自動控制實習

教材

日期：95 年 9 月

[教學進度]

週數	進度內容
一	開學(課程簡介與實驗室介紹)
二	Matlab 教學
三	Matlab 在控制上應用
四	Matlab 在控制上應用
五	Simulink 教學
六	Simulink 在控制上應用
七	基本 OP 電路實作
八	PID 類比控制系統模擬與實作
九	PID 類比控制系統模擬與實作
十	期中考
十一	一階系統參數量測(時域)
十二	一階系統參數量測(頻域)
十三	直流伺服馬達系統參數量測
十四	直流伺服馬達系統參數量測
十五	直流伺服馬達 PID 控制器設計
十六	直流伺服馬達 PID 控制器設計
十七	直流伺服馬達 PID 控制器設計
十八	期末考

［參考資料］

1. 洪清寶等譯,控制系統工程(二版)/Norman S. Nise 原著,86 年 5 月,滄海書局。
2. 控制工程使用 Matlab,趙清風編譯,全華科技.
3. Matlab 5 專業設計技巧,蒙以正著,碁峰資訊.
4. Matlab 5.3 Simulink 3.0 範例入門,羅華強編著,全華科技.
5. Matlab 程式設計基礎篇，鄭錦聰編著，91 年 12 月二版，全華科技.

[1].Matlab 教學

1.1 MATLAB 的介紹

MATLAB (MATrix LABoratory)具有用法簡易、可靈活運用、程式結構強又兼具延展性。

以下為其幾個特色：

- 功能強的數值運算 - 在 MATLAB 環境中，有超過 500 種數學、統計、科學及工程方面的函數可使用，函數的標示自然，使得問 題和解答像數學式子一般簡單明瞭，讓使用者可全力發揮在解題方面，而非浪費在電腦操作上。
- 先進的資料視覺化功能 - MATLAB 的物件導向圖形架構讓使用者可執行視覺數據分，並製作高品質的圖形，完成科學性或工程性圖文並茂的文章。
- 高階但簡單的程式環境 - 做為一種直譯式的程式語言，MATLAB 容許使用者在短時間內寫完程式，所花的時間約為用 FORTRAN 或 C 的幾分之一，而且不需要編譯(compile)及聯結 (link) 即能執行，同時包含了更多及更容易使用的內建 功能。
- 開放及可延伸的架構 - MATLAB 容許使用者接觸它大多數的數學原使碼，檢視運算法，更改現存函數，甚至加入自己的函數使 MATLAB 成為使用者所須要的環境。
- 豐富的程式工具箱 - MATLAB 的程式工具箱融合了套裝前軟體的優點，與一個靈活的開放但容易操作之環境，這些工具箱提 供了使用者在特別應用領域所需之許多函數。現有工具箱有：符號運算（利用 Maple V 的計算核心執行 ）、影像處理、統計分析、訊號處理、神經網路、模擬分析、控制系統、即時控制、系統確認、強建控制、弧線分析、最佳化、模糊邏輯、mu 分析及合成、化學計量分析。

MATLAB 有幾種在不同電腦作業系統的版本，例如 MATLAB for Windows, SIMULINK，在麥金塔上的 MATLAB for Macintch，另外還有在 Unix 上的各種工作站版本。基本上這些版本主要是提供方便的操作環境。

1.2 基本功能介紹

以下介紹 MATLAB 的入門功能，包括數學算式、定義變數。

1.2.1 MATLAB 的視窗環境

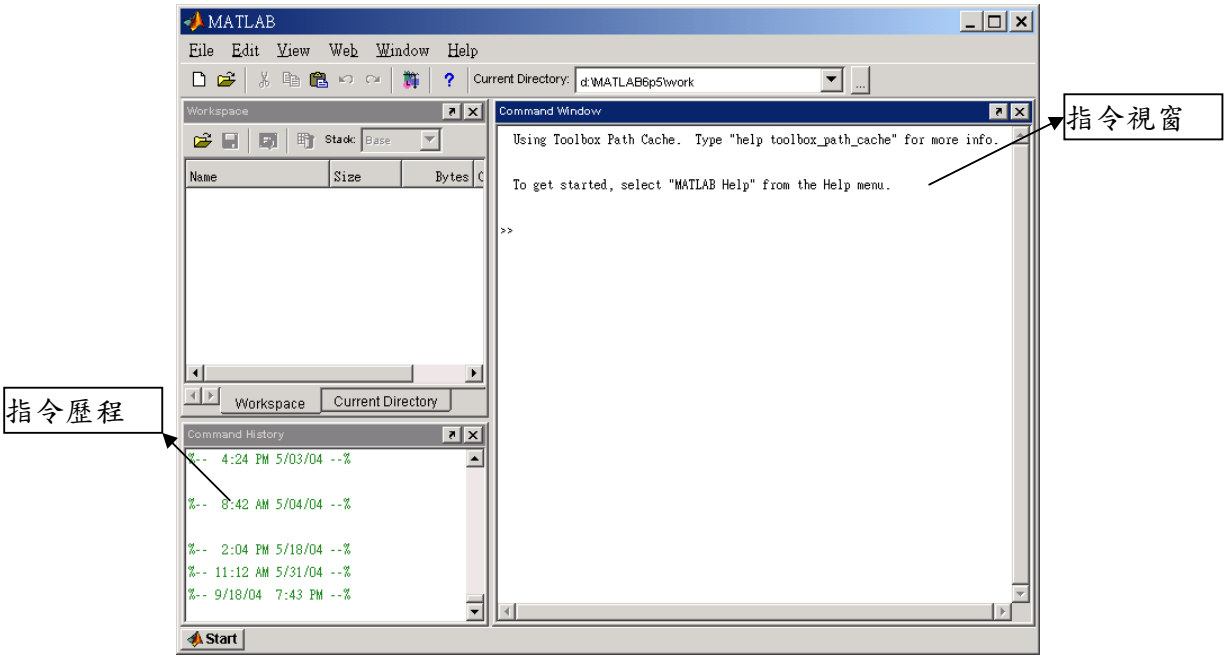
1.2.2 簡易數學

1.2.3 變數

1.2.4 其他功能

1.2.1 MATLAB 的視窗環境

進入 MATLAB 之後，會看到一個視窗 MATLAB Command Window 稱為指令視窗，它是你鍵入指令的地方也是 MATLAB 將計算結果顯示在此。而在它的功能選單一共有 **File**,**Edit**, **Options, Windows, Help** 五個主要功能，每一個之下各又有下一層的功能，我們會在後面相關的地方說明。



1.2.2 簡易數學

我們先從 MATLAB 的數學運算開始說明。就像你的計算器一樣，數學式的計算是直接了當。如果我們要算 1+2+3 及 1×10+2×20+3×30 這二個式子，以下例子接著提示符號>>之後的是要鍵入的算式，MATLAB 將計算的結果以 ans 顯示。如果算式是 x=1+2+3，MATLAB 將計算的結果以 x 顯示。

>> 1+2+3 ans = 6 >> 1*10 + 2*20 + 3*30 ans = 140	>> x=1+2+3 x = 6

如果在上述的例子結尾加上;，則計算結果不會顯示在指令視窗上，要得知計算值只須鍵入該變數值即可

>> x=1+2+3; >> x x= 6	

以下的例子，顯示 MATLAB 對使用變數的彈性

>> apple=5 apples = 5 >> orange=10 orange = 10	>> total_cost=apple*2+orange*4 total_cost = 50 >> average_cost=total_cost/(apple+orange) average_cost = 3.33334

MATLAB 提供基本的算術運算有：

加 (+)、減 (-)、乘 (*)、除 (/)、冪次方 (^)，範例為：5+3, 5-3, 5*3, 5/3, 5^3

其它在計算常用的功能我們來看一個算式來說明。要計算面積 Area = ,半徑 r = 2，則可鍵入

>> r=2; >> area=pi*r^2; >> area = 12.5664	

我們也可以將上述指令打在同一行，以, 或是; 分開，例如

>> r=2, area=pi*r^2 >> r=2; area=pi*r^2;	

請注意上述二式的差異，前者有計算值顯示，而後者則無。如果一個指令過長可以在結尾加上...（代表此行指令與下一行連續），例如

>> r=2; >> area = pi ... *r^2	

另外一個符號註解是由%起頭，也就是說在%之後的任何文字都被視為程式的註解。註解的功能是簡要的說明程式的內容，過多的註解在程式中或許沒有必要，但是我們寫程式時往往用了太少的註解。任何可能產生混淆的地方都應該省用註解，將適量的註解可在往後想了解程式時能節省一些不必要的時間與「有看沒有懂」的痛苦。例如

>> r=2; % 鍵入半徑 >> area=pi*r^2; % 計算面積	

MATLAB 可以將計算結果以不同的精確度的數字格式顯示，我們可以直接在指令視窗鍵入以下的各個數字顯示格式的指令，以 π 值為例

指令	數字值	說明
format short	3.1416	預設的 4 位有效小數位數
format long	3.14159265358979	15 位有效小數位數
format short e	3.1416e+000	4 位有效小數位數加上指數表格式

1.2.3 變數

MATLAB 對使用變數名稱的規定：

1. 變數名稱的英文大小寫是有區別的（apple, Apple, AppLe，三個變數不同）。
2. 變數的長度上限為 19 個字元。
3. 變數名的第一個字必須是一英文字，隨後可以摻雜英文字、數字或是底線。

以下列出 MATLAB 所定義的特別變數及其意義

變數名	意義
help	線上說明, 如 help quit
who	列出所有定義過的變數名稱
ans	預設的計算結果的變數名
eps	MATLAB 定義的正的極小值=2.2204e-16
pi	內建的 π 值
inf	$\frac{1}{0}$ ∞值，無限大
NaN	$\frac{0}{0}$ 無法定義一個數目

1.2.4 其他功能

指令	說明	例
↑	1. 將所下過的指令叫回來重覆使用 2. 按下↑則前一次指令重新出現，之後再按 Enter 鍵，即再執行前一次的指令。	
↓	1. 往後執行指令	
clear	1. 去除所有定義過的變數名稱	clear x y
Ctrl+C	1.用來中止執行中的 MATLAB 的工作。	

1.3 陣列與矩陣

在 MATLAB 的運算式是依據陣列與矩陣算。陣列即是一般通稱的向量，它代表一串數據將其寫成向量格式。而矩陣是依據線性代數做運算，其用途要留待後面再說明。

1.3.1 簡易陣列

1.3.2 建立陣列

1.3.3 陣列運算

1.3.4 特殊矩陣

1.3.5 陣列運算的特色

1.3.1 簡易陣列

MATLAB 的運算事實上是以陣列 (array) 及矩陣 (matrix) 方式在做運算，而這二者在 MATLAB 的基本運算性質不同，陣列強調元素對元素的運算，而矩陣則採用線性代數的運算方式。在此只說明如何定義矩陣，至於矩陣的詳細運算語法，參考相關手冊。

而宣告一變數為陣列或是矩陣時，如果是要個別鍵入元素，須用中括號[ ] 將元素置於其中。陣列為一維元素所構成，而矩陣為多維元素所組成，例如

```
>> x = [1 2 3]                  % 一維 1x3 陣列
>> x = [1 2 3; 4 5 6]           % 二維 2x3 矩陣，以;區隔各列的元素
>> x = [1 2 3                   % 二維 2x3 矩陣，各列的元素分二行鍵入
        4 5 6]
```

假設要計算 y = sin (x), $0 \leqq x \leqq \pi$ 而 x = 0, 0.2π, 0.4π,..., π ，即可用陣列方式運算，例如

```
>> x = [0 0.2*pi 0.4*pi 0.6*pi 0.8*pi pi]          % 注意陣列內也可作運算
x =
0 0.6283 1.2566 1.8850 2.5133 3.1416
>> y=sin(x)
y =
0 0.5878 0.9511 0.9511 0.5878 0.0000
```

要找出陣列的某個元素或數個元素，可參考以下的例子

```
>> x(3)                                          % 第三個 x 的元素
ans =
1.2566
>> y(5)                                          % 第五個 y 的元素
ans =
0.5878
>> x(1:5)                                        % 列出第一到第五個 x 的元素
ans =
0 0.6283 1.2566 1.8850 2.5133
>> y(3:-1:1)          % 列出第三到第一個 y 的元素，3 為起始值，1 為終止值，-1 為增量
ans =
0.9511 0.5878 0
>> x(2:2:6)           % 列出第二到第六個 x 的元素，2 為起始值，6 為終止值，2 為增量
```

```
ans =
0.6283 1.8850 3.1416
>> y([4 2 5 1])                    % 列出 y 元素，排列元素依序為原來 y 陣列的 4,2,5,1 個
ans =
0.9511 0.5878 0.5878 0
```

1.3.2 建立陣列

前一節提到陣列產生的方式須個別鍵入其元素，這方法只適用於陣列元素很少時。如果要建立的陣列的元素多達數百個，則須採用以下的數種方式

```
>> x=(0:0.02:1)          % 以:區隔起始值=0、增量值=0.02、終止值=1
>> x=linspace(0,1,51)    % 利用 linspace，以區隔起始值=0 終止值=1 之間的元素數目=51
>> x=(0:0.01:1)*pi       % 注意陣列外也可作運算
>> a=1:5, b=1:2:9        % 這二種方式更直接
a =
1 2 3 4 5
b =
1 3 5 7 9
>> c=[b a]               % 可利用先前建立的陣列 a 及陣列 b ，組成新陣列
c =
1 3 5 7 9 1 2 3 4 5
>> d=[b(1:2:5) 1 0 1]    % 由陣列 b 的三個元素再加上三個元素組成
d =
1 5 9 1 0 1
```

1.3.3 陣列運算

以下將陣列的運算符號及其意義列出，除了加減符號外其餘的陣列運算符號均須多加 . 符號。

陣列運算功能	功能
+	加
-	減
.*	乘
./	左除
.^	次方
.'	轉置

```
>> a=1:5; a-2                      % 從陣列 a 減 2
ans =
-1 0 1 2 3
>> 2*a-1                          % 以 2 乘陣列 a 再減 1
ans =
1 3 5 7 9
>> b=1:2:9; a+b                   % 陣列 a 加陣列 b
ans =
2 5 8 11 14
>> a.*b                           % 陣列 a 及 b 中的元素與元素相乘
```

```
ans =
1 6 15 28 45
>> a./b                    % 陣列 a 及 b 中的元素與元素相除
ans =
1.0000 0.66667 0.6000 0.5714 0.5556
>> a.^2                    % 陣列中的各個元素作二次方
ans =
1 4 9 16 25
>> 2.^a                    % 以 2 為底，以陣列中的各個元素為次方
ans =
2 4 8 16 32
>> b.^a                    % 以陣列 b 中的各個元素為底，以陣列 a 中的各個元素為次方
ans =
1 9 125 2401 59049
>> b=a'                    % 陣列 b 是陣列 a 的轉置結果
b =
1
2
3
4
5
```

1.3.4 特殊矩陣

有一些特別矩陣的定義，如元素皆為 0, 1 或是單位矩陣，因為在運算時常會用到，所以在此先介紹。zeros 函數是形成元素皆為 0 的矩陣;ones 函數是形成元素皆為 1 的矩陣; eye 則是產生一個單位矩陣，之所以稱為 eye 是取其發音與原來單位矩陣符號 I 相同，而又避免與定義複數中的虛部所用的符號 i 雷同，所以改以 eye 替代。上述三個函數的使用語法都相似，如 zeros(m)可以產生一個 m×m 的正方矩陣，而 zeros(m,n)產生的是 m×n 的矩陣。也可以使惡用這三個函數將一 m×n 矩陣原來元素全部取代成 0, 1 或是單位矩陣的值，不過要加上 size 指令來指出其矩陣大小是 m,n，所以語法為 zeros(size(A))，其中 A 是原來矩陣。

```
>> A=zero(2)                          % 0 的矩陣
A =
0 0
0 0
>> B=zeros(2,3)
B =
0 0 0
0 0 0
>> C=[1 2; 3 4; 5 6];
>> size(C)                            % 使用 size 指令得到 C 矩陣的大小
ans =
3 2
>> D=zeros(size(C))                   % 加上 size 指令將矩陣 C 原來的元素全部以 0 取代
>> A=ones(2), B=ones(2,3)             % 1 的矩陣
A =
1 1
```

```
1 1
B =
1 1 1
1 1 1
>> C=[1 2; 3 4; 5 6];
>> D=ones(size(C));
>> A=eye(2), B=eye(2,3)                                  % 單位矩陣
A =
1 0
0 1
B =
1 0 0
0 1 0
>> C=[1 2; 3 4; 5 6];
>> D=eye(size(C));
```

1.3.5 陣列運算的特色

MATLAB 在許多運算皆是以陣列為對象，即是以陣列的元素為對象。因此除了+, - 這二個運算外，其餘的運算符號（乘、除、次方）皆須加上.來強調陣列之間的運算。以下幾個例子可以說明陣列運算的特色。如果 a,b 各代表二個不同的陣列，a 與 b 之間的運算是元素對元素的方式，例如以下幾個例子：

```
>> x = 1.5;                                  % x 是純量
>> y = exp(x^2);                             % exp(x^2) 是純量運算
>> y1 = x/y                                  % x/y 是純量運算
>> x = 1:0.1:2;                              % x 是陣列
>> y = exp(x.^2);                            % exp(x.^2) 是陣列運算
>> y1= x./y                                  % x./y 是陣列運算
```

這個例子的算式較長，一樣也須注意純量與陣列運算的差別

```
>> x=2.0                                     % x 是一純量
>> nume = x^3 - 2*x^2 + x - 6.3;
>> deno = x^2 + 0.05*x - 3.14;
>> f = nume/deno
>> x=1:5;                                    % 注意 x 是一陣列
>> nume = x.^3 - 2*x.^2 + x - 6.3;
>> deno = x.^2 + 0.05*x - 3.14;
>> f = nume./deno
```

1.4 簡易繪圖

MATLAB 的繪圖功能很強，我們先從最簡單的二維繪圖指令 plot 介紹起。plot 是用來劃函數 x 對函數 y 的二維圖，例如要劃出 $y = \sin(x), 0 \leqq x \leqq 2\pi$。plot 可以在一個圖上劃數條曲線，且以不同的符號及顏色來標示曲線，其指令見線上說明 help plot。如要在 x 及 y 軸及全圖加註說明，則可利用指令 xlabel, ylabel, title，其指令見線上說明 help xlabel, help ylabel, help title。三維圖的指令為 plot3，其指令見線上說明 help plot3。此外二維圖及三維圖皆可使用指令 grid 加上格線。MATLAB 會將繪圖結果展示在另一個視窗稱為 MATLAB Figure Windows，

如果你看不到此視窗，別擔心它只是被蓋住，可以進入 Windows 再選擇 Figure。接著我們就來看以下的例子

```
>> v1=linspace(0,2*pi,20); v2=sin(v1);          % 建立 v1 及 v2 陣列
>> plot(v1,v2)                                  % 利用 plot，輸入的變數為 x 軸接著的變數為 y 軸
>> v3=cos(v1);                                  % 建立 v3 陣列
>> plot(v1,v2,v1,v3)                            % 劃二條曲線，一條代表 v1-v2 函數關係
                                                %一條代表 v1-v3 函數關係
>> plot(v1,v2,v1,v2,'+')                        % 一樣劃二條曲線，不過第二條曲線以符號 + 標示
>> plot(v1,v2,v1,v2.*v3,'--')                   % 劃二條曲線，一條代表 v1-v2 函數關係，一條
                                                % 代表 v1-(v2.*v3) 函數關係且以符號'標示
>> xlabel('x-axis')                            % 加上 x 軸的說明，在二個單引號 ' 之間鍵入文字的說明
>> ylabel('y-axis')                            % 加上 y 軸的說明
>> title('2D plot')                            % 加上圖的說明
>> plot3(v2,v3,v1), grid       % 將 v2-v1-v3 函數關係分別以 x 軸 y 軸及 z 軸劃，並加上格線
```

1.5 輸入及輸出

1.5.1 交談式輸入

我們來看一個已經講過的算式：要計算面積 Area= ，可利用指令 input 在螢幕印出提示文字做為交談式的輸入。

```
>> r = input('Type radius:')                    % 在兩個單引號 ' 之間鍵入提示文字
Type radius:                                    % 現在鍵入 2 做為半徑值
r =
2
>> area=pi*r^2;                                 % 鍵入面積算式
>> name = input('Your name please: ','s')       % 要鍵入文字則須在加上's'，s 是代表字串(string)
Your name please:                               % 鍵入名字 Charly Chen
name =
Charly Chen
```

1.5.2 輸出格式

至於輸出有二種格式：自由格式 (disp) 和格式化輸出 (fprintf)。要直接輸出文字或是一數值，可使用 disp，例如

```
>> temp=20;
>> disp(temp); disp('degrees C'); disp('度 C')                    %中文也接受呢！
20
degrees C
度 C
```

而指令 fprintf 則是用來控制輸出數據及文字的格式，它的基本格式如

```
>> fprintf('The area is %8.5f\n', area)
```

在二個單引號間包括輸出的字串 The area is，接著是輸出數據的格式%8.5f，再來是跳行符號以避免下一個輸出數據或是提示符號也擠在同一行，最後鍵入要輸出的數據名 area。例如

```
>> fprintf('The area is %8.5f\n', area) % 注意輸出格式前須有%符號，跳行符號須有\符號
The area is 12.56637 % 輸出值為 8 位數含 5 位小數
```

在此要稍加說明的是輸出數據的格式，以下的例子各說明了不同型態的輸出格式

```
>> fprintf('f_form: %12.5f\n',12345.2)          % 輸出值為 12 位數，含 5 位小數
f_form: 12345.20000
>> fprintf('f_form: %12.3f\n',1.23452)          % 輸出值為 12 位數，含 3 位小數
f_form: 1.235
>> fprintf('e_form: %12.5e\n',12345.2)          % 輸出值為指數格式的 12 位數，含 5 位小數
e_form: 1.23452e+004
>> fprintf('f_form: %12.0f\n',12345.2)          % 輸出值為整數格式的 12 位數
f_form: 12345
```

1.6 如何撰寫 Matlab 程式

我們前面各節所介紹在 MATLAB 所做的運算,是適合於所要計算的算式不太長或是想以交談式方式做運算，如果要計算的算式很長有數十行或是須要一再執行的算式，則那樣的方式就行不通了。MATLAB 提供了所謂的 M-file 的方式，可讓使用者自行將指令及算式寫成巨集程式然後儲存成個特別的檔案，其附加檔是 m，譬如 test.m，其中的 test 就是檔案名稱。至於要撰寫程式可以用任何一種編輯軟體（如 Windows 的記事本或）或是 Matlab 提供之編輯器，但是儲存格式必須是 Ascii 的格式。在指令視窗中的功能選單可以選擇 File 再選擇 New，即進入指定的編輯軟體或是文書處理軟體。當程式寫完後要存檔時，必須以.m 檔名稱儲存。要執行 M-file 可以在指令視窗下直接鍵入該檔名如 test；或是選擇功能表上的 Run M-file 來找到 M-file 的所在目錄再執行 M-file。Open M-file, Run M-file。如果要修改 M-file 可以選擇功能表上的 Open M-file ，即可搜尋要修改的 M-file，修改後再存檔。
以下的 tutex1.m 檔是一個簡易繪圖程式做為示範使用 M-file

```
% M-file, tutex1.m
% Simple plot for illustration of using M-file.
% 簡易繪圖以做為示範使用 M-file
x=linspace(0,2*pi,20); y=sin(x);
plot(x,y,'r+')
xlabel('x-value')
ylabel('y-value')
title('2D plot')
```

寫好上述程式後即可在指令視窗下鍵入 tutex1，即可執行已建立的 tutex1.m 程式。
再來看另一個 M-file: tutex2.m 的例子

```
% M-file, tutex2.m
% 計算一個球的體積
r = input('Type radius:');
area=pi*r^2;
volume=(4/3)*pi*r^3;
fprintf('The radius is %12.5f\n',r)
fprintf('The area of a circle is %12.5f\n',area)
fprintf('The volume of a sphere is %12.5f\n',volume)
```

1.6.1 如何在自己的目錄執行程式

當在執行 M-file 時，我們最好是將自己的 M-file 儲存在自己的工作目錄下，而不要放在 MATLAB 內建的目錄下，這樣做的好處是不會干擾到 MATLAB 程式的目錄下的各個檔案。要在自己的工作目錄執行程式可分為二個步驟：(1)建立搜尋路徑，(2) 切換目錄。
(1) 建立搜尋路徑
MATLAB 將許多內建函數分門別類放在不同的次目錄下,因此它在工作時須依序的搜尋這些次目錄，這個過程稱為「搜尋路徑」。MATLAB 的指令 path 可以讓我們將自己的工作目錄加在原來 MATLAB 的搜尋路徑之前或之後，如此

```
>> path(path,'c:\wufile\my_work')        % 將自己的目錄 \wufile\my_work 加在
                                         % MATLAB 的搜尋路徑之後
>> path('c:\wufile\my_work',path)        % 將自己的目錄 \wufile\my_work 加在
                                         % MATLAB 的搜尋路徑之前
```

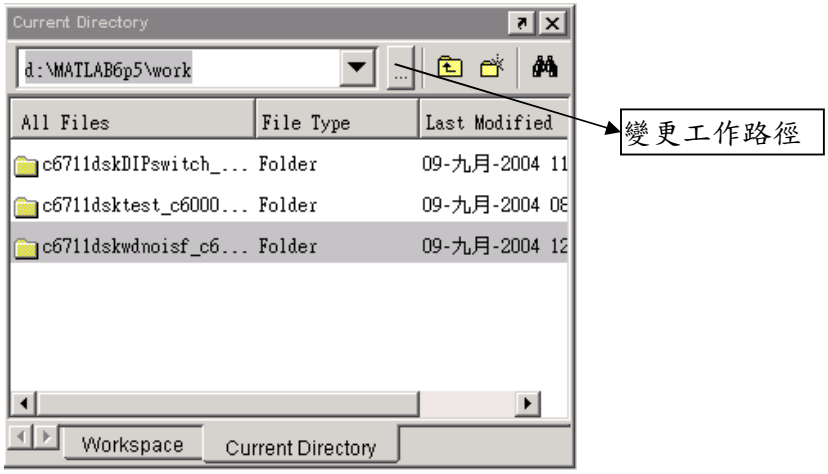
如果你不想每次進入 MATLAB 都要鍵入 path 指令，下面的方式可以將 path 指令設為自動啟用。你可以定義一個特別的 M-file 稱為 startup.m 內容如下，將它存在 MATLAB 的主目錄下，這樣每次 MATLAB 啟動時就會自動的執行這個 startup.m 檔，即與上述的自行設定 path 的作用是相同的。

(2) 切換目錄

1.設好搜尋路徑後，接著即可用 cd 指令將目錄切換到自己的工作目錄之下來安安心心的使用 MATLAB。以下的範例說明如何使用與切換目錄相關的指令：

```
>> cd \wufile\my_work                    % 切換至目錄\wufile\my_work
>> cd                                    % 如果只用 cd 則會顯示目前的目錄
c:\WUFILE\MY_WORK
>> dir                                   % 列出目錄下的檔案
. tutex1.m tutex2.m
.. test.txt
>> delete test.txt                       % 刪除 test.txt
```

2.



1.7 儲存及讀取數據

我們在使用 MATLAB 過程中，免不了希望將運算過程中的某些數據「儲存」起來，以便下次使用再「讀取」利用。「儲存」和「讀取」的指令分別是 save 及 load，而 save 的數據型態又分為：(1)二進制格式 (binary format) 的 MAT-file，(2) ASCII 格式的 ASCII-file。MAT-file 是以二進制方式儲存，可讓電腦在讀出/入(input/output) 速率加快，其格式為 test.mat（test 為檔名），MATLAB 將檔案的型態預設為 MAT-file；而 ASCII-file 則是以可辨識的字元儲存，但會降低電腦在讀出/入的速率，其格式為 test.dat（test 為檔名）。如果你的數據是只在 MATLAB 中產生及被使用，那最好使用 MAT-file。ASCII-file 則必須用在當數據檔要為其它不是 MATLAB 的應用軟體讀取時。

另外要注意，當 save 成 MAT 檔是儲存變數本身，而非直接儲存變數的數據；而 save 成 ASCII 檔則是直接儲存變數的數值。

這二者儲存的差異，造成在讀取 MAT 檔和 ASCII 檔的數據有所不同，詳見以下的範例。須注意的是在儲存及讀取數據時，MAT-file 或是 ASCII-file 的檔最好為矩陣型態，否則可能在讀取時有困難。數據儲存成矩陣的大小可以為 mxn，其中 m 是列的數目，n 則為行的數目。以下就是幾個 save, load 的使用範例

```
>> x=1:5; y=11:15;                  % 先產生二個列陣列 (row array) x, y
>> save data1 x y                   % 是將 x,y 二個變數的數值存入 data1 這個 MAT-file，
      %即 data1 其實是 data1.mat。data1.mat 的內容為變數 x, y，而非(1:5, 11:15) 的數據
>> save data2.dat x y -ascii        % 如果要將 data1 改以 ASCII 格式儲存，則須加上-ascii
                                    % 的選項。data2.dat 的內容為(1:5, 11:15) 的數據
>> type data2.dat                   % type 指令可以將 data2.dat 的內容列出
>> load data1                       % 讀取 data1.mat 檔
>> x, y                             % 叫出 data1.mat 中的變數來讀取其內容(1:5, 11:15)
>> load data2.dat                   % 讀取 data2.dat 檔
>> x2=data2(1,:); y2=data2(2,:);    % 將 data2 中的第一及第二列數據分別以 x2 及 y2
                                       %變數讀入，之後在運算中即可使用這二列數據
>> x=21:25; y=31:35;
>> save data3.dat x y -ascii
>> load data3.dat;
>> x3=data3(1,:); y3=data3(2,:); % 將 data3 中的第一及第二列數據分別以 x3 及 y3 變數讀入
                                 %，之後在運算中即可使用這二列數據
>> A=[1 2 3; 4 5 6];
>> save data4.dat A -ascii          %是將 A 陣列的數值存入 data4 這個 ASCII-file
>> load data4.dat
>> x4=data4(:,1);                     % 令 x4 為 data4 的第一行數據
>> y4=data4(:,2);                   % 令 y4 為 data4 的第二行數據
>> z4=data4(:,3);                     % 令 z4 為 data4 的第三行數據
```

1.8 其他繪圖功能

我們在前面多少都說明過簡易的二維繪圖功能，例如在圖上加註說明的指令有 title, xlabel,ylabel，除此之外還有二個指令 text, gtext 可以在圖中加上文字用以說明圖中的曲線或圖形代表什麼。text 是依據所繪圖的座標來放置文字說明，其語法為 text(x,y, 'string')，x, y 是要放置說明的座標值，string 是說明的文字。gtext 則是依據滑鼠或上下左右游標鍵來放置文字說明，其語法為 gtext('string')。我們來看幾個例子：

```
>> x=linspace(0,2*pi,30); y=sin(x); z=cos(x);
>> plot(x,y,x,z)                          % 劃二條曲線 y=sin(x), z=cos(x)
>> text(2.5,0.7,'sin(x)')                 % (2.5,0.7)是依據繪圖大小的座標值
>> gtext('cos(x)')                        % 將滑鼠移至適當位置再按滑鼠鍵
```

一般的 x-y 圖在橫軸及縱軸皆是以線性尺度來繪圖，如果要繪圖的數據的 x 或 y 值變化範圍太大，就須要改用對數 (log) 尺度來繪圖才可得到合理的圖。MATLAB 提供三種對數尺度的繪圖指令：semilogx,semilogy, loglog，它們的作用分別是 x 軸以對數尺度繪圖，y 軸以對數尺度繪圖，x 和 y 軸以對數尺度繪圖，而 logspace 指令則是配合對數刻度來取資料點。還有當需要有兩個 Y 軸刻度時，可使用指令 plotyy 我們來看幾個例子，藉以說明在何種場合須要用對數尺度繪圖。

```
>> y=0:0.1:10; x=10.^y
>> plot(x,y)
>> semilogx(x,y)                          % 改以對數尺度繪圖
>> x=[0 2 5 7 10 12 15 17 20 21];
>> y=[0.1 0.2 0.5 0.6 0.9 1 1.2 1.26 1.22 1.2];
>> plot(x,y)                              % 先以線性尺度繪圖，再分別以三種對數尺度繪
>> semilogx(x,y)                          % 圖，注意各個圖長像會改變
>> semilogy(x,y)
>> loglog(x,y)
```

繪圖樣式選擇:

色彩	樣式	線條	樣式	標記	樣式
b	藍	-	實線	+	加號
c	青	--	虛線	0	圓圈
g	綠	:	點線	*	星號
k	黑	-.	點虛線	x	乘號
m	紫紅	none	無線條	.	點號
r	紅			^	上三角
w	白			v	下三角
y	黃			>	右三角
				<	左三角
				square	正方形
				diamand	菱形
				pentagram	五角形
				hexagram	六角形

1.8.1 繪圖選項

MATLAB 有許多的繪圖選項，以下一一做介紹。如果要將二條曲線劃在同一個圖，可參考以下的例子：

```
>> x=linspace(0,2*pi,30);
>> y=sin(x); z=cos(x);
>> plot(x,y,x,z)                    % 將 y=sin(x) 及 z=cos(x) 二函數分佈繪圖
>> plot(x,y,'g:',x,z,'r--')         % 加上不同的顏色及符號來區別二條曲線
```

plot 還有許多繪圖控制的選項，可以參考 plot 的線上說明。

1.8.1.1 橫軸和縱軸的控制

要控制繪圖的橫軸及縱軸比例，可以用 axis 配合下列的相關的選項：

axis([xmin xmax ymin ymax])	以 xmin xmax 設定橫軸的下限及上限，以 ymin ymax 設定縱軸的下限及上限
axis auto	橫軸及縱軸依照數據大小的上下限來訂定，橫軸及縱軸比例是 4:3
axis square	橫軸及縱軸比例是 1:1
axis equal	將橫軸縱軸的尺度比例設成相同值
axis xy	預設值使用卡氏座標即是將圖原點設在左下角橫軸由左往右增縱軸由下往上遞增
axis ij	使用矩陣格式即是將圖原點設在左上角橫軸不變縱軸由上往下遞增
axis normal	以預設值畫縱軸及橫軸
axis off	將縱軸及橫軸取消
axis on	恢復縱軸及橫軸

上述的各個指令的語法也可以將關鍵字放在括弧內的單引號之間，如 axis(' ')。

以下是應用 axis 的範例：

```
>> x=linspace(0,2*pi,30); y=sin(x); z=cos(x);
>> plot(x,y,x,z)
>> axis off
>> axis on
>> axis('square','equal')
>> axis('xy','normal')
```

1.8.1.2 子圖

要將數個相關的圖畫在同一頁時，可以用 subplot 這個指令。其語法為 subplot(m,n,p)，其中 m, n 代表繪圖成 m x n 個子圖，m 表示在 y 方向有 m 個圖，n 表示在 x 方向有 n 個圖，p 是代表第幾個子圖。下例是以 subplot 分別畫出線性及對數尺度的四個子圖：

```
>> x=[0 2 5 7 10 12 15 17 20 21];
>> y=[0.1 0.2 0.5 0.6 0.9 1 1.2 1.26 1.22 1.2];
```

```
>> subplot(2,2,1), plot(x,y) % 畫左上角的圖
>> subplot(2,2,2), semilogx(x,y) % 畫右上角的圖
>> subplot(2,2,3), semilogy(x,y) % 畫左下角的圖
>> subplot(2,2,4), loglog(x,y) % 畫右下角的圖
```

1.8.1.3 圖形放大及縮小

zoom 指令可以將圖形放大或縮小，若要將圖形放大時用 zoom on，zoom out，當不再須要放大或縮小圖形時用 zoom off。

```
>> M=peaks(25);        % peaks 是 MATLAB 內建的一個像山峰的特別函數，25 是這個
>> plot(M)             % 函數矩陣的大小，如果數值愈大則畫出的山峰圖愈平滑
>> zoom on             % 開始放大圖形，每按一次 Enter 鍵圖形就放大一次
>> zoom out            % 開始縮小圖形，每按一次 Enter 鍵圖形就縮小一次
>> zoom off            % 停止圖形放大或縮小功能
```

1.8.1.4 函數分佈的快速繪圖

fplot 的指令可以用來自動的畫一個已定義的函數分佈圖，而無須產生繪圖所須要的一組數據做為變數。其語法為 fplot('fun',[xmin xmax ymin ymax])，其中 fun 為一已定義的函數名稱，例如 sin, cos 等等；而 xmin, xmax, ymin, ymax 則是設定繪圖橫軸及縱軸的下限及上限。以下的例子是將一函數 f(x)=sin(x)/x 在-20≦x≦20,-0.4≦y≦1.2 之間畫出：

```
>> fplot('sin(x)./x',[-20 20 -0.4 1.2])
>> title('Fplot of f(x)=sin(x)/x')
>> xlabel('x'), ylabel('f(x)')
```

1.8.1.5 列印功能

MATLAB 的指令 print 有二個功能，它可以直接將所畫的圖列印，也可以將所畫的圖以指定的圖檔格式儲存起來。前者是隨畫隨印；而後者是先將圖畫好再儲存起來，事後可將圖形檔案依指定的方式再列印或是與其它檔案整合。如果是前者，則可直接用繪圖視窗中的列印選單。以下的指令則是針對後者的方式，其的基本語法為 print[filename][-device][-options]，其中的 [ ]代表附加的選項。如果指令是 print，則表示列印現在繪圖視窗的圖。如果是 print filename，則表示將現在繪圖視窗的圖存成 filename 其附檔名依預設的印表機種類有不同的圖檔格式。印表機種類的設定即是由-device 決定，以下分項來說明。

MATLAB 直接支援的印表機種類如下

device 說明

-dps 黑白的描頁式 (PostScript) 印表機，附檔名為 .ps

-dps2 黑白的第二代描頁式 (PostScript II) 印表機，附檔名為 .ps

-deps 黑白的描頁式 (Encapsulated PostScript) 印表機，附檔名為 .eps

-deps2 黑白的第二代描頁式 (Encapsulated PostScript II) 印表機，附檔名為 .eps

其實 MATLAB 還支援有彩色的描頁式印表機，但這類的印表機太昂貴，一般人是不太可能會使用，所以不做介紹，可參考使用手冊或是線上說明的介紹。

device	說明
-dcdjcolor	24 bits 全彩的 HP DeskJet 500C

-depson	Epson 點矩陣式印表機
-dgif8	8 bits 彩色 GIF 圖檔，附檔名為 .gif
-dpcx256	256 色的彩色 PCX 圖檔，附檔名為 .pcx

以下例子是將圖存成三種圖形格式：

```
>> print fig1 -dps % 將圖存成 PostScript 格式，圖檔為 fig1.ps
>> print fig1 -dgif8 % 將圖存成 GIF 格式，圖檔為 fig1.gif
>> print fig1 -pcx256 % 將圖存成 256 色 PCX 格式，圖檔為 fig1.pcx
```

而 [-option] 的選項多半可以用繪圖視窗的列印功能來替代，所以不在此多做說明。

1.8.1.6 其它的功能

如果我們須要在所畫的圖中的曲線的某處加上符號，而又可以隨意的放置這些符號，則可以用指令 ginput 方式，它容許我們以滑鼠或上下左右游標在螢幕上輸入要加上符號的座標。下面的例子是一個有 8 個峰頂及峰谷的函數分佈圖(y = sin (x) / x)，我們以滑鼠方式將符號加在這些峰值上，藉以突顯這些極值，其語法為 [x,y]=ginput(n)。

```
>> x=linspace(-2*pi,2*pi,60);
>> y=sin(x).^2./(x+eps);          % 注意加上 eps 可避免當 x 趨近零時，y 會無法定義
>> plot(x,y)
>> [a,b]=ginput(8);               % 依序從螢幕輸入 8 點的座標
>> hold on
>> plot(a,b,'co')                 % 依據輸入的座標值將符號畫在圖上適當位置
>> hold off
```

1.8.2 三維繪圖

1.8.2.1 三維繪圖

plot3 可以用來畫一個三維的曲線，它的格式類似 plot，不過多了 z 方向的數據。其與法可以是 plot3(X,Y,Z)或是 plot3(X,Y,Z,'linetype')，其中的 linetype 是設定畫線的符號和顏色。下面的例子說明一個三維的曲線圖：

```
>> t=0:pi/50:10*pi;
>> plot3(sin(t),cos(t),t)
>> title('Helix'), xlabel('sin(t)', ylabel('cos(t)'), zlabel('t')
>> axis('ij')                          % 加上這個指令，注意圖的 y 軸及曲線方向改變了
```

1.8.2.2 曲面及等值線繪圖

如果要畫一個三維的曲面，MATLAB 是以 meshgrid 配合與 mesh 或 surf 指令來繪圖。先要以 meshgrid 產生在 x-y 平面的二維的網格數據，再以一組 z 軸的數據對應到這個二維的網格，即可畫出三維的曲面。以下的例子可說明上述的繪圖過程。

```
>> x=-7.5:0.5:7.5; y=x;                % 先產生 x 及 y 二個陣列
>> [X,Y]=meshgrid(x,y);                % 再以 meshgrid 形成二維的網格數據
>> R=sqrt(X.^2+Y.^2)+eps;              % 加上 eps 可避免當 R 在分母時趨近零時會無法定義
>> Z=sin(R)./R;                        % 產生 z 軸的數據
>> mesh(X,Y,Z)                         % 將 z 軸的變化值以網格方式畫出
>> surf(X,Y,Z)                         % 將 z 軸的變化值以曲面方式畫出
>> mesh(peaks)                         % 直接將以定義的 peaks 函數以網格方式畫出
>> title('Mesh plot of peaks')
```

與三維繪圖有關的還有等值線圖，相關指令為 contour,contour3。contour 是將等值線圖以二維圖表示，其語法有幾個方式。一是 contour(Z),contour(Z,n)，其中 Z 是一個二維矩陣，而 n 為等值線的數目（如果不給即以自動方式設定）。另一種語法則是將 z 軸的值對應到指定的 x,y 軸的值，語法為 contour(X,Y,Z),contour(X,Y,Z,n)，其中 X,Y, Z 代表 x,y,z 軸的數據。contour3 則是將等值線以三維圖表示，其語法與 contour 類似，只是將對應的關鍵字 contour 改成 contour3，其餘部份相同。

以下的例子可以比較 contour,contour3 圖示的不同：

```
>> [X,Y,Z]=peaks;                      % x,y 及 z 軸的數據由 peaks 函數定義
>> subplot(2,2,1)
>> contour(Z,20)                       % 畫出 peaks 的 Z 軸二維等值線圖，20 為等值線的數目
>> subplot(2,2,2)
>> contour(X,Y,Z,20)                   % 畫出 peaks 的二維等值線圖，注意 x,y 軸與上圖不同
>> subplot(2,2,3)
>> contour3(Z,20)                      % 畫出 peaks 的 Z 軸二維等值線圖
>> subplot(2,2,4)
>> contour3(X,Y,Z,20)                  % 畫出 peaks 的三維等值線圖，注意 x,y 軸與上圖不同
```

1.9 Matlab 提供之內建函數介紹

在這章我們將介紹 MATLAB 的內建函數，包括數學函數〔例如三角函數、複數函數、多項式函數等〕，分析數據相關函數〔計算平均值、最大最小值、排序等〕、邏輯/選擇函數〔if-else 等〕和產生亂數方法〔用來模擬隨機發生事件〕。雖然 MATLAB 提供上百種內建函數，萬一你需要使用的特別函數並不在其中，你也可以自行定義函數。

1.9.1 數學函數

在這一節中所介紹的函數是屬於基本的數學函數，包括三角函數、雙曲線函數、複數函數和多項式函數等。

1.9.1.1 常見數學函數

我們在第二章已介紹了加、減、乘、除等簡易的代數運算，除此之外 MATLAB 還提供許多內建函數，如對數函數、三角函數、多項式函數等，方便我們計算。舉例來說，要計算一角度的 sine 值，過程如下：

```
>> angle1=pi/2;
>> b=sin(angle1);                %注意 angle1 為徑度，sin 函數計算值需以徑度表示
>> angle2=90;                    %注意 angle2 為角度
>> b=sin(angle2*pi/180);         %也可在函數內作角度與徑度轉換
>> x=sqrt(2)/2; y=asin(x); y_deg=y*180/pi
>> x =
0.7071
>> y =
0.7854
>> y_deg =
45.0000
```

使用函數須注意幾點。首先函數一定出現在計算等式的右邊，等式左邊是代表這個函數的計算值。此外，一個函數可以被當做另一個函數的引數(argument)。例如：log_x=log(abs(x)) 其中 abs 和 log 皆為內建函數，其意思是先計算 abs(x)，所得值再代入 log 函數。

指令	意義
round(x)	將 x 值進位至最接近的整數
fix(x)	將 x 值進位至最接近 0 的整數
floor(x)	將 x 值進位至最接近-∞的整數
ceil(x)	將 x 值進位至最接近∞的整數
sign(x)	如果 x <0 傳回值為-1，如果 x =0 傳回值為 0，如果 x >0 傳回值為 1
rem(x,y)	傳回 x/y 的餘數，例如 rem(25,4)的值為 1

exp(x)	指數函數
log(x)	以 e ≒2.718282 為底的對數函數，及自然對數
log10(x)	為 10 底的對數函數

其餘的內建函數，用法可以參考 MATLAB 的線上說明或使用手冊。

1.9.1.2 三角和雙曲線函數

至於三角函數和雙曲線函數的使用，和一般數學式相似，其語法也很直接易懂。例如三角函數有：sin(x), cos(x), tan(x), asin(x), acos(x), atan(x), atan2(y,x)。常用到的雙曲線函數有：sinh(x), cosh(x), tanh(x), asinh(x), acosh(x), atanh(x)。上述函數用法請參考 MATLAB 的線上說明或使用手冊。

1.9.1.3 複數

要說明複數的運算，先從解以下的二次方程式的複數根談起

$$f(x)=x^2+4x+13$$
$$x_{1,2}=-2\pm 3\sqrt{-1}=-2\pm 3i$$

上式的根有實部 (-2) 及虛部 (±3)，我們就這個複數的表示法來說明 MATLAB 的複數功能。MATLAB 是以 i 或 j 字元來代表虛部，其它的複數相關函數有 real, imag, conj, abs, angle 等等，詳見線上說明 lookfor complex。如果複數表示為 $x=a+bi$

共軛複數 $\bar{x}=a-bi$, 複數大小 r = $\sqrt{a^2+b^2}$, 複數向量的夾角 θ= $\tan^{-1}(b/a)$

複數實部 $a = r\cos\theta$, 複數虛部 $b = r\sin\theta$, 複數指數表示法 $x=r\,e^{i\theta}$

上述各函數對應 MATLAB 的複數指令為

a=real(x), b=imag(x), $\bar{x}$=conj(x), r=abs(x), θ=angle(x), x=r*exp(i*angle(x))

以下是幾個複數表示式的例子：

```
>> x=1-2*i;                                  % 注意是 2*i 不是 2i
>> real(x)                                   % 列出實部
ans =
1
>> imag(x)                                   % 列出虛部
ans =
-2
>> conj(x)                                   % 計算共軛複數
ans =
1.0000 + 2.0000i
>> abs(x)                                    % 計算複數的大小
ans =
2.2361
>> angle(x)                                  % 計算複數向量的夾角（以徑度表示）
```

```
ans =
-1.1071
>> a=1; b=4; c=13;
>> x1=(-b+sqrt(b^2-4*a*c))/(2*a)                % 以解二次方程式根的公式計算複數根
x1 =
-2.0000 + 3.0000i
>> x2=(-b-sqrt(b^2-4*a*c))/(2*a)
x2 =
-2.0000 - 3.0000i
>> y=exp(i)                                     % 以複數指數方式表示一個複數
y =
0.5403 + 0.8415i
>> y=exp(i*pi*0.75)
y =
-0.7071 + 0.7071i
```

和複數有關的圖以極座標來表示會比一般的卡氏座標要合適，polar 指令可以將數據以極座標方式加以繪圖，其語法為 polar(theta,r)，(theta,r)分別代表極座標上的角度及半徑值。以下的例子說明了 polar 用法：

```
>> t=0:0.01:2*pi;
>> r=sin(2*t).*cos(2*t);
>> polar(t,r)
>> title('Polar plot of sin(2t)cos(2t)')
>> angle=0:2*pi/100:2*pi;
>> r=angle/(2*pi);
>> polar(angle,r)
>> title('Polar plot')
>> grid
```

1.9.1.4 多項式函數

在工程及科學分析上，多項式常被用來模擬一個物理現象的解析函數。之所以採用多項式，是因為它很容易計算。在這章中我們將說明如何做多項式的計算及解多項式的根。
令 $p(x)$ 代表一個多項式如下

$$p(x) = x^3 + 4x^2 - 7x - 10$$

MATLAB 以一最簡便方式代表上述的多項式 p=[1 4 -7 -10]，其中的數值是多項式的各階項（從高到低）的 各個係數，其實 p 也是一個陣列不過是用以代表這個多項式。

有了多項式的表示式後，我們即可來計算其函數值。假設要計算一組數據 x 對應的多項式值，依照一般的函數計算須以下列式子計算：

```
>> p=x.^3+4*x.^2-7*x-10
```

為了能直接運用多項式，可以用函數 polyval 直接做運算，語法為 polyval(p,x)，其中 p 即是代表多項式各階係數 的陣列。因此

```
>> x=linspace(-1,3);
>> p=[1 4 7 -10];
>> v=polyval(p,x);
```

我們接著說明如何對二個多項式做加減乘除運算，以及對多項式微分。當二個多項式間要做加減乘除時，加減運算可以直接進行。假設有二個多項式$a(x)$ 和 $b(x)$定義如下：

$a(x)=x^3+2x^2+3x+4,\quad b(x)=x^3+4x^2+9x+16$

如果多項式 $c(x)$為上述二多項式相加，即 $c(x)=a(x)+b(x)$，因此

$c(x)=2x^3+6x^2+12x+20$

如果是二多項式相減得到的多項式為 $d(x)=a(x)-b(x)$則

$d(x)=-2x^2-6x-12$

而將兩個多項式相乘可以得到一新的多項式$e(x)=a(x)b(x)$：

$e(x)=x^6+6x^5+20x^4+50x^3+75x^2+84x+64$

如果是兩個多項式相除，即 $f(x)=\dfrac{e(x)}{b(x)}=a(x)$：

$f(x)=x^3+2x^2+3x+4$

上述二個運算式不能直接運算，須要另外定義函數 conv 做乘法運算以及函數 deconv 做除法運算。當二多項式相乘，在數學上等於二個陣列做旋積(convolution)運算（因為我們是以陣列來代表一個多項式的各階係數），因此可利用 conv 函數做乘法運算，其語法為 conv(a,b)，其中 a, b 代表二個多項式的陣列。而二多項式相除就相 當於反旋積(de-convolution) 運算，因此有 deconv 函數，其語法稍有不同 [q,r]=deconv(a,b)，其中 q,r 分別代表整除多項式及餘數多項式。

以下就介紹相關範例，來說明二個多項式的加減乘除運算：

```
>> a=[1 2 3 4]; b=[1 4 9 16];
>> c=a+b
c =
2 6 12 20
>> d=a-b
d =
0 -2 -6 -12
>> e=conv(a,b)
e =
1 6 20 50 75 84 64
>> g=e+[0 0 0 c]
g =
1 6 20 52 81 96 84
>> [f,r]=deconv(e,b)
f =
1 2 3 4
r =
0 0 0 0 0 0 0                          % 因為是整除所以餘數多項式的各係數皆為零
>> [h,r]=deconv(g,a)
```

```
f =
1 4 9 18
r =
0 0 0 0 2 6 12                    % 餘數多項式為 2*x^2 + 6*x + 12
```

ex1:
兩個多項式 p1(x)=6x2+2x-7,p2(x)=3x4-5
計算 a. p1*p2　　b.p2 除以 p1 的商及餘式

1.9.2 數據分析函數

將工程及科學實驗所量測的數據做分析，是實驗評估一項的極重要的工作。這樣的分析工作可以從從簡單的運算例如計算平均值，到繁複的矩陣運算例如計算標準差(deviation)。這些量測可稱為統計量測，因為量測這些數據即含有統計性質。比方說我們量測每日的相對溼度，它的變化是和氣溫高低、晴天或是下雨、地形、緯度等息息相關，這些因素都會不時的改變。就像我們可以從統計資料中計算其特性，我們亦可以利用電腦依照預設的統計特性來產生特定的數據（例如亂數）。在這章我們將介紹一些基礎的統計概論，及一些 MATLAB 基本分析數據的函數，以及如何產生亂數。

1.9.2.1 極值、平均、總和、連乘及排序

首先介紹幾個分析函數，利用這些函數可以讓我們在分析數據極為方便。這些函數有：最大值 max，最小值 min，平均值 mean，一組數據的中位數 median，總和值 sum，連乘值 prod，累積總和值 cumsum，累積連乘值 cumprod，排序函數 sort。它們的使用方式如下

max(x)	找出 x 陣列的最大值
max(x,y)	找出 x 及 y 陣列的最大值，會有二個極值分屬 x 及 y 陣列
[y,i]=max(x)	找出 x 陣列的最大值以 y 顯示，其在 x 陣列的位置以 i 顯示
min(x)	找出 x 陣列的最小值
min(x,y)	找出 x 及 y 陣列的最小值，會有二個極值分屬 x 及 y 陣列
[y,i]=min(x)	找出 x 陣列的最小值以 y 顯示，其在 x 陣列的位置以 i 顯示
mean(x)	找出 x 陣列的平均值
median(x)	找出 x 陣列的中位數
sum(x)	計算 x 陣列的總和值
prod(x)	計算 x 陣列的連乘值
cumsum(x)	計算 x 陣列的累積總和值
cumprod(x)	計算 x 陣列的累積連乘值

以下是幾個例子：

```
>>rains =[126.8 148.5 173.0 148.4 194.7 208.9;328.8 300.7 268.3 210.5 278.4 321.5 ]
>> avg_rain=mean(rains)          % 將 rains 陣列中的每一行的平均值列出
avg_rain =
227.8000 224.6000 220.6500 179.4500 236.5500 265.2000
```

```
>> avg_rain=mean(avg_rain)      % 將上述陣列中的平均值列出
avg_rain =
225.7083
>> max_rain=max(rains)          % 將 rains 陣列中的每一行的最大值列出
max_rain =
328.8000 300.7000 268.3000 210.5000 278.4000 321.5000
>> [max_rain,x]=max(rains)      % 將 rains 陣列中的每一行的最大值及其位置列出
max_rain =
328.8000 300.7000 268.3000 210.5000 278.4000 321.5000
x =
2 2 2 2 2 2
>> min_rain=min(rains)          % 將 rains 陣列中的每一行的最小值列出
min_rain =
126.8000 148.5000 173.0000 148.4000 194.7000 208.9000
>> s_sort=sort(rains)           % 將 rains 陣列的值由小到大做排序
s_sort =
126.8000 148.5000 173.0000 148.4000 194.7000 208.9000
328.8000 300.7000 268.3000 210.5000 278.4000 321.5000
>> x=[1 2 3 4 5];
>> sum(x)                       % 將 x 陣列的值做總和
ans =
15
>> prod(x)                      % 將 x 陣列的值做連乘
ans =
120
>> cumsum(x)                    % 將 x 陣列的值累積後做總和
ans =
1 3 6 10 15
>> cumprod(x)                   % 將 x 陣列的值累積後做連乘
ans =
1 2 6 24 120
```

1.9.2.2 歷程分佈圖函數

有一個繪圖函數與數據分析有關，稱為歷程分佈圖函數 (histogram)，我們可以用它畫出一組數據的範圍及其如何分佈。它是將數據中的極小到極大值標示在橫軸（即是數據的範圍），再將各個數據出現的次數對應該數據值（橫軸）來畫在縱軸（即是數據分佈的比例）。histogram，預設值為 10 個長條。MATLAB 用來產生歷程分佈圖函數指令為 hist。以下是幾個例子：

```
>> x=-3:0.1:3;
>> y=sin(x);        % 注意 x 是徑度
>> hist(y)          % 畫出 sin(y)的 histogram，橫軸代表 y 的極值[-1,1]，縱軸代表 y 的個數
>> hist(y,25)       % 將預設 10 個長條改為 25 個，注意縱軸的值改變，為什麼？
>> hist(y,x)        % 將橫軸上下限改為-3 到 3，注意縱軸的值也改變，為什麼？
```

1.9.2.3 曲線擬合

若有一組實驗數據如下：

x=[-3 -1 0 2 5.5 7]　　y=[3.3 4.5 2.0 1.5 2.5 1.2]

希望分別利用 3,4,5 次多項式對此資料作曲線擬合(curve fitting)

```
x=[-3 -1 0 2 5.5 7];
y=[3.3 4.5 2.0 1.5 2.5 1.2];
p3=polyfit(x,y,3);
p4=polyfit(x,y,4);
p5=polyfit(x,y,5);
xaxis=-3.5:0.1:7.2;
p3curve=polyval(p3,xaxis);
p4curve=polyval(p4,xaxis);
p5curve=polyval(p5,xaxis);
plot(xaxis,p3curve,'--',xaxis,p4curve,'-.',xaxis,p5curve,'-',x,y,'*');
```

1.9.3 選擇指令及函數

到目前為止所介紹 MATLAB 程式的執行方式都是序列式的 (sequential)，即是一個指令接著一個指令的執行。但是有許多時候不能僅以序列式方式執行指令，例如我們須要一個有選擇的指令能讓我們依設定的條件來決定是否執行某些指令，或是要做些重複性或是有規則變化的運算或是資料處理，這些工作是以「控制流程」來做到。透過控制流程的指令可以將整個程式精簡許多，且變得更有效率。

1.9.3.1 關係及邏輯運算

在執行關係及邏輯運算時，MATLAB 將輸入的不為零的數值都視為真 (True)而為零的數值則視為否 (False)。 運算的輸出值將判斷為真者以 1 表示而判斷為否者以 0 表示。

MATLAB 提供以下的關係判斷及邏輯的運算元：

符號	關係的意義	符號	關係的意義
<	小於	~=	不等於
<=	小於等於	&	邏輯 and
>	大於	|	邏輯 or
>=	大於等於	~	邏輯 not
==	等於		

上述的各個運算元須用在二個大小相同的陣列或是矩陣的比較，以下有幾個例子：

```
>> a=1:5, b=5-a,
a =
1 2 3 4 5
b =
4 3 2 1 0
>> tf= a>4
tf =
0 0 0 0 1
>> tf= a==b
tf =
0 0 0 0 0
>> tf= b-(a>2)
tf =
4 3 1 0 -1
>> tf= ~(a>4)
```

```
tf =
1 1 1 1 0
>> tf= (a>2)&(a<6)
tf =
0 0 1 1 1
```

以下是算式利用關係及邏輯運算產生一不連續的訊號

```
>> x=linspace(0,10,100);                % 產生數據
>> y=sin(x);                            % 產生 sine 函數
>> z=(y>=0).*y;                         % 將 sin(x) 的負值設為零
>> z=z + 0.5*(y<0);                     % 再將上式的值加上 0.5
>> z=(x<8).*z;                          % 將大於 x=8 以後的值設為零
>> hold on
>> plot(x,z)
>> xlabel('x'),ylabel('z=f(x)')
>> title('A discontinuous signal')
>> hold off
```

除了上述的運算元之外，尚有以下的邏輯關係函數：xor(x,y), any(x), all(x), isnan(x), isinf(x), finite(x), find(x)，其 使用方式詳見線上說明。

1.9.3.2 if-else-end 語法

與 if 有關的語法有：只含 if 的結構，包括 if 及 else 的結構，包括 if、else 及 elseif 的結構。以下就是幾個例子。

簡易 if 結構：
if 條件式
  命令敘述
end
	例:
	if a < 50
	count = count + 1;
	sum = sum + a;
	end

巢狀 if 結構：
if 條件 1
  命令敘述 1
  if 條件 2
  命令敘述 2
  end
  命令敘述 3
end
	例:
	if a < 50
	  count = count + 1;
	  sum = sum + a;
	  if b > a
	  b= 0;
	  end
	end

else 結構：
if 條件
  命令敘述 1
else
  命令敘述 2
end
	例:
	if d <= 10
	  speed = 0.425 + 0.00175*d.^2;
	else
	  speed = 0.625 + 0.12*d - 0.00025*d.^2;
	end

elseif 結構： if 條件 1	

```
   命令敘述 1
elseif 條件 2
   命令敘述 2
elseif 條件 3
   命令敘述 3
end
```

1.9.3.3 迴圈

在一些運算過程中我們常須要做重複的計算，這些算式只是其中變數改變但是運算式相同，我們可以迴圈 (loop) 來計算。MATLAB 提供二種迴圈函數：for 迴圈, while 迴圈。

a.For

for 迴圈是用在須重複執行且執行次數有一定的算式，它的結構如下：

```
for index = array
 命令敘述
end
```

如果我們要計算一纜車離鐵塔的速度 (v)，它的速度計算方式與且鐵塔的距離 (d)有關，假設以 10 公尺為判斷值，則速度計算分為二個算式：

假設有一個陣列 d 為纜車到鐵塔的距離，則以下的 for 迴圈可計算速對應的速度

```
d=[1:1:10];
for k = 1:length(d)
   if d(k) <= 10
      velocity = 0.425 + 0.00175*d(k)^2;
   else
      velocity = 0.625 + 0.12*d - 0.00025*d(k)^2;
   end
   fprintf('d= %f velocity= %f\n',d(k),velocity)
end
```

另外幾個例子

```
>> for n=1:10
x(n)=sin(n*pi/10);
end
>> disp(x)
>> for n=1:5
for m=5:-1:1
A(n,m)=n^2+m^2;
end
disp(n)
end
>> disp(A)
```

但是如果可以用陣列或是矩陣運算來取代以 for 迴圈計算，就應採用前者因為計算速度快多了。上述的例子可改為

```
>> n=1:10;
>> x=sin(n*pi/10);
```

使用 for 迴圈的規則如下：

1. 上述的 for 迴圈中的指標 (index) 須為是一變數。

2. 如果 array 代表陣列是空無一物，則迴圈不會被執行，例如 k=1:0。
3. 如果 array 代表陣列是一純量，則迴圈會被執行一次，例如 k=1:1。
4. 如果 array 代表陣列是一向量，則迴圈會被依序的執行，例如 k=1:b, b=[1 3 5]。
5. 如果 array 代表陣列是一矩陣，則迴圈會被逐行依序的執行，例如 k=1:B, B=[1 2; 3 4]。
6. for 完整的語法為： for k = first:increment:last，其中的 first, increment, last 分別為初始值，增量，終止值。而迴圈被執行的次數由以下的算式決定：

floor((last-first)/increment)+1

如果計算得到的值為負，則迴圈不被執行。

b.While

while 語法如下：

```
while  條件式
  指令敘述
end
```

例

```
sum = 0;
k = 1;
while x(k) >= 0 & k <= length(k)
sum = sum + x(k);
k = k+1;
end
```

1.9.4 使用者自定函數

我們在前面提過的 M-file 除了可以撰寫程式外，還有另一個重要的用途，就是可以用來定義函數。這樣的函數稱為 M-檔定義的函數，然後儲存起來，就可以和那些內建的函數（如 sin, cos,log 等）一樣的自由使用。舉例來說，我們可以定義一函數 cirarea 是計算圓的面積，以下的 M-file: cirarea.m 就是定義這個函數

```
% M-file function, cirarea.m
% Calculate the area of a circle with raduis r
% r can be a scalar or an array
function c=cirarea(r)
c=pi*r.^2;
```

令一個例子是 MATLAB 內建的函數 linspace

```
function y = linspace(d1, d2, n)
% LINSPACE Linearly spaced vector.
% LINSPACE(x1, x2) generates a row vector of 100 linearly
% equally spaced points between x1 and x2.
% LINSPACE(x1, x2, N) generates N points between x1 and x2.
%
% See also LOGSPACE, :.
% Copyright (c) 1984-94 by The MathWorks, Inc.
if nargin == 2
n = 100;
end
y = [d1+(0:n-2)*(d2-d1)/(n-1) d2];
```

M-file 定義的函數有其語法的一些規定：

1. 第一行指令以 function 這個字做為起頭，接著是輸出的變數，等號，函數名稱，輸入的變數是接著函數名 稱放在括號之內。function out1=userfun(in1)，這行的 out1 是輸出的變數，userfun 是函數名稱，in1 是輸入的 變數。function [out1, out2]= serfun(in1, in2) 如果輸出變數 [out1,out2] 和輸入變數 (in1, in2)不只一個時，則在 輸出變數部份須加上 [ ]。
2. 上述的輸入變數是經由使用函數時輸入的，而輸出的變數即是函數傳回的值。
3. 函數名稱的取法的規定與一般變數相同。
4. 在定義函數程式之前，最好加上註解行來說明這個函數的特色及如何使用，如此的話使用指令如 help cirarea，該函數的註解行會出現在指令視窗。

```
>> r=1:3;
>> ar=cirarea(r)                    % 呼叫 cirarea.m 函數，以陣列 r 為輸入變數
ar =
3.1416 12.5664 28.2743
>> disp(ar)                         % 指令 disp 可以將變數值直接列出
3.1416 12.5664 28.2743
```

1.9.5 矩陣運算函數

1.9.5.1 基本矩陣運算元

MATLAB 的運算是以陣列(array)及矩陣 (matrix) 方式在做運算，而這二者在 MATLAB 的基本運算性質不同，陣列強調元素對元素的運算，而矩陣則採用線性代數的運算方式。我們就來說明矩陣運算的特點。

以下將陣列及矩陣的運算符號及其意義列出

陣列運算符號	矩陣運算符號	功能
+	+	加
-	-	減
.*	*	乘
./	/	左除
.\	\	右除
.^	^	次方
.'	'	轉置

利用這些運算符號即可進行以下的矩陣運算。

```
>> A=[2 5 1; 7 3 8; 4 5 21; 16 13 0];
>> A'                               % A 的轉置矩陣
A =
2 7 4 16
5 3 5 13
1 8 21 0
>> A=[4 -1 3]; B=[-2 5 2];
>> dot_prod = sum(A.*B)             % 二個陣列做內積
dot_prod =
-7
>> c=dot(A,B)                       % 以 dot 函數也可做內積運算
c =
-7
```

```
>> A=[4; -1; 3];
>> dot_prod = sum(A'.*B);                    % 如果 A 是行陣列則先做轉置，再做內積
>> F=[2 5 -1]; G=[0 1 -3];
>> out_prod=F'*G;                            % 二矩陣做外積
>> A=[2,5,1; 0,3,-1];
>> B=[1,0,2; -1,4,-2; 5,2,1];
>> C=A*B                                     % 矩陣相乘，注意二個矩陣的大小須相容
C =
2 22 -5
-8 10 -7
>> A=[2 1; 4 3];
>> A^2                                       % 矩陣次方
ans =
4 1
16 9
```

1.9.5.2 矩陣多項式

函數 polyvalm 是以矩陣方式做多項式函數計算，有別於 polyval 是以陣列方式計算函數值。它的語法為 polyvalm(a,X)，其中 X 為一矩陣而 a 則是一多項式。以下的例子可說明其用法。

```
>> X=[1 1 1; 2 2 2; 3 3 3];
>> a=[1 1 1];                          % 注意 a=X*X+X+I
>> f=polyvalm(a,X)
f =
8 7 7
14 15 14
21 21 22
```

1.10 求方程式的根

1.10.1 多項式的根

一個多項式視其階數而定，它的根可以有一個到數個，可能為實數也可能是複數。要求一高階多項式的根往往須借助數值方法，所幸 MATLAB 已將這些數值方法寫成一函數 roots(p)，我們只要輸入多項式的各階係數（以 p 代表）即可求解到對應的根。

```
>> p=[1 3 2];
>> r=roots(p)
r =
-2
-1
>> p=[1 -12 0 25 116];        % 注意二階項係數為零須要輸入，否則多項式的階數就不對
>> r=roots(p)                 % 有實數根及複數根
r =
11.7473
2.7028
-1.2251 + 1.4672i
-1.2251 - 1.4672i
```

與 roots 相關的函數尚有 poly, real，這二個函數的用途是要驗算求解的根展開能求得原多項式。例如有一個二次方程式的根為 2, 1，則以下式計算原多項式

poly 函數就是在求出多項式的各階係數，其語法為 poly(r)，其中 r 是代表根的陣列。而 real 則是用來去除因計算時產生的假虛部係數，為何會有此種情形請參考以下的例子。

```
>> r=[-2 1];
>> pp=poly(r)                      % pp=(x+2)(x-1)=x^2+3x+2
pp =
1 3 2
>> p=[1 -4 6 -4];
>> r=roots(p)
r =
2.0000 1.0000 + 1.0000i 1.0000 - 1.0000i
>> pp=poly(r)                      % 這個多項式的係數與原多項式 p 相同
pp =
1 -4 6 -4
>> pp=[1 7 12 9];                  % 再看另一個多項式
>> r=roots(pp)
r =
-4.9395
-1.0303 + 0.8721i
-1.0303 - 0.8721i
>> pp=poly(r)                      % 注意因計算的誤差會有假虛部產生
pp =
1.0000 7.0000 12.0000 9.0000 + 0.0000i
>> pp=real(pp)                     % 可以 real 將假虛部去除，將原多項式還原
pp =
1.0000 7.0000 12.0000 9.0000
```

1.11 練習

1.8.1 二維繪圖練習一
1.8.2 三維繪圖練習一
1.8.3 三維繪圖練習二
1.11.1 解常微分方程
1.11.2 解積分方程
1.11.3 將數據由檔案讀入/寫入檔案
1.11.4 方塊圖化簡
1.11.5 二階系統步階響應圖
1.11.6 二階系統暫態特性
1.11.7 根軌跡圖

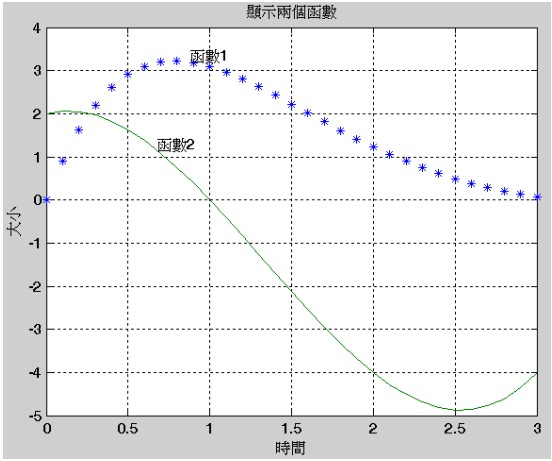
範例:

[1_8_1].繪出如下要求之圖形

1.將函數 f1(t)=10sin(t)$e^{(-t)}$ 之值用"*"表示，f2(t)=t^3-4t^2+t+2 之值用實線表示

2.大區間為 0 到 3,間隔為 0.1

3.其餘參考下圖



sol:

程式如下:

```
t=0:0.1:3;
f1=exp(-t).*sin(t)*10;
f2=t.^3-4*t.^2+t+2;
plot(t,f1,'*',t,f2)
title('顯示兩個函數')
xlabel('時間')
ylabel('大小')
grid
gtext('函數 1')
gtext('函數 2')
pause
```

練習:

[1_8_1P].

使用全對數刻度(loglog)及半對數刻度(semilogx,semilogy)把

f1(t)=100$e^{(-t)}$

f2(t)=t^3+4t^2+t+2

在區間由 0.1 到 10 取 50 點時的兩條曲線繪出，其他要求如範例。

[1_8_2P].

考慮兩個函數

$f_1(t)=10e^{(-0.2t)}\cos(t)$

$f_2(t)=\sin(t)$ 其中 t 由 0 到 10 間隔 0.1

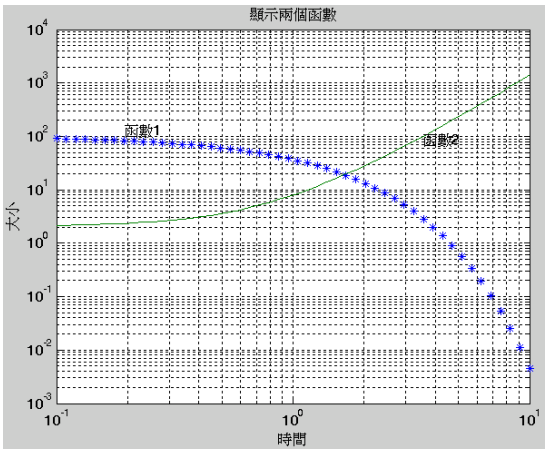
是利用 Plotyy 繪出此兩個不同刻度的 f1 及 f2

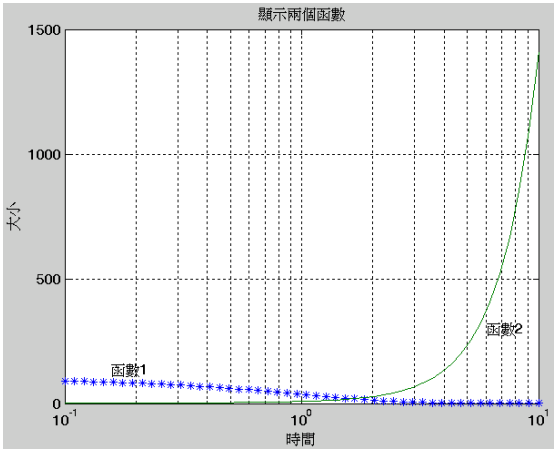
[1_8_3P].

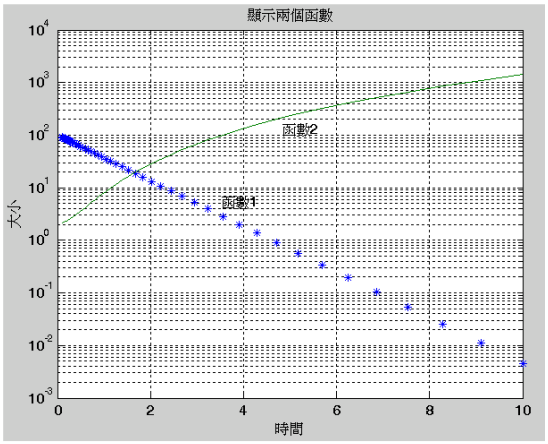
利用指令 subplot 同時繪出兩個圖形,左半邊的圖為[1_8_1P]中的全對數刻度(loglog)而右半邊的圖形為半對數刻度(semilogx)。

[1_8_1P]-參考答案

```
t=logspace(-1,1);
f1=exp(-t)*100;
f2=t.^3+4*t.^2+t+2;
loglog(t,f1,'*',t,f2)
title('顯示兩個函數')
xlabel('時間')
ylabel('大小')
grid
gtext('函數 1')
gtext('函數 2')
pause
semilogx(t,f1,'*',t,f2)
title('顯示兩個函數')
xlabel('時間')
ylabel('大小')
grid
gtext('函數 1')
gtext('函數 2')
pause
semilogy(t,f1,'*',t,f2)
title('顯示兩個函數')
xlabel('時間')
ylabel('大小')
grid
gtext('函數 1')
gtext('函數 2')
pause
```

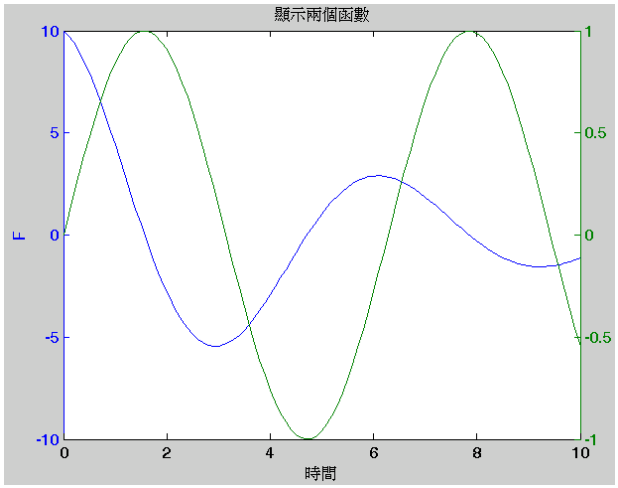






[1_8_2P]-參考答案

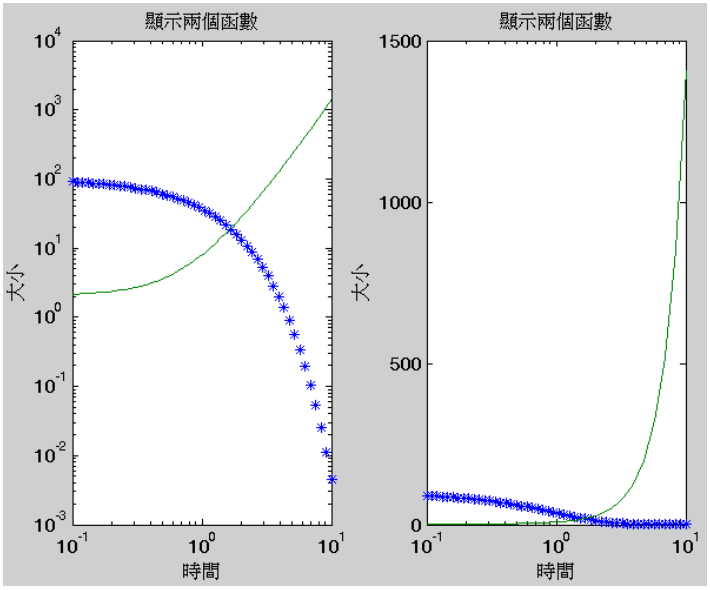
```
t=0:0.1:10;
f1=10*exp(-0.2*t).*cos(t);
f2=sin(t);
plotyy(t,f1,t,f2)
title('顯示兩個函數')
xlabel('時間')
ylabel('F')
pause
```



[1_8_3P]

```
t=logspace(-1,1);
f1=exp(-t)*100;
f2=t.^3+4*t.^2+t+2;
subplot(121)
loglog(t,f1,'*',t,f2)
title('顯示兩個函數')
xlabel('時間')
ylabel('大小')
subplot(122)
semilogx(t,f1,'*',t,f2)
```

title('顯示兩個函數') xlabel('時間') ylabel('大小') pause	



[1_8_2].繪製函數圖形，其函數為

$z=e^{0.2Y}\sin(Y)\cos(X)$

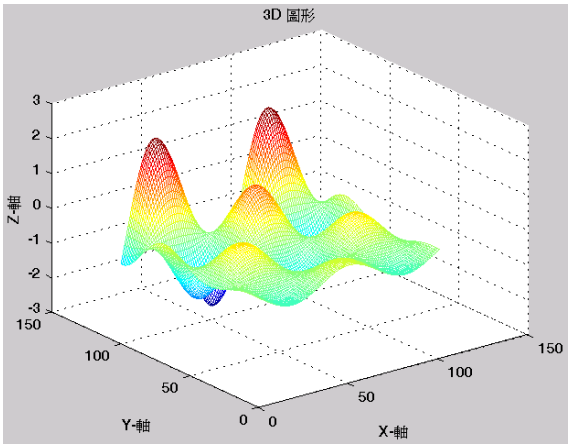
其中-5<X<5,-5<Y<5,間隔均為 0.1

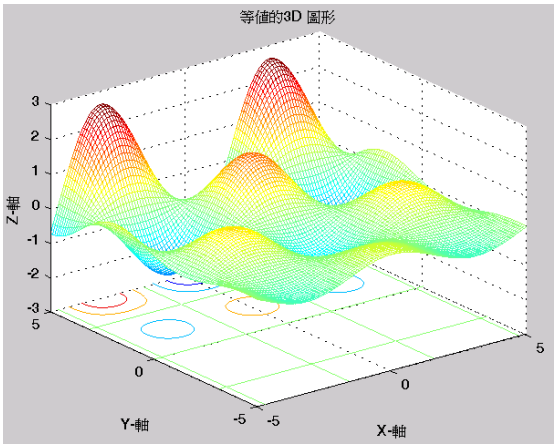
分別使用指令 mesh、meshc 及 meshz 來繪製此函數

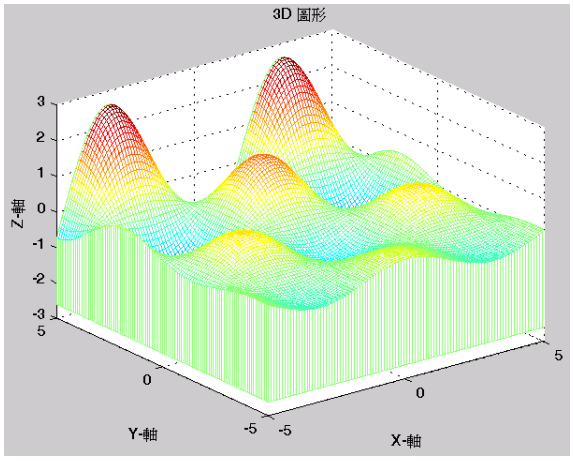
sol:

程式如下:

```
clear
x=-5:0.1:5;
y=-5:0.1:5;
[X Y]=meshgrid(x,y);
Z=exp(0.2*Y).*sin(Y).*cos(X);
mesh(Z)
xlabel('X-軸')
ylabel('Y-軸')
zlabel('Z-軸')
title('3D 圖形')
pause
meshc(X,Y,Z)
xlabel('X-軸')
ylabel('Y-軸')
zlabel('Z-軸')
title('等值的 3D 圖形')
pause
meshz(X,Y,Z)
xlabel('X-軸')
ylabel('Y-軸')
zlabel('Z-軸')
title('3D 圖形')
pause
```







練習:

[1_8_4P]

已知函數

$f(x,y)=(x^2-y^2)\sin(0.5x)$

其中 x,y 均從 1 到 10 間隔 0.1

試利用 mesh 及 surfc 繪出此函數來，並要求得加上 xlabel,ylabel,zlabel,title 及 grid 在圖形上。

[1_8_3].考慮一非線性函數為

$Z=e^{0.1y}\sin(y)\cos(x)$

其中: -10<x<10,間隔 0.1;-10<y<10,間隔 0.1

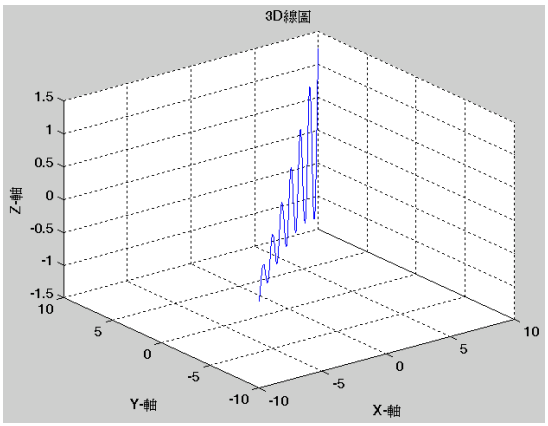
試利用 plot3 繪出，比較二維圖及三維圖的差異？並利用 mesh 繪出此函數的立體圖及以 view 看不同角度的 mesh 圖。

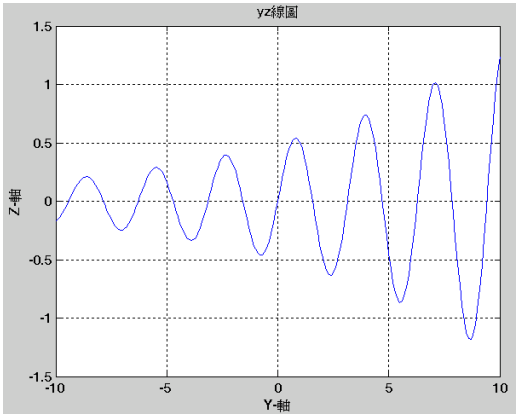
sol:

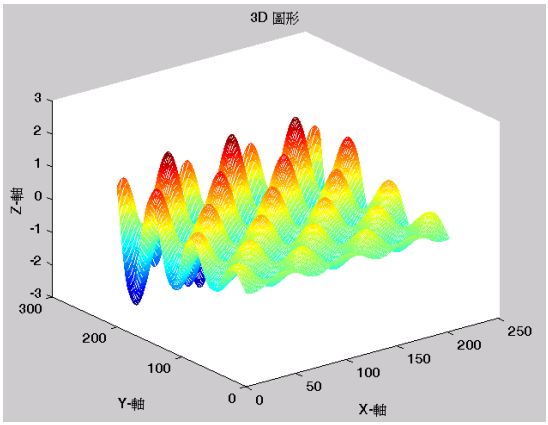
程式如下:

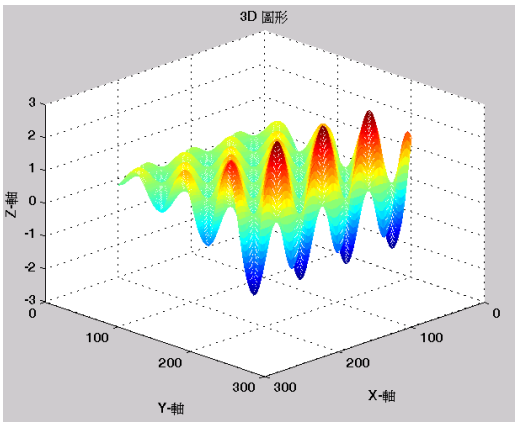
```
clear
x=-10:0.1:10;
y=-10:0.1:10;
z=exp(0.1*y).*sin(y).*cos(x);
plot3(x,y,z)
xlabel('X-軸')
ylabel('Y-軸')
zlabel('Z-軸')
title('3D 線圖')
grid
pause
plot(y,z)
xlabel('Y-軸')
ylabel('Z-軸')
title('yz 線圖')
grid
pause
x=-10:0.1:10;
y=-10:0.1:10;
[X Y]=meshgrid(x,y);
Z=exp(0.1*Y).*sin(Y).*cos(X);
mesh(Z)
xlabel('X-軸')
ylabel('Y-軸')
zlabel('Z-軸')
title('3D  圖形')
grid
pause
x=-10:0.1:10;
y=-10:0.1:10;
[X Y]=meshgrid(x,y);
Z=exp(0.1*Y).*sin(Y).*cos(X);
mesh(Z)
xlabel('X-軸')
ylabel('Y-軸')
zlabel('Z-軸')
title('3D  圖形')
xview=2;
yview=2;
zview=1.5;
view([xview,yview,zview])
```

pause	









1.11.1 解常微分方程

(1).範例說明

1.求 $y''' + y'' + y' = 0$ 的 ODE

令 $y_1=y$　　$y_2=y'$　　$y_3=y''$

則原式可寫成 $y_1' = y_2$　$y_2' = y_3$　　$y_3' = -y_3 - y_2$

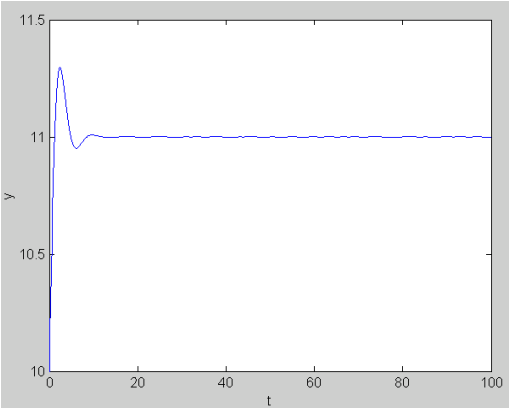
ODE 檔可寫成

```
function ydot=d1fun(t,y);
ydot=[y(2);y(3);-y(3)-y(2)]
```

解題

```
clear
y0=[10;1;0];
[t,y]=ode45('d1fun',[0 100],y0);
plot(t,y(:,1))
xlabel('t')
ylabel('y')
```

執行結果



(2).練習

求以下的常微分方程

1. $y''' - 3y'' + 4y' = 0$

1.11.2 解積分方程

(1).範例說明

1.求 $y=\int_0^{10}\frac{1}{x^3-2x+4}$ 的積分程式

(a).Matlab 程式

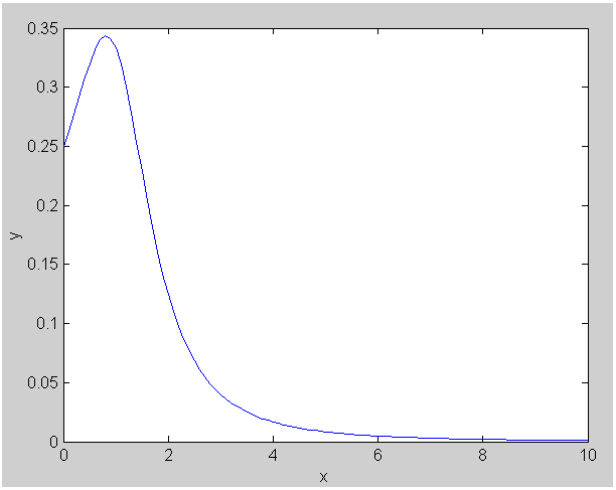
積分方程

```
function y=d2fun(x);
y=1./(x.^3-2.*x+4);
```

解題

```
clear
x_a=0:0.1:10;
for i=1:length(x_a);
      x=x_a(i);
      int(i)=quad('d2fun',0,x);
      y(i)=d2fun(x);
end
plot(x_a,y)
xlabel('x')
ylabel('y')
```

(b).執行結果



(2).練習

求以下之積分方程

1. $y=\int_0^{5}\frac{1}{x^3+3x^2-2x+4}$

1.11.3 將數據由檔案讀入/寫入檔案

(1).範例說明

$$A=\begin{bmatrix}10 & 20 & 30\\ 40 & 50 & 60\\ 70 & 80 & 90\end{bmatrix}\quad B=\begin{bmatrix}100 & 110 & 120\\ 130 & 140 & 150\\ 160 & 170 & 180\end{bmatrix}\quad C=\begin{bmatrix}C_{11} & C_{12} & C_{13}\\ C_{21} & C_{22} & C_{23}\\ C_{31} & C_{32} & C_{33}\end{bmatrix}=A*B$$

1.A,B 矩陣值如上所示,存於 testdata1.txt 檔案中,將其讀入 Matlab 計算兩矩陣相乘的結果存於 C 矩陣並存於 outdata1.txt 中.

testdata1.txt	outdata1.txt
10 20 30 40 50 60 70 80 90 100 110 120 130 140 150 160 170 180	C_{11} C_{12} C_{13} C_{21} C_{22} C_{23} C_{31} C_{32} C_{33}

(a).Matlab 程式

```
function ch1_6_1
fid=fopen('testdata1.txt','r+');
A=fscanf(fid,'%d',[3 3])'
B=fscanf(fid,'%d',[3 3])'
C=A*B
fclose(fid);
fid=fopen('outdata1.txt','w');
fprintf(fid,'%4d %4d %4d \r\n',C');
fclose(fid);
```

(b).輸出結果

outdata1.txt		
8400	9000	9600
20100	21600	23100
31800	34200	36600

(2).練習

1.假設實驗得到一組數據存於 Test1.txt 中，內容如下所示(共 20 筆)

t	y	t	y
0.1	10	3.6	7.5
0.5	13	3.9	3.6
0.7	8	4.1	7.2
1.2	9	4.2	2.9
1.6	1.7	4.3	9.1
2.3	3.9	4.5	2.9
2.6	11	4.7	2.7
2.9	10.6	5.2	2.0
3.0	8.5	5.5	1.8
3.2	3.4	5.7	1.5

將以上資料輸入 matlab 中以 t 為橫座標,y 為縱座標繪出曲線圖.

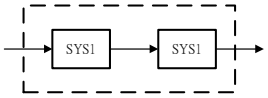
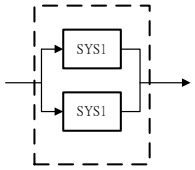
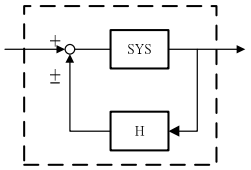
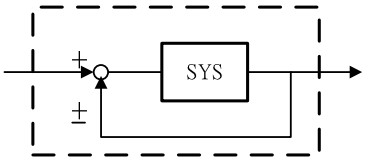
1.11.4 方塊圖化簡

(1).相關指令說明

1.系統基本方塊圖連結方式

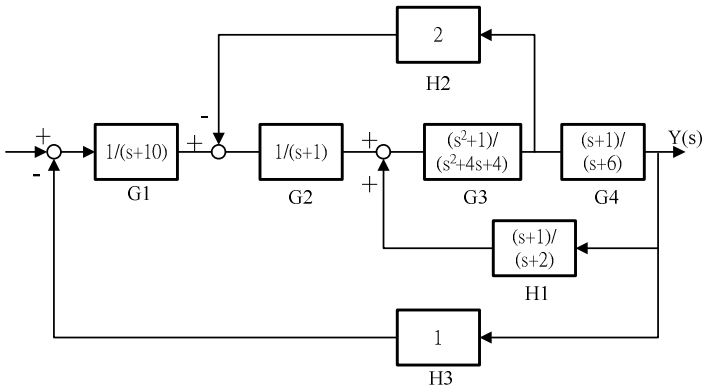
函數	說　明
series	系統以串聯方式連接
paralell	系統以並聯方式連接
feedback	系統以回授方式連接
cloop	系統以閉迴路方式連接

2.使用方式

函數	用　法	示意圖
series	sys=series(sys1,sys2,...)	
paralell	sys=paralell(sys1,sys2,...)	
feedback	sys=feedback(sys,H,±1)	
cloop	[numc,denc]=cloop(num,den, ±1)	

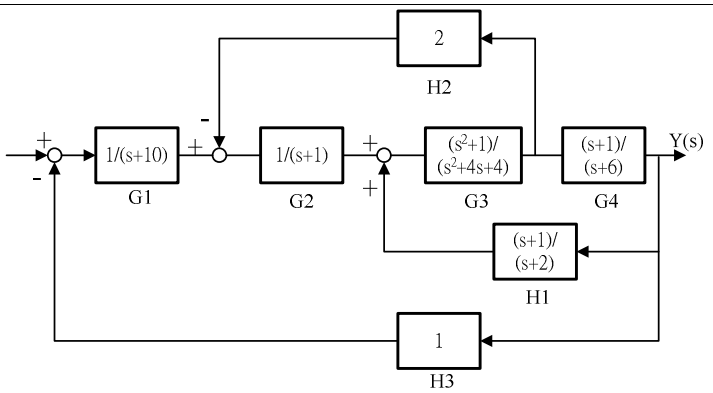
(2).範例說明

1.求轉移函數 $G(s)=\dfrac{Y(s)}{R(s)}$?



(a).Matlab 程式

```
1.function ch1_7
2.g1n=[1];g1d=[1 10];
3.g2n=[1];g2d=[1 1];
4.g3n=[1 0 1];g3d=[1 4 4];
5.g4n=[1 1];g4d=[1 6];
6.h1n=[1 1];h1d=[1 2];
7.h2n=[2];h2d=[1];
8.h3n=[1];h3d=[1];
9.h2np=conv(h2n,g4d);h2dp=conv(h2d,g4n);
10.G1=tf(g1n,g1d);G2=tf(g2n,g2d);
11.G3=tf(g3n,g3d);G4=tf(g4n,g4d);
12.H1=tf(h1n,h1d);H2p=tf(h2np,h2dp);H3=tf(h3n,h3d);
13.G34=series(G3,G4);
14.G34c=feedback(G34,H1,+1);
15.G2p=series(G2,G34c);
16.G2c=feedback(G2p,H2p,-1);
17.G1p=series(G1,G2c);
18.G1c=feedback(G1p,H3,-1);
19.Gs=G1c
```

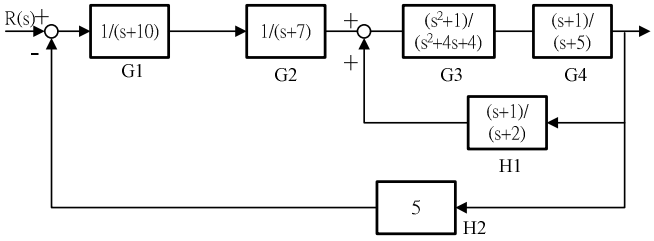


(b).結果

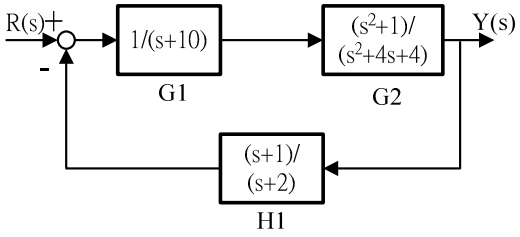
```
>>
Transfer function:
            s^5 + 4 s^4 + 6 s^3 + 6 s^2 + 5 s + 2
-----------------------------------------------------------------
12 s^6 + 205 s^5 + 1066 s^4 + 2517 s^3 + 3128 s^2 + 2196 s + 712

>>
```

(3).練習

1.



2.



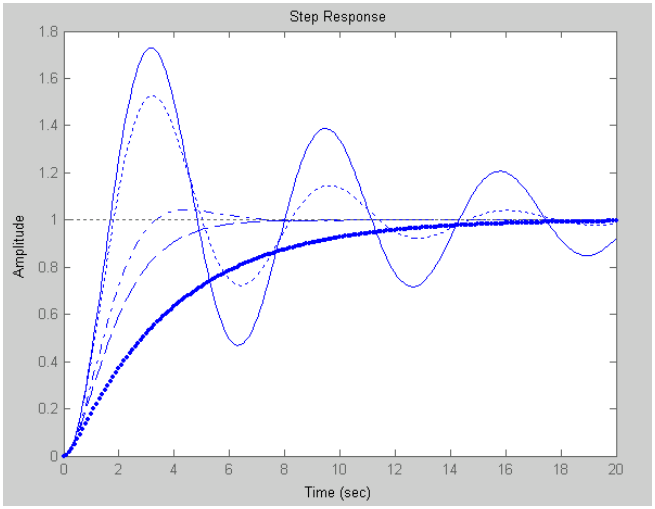
1.11.5 二階系統步階響應圖

(1).範例說明

1.Matlab 程式

```
1.function ch1_1
2. t=[0:0.1:20];wn=1;
3.z1=0.1;[num1,den1]=ord2(wn,z1);sys1=tf(num1,den1);
4.z2=0.2;[num2,den2]=ord2(wn,z2);sys2=tf(num2,den2);
5.z3=0.707;[num3,den3]=ord2(wn,z3);sys3=tf(num3,den3);
6.z4=1;[num4,den4]=ord2(wn,z4);sys4=tf(num4,den4);
7.z5=2;[num5,den5]=ord2(wn,z5);sys5=tf(num5,den5);
8.step(sys1,'-',sys2,':',sys3,'-.',sys4,'--',sys5,'.',t)
```

2.執行結果



1.11.6 二階系統暫態特性

(1).範例說明

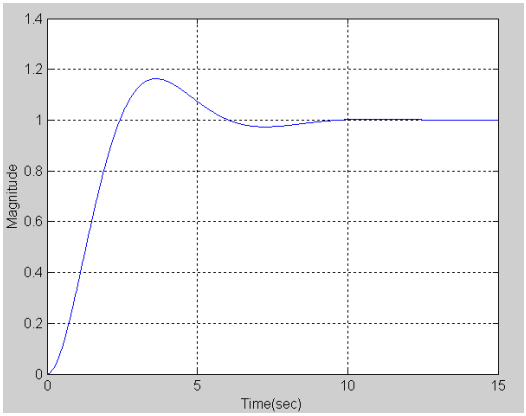
1. $L(s)=\dfrac{1}{s(s+1)}$ 時，顯示此閉迴路系統的步階響應，由步階響應波形讀取 T_r,T_d,T_s,O_s,T_p 之值

$$W(s)=\frac{L(s)}{1+L(s)}=\frac{1}{s^2+s+1}$$

(a).Matlab 程式

```
1.function ch1_2
2.num=[1];den=[1 1 1];
3.t=0:0.01:15;
4.[y,x,t]=step(num,den,t);
5.plot(t,y),grid
6.xlabel('Time(sec)'),ylabel('Magnitude')
7.[peak,M]=max(y);
8.Tp=t(M)
9.PO=100*(peak-1)
10.L=find(abs(y-1)>=0.02);
11.Ts=t(length(L))
12.T1=find(y<0.1);T2=find(y<0.5);
13.T3=find(y<0.9);
14.Tr=t(length(T3))-t(length(T1))
15.Td=t(length(T2))
```

(b).執行結果



Tp	3.6300
PO	16.3033
Ts	5.28
Tr	1.6400
Td	1.2900

(2).練習

顯示以下轉移函數步階響應，並由步階響應波形讀取 T_r,T_d,T_s,O_s,T_p 之值

$$1.W(s)=\frac{24.542}{s^2+4s+24.542}\quad 2.W(s)=\frac{245.42}{(s+10)(s^2+4s+24.542)}$$

$$3.W(s)=\frac{73.626}{(s+3)(s^2+4s+24.542)}$$

1.11.7 根軌跡圖

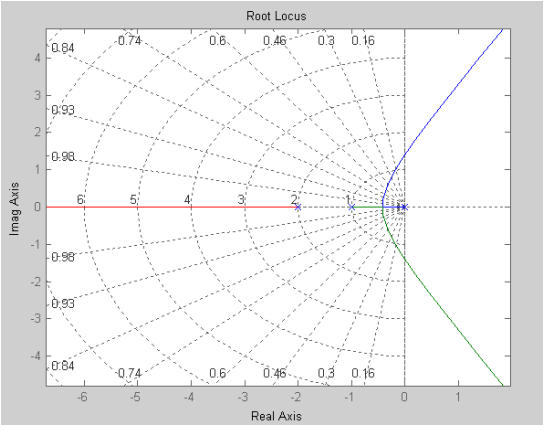
(1).範例說明

1.繪出轉移函數 $L(s)=\dfrac{K}{s(s+1)(s+2)}$ 的回授控制系統之根軌跡

(a).Matlab 程式

```
1.function ch1_3
2.num=[1];den=[1 3 2 0];
3.sys=tf(num,den);
4.rlocus(sys),sgrid
```

(b).執行結果



(2).練習

畫出以下轉移函數的控制系統之根軌跡並求出系統穩定之 K 的條件！

1. $\dfrac{K}{s(s+2)(s+4)}$　2. $\dfrac{K(s+1)}{s^2(s+3)}$　3. $\dfrac{K(s-1)}{s(s+1)(s+2)}$

1.11.8 Matlab 實作 PID 控制器

(1).PID 控制器介紹

在討論系統的穩定度分析與性能（或規格）分析方面，一個好的控制系統之設計必須滿足下列之需求：

1. 穩定性　2.精確性　3.暫態響應

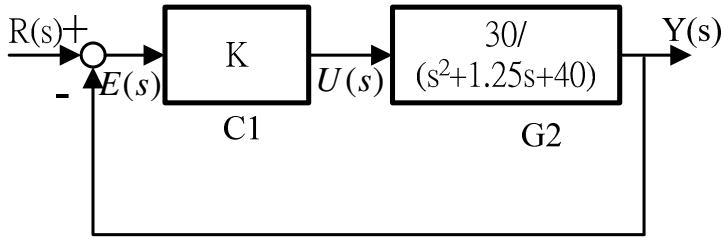
如果一個系統不能滿足上項三者的需求，那麼這個系統就是不可用的系統或是一個性能不好的系統；在執行特定的工作時，該系統可能就達成不了我們所預期的目標。反過來說，如果一個系統能滿足該系統所需的上項三者之需求，那麼該系統必能執行某項特定的工作，而可達成預期的目標。所以設計之目的就是要使一個不能滿足需求的系統，能使之符合需求，以執行特定的工作。

控制器有相當多種形式，而在實用上多數仍採用比例-積分-微分(PID)控制器，尤其在溫度、壓力、流量之定值控制上應用最為廣泛。以下將對此類型之 PID 控制器的設計做介紹。

1.比例控制器(P)

比例控制器（Proportional controller）為固定增益控制器，它可以用來控高系統迴路增益值；藉此提升系統響應的速度與對外擾及不確定性的抵抗力。比例控制器的轉移函數為：

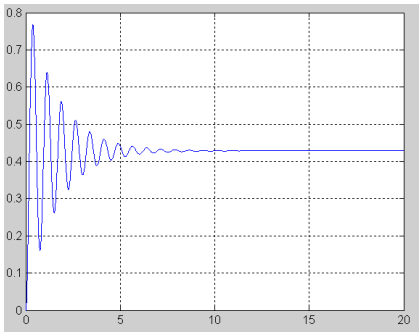
$G_1 = C(s) = \dfrac{U(s)}{E(s)} = K$ ，其中 K 為常數。



(a).比例控制器(Matlab 程式)

```
1.function ch1_8_1
2.g1n=[1];g1d=[1];%k=1
3.g2n=[30];g2d=[1 1.25 40];
4.G1=tf(g1n,g1d);G2=tf(g2n,g2d);
5.G1p=series(G1,G2);
6.Gs=feedback(G1p,1,-1);
7.t=0:0.01:20;
8.[y,t]=step(Gs,t);
9.plot(t,y),grid
```

(b).執行結果



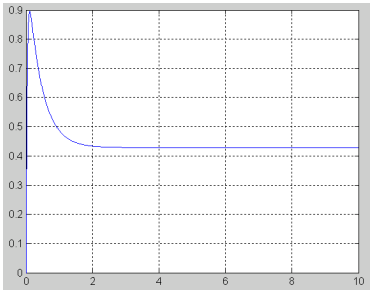
2. 比例控制器+微分控制器(P+D)

PD 控制器會增加阻尼，反應增快，由 K_p 值決定穩態誤差 e_{ss}，K_p 大 e_{ss} 小但穩定度差。
PD 控制器轉移函數為 $G_c(s) = K_p + K_D s$

(a).比例控制器+微分控制器(matlab 程式)：

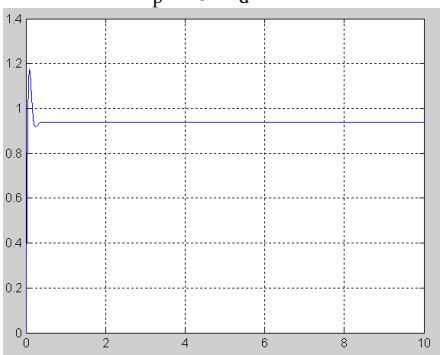
```
1.function ch1_8_2
2. g1n=[1 1];g1d=[1];%kp=1, kd=1
3.g2n=[30];g2d=[1 1.25 40];
4.G1=tf(g1n,g1d);G2=tf(g2n,g2d);
5.G1p=series(G1,G2)
6.Gs=feedback(G1p,1,-1)
7.t=0:0.01:10;
8.y=step(Gs,t);
9.plot(t,y),grid
```

(b).輸出結果

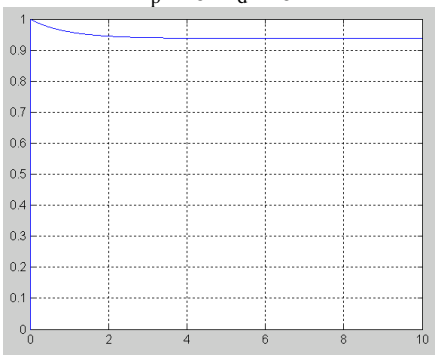


K_p=1 K_d=1

當比例控制器變大，
而微分控制器沒有變動時輸出如下圖
K_p=20 K_d=1



當比例控制器沒有變動，
而微分控制變大時如下圖
K_p=20 K_d=20



3. 比例控制器+積分控制器(P+I)

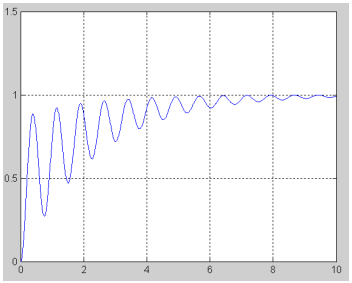
PI 控制器之轉移函數為

$$G_c(s) = K_p + \frac{K_I}{s} = \frac{K_p s + K_I}{s}$$

(a). 比例控制器+積分控制器(matlab 程式)：

```
1.function ch1_8_3
2. g1n=[1 1];g1d=[1 0];%kp=1, ki=1
3.g2n=[30];g2d=[1 1.25 40];
4.G1=tf(g1n,g1d);G2=tf(g2n,g2d);
5.G1p=series(G1,G2)
6.Gs=feedback(G1p,1,-1)
7.t=0:0.01:10;
8.y=step(Gs,t);
9.plot(t,y),grid
```

K_p=1 K_I=1



4. 比例控制器+積分控制器+微分控制器(PID)

PID 控制器之轉移函數為

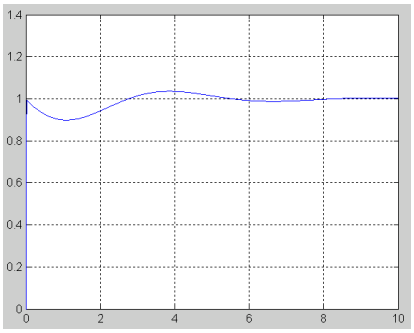
$$G_C(s) = K_P + K_D s + \frac{K_I}{s} = \frac{K_D s^2 + K_p s + K_I}{s}$$

低頻時高增益穩態誤差小，中頻時低增益超出量小，高頻時高增益反應速度快，PID 控制器可稱為帶通（band pass）或頻率衰減（band-attenuate）濾波器。

(a).比例控制器+積分控制器+微分控制器(matlab 程式)：

```
1.function ch1_8_4
2. g1n=[7 4 10];g1d=[1 0]; % Kd=7, Kp=4 KI=10
3.g2n=[30];g2d=[1 1.25 40];
4.G1=tf(g1n,g1d);G2=tf(g2n,g2d);
5.G1p=series(G1,G2)
6.Gs=feedback(G1p,1,-1)
7.t=0:0.01:10;
8.y=step(Gs,t);
9.plot(t,y),grid
```

K_p=4 K_I=10 K_d=7



(2).練習

1.PI 控制器之 K_p=5,K_I=1, 5 10, 20

比較 K_I 不同之響應圖，並說明其差別

2.PID 控制器

(a).K_p=4,K_I=20,K_d=5

(b). K_p=4,K_I=20,K_d=10

(c). K_p=10,K_I=20,K_d=10

(d). K_p=10,K_I=4,K_d=10

比較(a), (b), (c), (d)之響應圖，並說明其差別與影響

下圖為PID控制律對系統的影響

控制律 / 比較項目	比例（P）控制律 $u(t)=K_p e(t)$	積分（I）控制律 $u(t)=K_i\int e(t)dt$	微分（D）控制律 $u(t)=K_d\frac{d}{dt}e(t)$
控制器轉移函數	$C(s)=K_p$	$C(s)=K_i/s$	$C(s)=K_d s$
時域觀點	放大誤差可增快反應速度。可能導致系統響應阻尼比下降。	累積誤差，系統反應初期作用不大，但累積後可改善穩態誤差。可能導致系統響應阻尼比下降。	對誤差微分，以提供誤差時間變化的趨勢，有預測系統反應的作用，可改善暫態響應，增加系統隱定性。
頻域性能之影響	頻率響應特性大致不變，僅提高了系統的頻寬，故暫態反應加快。系統相位裕度變小。	低頻之增益提高，使穩態精度也相對的提高。系統相位裕度變小。	高頻之增益提高，使暫態性能也相對的提高；但相對亦減低了對雜訊的抵抗能力。系統相位裕度變大。
頻域性能之影響	全通濾波器	低通濾波器	高通濾波器
備註	對於穩態誤差的消弭相當有限，只能適度提升響應速度，補償僅靠系統的迴路增益，但增加比例控制易產生飽如現象。	系統階次控高，加大相位落後之可能。增大系統振盪的現象，降低系統的相對穩定性。	當系統有雜訊進入時，微分的效果會放大雜訊對系統的影響，因此實際使用時常與其他控制器結合，而不單獨使用。

[2].Simulink 教學

2.1 Simulink 的基本介紹

(1).何謂 SIMULINK?

一種用於對動態系統進行建模(modeling)、模擬(simulating)和分析(analyzing)的軟體。

它支援連續時間、離散時間或兩者混合的線性和非線性系統。也支援具有不同取樣速率的多重速率系統。

(2).simulink 功能方塊

主要分成以下幾個方塊庫：

方塊庫圖示	方塊庫名稱	方塊庫圖示	方塊庫名稱
	Continuous		Signals&Systems
	Discrete		Sinks
	Functions&Tables		Sources
	Math		Subsystems
	Nonlinear		

方塊庫分別說明

a. Continuous 方塊庫

方塊圖	方塊名稱	功能說明
du/dt	Derivate	求導數
$\frac{1}{s}$	Integrator	積分器
$\frac{(s-1)}{s(s+1)}$	Zero-Pole	轉移函數(零極點形式)
$\dot{x} = Ax+Bu$ $y = Cx+Du$	State-Space	狀態空間方程式
$\frac{1}{s+1}$	Transfer Fcn	轉移函數(多項式形式)

b.Functions&Tables 方塊庫

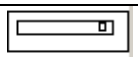
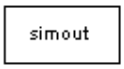
方塊圖	方塊名稱	功能說明
f(u)	Fcn	函數
MATLAB Function	Matlab Fcn	Matlab 函數
P(u) O(P) = 5	Polynomial	多項式

c. Math

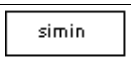
方塊圖	方塊名稱	功能說明
|u|	Abs	絕對值

	Complex to Magnitude-Angle	複數的相位角和大小
	Complex to Real-Imag	複數的實部和虛部
	Dot Product	內積
	Gain	增益
	Math Function	數學函數
	Matrix Gain	增益矩陣
	MinMax	極值
	Product	乘積
	Sum	加法器
	Trigonometric Function	三角函數

d. Sinks

方塊圖	方塊名稱	功能說明
	Display	顯示輸入值
	Scope	示波器
	Out1	提供來自子系統或模型的輸出
	Floating Scope	顯示多條線之示波器
	Stop Simulation	當輸入非零時停止模擬
	To File	將數據寫入檔案
	To Workspace	將數據寫入工作空間中之矩陣
	XY Graph	顯示 XY 訊號圖

e. Sources

方塊圖	方塊名稱	功能說明
	Clock	時鐘
	Constant	常數
	From Workspace	從工作空間中矩陣讀取數據

	From File	從檔案讀取數據
	Pulse Generator	脈波產生器
	Ramp	斜坡訊號
	Random Number	隨機數(常態分布)
	Repeating Sequence	週期序列
	Signal Generator	訊號產生器
	Sine Wave	正弦波訊號
	Step	步階訊號
	Uniform Random Number	隨機數(均勻分布)

2.2 練習

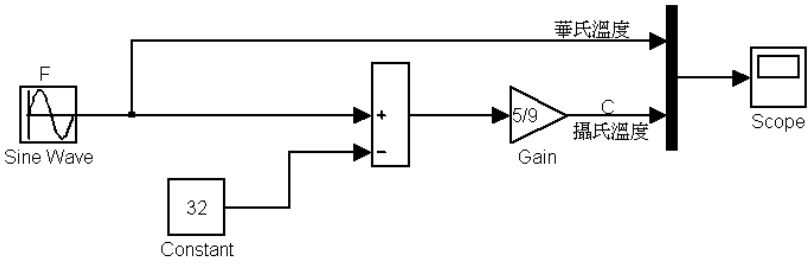
2.2.1 溫度轉換

2.2.2 RLC 系統

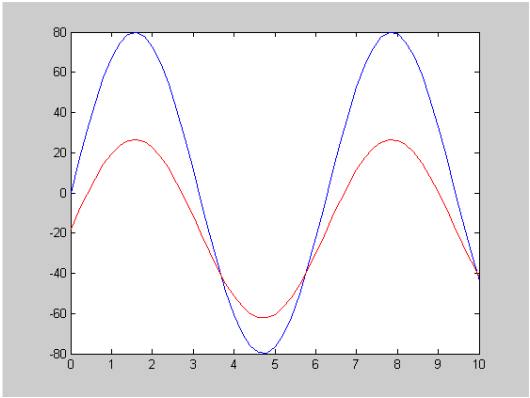
2.2.3 Matlab 之 Simulink 實作 PID 控制器

2.2.1 溫度轉換

1. 題目：將華氏溫度(輸入)轉成攝氏溫度(輸出)
   計算公式:℃=(˚F-32)*5/9,其輸入為 Sin 波形且最大值為 80
2.模型



3.輸出結果



4.練習
(a).計算攝氏溫度轉換成華氏溫度,計算公式˚F=℃*9/5+32,其輸入為 Sin 波形且最大值為 50

2.2.2 RLC 電路系統

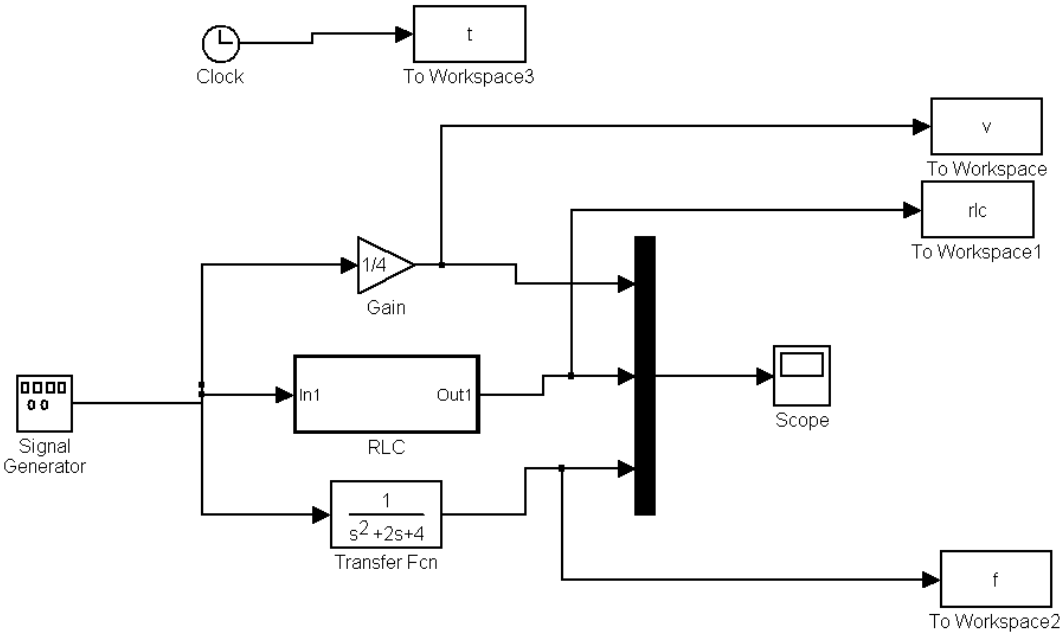
1.題目

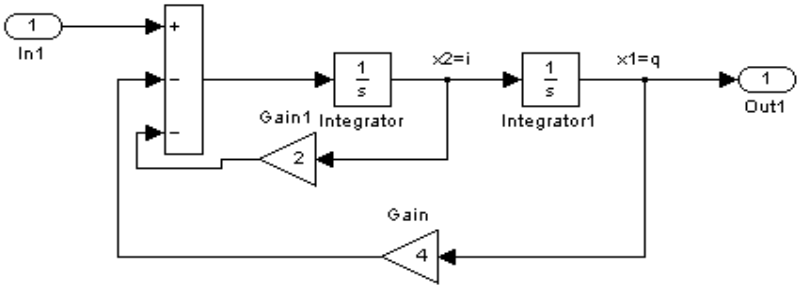
$$v(t) = L\frac{di}{dt} + Ri + \frac{1}{C}\int_0^t i(\tau)d\tau = L\frac{d^2q}{dt^2} + R\frac{dq}{dt} + \frac{1}{C}q$$

$$1\frac{d^2q}{dt^2} + 2\frac{dq}{dt} + 4q = v(t)$$

$$\frac{Q(s)}{V(s)} = \frac{Y(s)}{U(s)} = \frac{1}{Ls^2 + Rs + 1/C} = G(s)$$ 假設 L=1,R=2,C=1/4

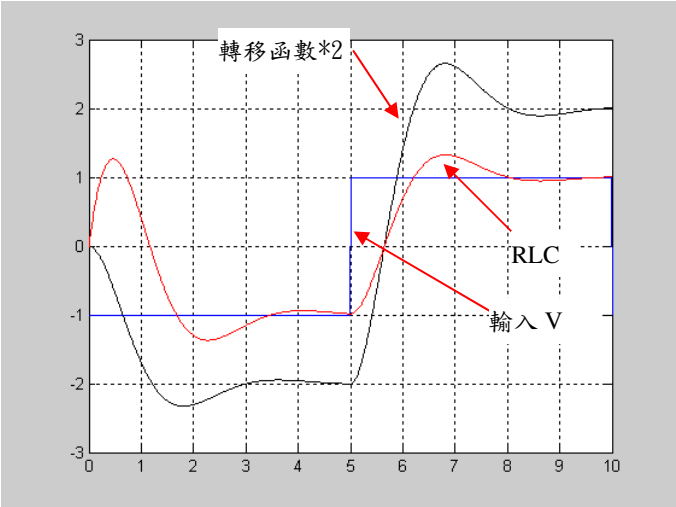
2.模型





RLC 子系統

3.輸出結果



4.練習

(a).同以上之 RLC 系統但假設 R=3,L=5,C=1/7(轉移函數*2，初始值全為 0)

(b).同以上之 RLC 系統但假設 R=5,L=7,C=1/9(轉移函數*2，初始值全為 0)

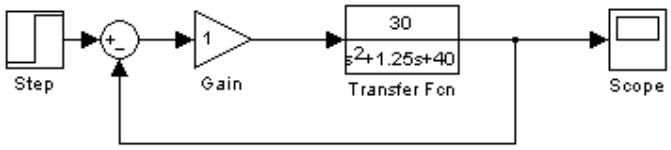
(c)同(a)之條件，但轉移函數*1，RLC 系統之初始值不為 0

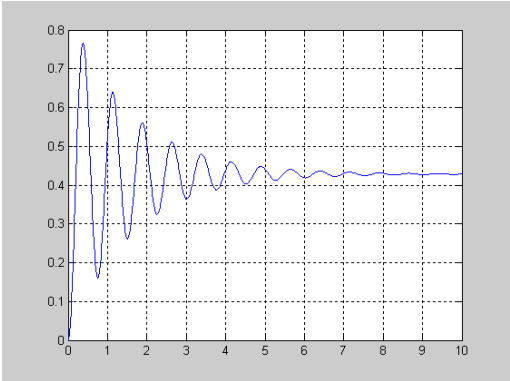
2.2.3 Matlab 之 Simulink 實作 PID 控制器

1. 比例控制器(P)

$$C(s)=\frac{U(s)}{E(s)}=K$$ ，其中 K 為常數。

下圖為 Matlab 實作比例控制器：

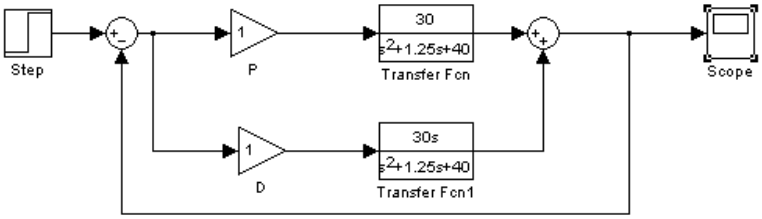




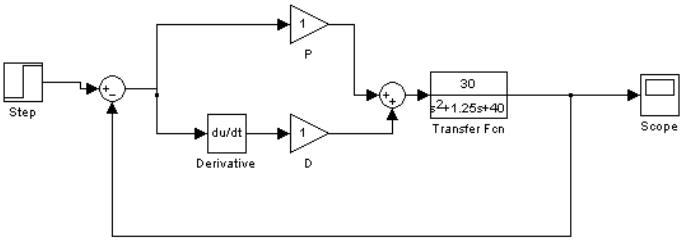
2. 比例控制器+微分控制器(P+D)

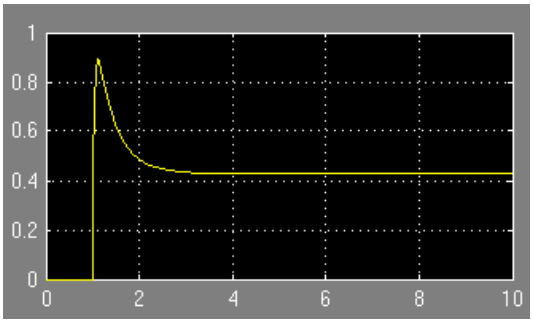
PD 控制器轉移函數為 $G_c(s)=K_p+K_Ds$

下圖為 Matlab 實作比例控制器+微分控制器：

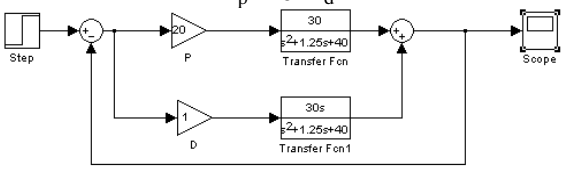


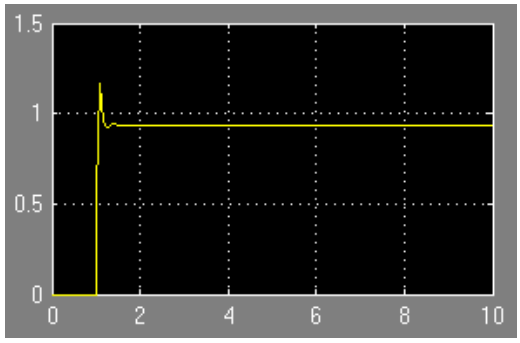
以下是錯誤作法



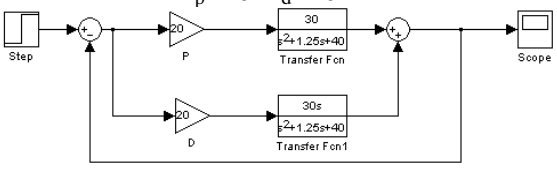


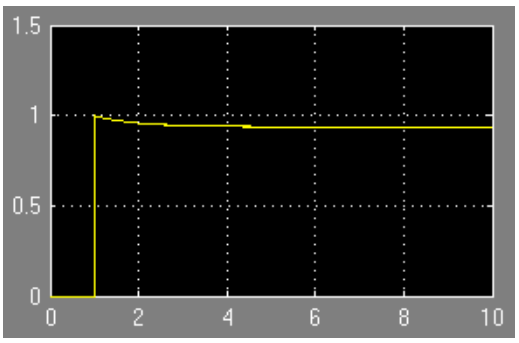
當比例控制器變大，
而微分控制器沒有變動時輸出如下圖
K_p=20 K_d=1





當比例控制器沒有變動，
而微分控制變大時如下圖
K_p=20 K_d=20

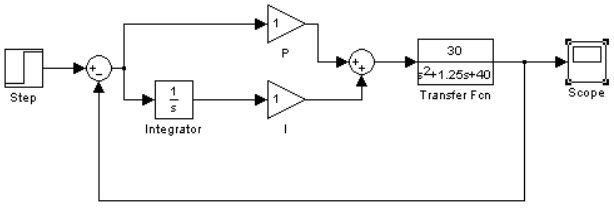


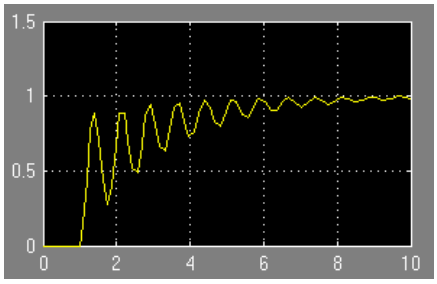


3. 比例控制器+積分控制器(P+I)

PI 控制器之轉移函數為

$$G_c(s) = K_p + \frac{K_I}{s}$$

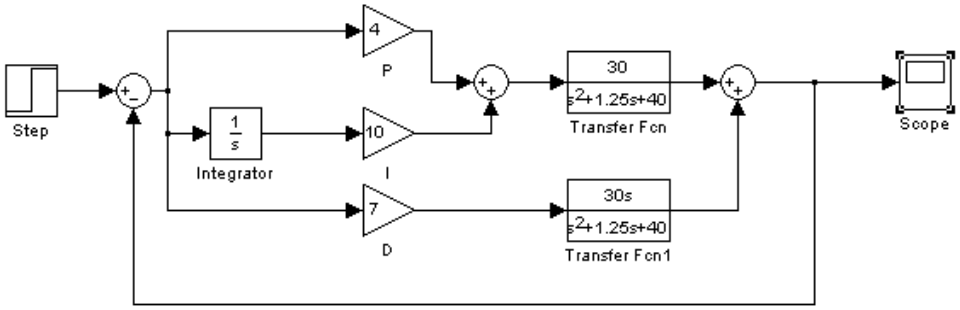


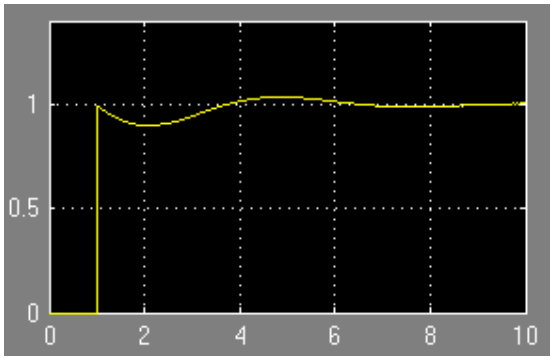


4. 比例控制器+積分控制器+微分控制器(PID)

PID 控制器之轉移函數為

$$G_C(s) = K_P + K_D s + \frac{K_I}{s}$$





5. 練習

(a).PI 控制器之 K_p=5,K_I=20

(b).PID 控制器

a.K_p=4,K_I=20,K_d=7

b. K_p=4,K_I=20,K_d=10

c. K_p=10,K_I=20,K_d=10

[3].系統實作

3.1 基本 OP 電路實習

3.2 輸入/輸出模組實作

3.3 整合模組電路

3.4 PID 類比控制系統實作

3.1 基本 OP 電路實習

(1).741 運算放大器(OP)介紹

1.作動方式與原理

741 放大器為運算放大器中最常被使用的一種，擁有反相向與非反相兩輸入端，由輸入端輸入欲被放大的電流或電壓訊號，經放大後由輸出端輸出。放大器作動時的最大特點為需要一對同樣大小的正負電源，其值由±12V_{dc} 至±18V_{dc} 不等，而一般使用±15V_{dc} 的電壓。741 運算放大器的外型與接腳配置分別如圖 1、2 所示。

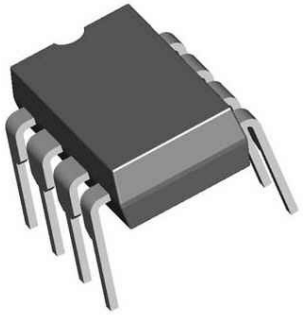


圖 1. 741 運算放大器外型圖

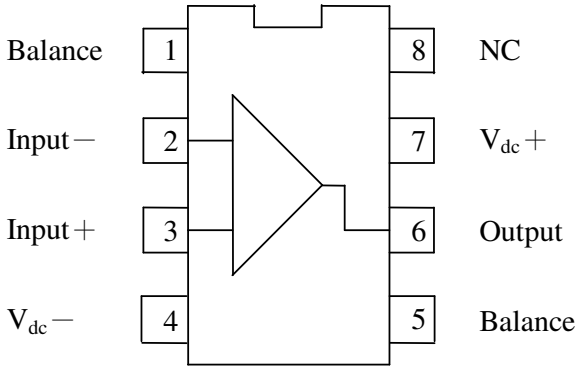


圖 2. 741 放大器輸出入腳位圖

741 運算放大器使用時需於 7、4 腳位供應一對同等大小的正負電源電壓＋V_{dc} 與－V_{dc}，一旦於 2、3 腳位即兩輸入端間有電壓差存在，壓差即會被放大於輸出端，唯 Op 放大器具有一特色，其輸出電壓值決不會大於正電源電壓＋V_{dc} 或小於負電源電壓－V_{dc}，輸入電壓差經放大後若大於外接電源電壓＋V_{dc} 至－V_{dc} 之範圍，其值會等於＋V_{dc} 或－V_{dc}，故一般運算放大器輸出電壓均具有如圖 3 之特性曲線，輸出電壓於到達＋V_{dc} 和－V_{dc} 後會呈現飽和現象。

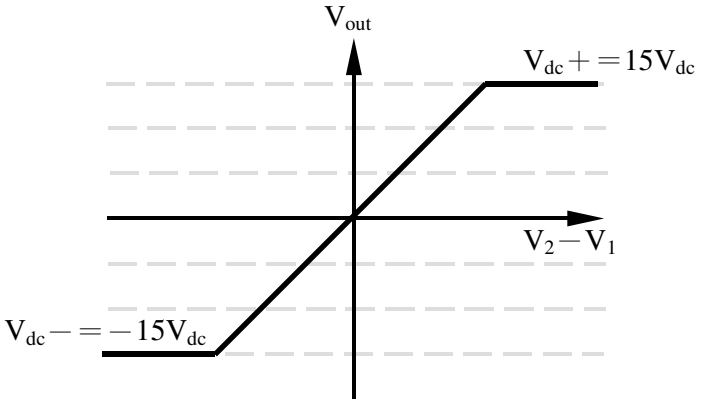
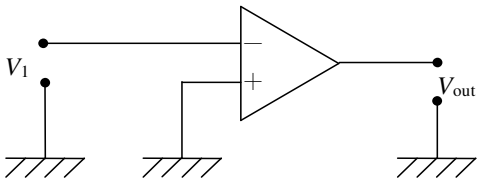


圖 3. 放大器輸出入電壓關係圖

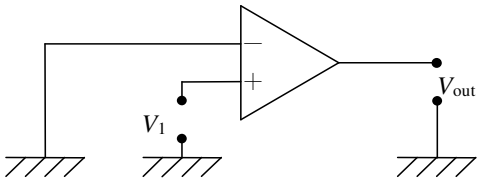
741 運算放大器之基本動作如圖 4 所示，若在非反相輸入端輸入電壓，會於輸出端得到被放大的同極性輸出；若以相同電壓訊號在反相輸入端輸入，則會在輸出端獲得放大相同倍

率後但呈逆極性之訊號輸出。而當對放大器兩輸入端同時輸入電壓時，則是以非反相輸入端電壓值(V_1)減去反相輸入端電壓值(V_2)，可於輸出端得到(V_1-V_2)經過倍率放大後之輸出。

1. $Vo=-A\times V_1$



2. $Vo=A\times V_1$



3. $Vo=A\times(V_1-V_2)$

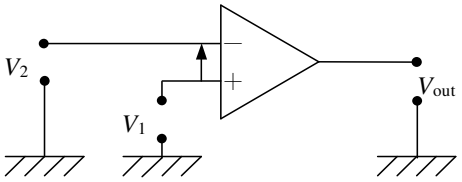
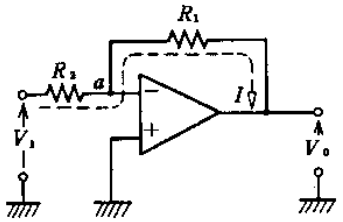
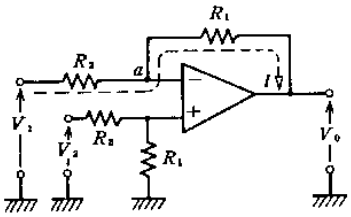
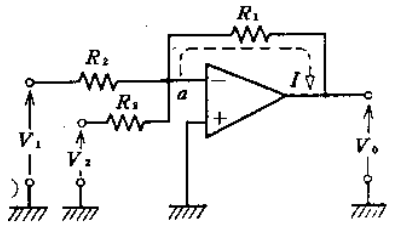
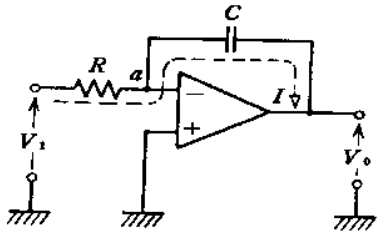
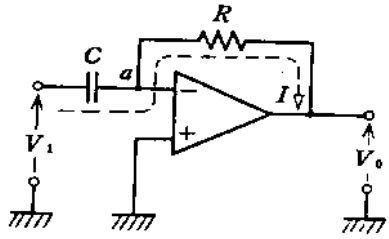
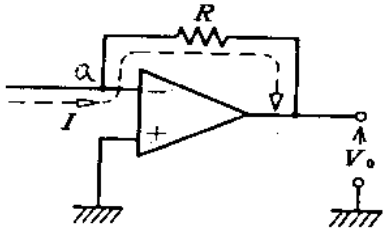
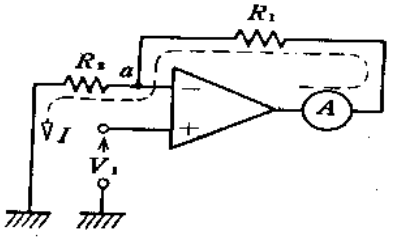


圖 4. 放大器基本輸出入關係圖

(2).常用運用電路

1.反相放大器 得到與輸入逆相之$\frac{R_1}{R_2}$倍輸出 -點 a 的電位=0V $-I=\frac{V_1}{R_2}$ $-V_o=-IR_1=-\frac{R_1}{R_2}V_1$	
2.差動放大器 得到(V_2-V_1)放大$\frac{R_1}{R_2}$倍之輸出 -點 a 的電位=$\frac{R_1}{R_1+R_2}V_2$ $-I=\frac{V_1-\frac{R_1}{R_1+R_2}V_2}{R_2}$	

$-V_o = \frac{R_1}{R_1+R_2}V_2 - IR_1 = \frac{R_1}{R_2}(V_2 - V_1)$	
3.加法器 得到輸入電壓 V_1 及 V_2 之加算輸出 -點 a 的電位=0V $-I = \frac{V_1}{R_2} + \frac{V_2}{R_2} = \frac{1}{R_2}(V_1 + V_2)$ $-V_o = -IR_1 = -\frac{R_1}{R_2}(V_1 + V_2)$	
4.積分器 得到將輸入電壓加以積分之輸出 -點 a 的電位=0V $-I = \frac{V_1}{R}$ $-V_o = \frac{1}{C}\int Idt = \frac{1}{CR}\int V_1 dt$	
5.微分器 得到將輸入電壓加以微分的輸出 -點 a 的電位=0V $-I = \frac{dQ}{dt} = C\frac{dV}{dt}$ $-V_o = -IR = -CR\frac{dV}{dt}$	
6.I-V 變換器 得到輸出電壓對應於輸入電流 -點 a 的電位=0V $-V_0 = -IR$	
7.V-I 變換器 得到輸出電流對應於輸入電壓 -點 a 的電位=V_1 $-I = \frac{V_1}{R_2}$	

(3).實習部分

1.預備知識

(a).數位示波器操作

(b).訊號產生器操作

練習:

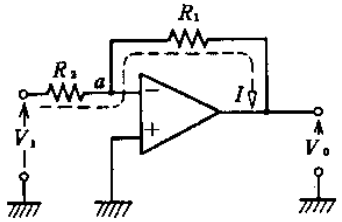
a.產生 50Hz,振幅為 2V 之方波訊號

b.產生 1KHz,振幅為 1V 之正弦波訊號

2.反相放大器(比例器)實作

(a).條件:

a.如圖



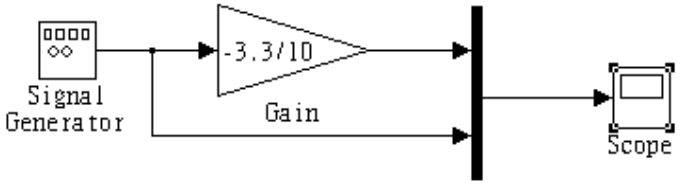
b.其中 R1=3.3KW,R2 為 10KW

c.V_i 的輸入訊號為 20Hz,振幅為 3V 之方波訊號

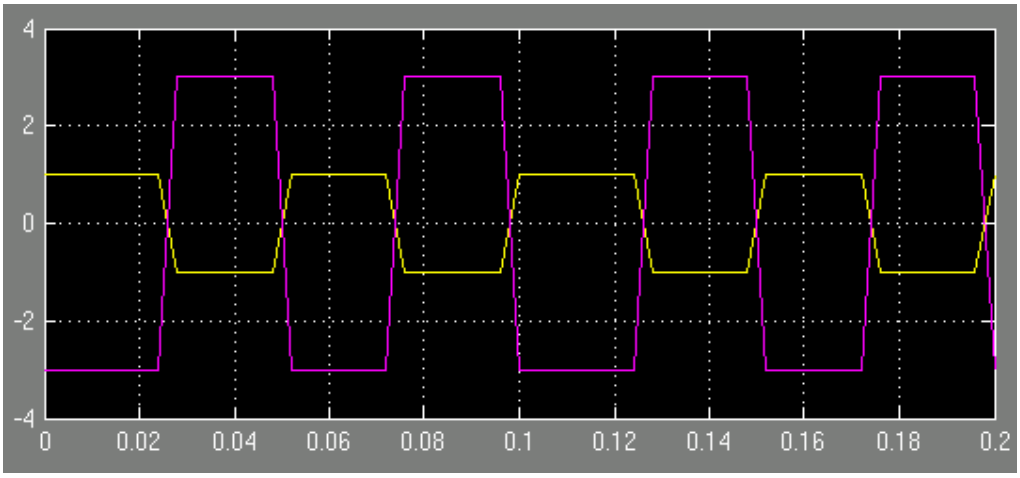
(b).參考答案

a.利用 Matlab 的 Simulink 模擬輸出結果

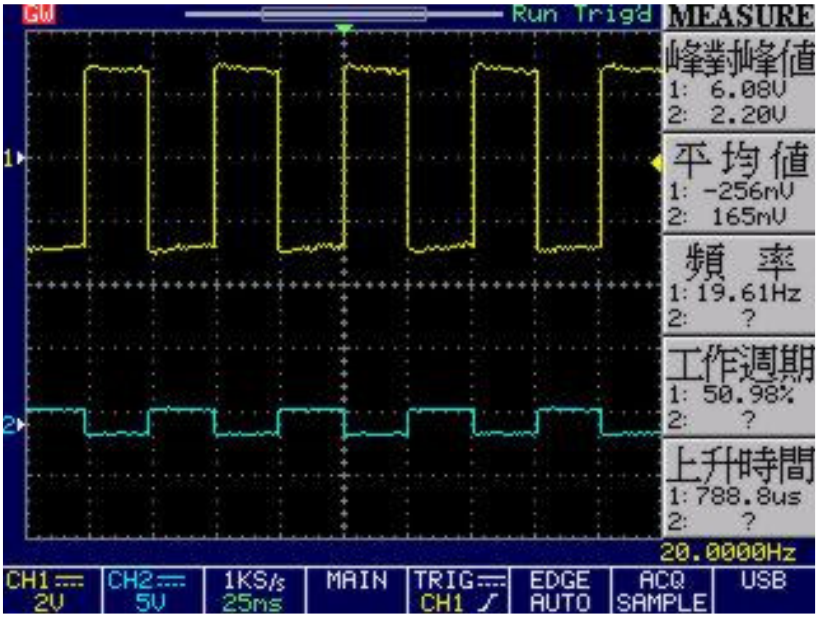
-模型圖



-輸出結果



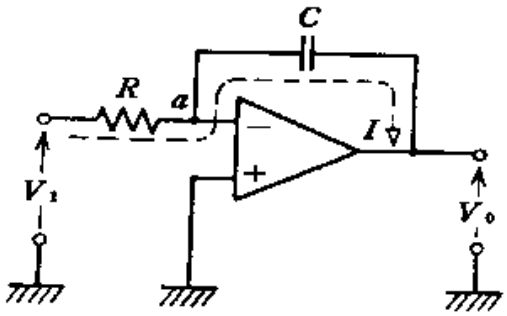
b.實際建立以上電路及輸入訊號，利用示波器量出 V_o 輸出結果



3.積分器實作

(a).條件:

a.如圖



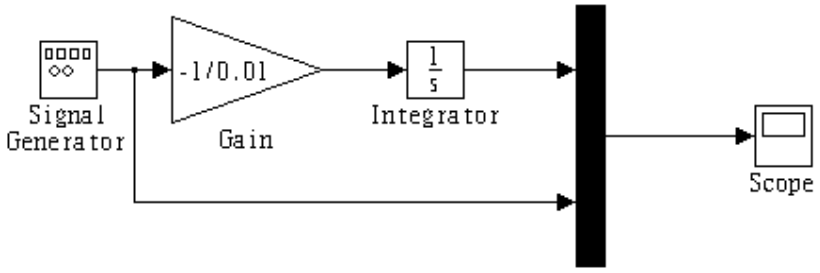
b.其中 C 為 0.1mF,R 為 100KW

c.V_i 的輸入訊號為 20Hz,振幅為 2V 之方波訊號

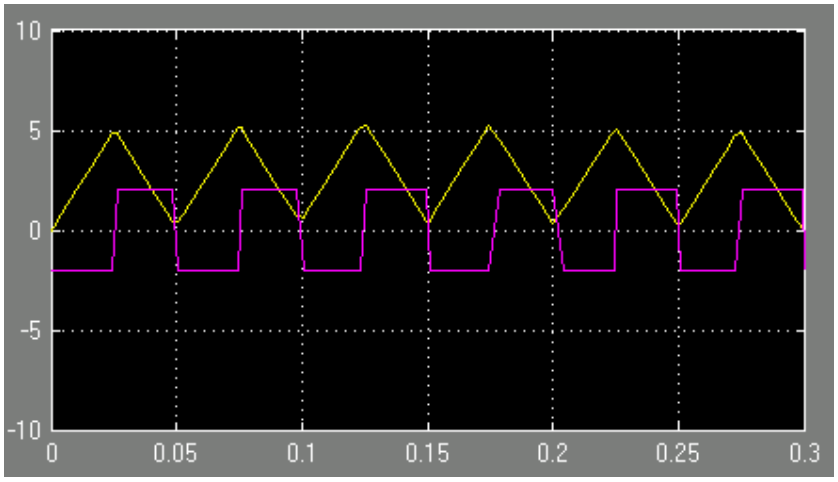
(b).參考答案

a.利用 Matlab 的 Simulink 模擬輸出結果

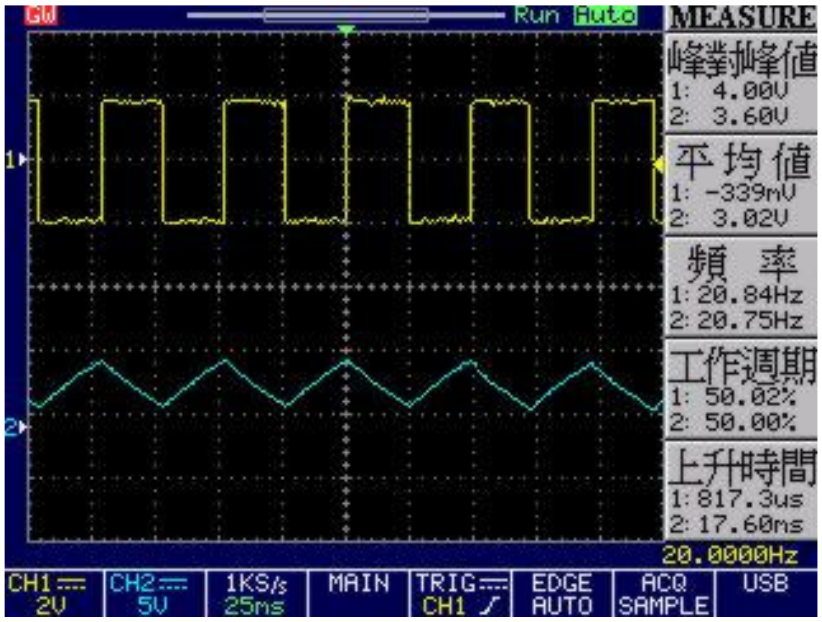
-模型圖



-輸出結果



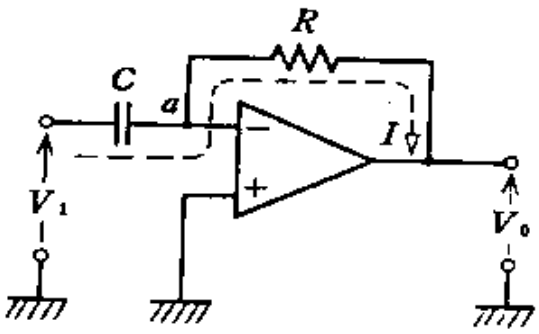
b.實際建立以上電路及輸入訊號，利用示波器量出 V_o 輸出結果

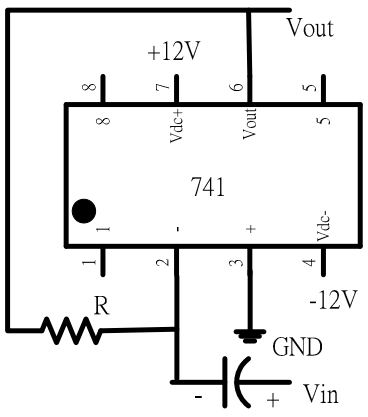


4.微分器實作

(a).條件:

a.如圖





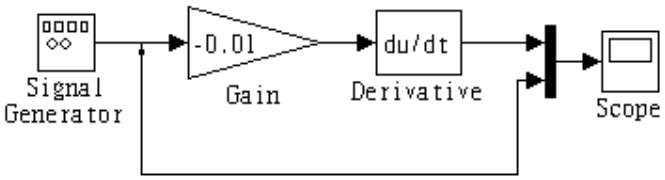
b.其中 C 為 0.1mF,R 為 100KW

c.V_i 的輸入訊號為 20Hz,振幅為 2V 之方波訊號

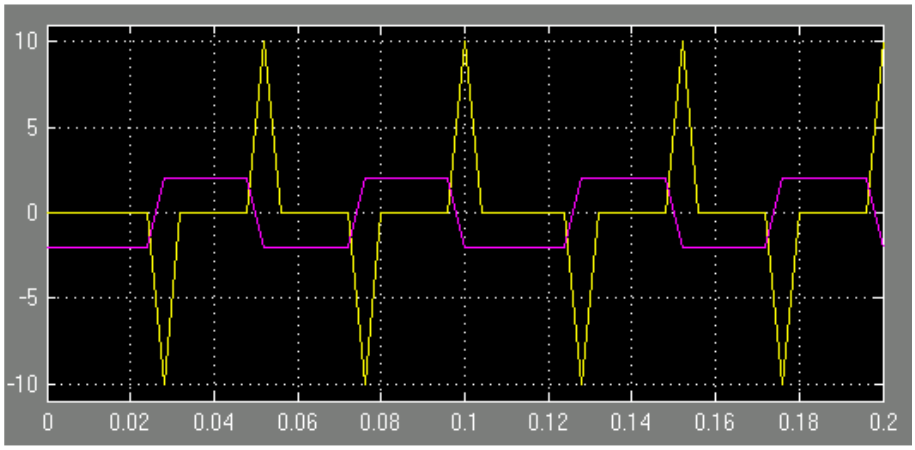
(b).參考答案

a.利用 Matlab 的 Simulink 模擬輸出結果

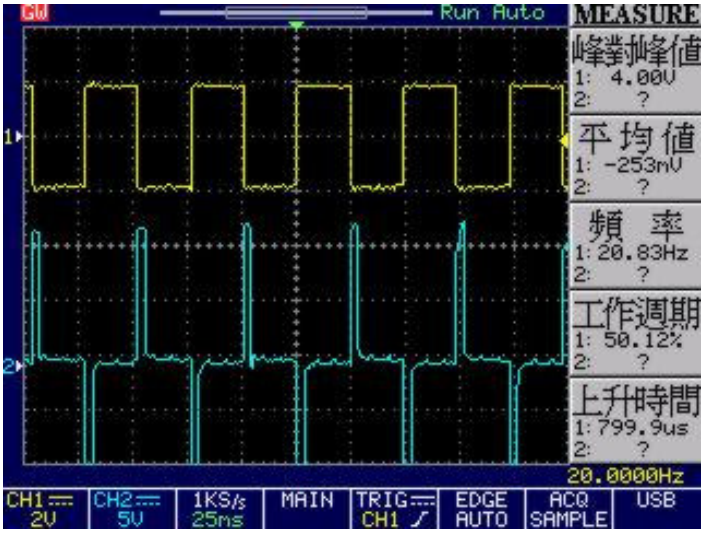
-模型圖



-輸出結果

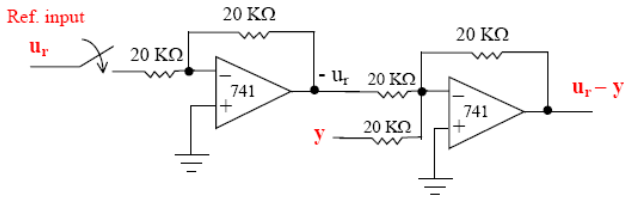


b.實際建立以上電路及輸入訊號，利用示波器量出 V_o 輸出結果

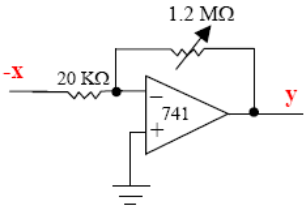


3.2 輸入/輸出模組實作

1. 輸入模組：



2. 輸出模組：



零件清單:

項　目	規　格	數　量	項　目	規　格	數　量
電阻	20kΩ	6	switch	1	1
電阻	1.2MΩ	1	運算放大器	741	3
電容	10μF	2			

[3].模擬標準二階電路系統之 **OP** 電路設計製作

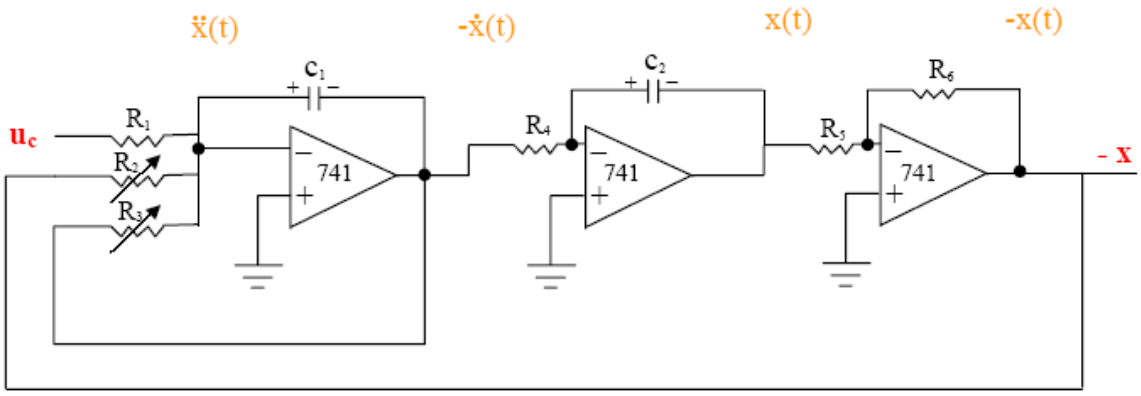
電路設計：

$$\ddot{x}(t) + 2\zeta\omega_n\dot{x}(t) + \omega_n{}^2 x(t) = v(t)$$

或

$$\ddot{x}(t) = -2\zeta\omega_n\dot{x}(t) - \omega_n{}^2 x(t) + v(t)$$

- 二階系統模組(受控體)：



其中

$$\frac{1}{R_1 c_1} = 1 \qquad (1)$$

$$\frac{1}{R_2 c_1} = \omega_n{}^2 \qquad (2)$$

$$\frac{1}{R_3 c_1} = 2\zeta\omega_n \quad (3)$$

$$\frac{1}{R_4 c_2} = 1 \quad (4)$$

$$R_5 = R_6 \quad (5)$$

(b) 參數設計：

(i) 先取**$R_1 = 100\ k\Omega$, $c_1 = c_2 = 10\ \mu F$, $R_5 = R_6 = 20\ k\Omega$**

$\Rightarrow R_1 c_1 = 100\ k\Omega \times 10\ \mu F = 1 \rightarrow$ 符合(1) 式

(ii) 由(2)式 $\frac{1}{R_2 c_1} = \omega_n^2 \Rightarrow \frac{1}{R_2} = (2\pi)^2 \times 10\ \mu F$

$\Rightarrow R_2 = 2.53\ k\Omega$ (故可選用 $R_2 = 50\ k\Omega$ 之可變電阻)

(iii) 由(3)式 $\frac{1}{R_3 c_1} = 2\zeta\omega_n \Rightarrow \frac{1}{R_3} = 2 \times 0.1 \times 2\pi \times 10\ \mu F$

$\Rightarrow R_3 = 79.5\ k\Omega$ (故可選用 $R_2 = 100\ k\Omega$ 之可變電阻)

(iv) 由(4)式 $\frac{1}{R_4 c_2} = 1 \Rightarrow R_4 = 100\ k\Omega$

(c) 振盪頻率分析：

$$\omega_d = \omega_n \sqrt{1 - \zeta^2}$$

$$\Rightarrow \omega_d = 2\pi\sqrt{1 - (0.1)^2} = 6.252$$

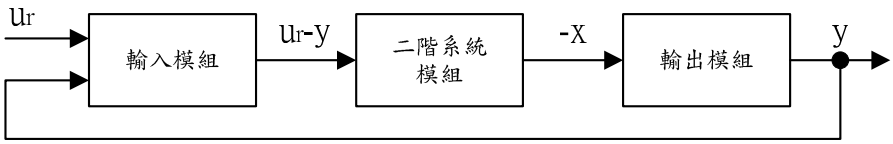
$$\because \omega_d = 2\pi f \Rightarrow f = \frac{\omega_d}{2\pi} = 0.99Hz \cong 1Hz$$

週期 $T = \frac{1}{f} = 1$ (sec)

(d).零件清單

項　目	規　格	數　量	項　目	規　格	數　量
電阻	20kΩ	2	可變電阻	50KΩ	1
電阻	100kΩ	2	可變電阻	100kΩ	1
電容	10μF	2	運算放大器	741	3

3.3 整合模組電路

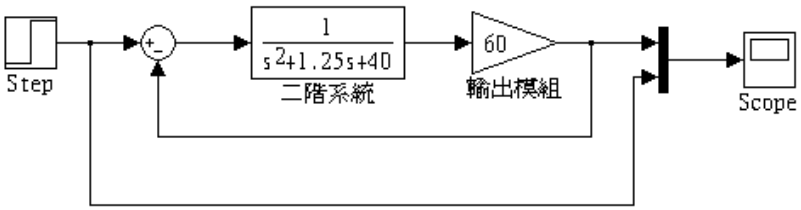


[5].練習

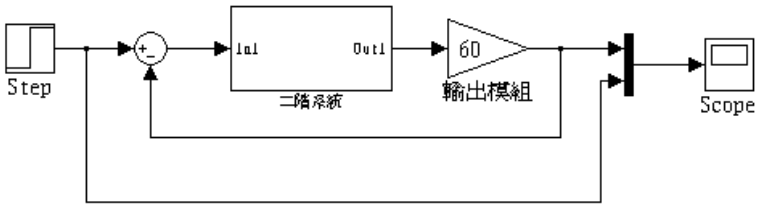
1. 輸入 0->5V 的步階訊號

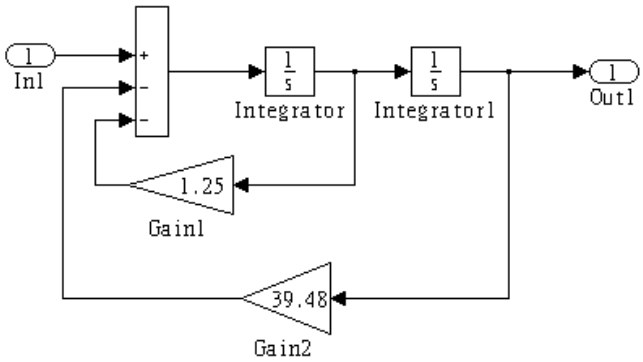
a.利用 Matlab 的 Simulink 模擬輸出結果

-模型圖

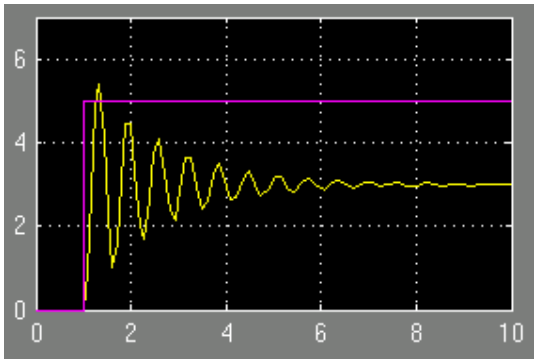


or

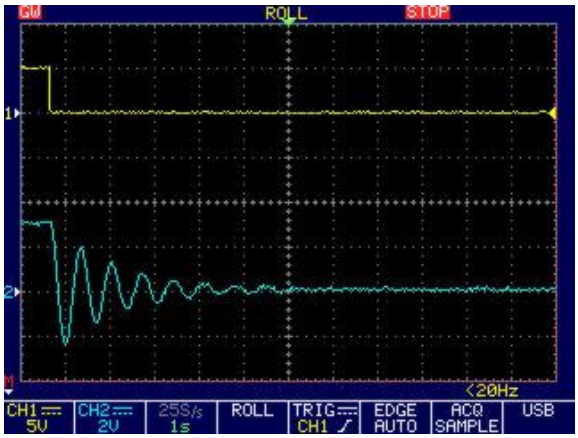




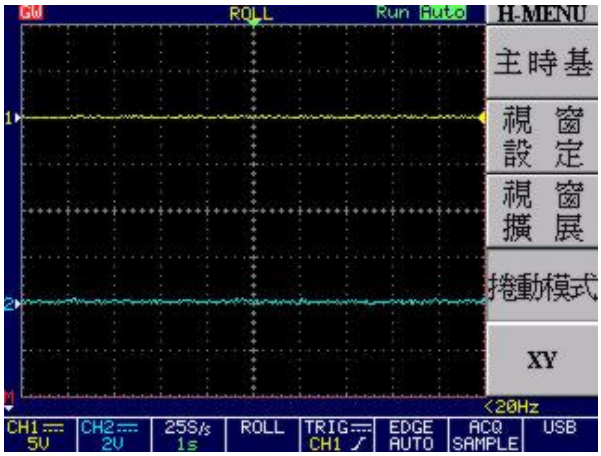
-輸出結果



b.實際建立以上電路及輸入訊號，利用示波器量出 V_o 輸出結果



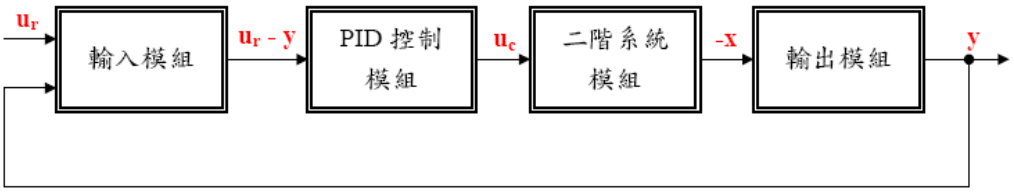
**示波器觀察時,需改變其捲動模式才有辦法觀察以上圖形



3.4 PID 類比控制系統實作

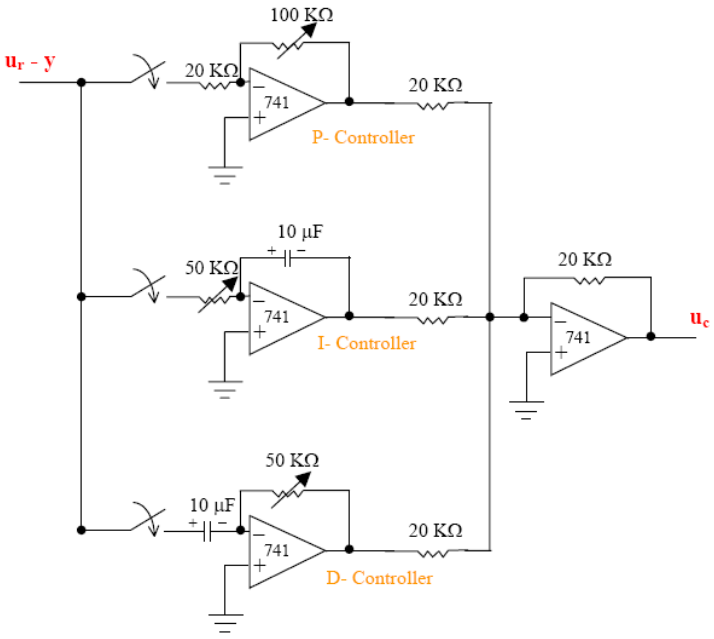
前面我們介紹一二階之電路，在此再結合一PID控制器的實現類比電路，以期學生能對 PID 控制器有更深刻的體認。

整個系統如下所示:



和前一實驗系統比較，只是多了PID控制模組。以下就該模組詳細說明:

[1].PID 控制器模組類比電路：



零件清單:

項　目	規　格	數　量	項　目	規　格	數　量
電阻	20kΩ	5	可變電阻	100kΩ	1
電阻	100kΩ	2	switch	3	1
電容	10μF	2	運算放大器	741	4
可變電阻	50KΩ	2			

[2].整合模組電路零件清單:

項　目	規　格	數　量	項　目	規　格	數　量
電阻	20kΩ	11	可變電阻	50KΩ	3
電阻	100kΩ	4	可變電阻	100kΩ	2

電阻	1.2MΩ	2	switch	1*4	1
電容	10μF	6	運算放大器	741	10

[3].練習

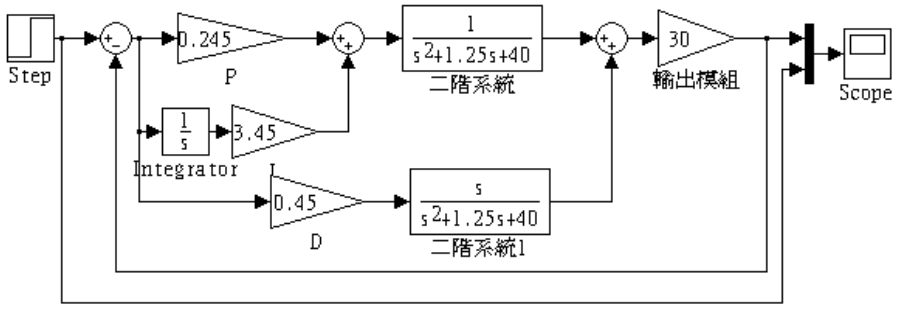
1. 輸入 0->5V 的步階訊號

2.PID 之值分別為

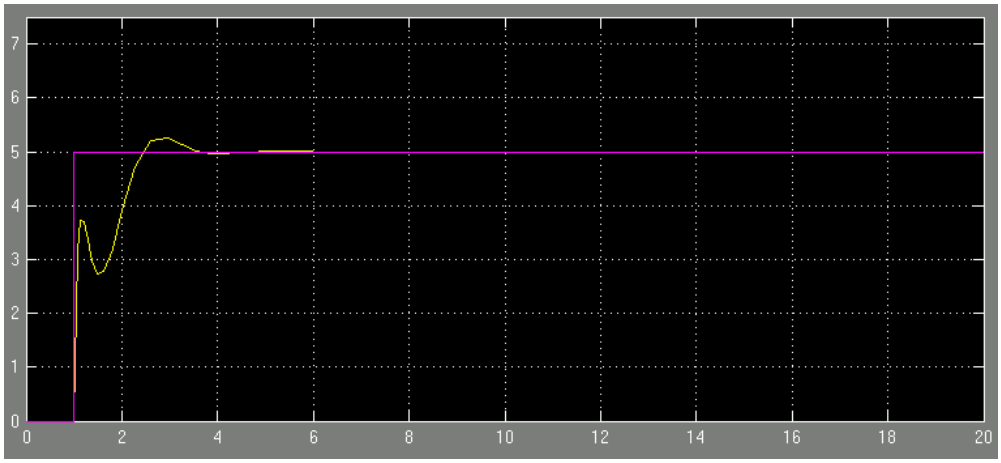
PID 值		可變電阻值(KΩ)	備註
P	0.245	100KΩ↓	↓,由 R1/R2 計算
I	3.45	50KΩ↓	↑,由 1/(CR)計算
D	0.45	50KΩ↓	↓,由 CR 計算

a.利用 Matlab 的 Simulink 模擬輸出結果

-模型圖



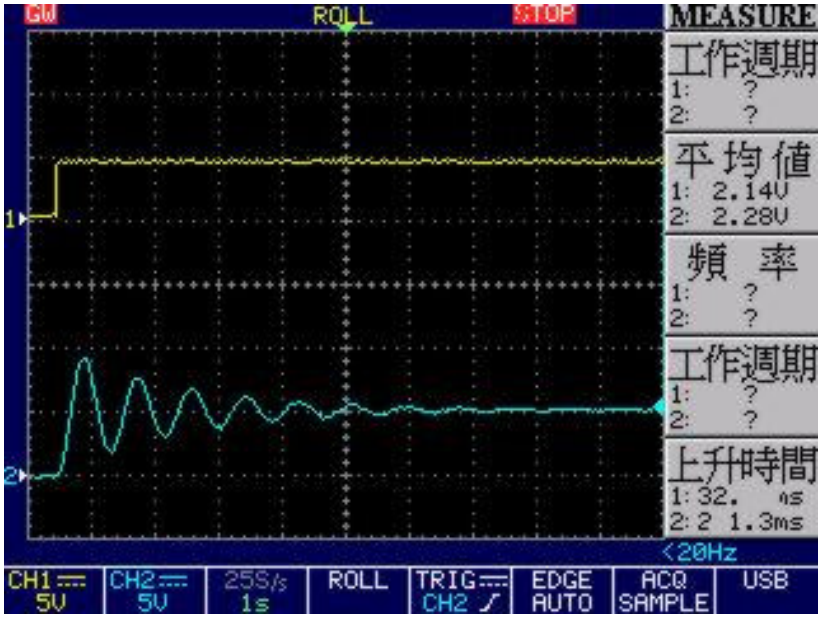
-輸出結果



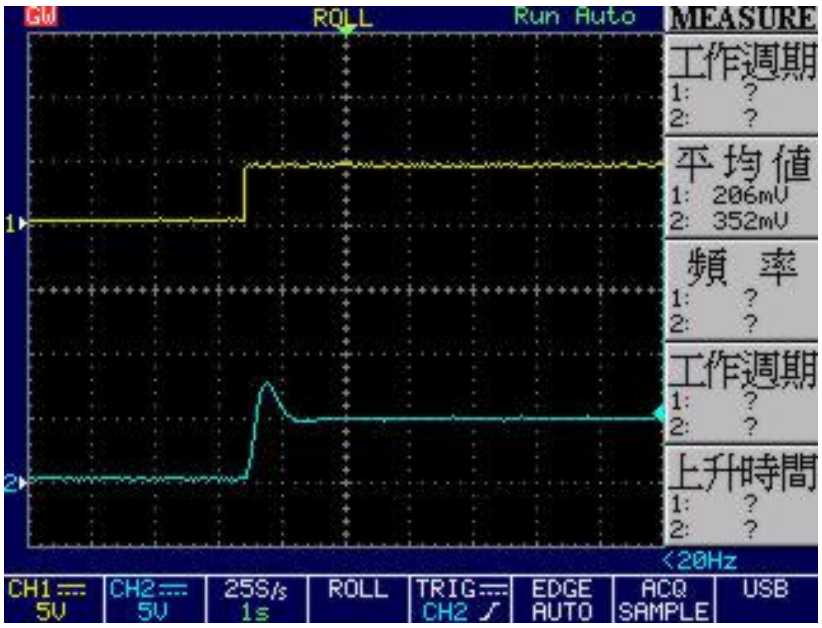
b.實際建立以上電路及輸入訊號，利用示波器量出 V_o 輸出結果

(其中 PID 之可變電阻分別設為以下值)

PID	可變電阻值(KΩ)
P	4.9
I	29
D	45



未接 PID 控制器時,輸出訊號情形



接上 PID 控制器時,輸出訊號情形

實驗　直流伺服馬達一階系統參數量測

目的：

利用一階系統參數量測方法量測馬達之一階系統參數

原理：

一階系統可用一階微分方程式表示如下：

$$\frac{dy(t)}{dt} + ay(t) = br(t) \qquad (1)$$

其中 a,b 爲常數。
則(1)之拉氏轉換如下

$$sY(s) - y(0) + aY(s) = bR(s)$$

$$Y(s) = \frac{b}{s+a}R(s) + \frac{y(0)}{s+a} \qquad (2)$$

所以轉移函數

$$G(s) = \frac{Y(s)}{R(s)} = \frac{b}{s+a} \qquad (y(0) = 0) \qquad (3)$$

直流伺服馬達速度和輸入電壓之關係是最常見的一階系統。

若輸入爲步級信號 r(t)=A，則

$$R(s) = \frac{A}{s}; A\text{是振幅大小} \qquad (4)$$

則

$$Y(s) = \frac{bA}{s(s+a)} = \frac{bA}{a}\left(\frac{1}{s} - \frac{1}{s+a}\right) \qquad (5)$$

取反拉氏轉換可得

$$y(t) = \frac{bA}{a}(1 - e^{-at}) \qquad (6)$$

定義兩個重要名詞

時間常數 (time constant);系統輸出上升至輸出最終值的 0.632 倍時所需之時間。

$$\tau = \frac{1}{a} \qquad (7.a)$$

增益常數 *k*: 穩態時倍率
利用終值定理可得穩態值如下

$$y(\infty) = \lim_{s\to 0} sY(s) = \lim_{s\to 0} s(\frac{bA}{s(s+a)}) = \frac{bA}{a}$$

$$\therefore k = \frac{Ab}{a} / A = \frac{b}{a} \qquad (7.b)$$

$$a = \frac{1}{\tau}$$
$$b = k \cdot a$$

實習步驟：

(1)從 user 資料夾中，點選 Setup_SRVO2_Exps.m run 一次

(2)點選 q_tacho_lead_Q4.mdl ，出現電路圖，修改電路圖，如圖一，更改馬達的轉速 A 值。

(3)點選 build all 開始 run，出現一個工作視窗，再點選示波器的圖示

(4)之後出現 Measured Velocity(RPM)按確定，會有波型隨著馬達運轉而起伏變化呈現出來。

(5)存取波型：在顯示波形的視窗 FILE 中選取 save to workspace

(6)再到 MATLAB 視窗查詢時間和變數是否存在

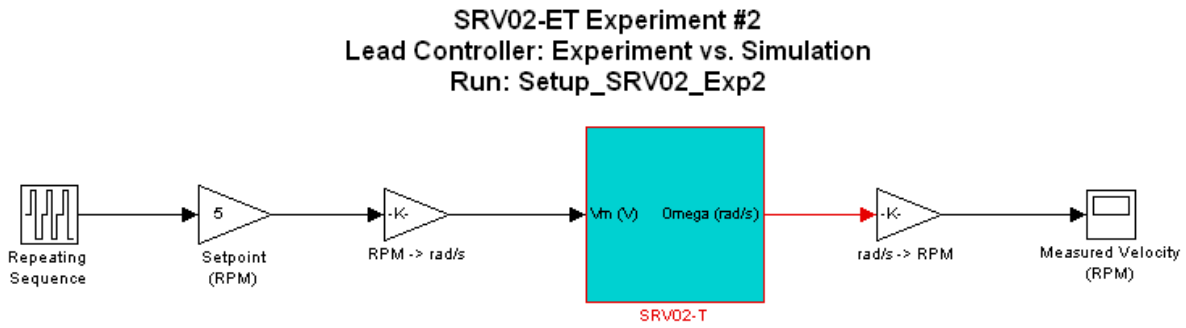
(7)最後再把變數資料和時間資料存取成.dat 資料型式

(8)要再做下一個調整轉速前，要先 clear 清除掉之前所儲存的變數

(9)算出 a 和 b 的平均值，加入在新加的一轉移函數，a、b 不變，改變轉速 A

值，如圖二，驗證 k 值誤差程度如何

4 - 2



圖（一）實驗部分 simulink 接線圖

第一部分：實驗過程之波形

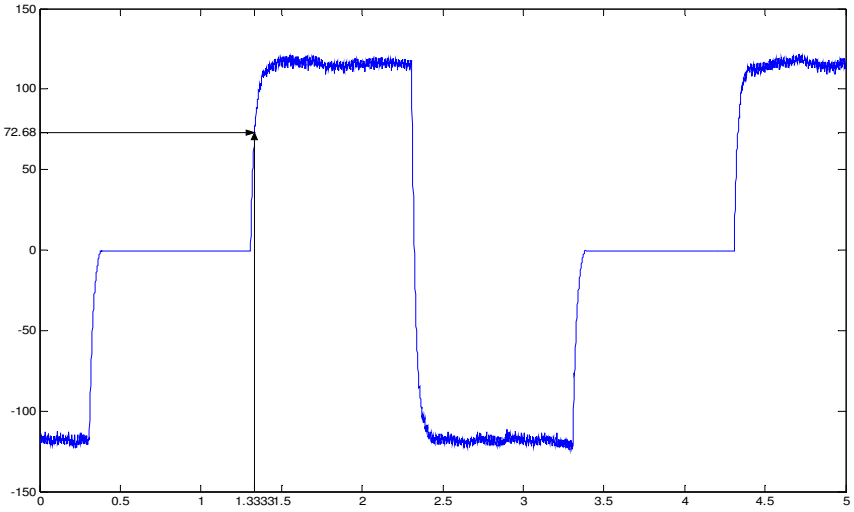
（參考圖(一)之電路圖）

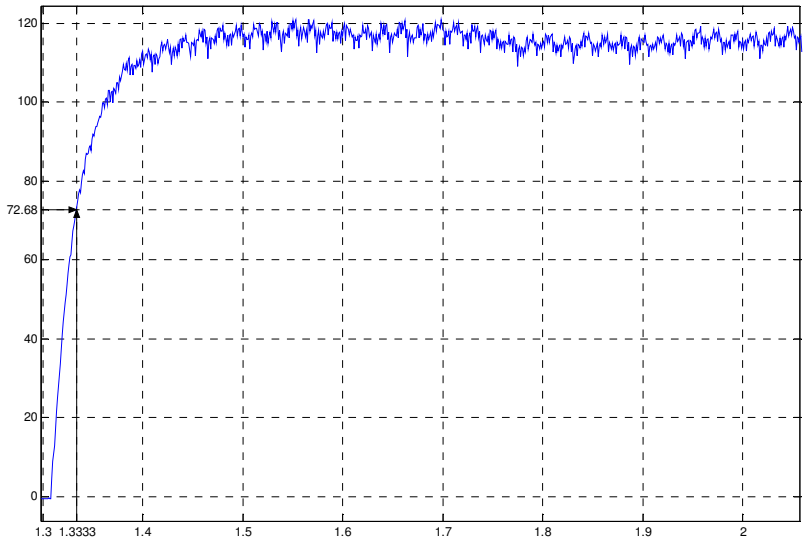
比較圖

轉速 A	15	20	25	30	35	40	平均
k 值							
a							
b							

τ							

轉速 A=15





轉速 A=15

$$k=\frac{110}{15}=7.33$$

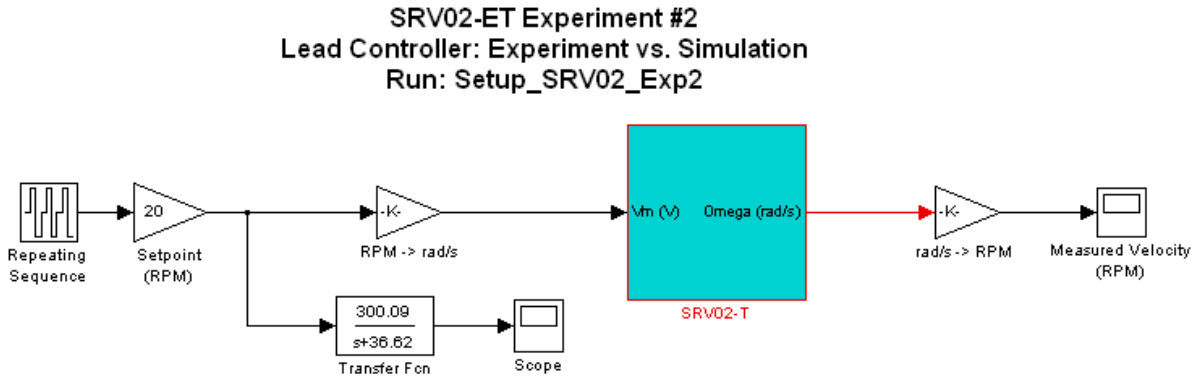
$$\tau = 1.3333-1.3156= 0.0267$$

$$a=\frac{1}{0.0267}=37.45$$

$$b=ak= 274.63$$

第二部分:驗證

（參考圖(二)之電路圖）



圖（二）驗證部分 simulink 接線圖

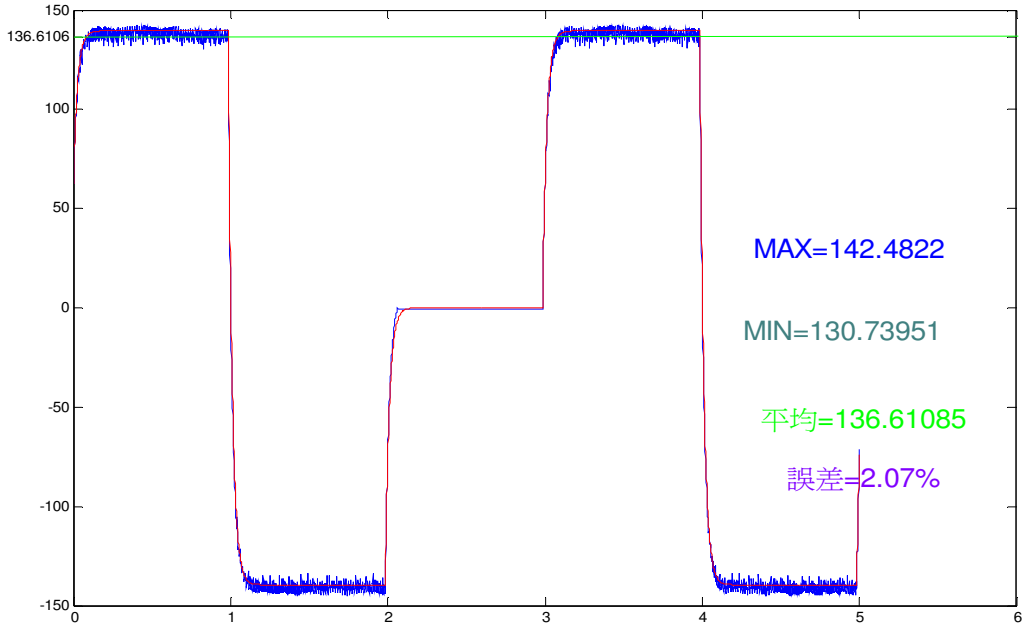
（實　驗　+　驗　證）　穩　態　誤　差　表　格

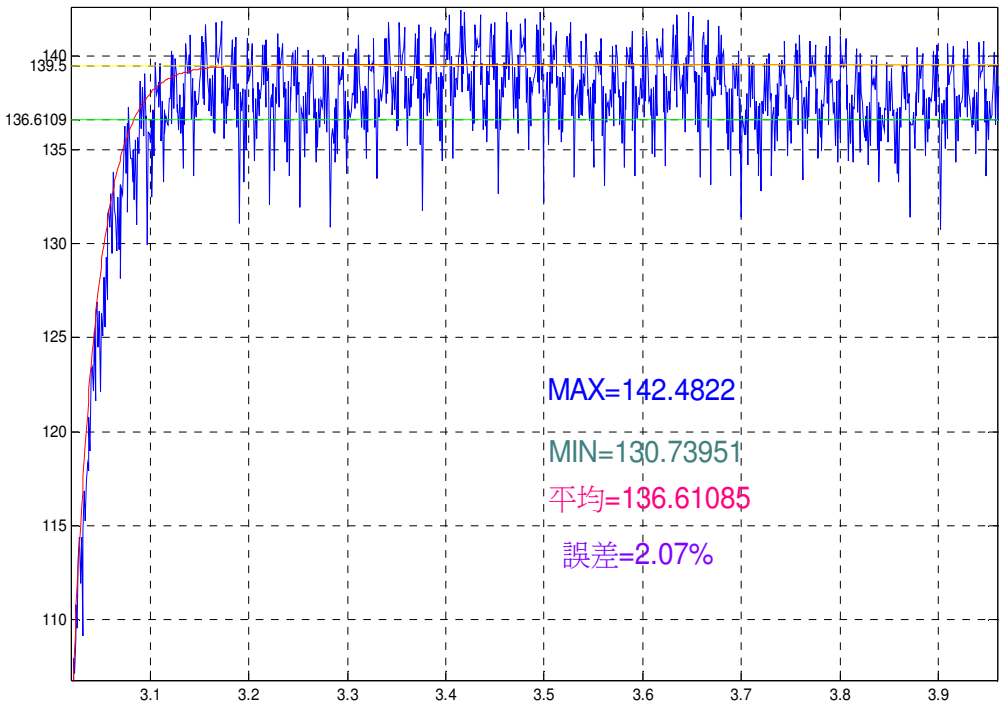
轉速 A	17	22	27	32	37	42
實驗穩態值						
驗證穩態值						
穩態百分誤差						

$$百分誤差(\%)=\frac{|驗證值-實驗值|}{實驗值}\times 100\%$$

A=17

A=17


150
136.6106
100
50
0
-50
-100
-150
0
1
2
3
4
5
6
MAX=142.4822
MIN=130.73951
平均=136.61085
誤差=2.07%


140
139.5
136.6109
135
130
125
120
115
110
3.1
3.2
3.3
3.4
3.5
3.6
3.7
3.8
3.9
MAX=142.4822
MIN=130.73951
平均=136.61085
誤差=2.07%

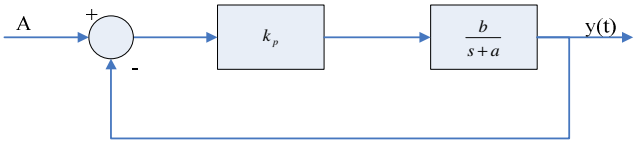
實習:直流伺服馬達速度回授 PID 控制器設計

1. 目的：

利用 P 控制，PI 控制器，解決馬達速度迴授的穩態誤差，並討論其暫態響應。

2. 原理：

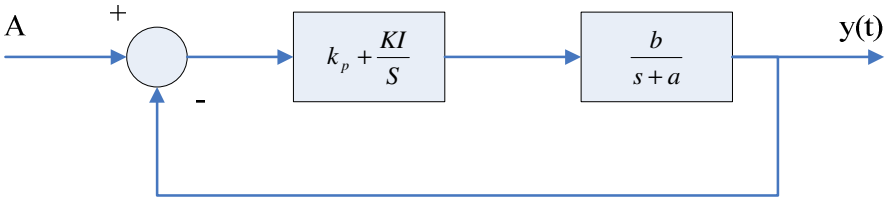
(1)P 控制器



$$G(s) = k_P \cdot \frac{b}{s+a} \qquad G_{Kp} = \lim_{s\to 0} G(s) = \frac{b \cdot kp}{a}$$

穩態誤差 $e_{ss} = \frac{1}{1+G_{k_P}} = \frac{a}{a+b\cdot kp}$ ，Kp↑ ， e_{ss}↓ 系統必存在穩態誤差

(2)PI 控制器



$$G(s) = (k_P + \frac{k_I}{s}) \cdot \frac{b}{s+a} = \frac{(k_P s + k_I)b}{s(s+a)}$$

$$Gk_p = \lim_{s\to 0} G(s) = \infty$$

穩態誤差 $e_{ss} = \frac{1}{1+Gk_P} = 0$

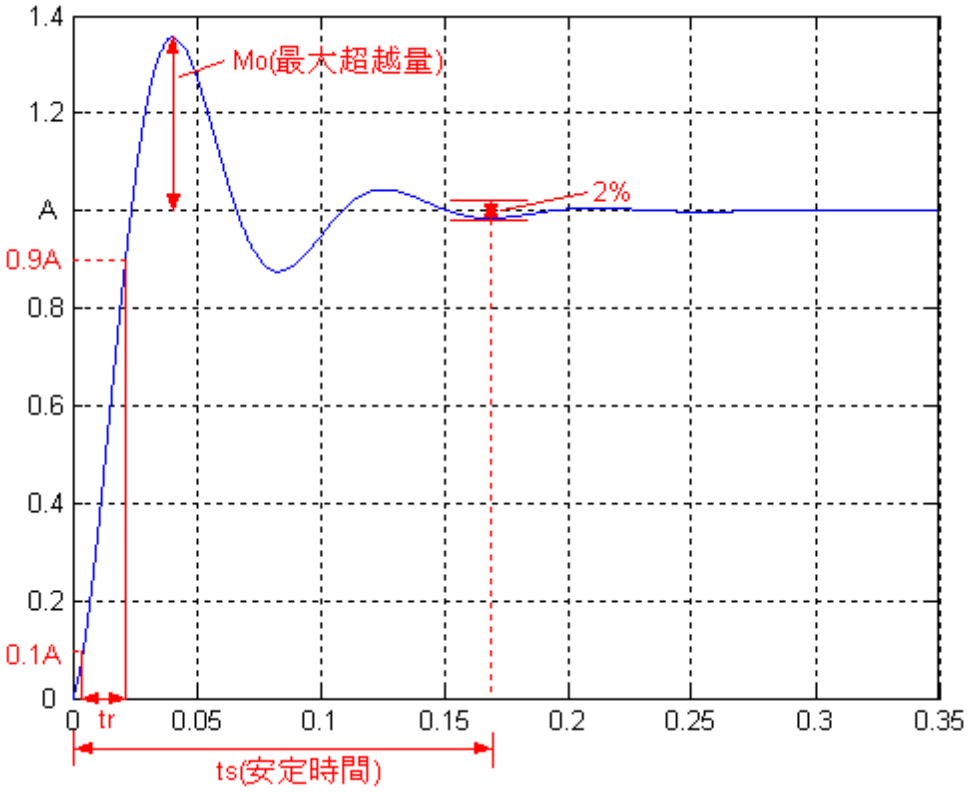
(3)同理 PID 控制器穩態誤差 e_{ss} = 0

(4) PI 控制器设计

$$G(s) = \frac{(k_P S + k_I)b}{s(s+a)}$$

$$T(s)=\frac{G(s)}{1+G(s)}=\frac{(k_p s+k_I)b}{s^2+as+k_p bs+k_I b}$$

$$\begin{cases}\omega_n{}^2=k_I b\\ 2\zeta\,\omega_n=k_p b+a\end{cases}$$



最大超越量：$M_0=e^{\frac{-\pi\,\zeta}{\sqrt{1-\zeta^2}}}\times A$　　最大超越量百分比：$P.O.=e^{\frac{-\pi\,\zeta}{\sqrt{1-\zeta^2}}}\times 100\%$

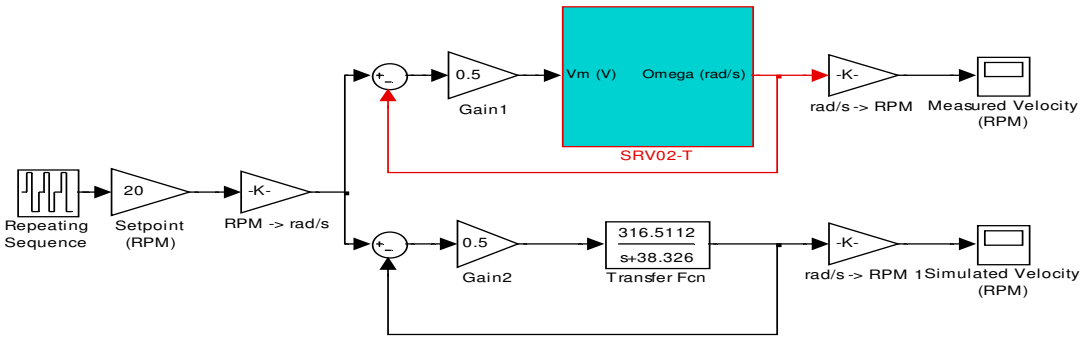
上升時間：$t_r\approx\frac{1.8}{\omega_n}$，(0.1～0.9)　　安定時間：$t_s\approx\frac{4}{\zeta\,\omega_n}$，(±2%)

3.實驗項目

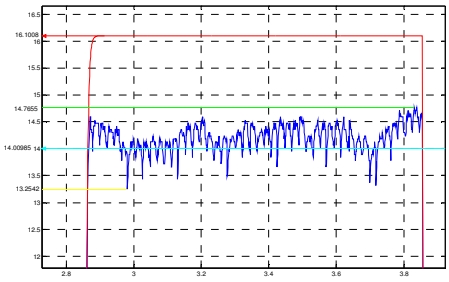
(1)P 控制器:

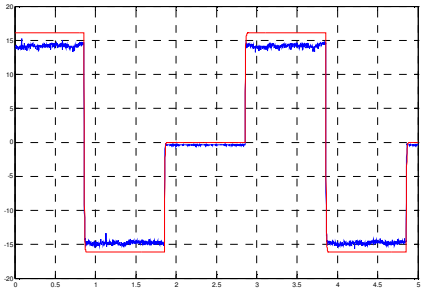
請畫出 k_p=0.5，1，2，5 之系統響應圖

電腦模擬與實驗值均要畫，且要作比較。(求出誤差百分比)



Kp=0.5





Kp=1

Kp=2

Kp=5

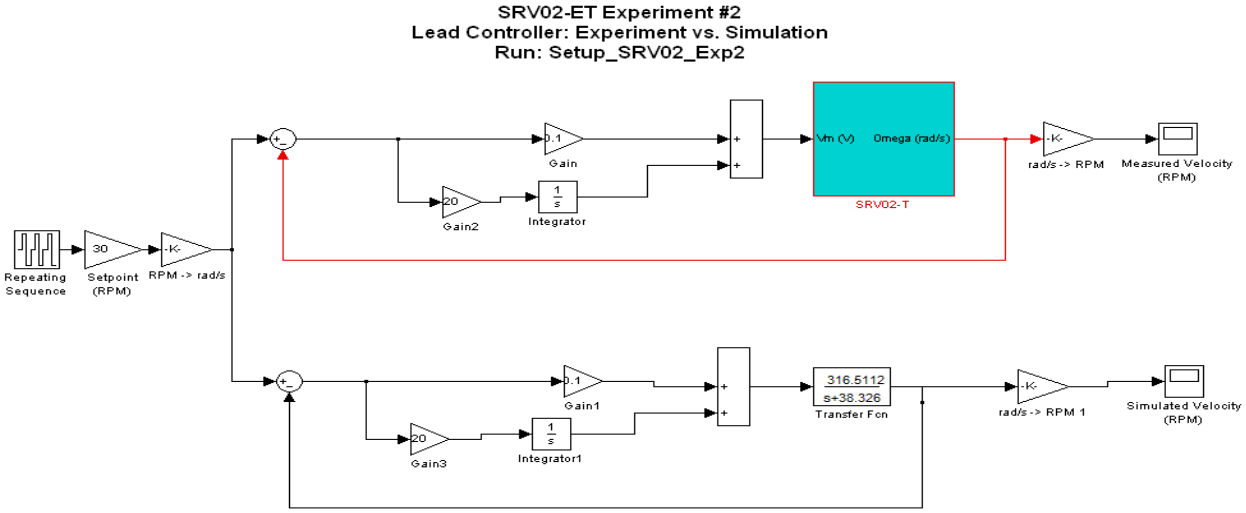
	Kp=0.5	1	2	5
實驗值	14.00985			
理論值	16.1008			
誤差百分比	14.9%			
實驗值穩態誤差	5.9			
理論值穩態誤差	3.9			

(2) PI 控制器

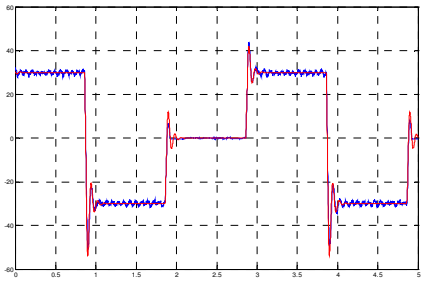
①k_I=20，k_p 可變:0.04，0.1，0.15，0.2

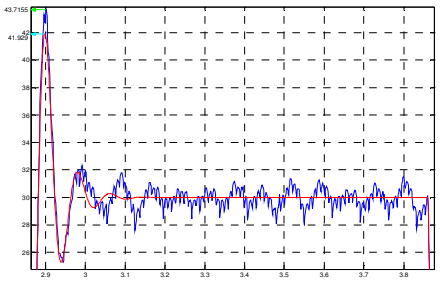
分別求出 P.O，t_r，t_s

比較電腦模擬與實驗值。(求出誤差百分比)



kp=0.04





Kp=0.1

KP=0.15

Kp=0.2

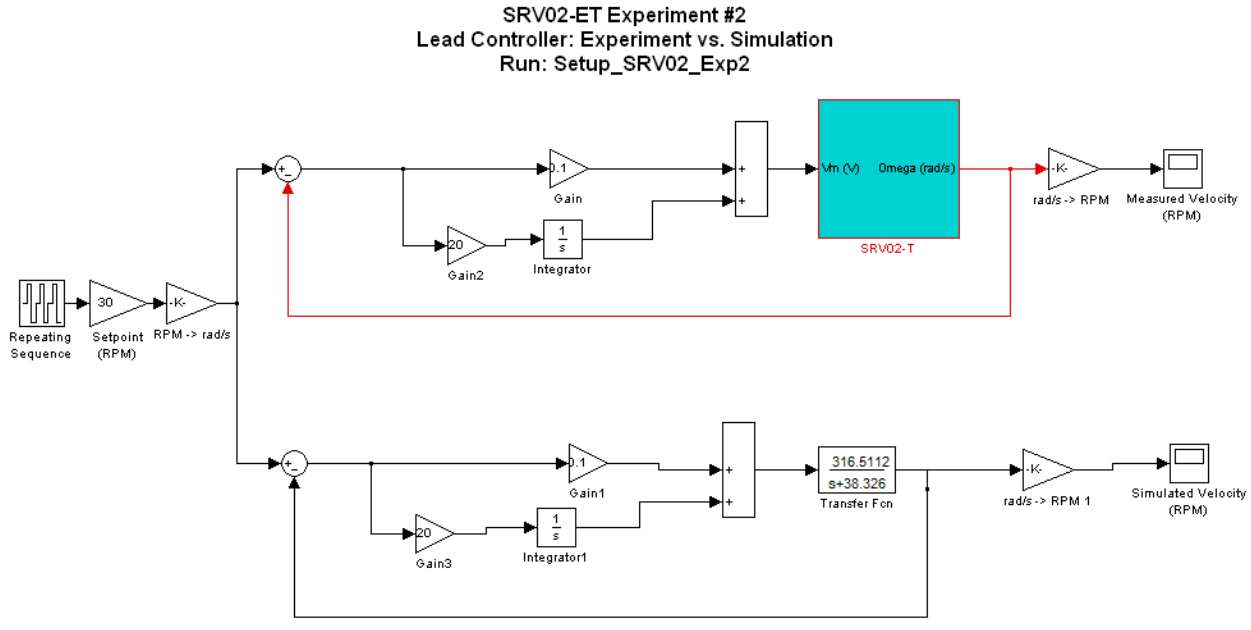
	Kp=0.04	0.1	0.15	0.2
M.O(實驗)	13.7155			
M.O(理論)	11.929			

M.O 誤差%	13%			
P.O	34.6%			
Tr	0.0226			
Ts	0.1571			

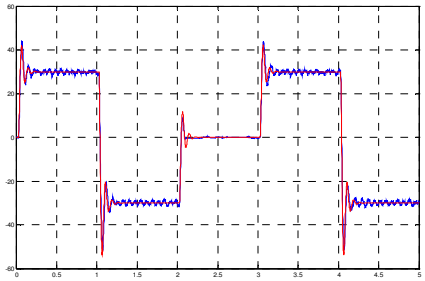
②k_p =0.1，k_I=10，20，50，100

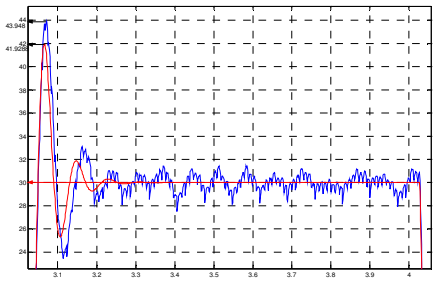
分別求出 P.O，t_r，t_s

比較電腦模擬與實驗值。(求出誤差百分比)



Ki=20





ki=10

Ki=50

ki=100

.

	Ki=10	20	50	100
M.O(實驗)	5.6932			
M,O(理論)	6.8617			
M.O 誤差%	20%			
P.O	0.18			
Tr	0.03199			
Ts	0.16			

3.利用根軌跡設計 PI 控制器

求出 k_p 與 k_I 滿足 P.O $\leq$ 5%，$t_s \leq$ 0.1sec

解:

$$P.O = 100e^{\frac{-\zeta\pi}{\sqrt{1-\zeta^2}}} \Rightarrow 0.1 = 100e^{\frac{-\zeta\pi}{\sqrt{1-\zeta^2}}} \Rightarrow 0.001 = e^{\frac{-\zeta\pi}{\sqrt{1-\zeta^2}}}$$

$$\Rightarrow -6.907 = \frac{-\zeta\pi}{\sqrt{1-\zeta^2}} \Rightarrow \frac{\zeta^2}{4.83} = 1-\zeta^2$$

$$\Rightarrow \frac{4.83}{5.83} = \frac{1}{\zeta^2} \Rightarrow \zeta = 0.91$$

$$Ts = \frac{4}{\varpi_n\zeta} \Rightarrow 0.1 = \frac{4}{\varpi_n\zeta} \Rightarrow \varpi_n\zeta = 40$$

$$\Rightarrow \varpi_n = 43.95$$

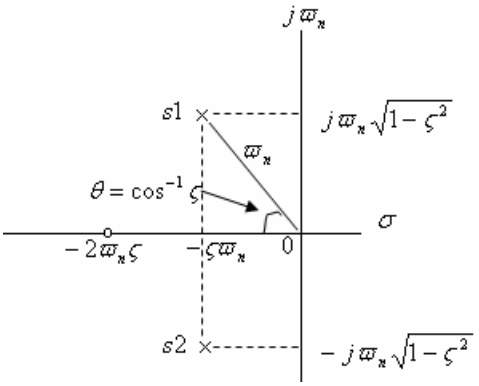
$$G(s) = \frac{(k_P S + k_I)b}{s(s+a)}$$

$$T(s) = \frac{G(s)}{1+G(s)} = \frac{(k_p s + k_I)b}{s^2 + as + k_p bs + k_I b}$$

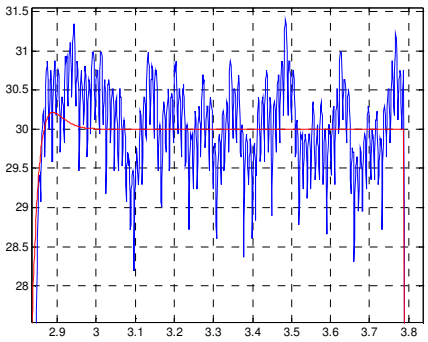
$$2\zeta\,\omega_n = k_p b + a$$

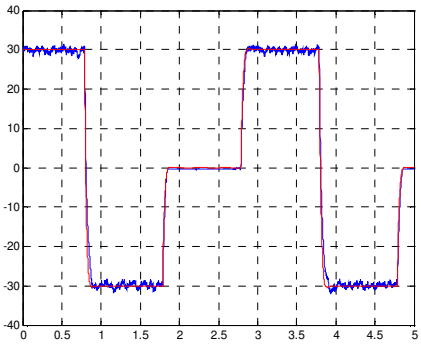
$$k_I = \frac{\omega_n^{\,2}}{b} = \frac{43.95^2}{316.5112} = 6.10447$$

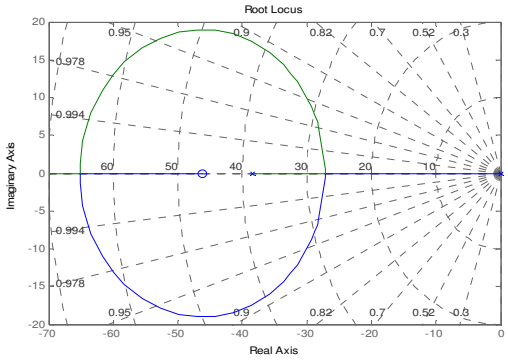
$$k_p = \frac{2\zeta\,\omega_n - a}{b} = \frac{2\times 0.91\times 43.95 - 38.326}{316.5112} = 0.13163$$



P.O%=0.1%　Wn=43.95　A=30　a=38.326　b=316.5112







左圖:

P.O=0.1%

Ts=0.1

Tr=0.04

把測試出來數據重新計算

$$k_I = \frac{\omega_n^{\,2}}{b} = \frac{43.95^2}{316.5112} = 6.10447$$

$$k_p = \frac{2\zeta\,\omega_n - a}{b} = \frac{2\times 0.91\times 43.95 - 38.326}{316.5112} = 0.13163$$

$$G(s) = (k_P + \frac{k_I}{s})\cdot\frac{b}{s+a} = \left(0.13163 + \frac{6.10447}{s}\right)\cdot\frac{316.5112}{s+38.326} = \frac{41.66s + 1923.133}{s^2 + 38.326s}$$

{二階系統頻率響應實習}

[1].實驗目的：

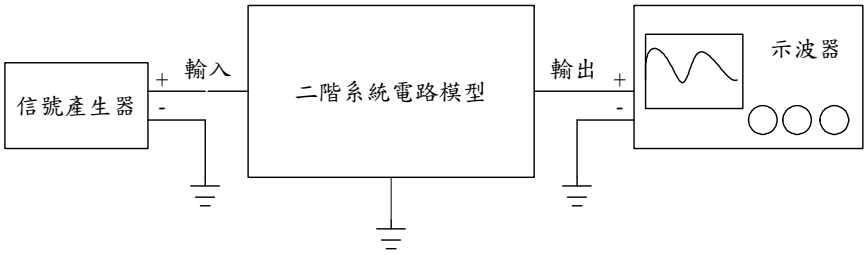
了解線性系統的頻率響關係

[2].實驗器材：

二階系統電路模型，信號產生器，數位示波器

[3].實驗步驟：

1. 線路架構：



2. 信號產生器之調整：正弦波

頻率：$f = \approx 0\,\text{Hz}$ to $5\,\text{Hz}$

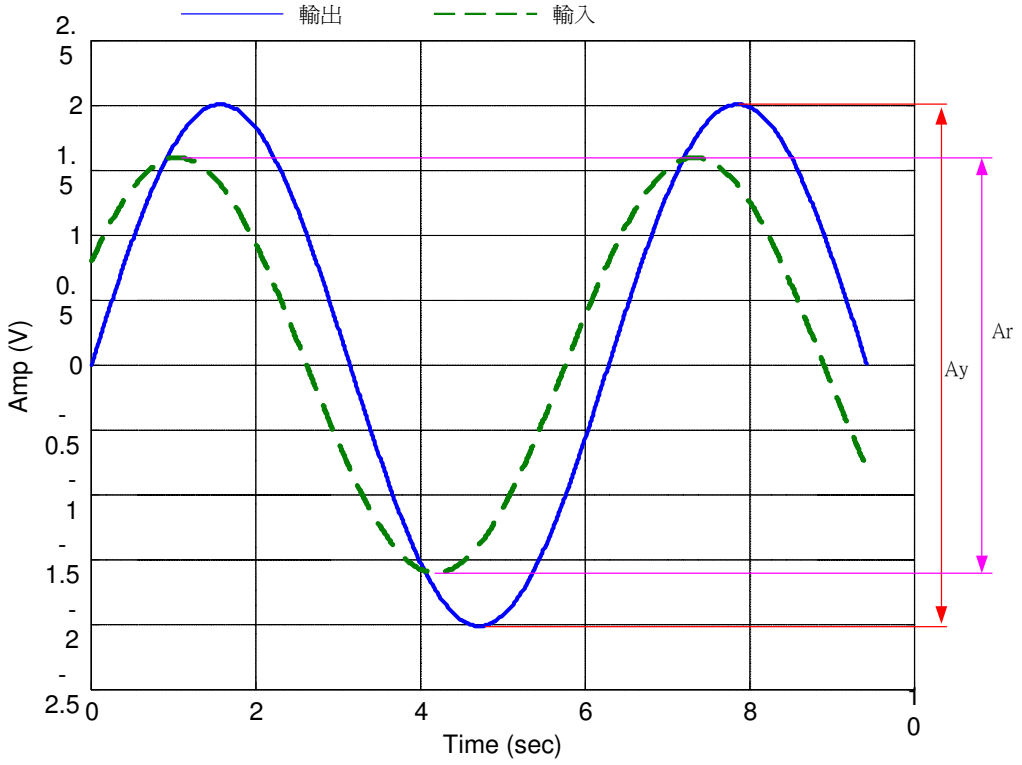
振幅：峰對峰值約 2V

量測資料點數約 20 組

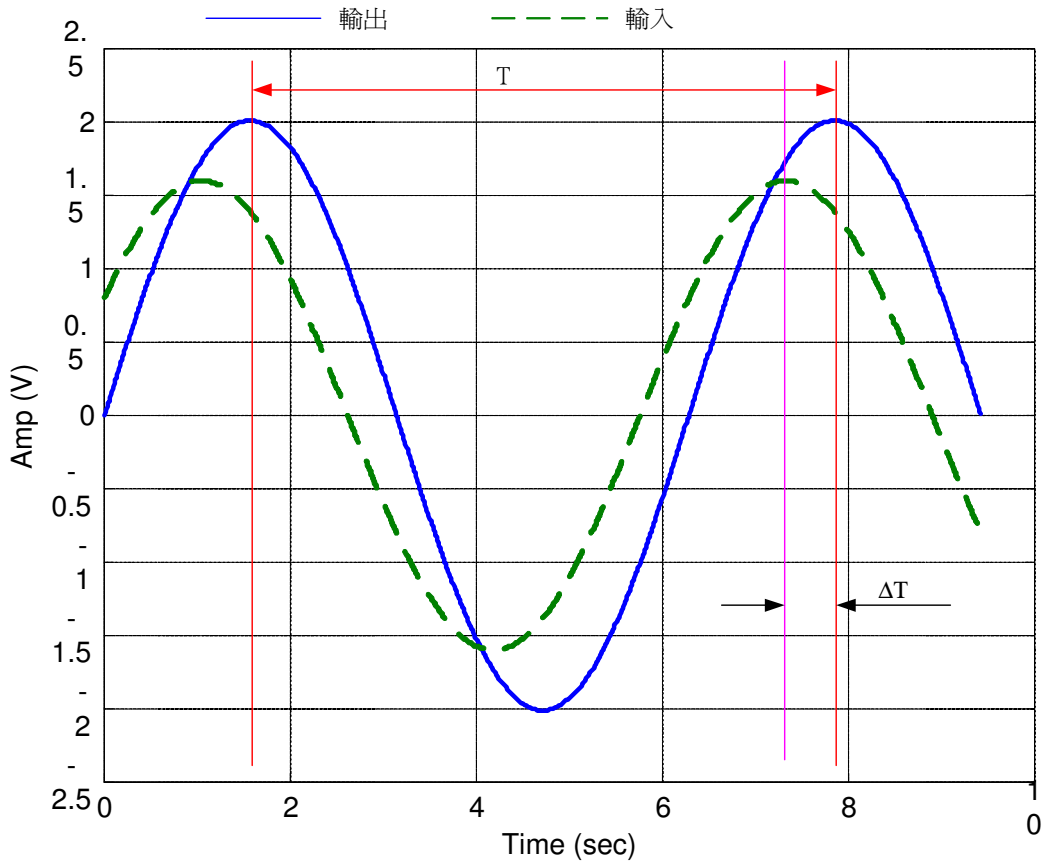
3. 數據資料量測：

每一組不同頻率的輸入訊號，量測其輸入和輸出信號之振幅及相角差

振幅量測 (Ar, Ay)：



週期(T)及相角時間差(ΔT)量測：



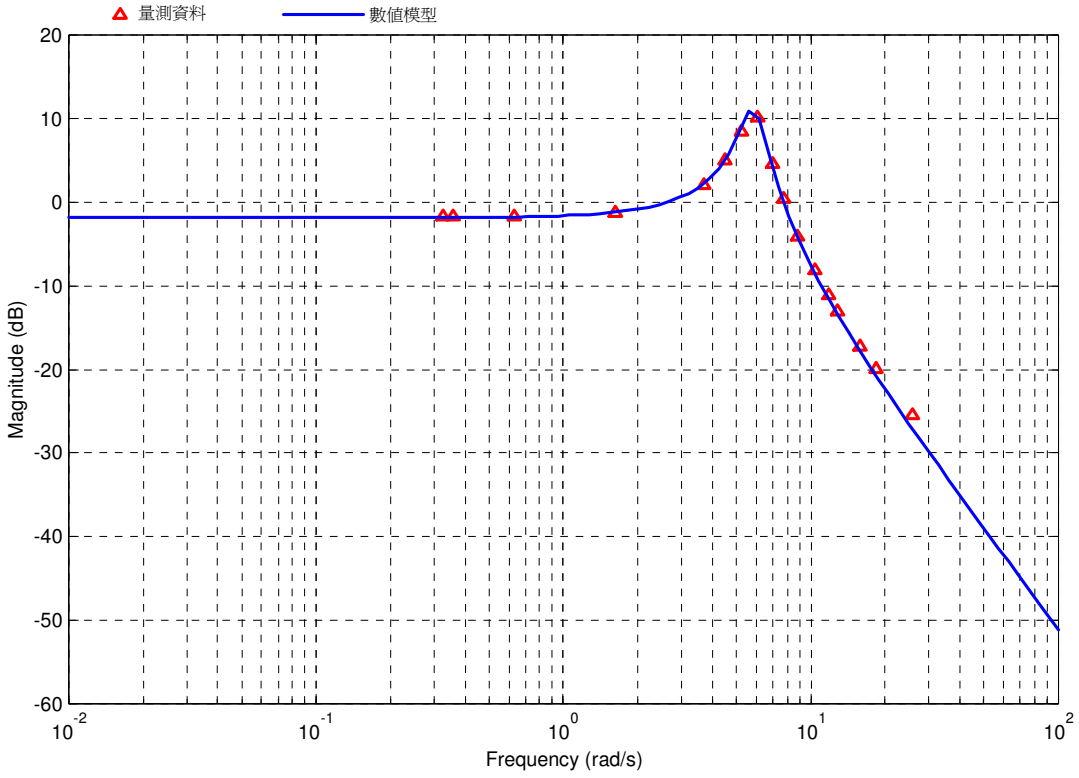
4. 記錄每一量測點的數值 (Ar, Ay, T,ΔT)，並計算此系統之增益 M，和相位差 θ ：

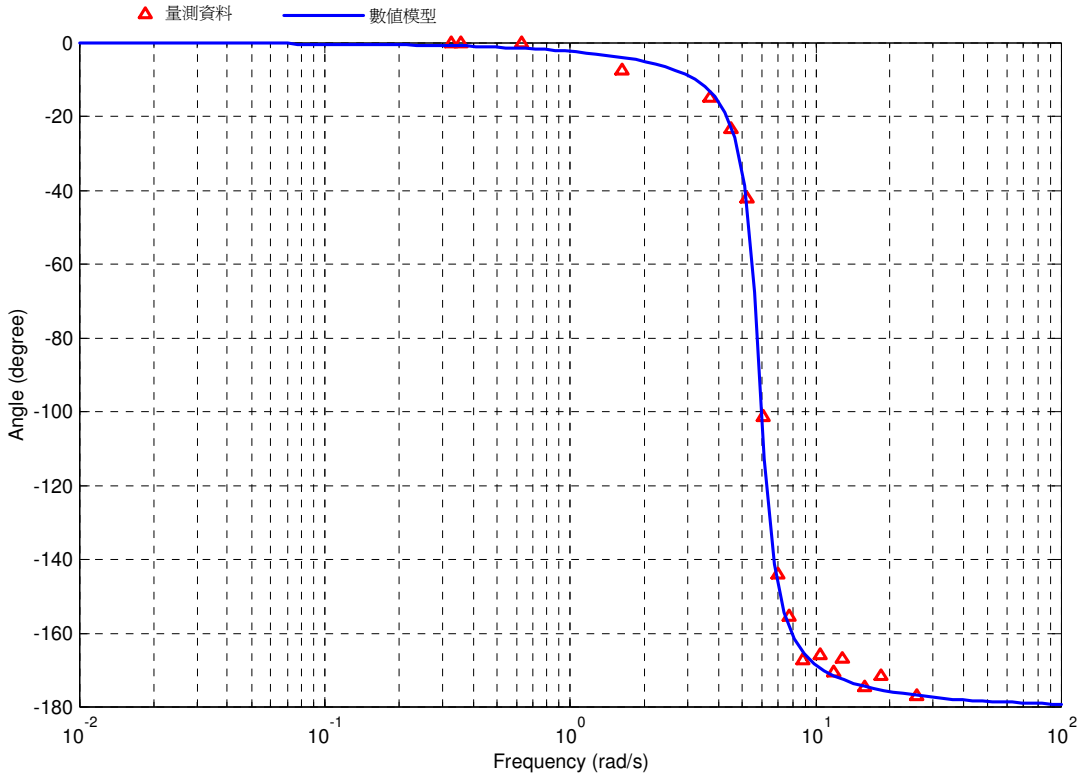
$$M(\omega)=\frac{Ay}{Ar}\ ,\qquad \theta(\omega)=\frac{\Delta T}{T}\times 360^{\circ}$$

5. 依據量測資料繪製波德圖，並和數值模型 G(s)加以比對，討論其結果。

範例：

頻率(Hz) f	角頻率(rad/s) f *2pi	輸入信號振幅 (Ar)	輸出信號振幅 (Ay)	增益 M=Ay/Ar	增益(dB) 20*log10(M)	dT (秒)	T (秒)	相位差 dT/T*360
0.052	0.32672536	2.02	1.66	0.821782178	-1.704865628	0	19.2308	0
0.057	0.35814126	2.02	1.66	0.821782178	-1.704865628	0	17.5439	0
0.101	0.63460118	2.02	1.66	0.821782178	-1.704865628	0	9.90099	0
0.257	1.61477726	2.02	1.76	0.871287129	-1.196774033	0.08	3.88	7.42268
0.588	3.69450984	2.02	2.58	1.277227723	2.12536673	0.07	1.70068	14.8176
0.719	4.51760642	2.02	3.6	1.782178218	5.019022626	0.09	1.39	23.30935
0.833	5.23388894	2.02	5.4	2.673267327	8.540847808	0.14	1.2	42
0.97	6.0946846	2.02	6.52	3.227722772	10.17792453	0.29	1.03	101.3592
1.11	6.9743298	2.02	3.44	1.702970297	4.624141462	0.36	0.9	144
1.235	7.7597273	2.02	2.1	1.03960396	0.337358506	0.35	0.81	155.5556
1.408	8.84671744	2.02	1.26	0.623762376	-4.099616487	0.33	0.71	167.3239
1.645	10.3358311	2.02	0.792	0.392079208	-8.132523757	0.28	0.608	165.7895
1.88	11.8123784	2.02	0.56	0.277227723	-11.14326685	0.252	0.532	170.5263
2.033	12.77370494	2.02	0.452	0.223762376	-13.00425869	0.228	0.492	166.8293
2.525	15.8650295	2.02	0.276	0.136633663	-17.28884575	0.192	0.396	174.5455
2.907	18.26520426	2.02	0.204	0.100990099	-19.91442404	0.164	0.344	171.6279
4.098	25.74847164	2.02	0.108	0.053465347	-25.43855228	0.12	0.244	177.0492




量測資料
數值模型
0
-20
-40
-60
-80
-100
-120
-140
-160
-180
Angle (degree)
10^-2
10^-1
10^0
10^1
10^2
Frequency (rad/s)

[控制系統之時域分析介紹]

對於控制系統時域行為之響應特性可分為暫態響應與穩態響應，影響系統暫態特性行為之主要因素來自系統本身之特性，而影響系統穩態特性行為之主要因素來自外加之輸入。以下將針對一階與二階系統之時域特性進行分析說明，而對於更高階系統之時域分析，由於目前並未有完整之數學理論分析與說明，因此將在最後兩小節中分別以系統加入零點與加入極點，對系統響應之影響來說明。

(1) 一階系統時域分析

一無零點一階系統可表示為

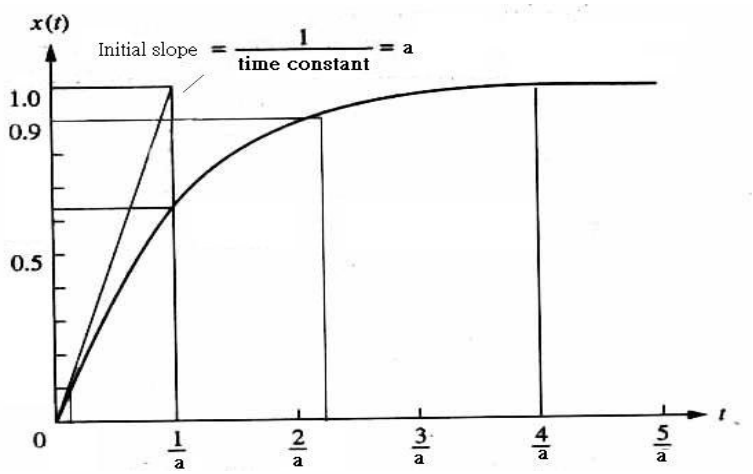
$$G(s)=\frac{a}{s+a}$$

若以單位步階函數輸入，即 $R(s)=\frac{1}{s}$，則

$$C(s)=\frac{a}{s(s+a)}=\frac{1}{s}-\frac{1}{s+a}$$

取反拉式轉換

$$c(t)=1-e^{-at}$$



(a) 響應時間常數 $\frac{1}{a}$：爲 e^{-at}的衰減指數特性，系統時間常數越小，則表示系統響應速度越快。

(b) 上升時間 Tr ：定義爲響應從系統終值的 0.1 倍至 0.9 倍所需的時間，即

$$Tr = T_{0.9} - T_{0.1} = \frac{2.31}{a} - \frac{0.11}{a} = \frac{2.2}{a}$$

(c) 安定時間 Ts ：定義爲響應到達系統終值的 2%誤差內所須的時間，即

$$C(Ts) \cong 0.98$$

$$Ts \cong \frac{4}{a}$$

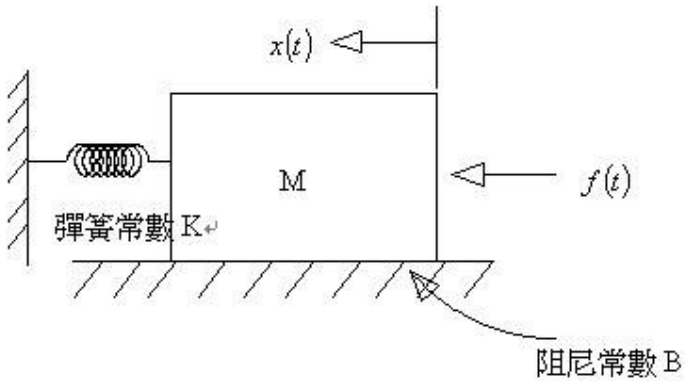
(2) 二階系統時域分析

一無零點二階系統可表示爲

$$G(s) = \frac{\omega_n^2}{s^2 + 2\zeta\omega_n s + \omega_n^2}$$

其中 ζ 爲系統阻尼比， ω_n 爲系統自然振動頻率。

若以彈簧-阻尼-質量運動系統來分析，則系統轉移函數爲：



圖彈簧、阻尼、質量運動系統

$$\frac{X(s)}{F(s)}=\frac{\frac{1}{M}}{s^2+\frac{B}{M}s+\frac{K}{M}}$$

$$=\frac{1}{K}\cdot\frac{K}{s^2+\frac{B}{M}s+\frac{K}{M}}$$

與二階系統表示式 $G(s)=\frac{\omega_n^2}{s^2+2\zeta\omega_n s+\omega_n^2}$ 比較，可知

$$\omega_n^2=\frac{K}{M}$$

$$2\zeta\omega_n=\frac{B}{M}$$

所以

$$\omega_n=\sqrt{\frac{K}{M}}\quad,\quad \zeta=\frac{B}{2\sqrt{KM}}$$

因此系統之自然振動頻率 ω_n 與 K 成正比，與 M 成反比；有阻尼比則與 B 成正比，與 $\sqrt{KM}$ 成反比。故自然振動頻率 ω_n 爲系統在無阻尼時之震動特性，即 $\zeta=0$，而阻尼比 ζ 則是與系統摩擦有關的系統特性。

對於二階系統轉移函數的兩極點爲

$$S_{1,2}=-\zeta\omega_n\pm\omega_n\sqrt{\zeta^2-1}$$

當 $\zeta>1$ 時爲兩相異根，爲過阻尼系統。

當 $\zeta-1$ 時爲兩重根，爲臨界阻尼系統。

當 $\zeta<1$ 時爲兩共軛複數根，爲欠阻尼系統。

當 $\zeta=C$ 時爲兩純虛根，爲無阻尼系統。

其響應圖形<如圖>

(a) 時間常數 $\frac{1}{\zeta\omega_n}$：爲 $e^{-\zeta\omega_n t}$的衰減指數特性。

(b) 上升時間 T_r：定義爲響應從系統終值的 0.1 倍至 0.9 倍所須的時間，即；

$$T_r = T_{0.9} - T_{0.1}$$

(c) 安定時間 T_s：定義爲響應到達系統值的 2%誤差內所須的時間，即

$$C(Ts) \cong 0.98$$

$$Ts \cong \frac{4}{\zeta\omega_n} \quad , \quad 0 < \zeta < 0.55$$ 誤差在 10%內。

(d) 峰值時間 T_p：定義爲響應達到第一個最大值的時間，即

$$\dot{C}(T_p) = 0$$

$$T_p = \frac{\pi}{\omega_n\sqrt{1-\zeta^2}}$$

(e) 超越量百分比 $\%OS$：定義爲在峰值時間的超越量，即

$$\%OS = \frac{C(T_p) - C(\infty)}{C(\infty)} \times 100\%$$

$$= e^{-\sigma/\sqrt{1-\zeta^2}} \times 100\%$$

(3) 增加極點之系統響應：

對於系統特性分析，如時間常數、上升時間、安定時間、峰值時間、超越量百分比等公式，僅適用於一階或二階無零點系統。

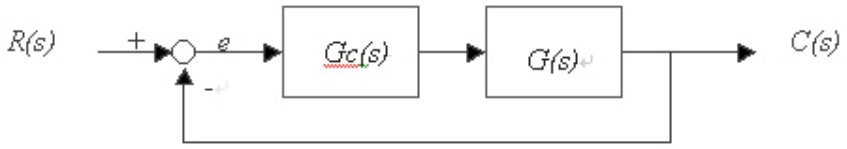
若系統爲二階以上且有零點，則其系統特性很難推導公式來分析系統特性，但是對於高階系統人可以一近似之二階系統或一階系統來分析。因此若假設三階系統，及二階系統在多一極點，即

$$G(s) = \frac{\omega_n^2}{(s+\sigma_r)(s^2 + 2\zeta\omega_n s + \omega_n^2)}$$

則其單位步階輸入之響應爲：

{PID 控制器實作}[12]

對使用 PID 控制器之閉迴路方塊圖如下



PID 控制器的轉移函數 *Gc(s)*如下式

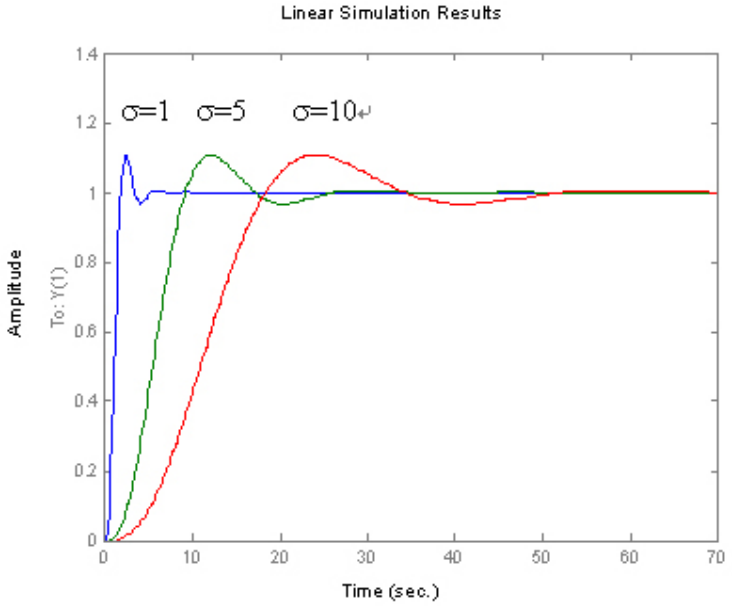
$$Gc(s) = Kp \cdot (1 + \frac{1}{Ki \cdot s} + Kd \cdot s)$$

其中，*Kp* 爲比例增益、*Ki* 爲積分常數、*Kd* 爲微分常數。

若以參考模型法假設三階控制系統之閉迴路響應特性爲

$$Rm(s) = \frac{1}{1 + \sigma \cdot s + 0.5(\sigma \cdot s)^2 + 0.15(\sigma \cdot s)^3}$$

其中，σ是與響應時間相關的常數，當σ分別爲 1, 5, 10 時的響應分別如下圖



由上圖可知當參考模式之參數 σ 越小時，其系統之暫態響應速度越快。因此，若要設計一響應速度快之系統時，其參考模式之參數應選擇較小值。

若一二階系統爲

$$G(s) = \frac{N(s)}{D(s)}$$

將其分子分母同除分子並將分母表示爲 s 之升幕，即

$$\begin{aligned} G(s) &= \frac{1}{\frac{D(s)}{N(s)}} \\ &= \frac{1}{\beta_0 + \beta_1 s + \beta_2 s^2 + \beta_3 s^3 + \ldots} \\ &= \frac{1}{\beta(s)} \end{aligned}$$

另外，將 PID 控制器 *Gc(s)*之分子同樣表示爲 s 之升幕，即

$$\begin{aligned} Gc(s) &= \frac{\frac{Kp}{Ki} + Kp \cdot s + Kp \cdot Kd \cdot s^2}{s} \\ &= \frac{\gamma_0 + \gamma_1 s + \gamma_2 s^2}{s} \\ &= \frac{\gamma(s)}{s} \end{aligned}$$

則閉迴路轉移函數 $M(s) = \frac{Gc(s) \cdot G(s)}{1 + Gc(s) \cdot G(s)}$ 可表示爲

$$\begin{aligned} M(s) &= \frac{\gamma(s)}{\gamma(s) + \beta(s) \cdot s} \\ &= \frac{1}{1 + \frac{\beta(s)}{\gamma(s)} s} \end{aligned}$$

若要設系此閉迴路系統之響應特性與參考模式相同，則須令上式等於參考模式，即

$$\frac{1}{1 + \frac{\beta(s)}{\gamma(s)} s} = Rm(s)$$

整理後可表示爲

$$\gamma(s) = \frac{\beta(s)}{\frac{1}{Rm(s)} - 1} s$$

將上式右邊表示爲 s 之升幕，故

$$\frac{\beta_0}{\sigma}+\frac{\beta_1-0.5\cdot\sigma\beta_0}{\sigma}s+\frac{\beta_2-0.5\cdot\sigma\beta_1+0.1\cdot\sigma^2\beta_0}{\sigma}s^2+\frac{\beta_3-0.5\cdot\sigma\beta_2+0.1\cdot\sigma^2\beta_1-0.025\cdot\sigma^3\beta_0}{\sigma}s^3$$

$$=\frac{Kp}{Ki}+Kp\cdot s+Kp\cdot Kd\cdot s^2$$

由於上式等於 PID 控制器之係數，因此 s^3 項係數需爲 0，解方程式後可解出σ，即

$$\beta_3-0.5\cdot\sigma\beta_2+0.1\cdot\sigma^2\beta_1+0.025\cdot\sigma^3\beta_0=0$$

由上市可解出三個根，可依系統設計之需求選擇適當之σ值。因此，將σ代入下列式子，則可求出 *Kp*、*Kd*、*Ki*

$$Kp\cdot Kd=\frac{\beta_2-0.5\cdot\sigma\beta_1+0.1\cdot\sigma^2\beta_0}{\sigma}$$

$$Kp=\frac{\beta_1-0.5\ \sigma\beta_0}{\sigma}$$

$$\frac{Kp}{Ki}=\frac{\beta_0}{\sigma}$$

以下以一例子說明，若控制系統 *G(s)*爲

$$G(s)=\frac{1}{s(1+0.2s)(1+0.5s)}$$

則

$$G(s)=\frac{1}{s+0.7s^2+0.1s^3}$$

故β_0= 0、β_1= 1、β_2= 0.7、β_3= 0.7 代入公式可得

$$0.1-0.5\times0.7\sigma+(0.5^2-0.15)\sigma^2=0$$

解上式可得二根

σ= 0.314, 3.185

採用其中較小之正根 0.314，則可得

$$Kp=\frac{\beta_1-0.5\cdot\sigma\beta_0}{\sigma}$$

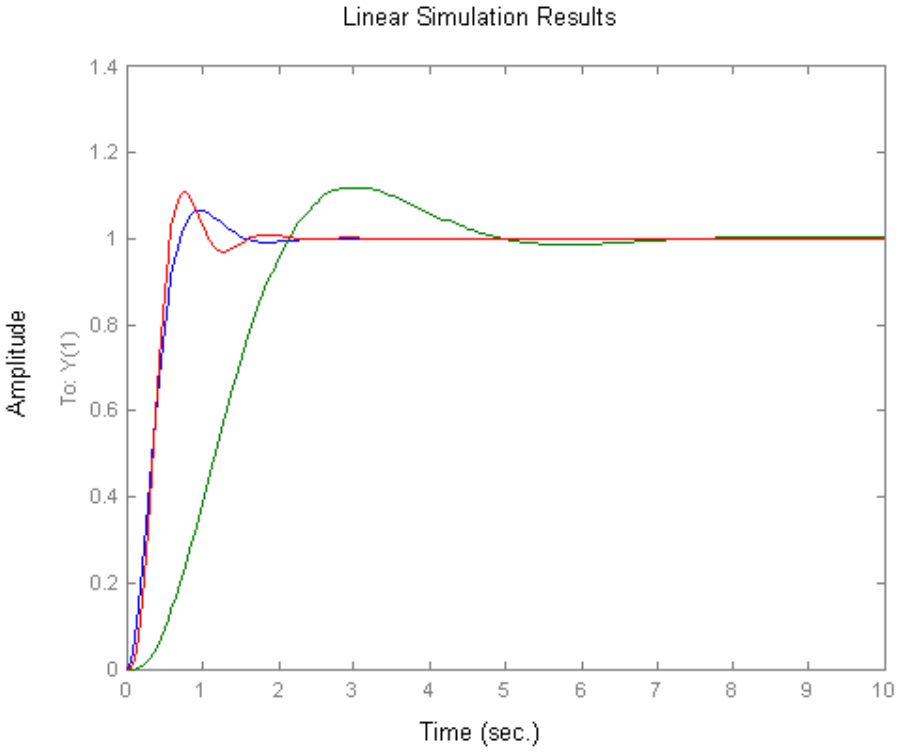
$$=3.185$$

$$Kd = \frac{(\beta_2 - 0.5\cdot\sigma\beta_1 + 0.1\cdot\sigma^2\beta_0)}{\sigma}$$
$$= \frac{(0.7 - 0.5\times 0.314\times 1)}{0.314}$$
$$= 1.729$$

$$Kp\,!\,Ki = \frac{\beta_0}{\sigma}$$
$$= 0$$

因此，可得 PID 控制器爲

$$Gc(s) = \frac{\frac{Kp}{Ki} + Kp\cdot s + Kp\cdot Kd\cdot s^2}{s}$$
$$= \frac{3.185s + 1.729s^2}{s}$$
$$= 3.185 + 1.729s$$



上圖中紅色爲參考模式之步階系統響，綠色爲系統未使用控制器之閉迴路響應，而藍色則爲利用參考模式設計 PID 控制器之閉迴路系統響應。比較使用控制與未使用控制器兩響應曲線可知，採用 PID 控制器之系統響應無論在暫態響應速度與達到穩態的時間，均有大幅的改善

{應用系統實習}[16]

[16].