lw = 2)
plt.xlabel(r'$Time Period(sec)$').set_fontsize(16)
plt.ylabel(r'$Amplitude(in)$').set_fontsize(16)
plt.title(r'$Time - Displacement$').set_fontsize(16)
plt.grid(axis = 'both')
plt.legend(loc =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

變位、速度、和加速度執行結果如下所示 ：

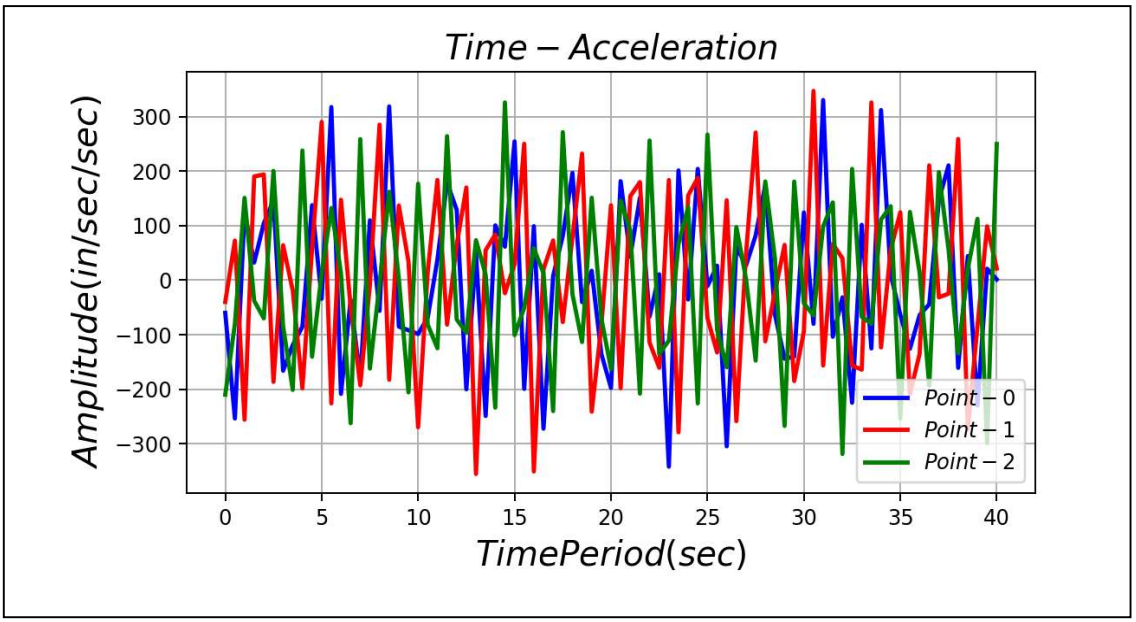
1　變位圖 ：



2　速度 ：



3　　加速度圖 ：


Time − Acceleration
Amplitude(in/sec/sec)
TimePeriod(sec)
300
200
100
0
−100
−200
−300
0
5
10
15
20
25
30
35
40
Point − 0
Point − 1
Point − 2