Scilab 簡介 4

2011/10/13

Chih-Han Lin 林致翰

clin@ltl.iams.sinica.edu.tw
or r99245002@ntu.edu.tw

編輯較大型的程式請遵守以下原則：

- 不希望將計算結果即時顯示在命令列上的中間計算指令請記得在該句末加上分號(新版 Scilab 的 scinote 已自動修正)

- 加上適當的註解使用連續兩個斜線 // (double-slash)

- 變數名稱要設的易懂且有意義，常用的命名方式有兩種:一種使用縮寫加手自大寫，如 DisplsmtOfArCrft， 一種使用縮寫加底線，如 Displsmt_of_ar_crft。

- 有意義的常數值要設成變數以方便可能的修正。
- 程式落入無窮迴圈想終止時在命令列視窗按 Ctrl + c，再輸入 abort 或 resume
- 合理縮排，區分內外層迴圈，增加可讀性

撰寫一個完整的程式

此段程式碼在 LNEP_programming.pdf page. 17

```
// constant table

grvy = 9.8; // m/s, acceleration of gravity
m = 1; // kg, mass of the ball
time = 2E2; // total simulation time, sec
t_step = 1E-3; // ms
```

大數值使用科學記號 aEb=a*10^b 可讓版面更簡潔

```
x0=0; // initial displacement, meter
v0=0; // initial velocity, m/s
```

註解中標示常數單位

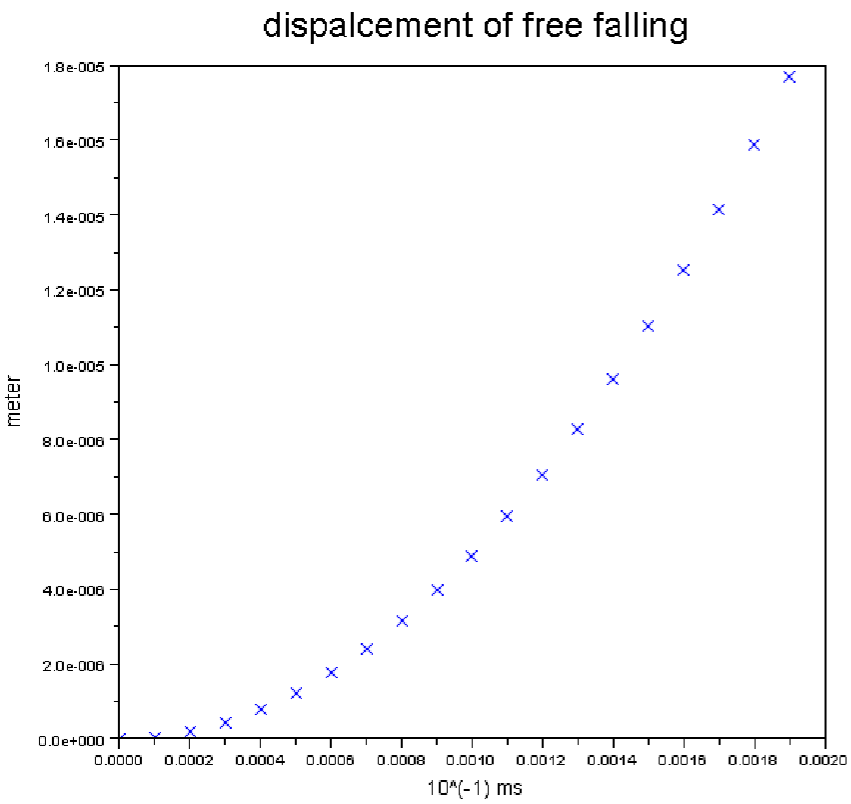
將計算中會用到的常數值與初始條件放至文件開頭

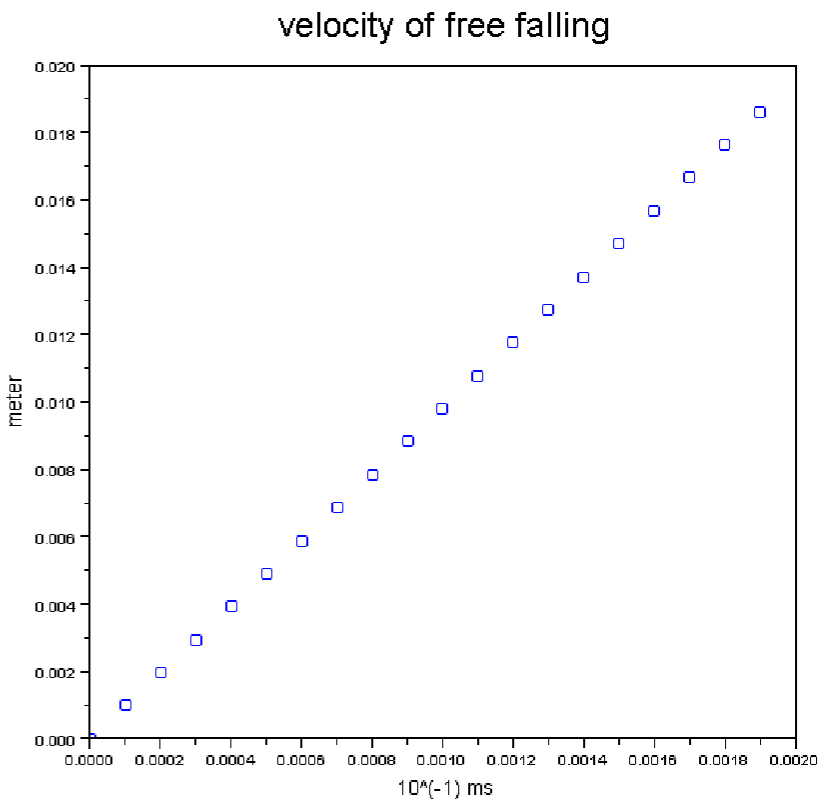
```
// ------parameter-------------
total_step = roundtime/t_step;
// total simulation step
t_scale = (1:total_step)*t_step ;
// create corresponding t-axis
// 此模擬預先設定總模擬時間與單步間距，總步數與對應的時間
// 軸由其導出 當然，用 linespace 來作切割更簡單

// ------free falling------
displacement = x0+0.5*grvy*t_scale.^2;
vlcty = v0+grvy*t_scale;
// 完成參數設定便進行計算
//此範例有解析解，直接帶入時間軸陣列即可推算出位移與速度
```

```
// ----------plot--------------------
// subplot 可設定子圖顯示，subplot(klm) 代表有
// k 列 x l 欄 = m 個子圖，m 為當前子圖編號，
// 由第一列左至右依序填入圖塊，第一列填滿後再填入下一列
// ex: | subplot(231) | subplot(232) | subplot(233) |
//     | subplot(234) | subplot(235) | subplot(236) |

subplot(121)
plot( t_scale,displacement,'x');
xtitle('dispalcement of free falling',...
'10^(-1) ms','meter');

// plot(a(1:n),b(1:n)) 代表以 (x1,y1)=(a(1,1),
// b(1,1)),(x2,y2)=(a(1,2),b(1,2)) 等等
// n 個數據點進行二維繪圖，plot(a,b,'s') 中的 's'
//   為繪圖控制字串，'x' 代表以 x 標記數據點

subplot(122)
plot(t_scale,vlcty,'o');
xtitle('velocity of free falling',...
'10^(-1) ms','meter');
```

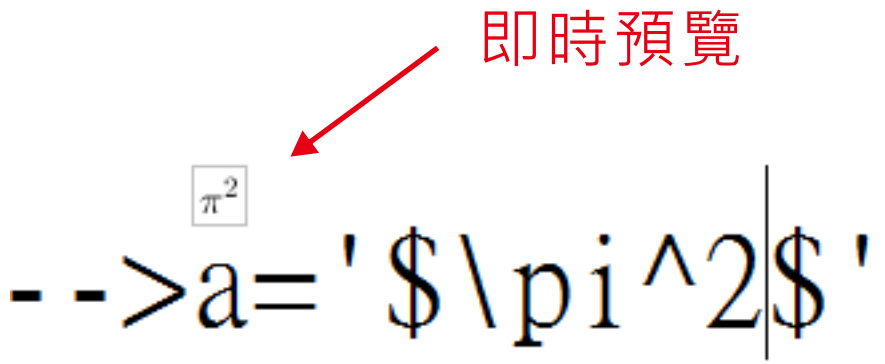
xtitle 可以設定標題、座標軸

執行結果


dispalcement of free falling
meter
10^(-1) ms
0.0e+000
2.0e-006
4.0e-006
6.0e-006
8.0e-006
1.0e-005
1.2e-005
1.4e-005
1.6e-005
1.8e-005
0.0000
0.0002
0.0004
0.0006
0.0008
0.0010
0.0012
0.0014
0.0016
0.0018
0.0020

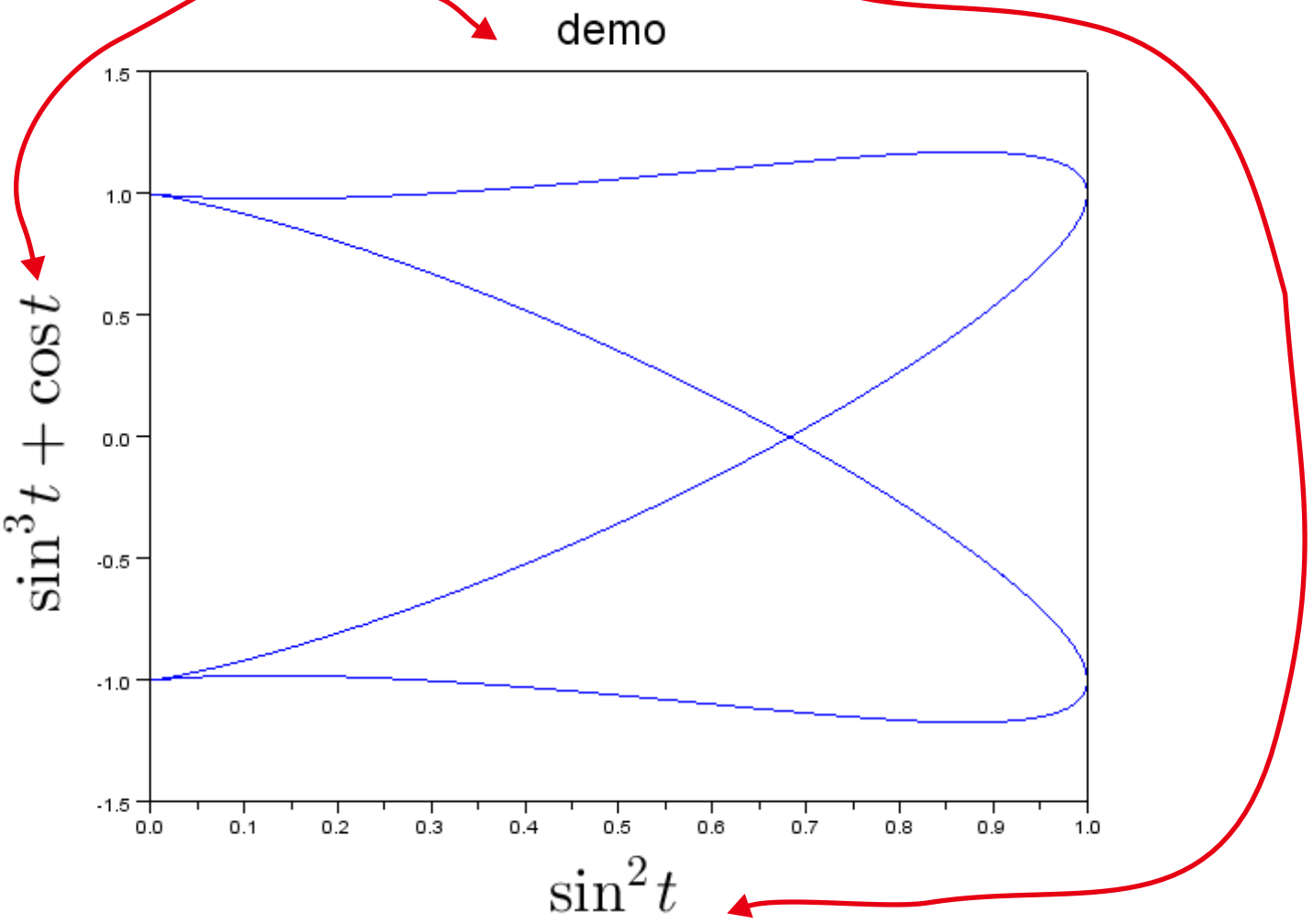

velocity of free falling
meter
10^(-1) ms
0.000
0.002
0.004
0.006
0.008
0.010
0.012
0.014
0.016
0.018
0.020
0.0000
0.0002
0.0004
0.0006
0.0008
0.0010
0.0012
0.0014
0.0016
0.0018
0.0020

Scilab + LaTeX

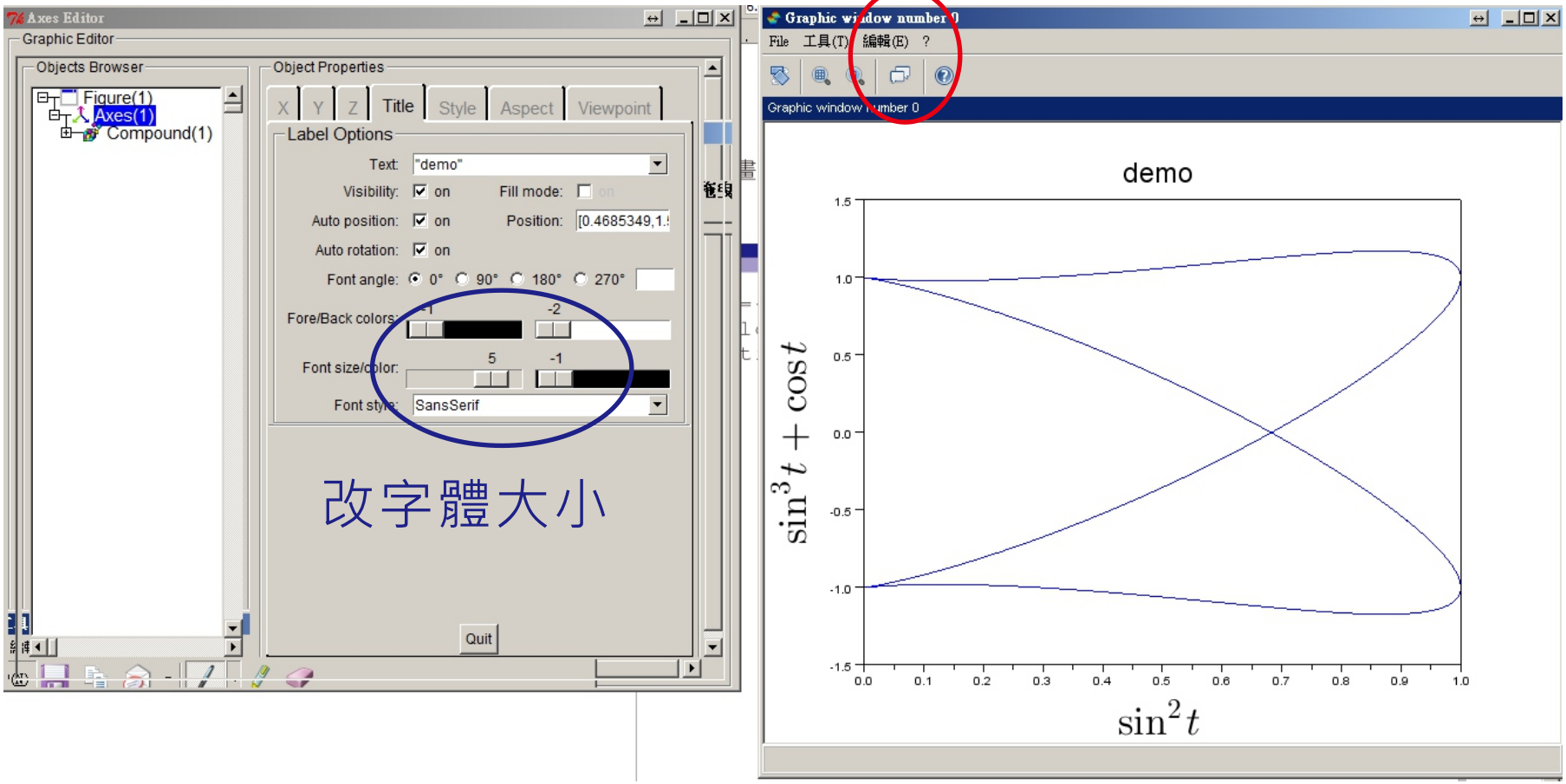


在 Scilab 中要輸入字串時，使用 $ $ 框住的代碼會被當成是 LaTeX inline Math 的代碼，在標示圖片資訊的時候我們很常使用它來打出美觀的數學式

```
-->t=-20:0.01:20;
-->plot(sin(t).^2,sin(t).^3+cos(t))
-->xtitle('demo','$\sin^2t$','$\sin^3t+\cos t$')
```



編輯 –> 座標軸特性


Axes Editor
Graphic Editor
Objects Browser
Figure(1)
Axes(1)
Compound(1)
Object Properties
X
Y
Z
Title
Style
Aspect
Viewpoint
Label Options
Text: "demo"
Visibility: on
Fill mode: on
Auto position: on
Position: [0.4685349,1.
Auto rotation: on
Font angle: 0° 90° 180° 270°
Fore/Back colors: -1 -2
Font size/color: 5 -1
Font style: SansSerif
改字體大小
Quit
Graphic window number 0
File 工具(T) 編輯(E) ?
Graphic window number 0
demo
$\sin^3 t + \cos t$
$\sin^2 t$
1.5
1.0
0.5
0.0
-0.5
-1.0
-1.5
0.0 0.1 0.2 0.3 0.4 0.5 0.6 0.7 0.8 0.9 1.0

微分方程模擬

考慮一個自由下墜的銅球，假設這個球除了重力之外，還受到空氣的黏滯力作用。先假設黏滯力與速度 v 成平方正比，則球的運動方程式可寫爲:

$$m\frac{dv}{dt} = mg - \gamma v^2 \tag{5.1}$$

γ 爲黏滯係數。我們知道這個微分其實就是

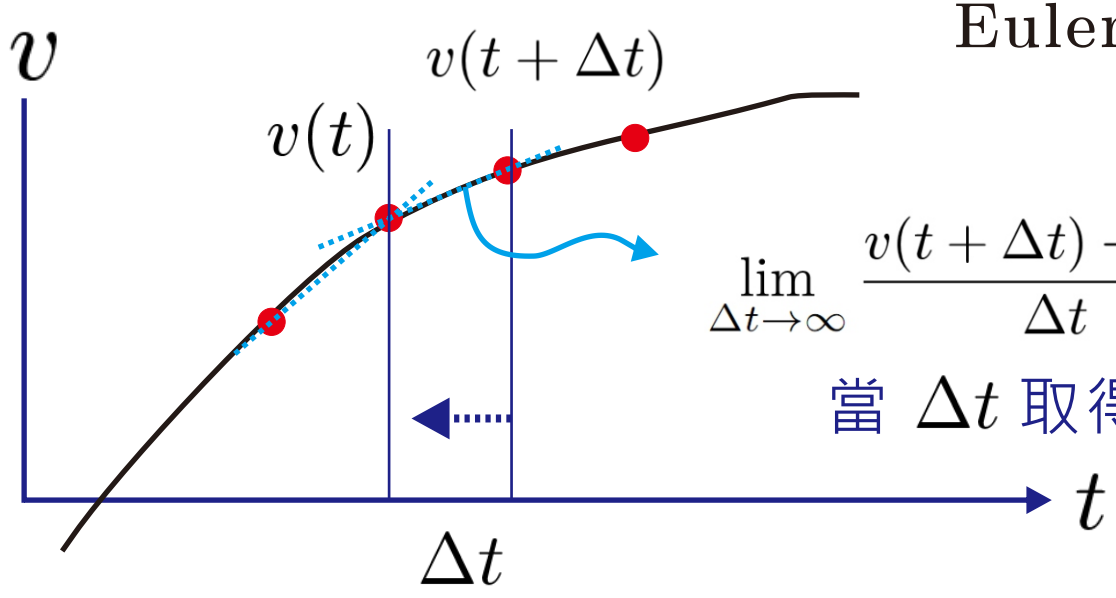
$$\lim_{\Delta t \to \infty} \frac{v(t+\Delta t) - v(t)}{\Delta t} \tag{5.2}$$

我們可以把運動方程式離散化（將 Δt 取爲足夠小的數，比方說 $\Delta t = 1\mu\text{s}$），

$$m\frac{v_{i+1} - v_i}{\Delta t} = mg - \gamma v_i^2 \tag{5.3}$$

課程網站->課程資料-> 課程講義 -> 2009 版講義 lnep1fall_2009-2.pdf

Euler Method (google&wiki)



$$\lim_{\Delta t \to \infty} \frac{v(t+\Delta t) - v(t)}{\Delta t}$$

當 Δt 取得夠小，會逼近 $v(t)$ 點斜率

相當於使用該點近似斜率(由微分方程決定)來求出時間上相鄰 v 的數值

設定任意的起始值 v_i，則經過 Δt 後的速度變爲

$$v_{i+1} = v_i + \left(g - \frac{\gamma v_i^2}{m}\right)\Delta t \qquad (5.4)$$

重複進行迭代以後我們可以算出一系列間隔 1μ s 的 v 值存在某個陣列裡，在透過適當地縮放指標釋回另外計算的單位即完成模擬。

程式碼

```
// constant table
gvy=9.8; // m/s
mass=1; // kg
gamm=1; // kg/m
order=0;
v0=0; // m/s
simu_times=10; // 10^(order) sec

sample_point=5000;
dt=simu_times/sample_point*10^(order); // *10^-3
// equation
```

宣告常數

總模擬時間

總模擬步數

根據模擬步數與模擬時間換算的 dt 大小

```
function [vlcty]=dynamic(v_initial ),
vlcty=v_initial+(gvy-gamm*v_initial/mass)*dt;
endfunction
```

根據（5.4）撰寫之函數，輸入 v(i), 得到 v(i+1)

```
// create velicity array

vlctyarray=zeros(1,sample_point);
vlctyarray(1)=v0;
```

給定初始條件

```
// calculation
for i=1:sample_point-1,
vlctyarray(i+1)=dynamic(vlctyarray(i));
end;
```

執行計算

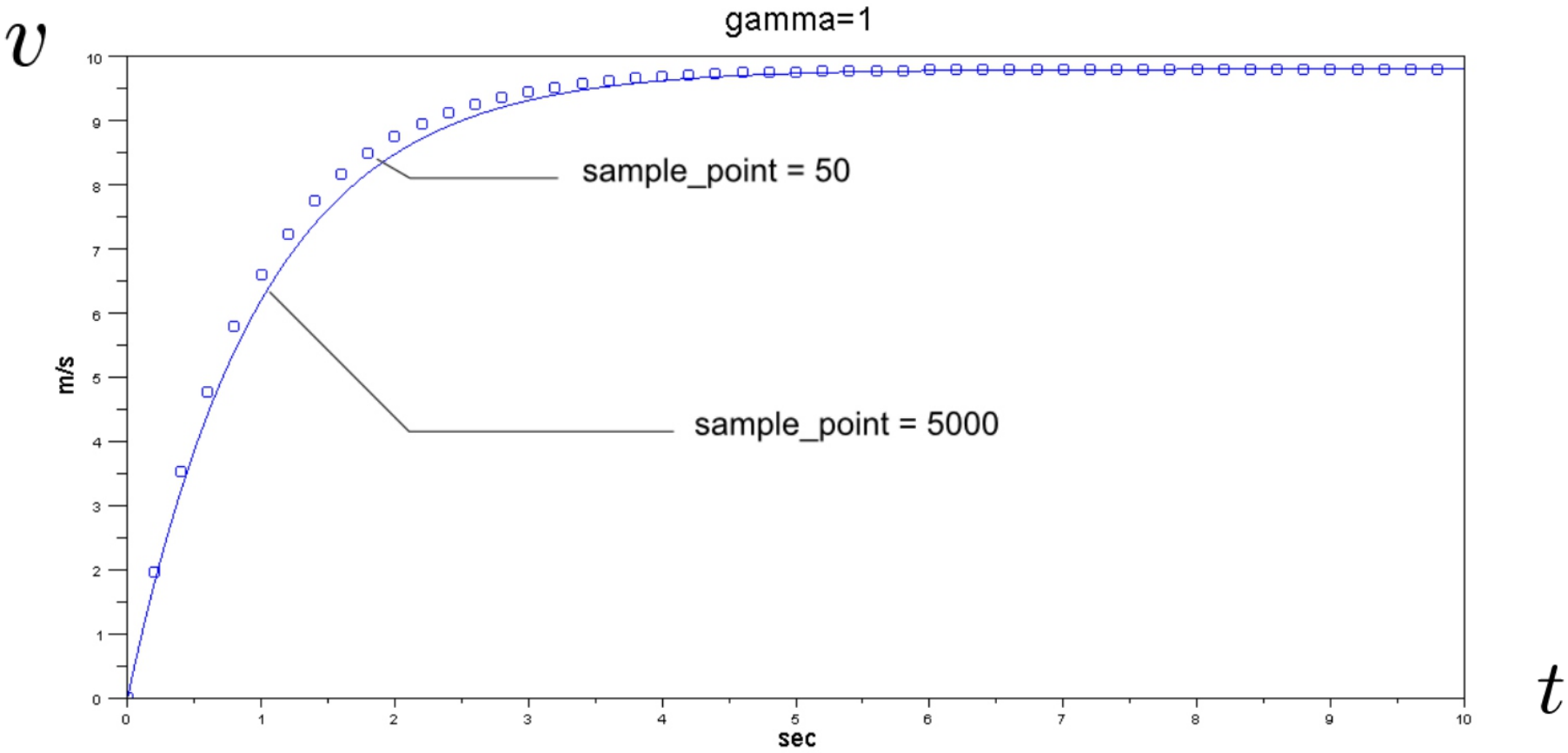
```
//plotting
t=[1:sample_point];
t_scale=(t-1)/sample_point*simu_times*10^(order);

plot(t_scale,vlctyarray_analytic,'bo-');
```

由給定參數製造對應的時間座標

由給定參數製造對應的時間座標

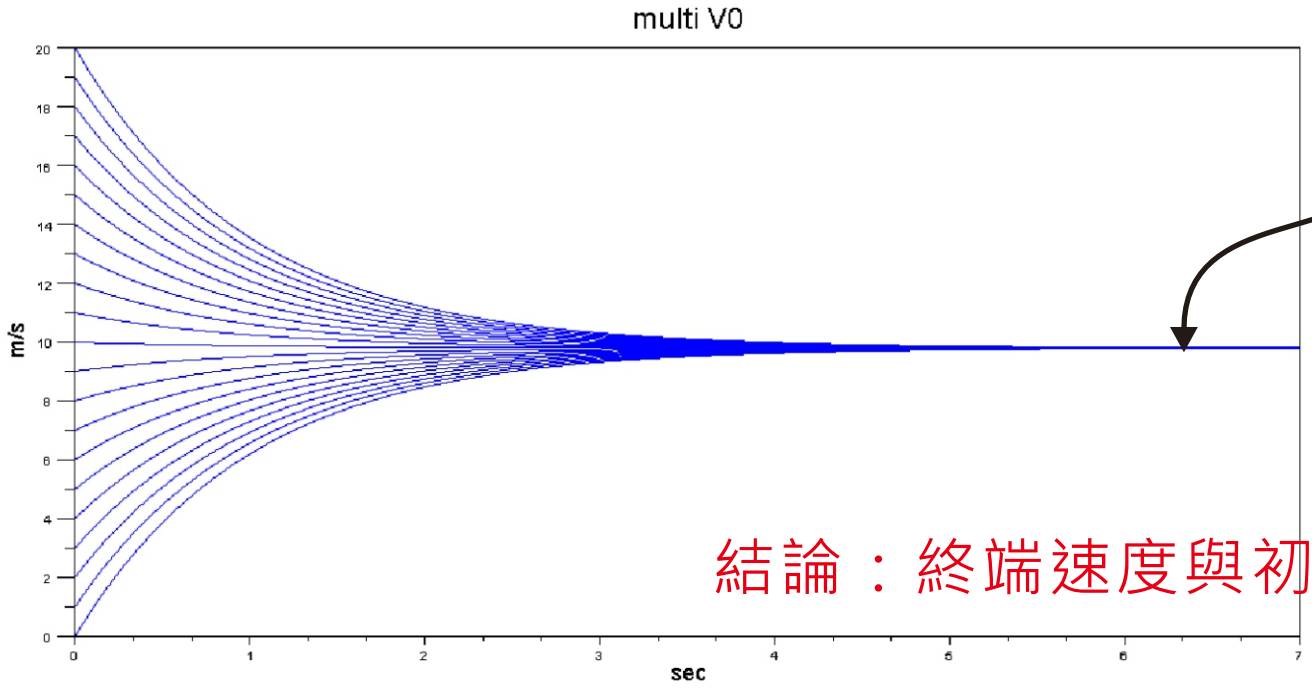
取 gamma=1, sample_point=50, 5000 繪出的結果，dt 越小，結果越準確，此方程會抵達穩態，所以兩個在穩態區的計算結果差異不大。

（若發散，計算誤差？）



```
for v0=0:20,
vlctyarray(1)=v0;
   for i=1:sample_point-1,
   vlctyarray(i+1)=dynamic(vlctyarray(i));
   end;
plot(t_scale,vlctyarray);
end;
```

改變不同起始速度



終端速度

結論：終端速度與初速無關

如何分割大型程式

會使用到的 function 與固定不變的常數單獨放到一個 scinote 檔
（計算與自訂批次繪圖函數）

主要執行運算與存取計算資料的部份獨立一份 scinote 檔
（不同類型的模擬可分開，比如掃描不同參數
ex. 自由落體模擬：改變 gamma，改變初速，改變 dt 等等）

繪製資料圖（或者在 console 中使用已撰寫好的自訂繪圖函數）
分析數據等等 獨立一份 scinote 檔

實作內容

1. 解三元一次方程式，比如

$$2x_1 + 4x_2 + 4x_3 = 4, \tag{1}$$

$$5x_1 + 5x_2 + x_3 = 7, \tag{2}$$

$$5x_1 + 4x_2 + 2x_3 = 10. \tag{3}$$

   答案爲 $x_1 = 2.857,\ x_2 = -1.714,\ x_3 = 1.286$。你有很多方法來計算這個問題，比如將上式改寫成矩陣形式，利用 `inv` 來計算反矩陣直接得到結果，或者使用 Cramer's rule 或其他 Scilab 內建的函數來做計算。

2. 求解多項式 $x^3 - x^2 + x + 1 = 0$ 的所有實根。

3. 求 $x^2 e^{-x}$ 的極大值。