G N U P L O T

使用手冊

中央研究院計算中心
ASPAC 計劃
Email:aspac@beta.wsl.sinica.edu.tw
技術報告: 94002

83 年 5 月 20 日 version:2.1

ASPAC 計畫版權聲明

因一般的 PoorMan $\LaTeX$中無法使用中文當成章節名稱,

所以本文中暫以英文來表示章節名稱.

Contents

List of Figures

List of Tables

1 Introduction

GNUPLOT 是在 1986 年由 Colin Kelley 和 Thomas Williams 所發展的繪圖公用程式，可將數學函數或數值資料以平面或立體的圖形畫在不同種類終端機或繪圖輸出裝置上。目前已廣泛地被人們使用，在 Internet 網路上尚有 newsgroup — comp.graphics.gnuplot 作為大家交換心得或學習的園地。[1]

GNUPLOT 是一個命令導向的交談式繪圖程式 (command-driven interactive function plotting program)。GNUPLOT 執行使用者輸入的每一項命令，可逐步設定或修改繪圖環境。它以圖形表達數據或函數，使我們藉由圖形做更進一步的分析。

在此以 GNUPLOT 3.5 版為討論主體。而本篇文章所引用範例皆整理自 GNUPLOT demo 中範例。在本篇文章中，第二章、介紹 GNUPLOT 的基本觀念。第三章、介紹 GNUPLOT 的計算環境。第四章、設定繪圖環境的設定。第五章、介紹繪圖指令。第六章、介紹輸入及輸出部分。第七章、介紹雜項指令。第八章、介紹由 GNUPLOT 所繪出不同類別的例子。第九章、結論。我們期望讀者能由此使用手冊中瞭解 GNUPLOT，進一步繪出期望的結果。

[1]此 newsgroup 將許多問題蒐集成 FAQ (Frequently Answered Questions) 並定期刊載，此亦可經由 anonymous ftp 從 ftp://rtfm.mit.edu/pub/usenet/news.answers/graphics/gnuplot-faq 中取得。

2 Getting start

2.1 How to get it

GNUPLOT 可在許多機器上執行，若你所使用的計算機內沒有 gnuplot 此項軟體。可透過 anonymous ftp 由下述地方取得：

版 本	ftp site
Source	phi.sinica.edu.tw:/pub/math/gnuplot35/src.zip ftp.edu.tw:/pub/X/contrib/gnuplot.3.5.gz ftp.x.org:/pub/contrib/gnuplot.3.5.tar.gz ftp.dartmouth.edu:/pub/gnuplot/gnuplot.3.5.tar.Z
MS-Windows binaries	phi.sinica.edu.tw:/pub/math/gnuplot35/win.zip ftp.edu.tw:/PC/simtel/plot/gpt35*.zip oak.oakland.edu:/pub/msdos/plot/gpt35*.zip
OS/2 2.x binaries	phi.sinica.edu.tw:/pub/math/gnuplot35/os2.zip ftp.edu.tw:/PC/os2/2_x/unix/gnuplt35.zip ftp-os2.nmsu.edu:/os2/2_x/unix/gnuplt35.zip
MS DOS 32bit binaries	phi.sinica.edu.tw:/pub/math/gnuplot35/dos.zip

或者透過郵寄服務 (ftp-mail server) 取得。方法為送一封信給 `ftpmail@ftp.dartmouth.edu`, 信的內容為:

```
open
cd pub/gnuplot
mode binary
get gnuplot3.5.tar.Z
quit
```

即可取得 uuencoded 版本的 gnuplot source (compressed tar file)。

2.2 Starting up GNUPLOT

以下是使用 GNUPLOT 繪出三角函數 $sin(x)$ 與數值資料圖形及 $sin(x) * cos(y)$ 函數 3D 圖形的命令。

```
  % gnuplot [2]

        G N U P L O T
        unix version 3.5
        patchlevel 3.50.1.17, 27 Aug 93
        last modified Fri Aug 27 05:21:33 GMT 1993

        Copyright(C) 1986 - 1993   Thomas Williams, Colin Kelley

        Send comments and requests for help to info-gnuplot@dartmouth.edu
        Send bugs, suggestions and mods to bug-gnuplot@dartmouth.edu

Terminal type set to 'x11'
gnuplot> plot sin(x), 'data.demo'
gnuplot> splot sin(x)*cos(y)
```

[2]本文中以 % 表示系統提示符號 (prompt)

```
gnuplot> quit
%
```

檔案 data.demo 為一數據資料檔案，每一行描述一點座標位置。內容如下：

```
#  file : data.demo
-20.000000 -3.041676
-19.000000 -3.036427
-18.000000 -3.030596
        .
        .
        .
```

執行 GNUPLOT 程式時，GNUPLOT 首先檢查是否設定環境參數 DISPLAY，若有則依其設定。當其確定為 X 環境時，將輸出模式設定為 X11。輸入

```
plot sin(x), 'data.demo'
```

產生圖1 結果 — 以線段繪出三角函數 $sin(x)$；以點繪出檔案 data.demo 內各點座標。接著輸入

```
splot sin(x)*cos(y)
```

產生圖2 結果 — 以縱橫各 10 條線組成的網格畫出 $sin(x) * cos(y)$ 的圖形。而 quit 指令結束 gnuplot 程式。

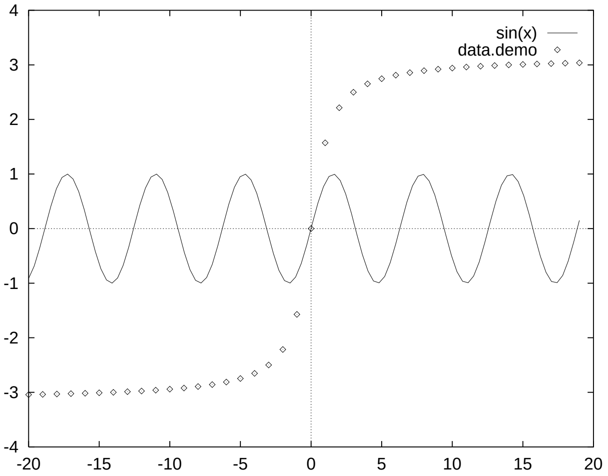


Figure 1: Plotting sin(x), data file – data.demo

由此例中，可知 GNUPLOT 具有下列特色 ：

- 不需複雜的設定即可畫出平面或立體的數學函數或數值資料圖形。
- 自動調整或可由使用者調整 X 軸、Y 軸 及 Z 軸的範圍和各軸的標點，顯示完整地結果。
- 將函數名稱自動標示在右上角。
- 以不同的線條顏色或不同的圖樣明顯地區別不同的函數圖形。

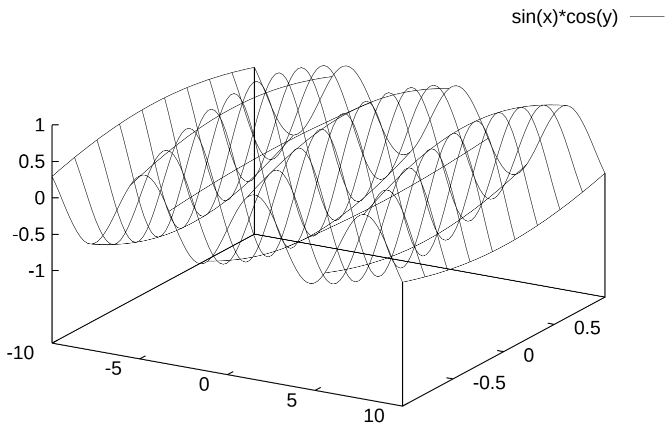


Figure 2: Plotting sin(x)*cos(y)

- GNUPLOT 有許多內建函數，亦可自建函數。

在 X 環境下，GNUPLOT 如同其他 X client 程式一般，可接受許多參數。如

```
gnuplot -font 8x13bold       # 設定字形為 8x13bold。
gnuplot -geometry 900x700    # 設定視窗的長為 900 pixel 寬為 700 pixel。
gnuplot -bg black            # 設定背景顏色為黑色。
```

等。[3]

2.3 Programming style

我們依據一般繪製數學函數或數值資料可能發生的步驟。提出一份 GNUPLOT 程式撰寫程序，如圖3。

1. 定義常數及函數：定義常數及函數，使程式模組化及易於瞭解。
2. 設定繪圖環境：GNUPLOT 繪圖之前已預先設定許多繪圖參數。[4] 這些設定做為 GNUPLOT 繪圖時的參考，包括座標軸、文字說明、圖形顯示範圍、3D 圖形投影到平面的投影角度、取樣點數、繪製圖形所用的圖案等等的設定。我們可根據個人所需逐一改變繪圖參數。GNUPLOT 的參數設定可見附錄 A，或在 GNUPLOT 程式內，鍵入 show all 命令得知目前的繪圖環境設定情況。

```
gnuplot> show all
                .
                .
    autoscaling is  x: ON, y: ON, z: ON
    border is drawn
```

[3]其餘參數可查gnuplot 之 manual page。
[4]見附錄 A

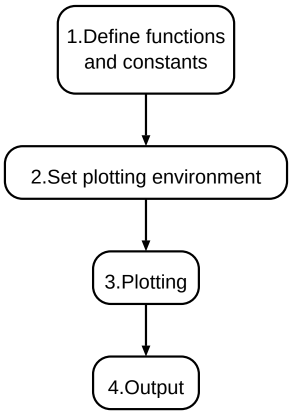


Figure 3: Programming flow chart

```
boxwidth is auto
point clip is OFF
drawing and clipping lines between inrange and outrange points
          .
          .
          .
```

3. 繪圖：在定義數學函數或設定繪圖環境之後，接著是繪出數學函數或數值資料圖形。GNUPLOT 提供操作方便的繪圖指令 — plot(2D) 或 splot(3D)，具有豐富的參數設定，可由參數的設定做個別調整。

4. 產生輸出結果：在終端機上繪出結果後，可經由一些步驟而產生不同輸出裝置所需的輸出語言，得到輸出結果。

依照上述過程，可快速地畫出圖形。剩下的步驟就是細心的調整繪圖環境參數或修改函數方程式、常數等，即可得到滿意的結果。

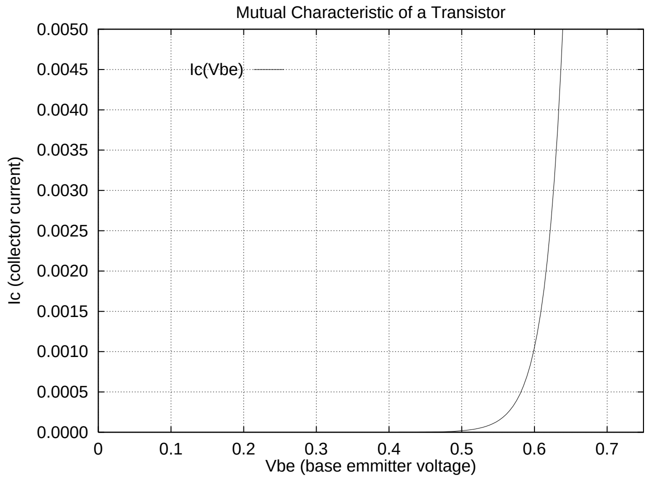
在圖4 中第1行由 # 符號開始為註解，第2至第7行定義方程式及設定參數，第8行之後是設定繪圖的狀態，第18行將圖畫出，其餘是將圖以 PostScript [5] 格式存於 'npn.ps' 檔案中。其中設定繪圖參數為：[6]

- set dummy Vbe : gnuplot 一般以 x 為變數，代表橫軸上的變數。GNUPLOT 可設定成其它名稱以增加可讀性。此例設定為 Vbe (表示 base emmitter 之間的電位差)。
- set grid : 將座標平面繪上格子。
- set title : 改變 title 名稱，將 title 改成 "Mutual Characteristic of a Transistor"。
- set xlable, set ylable : 設定橫軸與縱軸名稱，將橫軸名稱定為 "Vbe(base emitter voltage)"，將縱軸名稱定為 "Ic(collector current)"。
- set xrange, set yrange : 設定橫軸與縱軸顯示範圍，此調整顯示範圍至最想觀察的區域。

[5] PostScript is a device-independent page description language. Page description languages are programming lagnuage that are optimize to render document images on display device. Display devices may be workstation screen, laser printers, film recorders, or fax machine.

[6] 較詳細的解釋，見第 4 章

- set key :設定所繪的函數名稱所放置的座標位置。



# Bipolar Transistor (NPN) Mutual Characteristic	# comment
Ie(Vbe)=Ies*exp(Vbe/kT_q) Ic(Vbe)=alpha*Ie(Vbe)+Ico alpha = 0.99 Ies = 4e-14 Ico = 1e-09 kT_q = 0.025	defined functions and constants
set dummy Vbe set grid set samples 160 set title "Mutual Characteristic of a Transistor" set xlabel "Vbe (base emmitter voltage)" set xrange [0 : 0.75] set ylabel "Ic (collector current)" set yrange [0 : 0.005] set key .2,.0045 set format y "%.4f"	set plotting environment
plot Ic(Vbe)	plotting
set terminal postscript set output "npn.ps" replot	output

Figure 4: Example : NPN Mutual Characteristic

GNUPLOT 指令可分成五部份：

1. 使用者自定函數、常數：在繪圖前定義函數或常數，產生繪圖所需的函數。
2. 設定繪圖環境：GNUPLOT 的繪圖環境由一組參數構成，由設定參數可改變繪圖環境。
3. 設定輸出環境：GNUPLOT 提供許多種輸出裝置的輸出語言，可由命令設定輸出裝置。
4. 繪圖：GNUPLOT 提供操作方便的繪圖指令 — plot(splot)。
5. 雜項：不屬於前四類範圍的指令，如系統相關指令。

接下來的幾章就針對 GNUPLOT 指令加以描述。

3 Constants, Operators and Functions

我們在此討論 GNUPLOT 中數字、常數、運算子及函數。

3.1 Number

GNUPLOT 表示數字可分成整數、實數及複數三類：

- 整數 ： GNUPLOT 與 C 語言相同，採用 4 byte 儲存整數。故能表示 -2147483647 至 +2147483647 之間的整數。
- 實數 ： 能表示約 6 或 7 位的有效位數，指數部份為不大於 308 的數字。亦即

```
正數 ： 約 4.19e-307 至 1.67e+307
  零 ： 0
負數 ： 約 -1.67e+308 至 -4.19e-307
```

- 複數 ： 以 {<real>,<imag>} 表示 — 複數。其中 <real> 為此複數的實數部份，<imag> 為虛數部份，此兩部份均以實數型態表示。如 3 + 2i 即以 {3,2} 表示。

GNUPLOT 儲存數字的原則為，若能以整數方式儲存則以整數儲存數字，不然以實數方式儲存，其次以複數方式儲存。例如在 GNUPLOT 執行

```
print 1/3*3
print 1./3*3
```

[7] 分別得到 0 和 1.0 的結果。這是因前者使用整數計算，而後者採用實數計算的結果。執行

```
print 1234.567
print 12345 + 0.56789
print 1.23e300 * 2e6
print 1.23e300 * 2e8
```

分別得到 1234.57、12345.6、2.46e+304 和 undefined value 的結果。這些例子是受到實數的有效位數和所能表現最大數字的限制。這是我們要注意的。

3.2 Operators

GNUPLOT 所提供如 C 語言般的運算子 (operator)。[8] 所有的運算子均可做用在整數、實數或複數上。運算子可分作 Binary Operator 和 Unary Operator，就是分別作用在兩個或一個數學式子上，數學式子可為數字或方程式。

[7] print 將其後的數學式子印出其數值。
[8] 見表2、表1，此兩表引自 GNUPLOT manual。

Unary Operators		
Symbol	Example	Explanation
`-`	`-a`	unary minus
`~`	`~a`	* one's complement
`!`	`!a`	* logical negation
`!`	`a!`	* factorial

Table 1: Unary Operators

Binary Operators		
Symbol	Example	Explanation
`**`	`a**b`	exponentiation
`*`	`a*b`	multiplication
`/`	`a/b`	division
`%`	`a%b`	* modulo
`+`	`a+b`	addition
`-`	`a-b`	subtraction
`==`	`a==b`	equality
`!=`	`a!=b`	inequality
`&`	`a&b`	* bitwise AND
`^`	`a^b`	* bitwise exclusive OR
`|`	`a|b`	* bitwise inclusive OR
`&&`	`a&&b`	* logical AND
`||`	`a||b`	* logical OR
`?:`	`a?b:c`	* ternary operation

Table 2: Binary Operators

3.3 Functions

在 GNUPLOT 中函數的參數可以是整數，實數或是複數。表3 [9] 是 GNUPLOT 所提供的函數。GNUPLOT 畫條件函數的方法可由圖9 中觀察，其採用類似 C 語言的語法，畫出函數

$$F(x) = \begin{cases} -sin(x) & \text{if } sin(x) > sin(x+1), \\ -sin(x+1) & \text{if } sin(x) \leq sin(x+1) \end{cases}$$

[9]表3 引自 GNUPLOT manual。

Function	Arguments	Returns
abs(x)	any	absolute value of **x**, $\|x\|$; same type
abs(x)	complex	length of **x**, $\sqrt{\mathrm{real}(x)^2 + \mathrm{imag}(x)^2}$
acos(x)	any	$\cos^{-1} x$ (inverse cosine) in radians
arg(x)	complex	the phase of x in radians
asin(x)	any	$\sin^{-1} x$ (inverse sin) in radians
atan(x)	any	$\tan^{-1} x$ (inverse tangent) in radians
besj0(x)	radians	j_0 Bessel function of x
besj1(x)	radians	j_1 Bessel function of x
besy0(x)	radians	y_0 Bessel function of x
besy1(x)	radians	y_1 Bessel function of x
ceil(x)	any	$\lceil x \rceil$, smallest integer not less than x (real part)
cos(x)	radians	$\cos x$, cosine of x
cosh(x)	radians	$\cosh x$, hyperbolic cosine of x
erf(x)	any	Erf(real(x)), error function of real(x)
erfc(x)	any	Erfc(real(x)), 1.0 - error function of real(x)
exp(x)	any	e^x, exponential function of x
floor(x)	any	$\lfloor x \rfloor$, largest integer not greater than x (real part)
gamma(x)	any	Gamma(real(x)), gamma function of real(x)
ibeta(p,q,x)	any	Ibeta(real(p, q, x)), ibeta function of real(p,q,x)
inverf(x)	any	inverse error function of real(x)
igamma(a,x)	any	Igamma(real(a, x)), igamma function of real(a,x)
imag(x)	complex	imaginary part of x as a real number
invnorm(x)	any	inverse normal distribution function of real(x)
int(x)	real	integer part of x, truncated toward zero
lgamma(x)	any	Lgamma(real(x)), lgamma function of real(x)
log(x)	any	$\log_e x$, natural logarithm (base e) of x
log10(x)	any	$\log_{10} x$, logarithm (base 10) of x
norm(x)	any	normal distribution (Gaussian) function of real(x)
rand(x)	any	Rand(real(x)), pseudo random number generator
real(x)	any	real part of x
sgn(x)	any	1 if $x > 0$, -1 if $x < 0$, 0 if $x = 0$. imag(x) ignored
sin(x)	radians	$\sin x$, sine of x
sinh(x)	radians	$\sinh x$, hyperbolic sine x
sqrt(x)	any	$\sqrt{x}$, square root of x
tan(x)	radians	$\tan x$, tangent of x
tanh(x)	radians	$\tanh x$, hyperbolic tangent of x

Table 3: GNUPLOT functions

3.4 User-defined functions and Constants

在 GNUPLOT 中，使用者可自定函數。函數可有 1 至 5 個變數。其定義函數的語法如下：

```
<function-name> ( <dummy1> {,<dummy2> {, ...}}) = <expression>
```

而使用者定義常數的語法如下：

```
<variable-name> = <constant-expression>
```

下面舉一些例子：

```
# 常數 w 為 2。
w = 2
# 常數 q 為小於但最接近 tan(pi/2 - 0.1) 的整數。
q = floor(tan(pi/2 - 0.1))
# 函數 f(x) 為 sin(w*x), 其中 w 為常數。
f(x) = sin(w*x)
# 函數 sinc(x) 為 sin(pi*x)/(pi*x)。
sinc(x) = sin(pi*x)/(pi*x)
# 函數 delta(t) 為脈衝函數。
delta(t) = (t == 0)
# 函數 ramp(t) 當其小於零為零，當其大於零為斜率等於 1 的直線。
ramp(t) = (t > 0) ? t : 0
# 函數 min(a,b) 取兩者中較小的數。
min(a,b) = (a < b) ? a : b
comb(n,k) = n!/(k!*(n-k)!)
len3d(x,y,z) = sqrt(x*x+y*y+z*z)
```

GNUPLOT 已定義的常數僅有 **pi**。[10]

[10] pi = 3.14159

4 Plotting environment

如第二章所述，只要鍵入 `plot sin(x), 'data.demo'` 即可得到圖1 的結果。GNUPLOT 自動調整 X 軸、Y 軸的顯示範圍，使圖形顯示在適當的位置；選擇不同的顏色、圖形，用以區別不同的函數或數值資料 — 也就是 GNUPLOT 自動調整其所需的繪圖環境。若我們需要一些特別的，如在 3D 中加入 contour、設定消去隱藏線、改變 X 軸、Y 軸的座標點名稱 ... 等，可由改變繪圖參數而改變之。[11] 本章說明這些繪圖參數設定的方法與功能。

GNUPLOT 的繪圖參數可分作下列幾項：

名稱	說明
座標軸(axis)	改變座標軸的外觀，如 linear 或 logscale 等。
標題(label)	設定文字說明或線條，產生時間等。
座標系統	改變座標系統。
圖形處理	處理欲繪出的圖形，使其易於觀察。
3D	調整 3D 圖形的參數。
輸出	設定輸出的選。
雜項	不屬於上述幾項的設定。

我們可用 set 命令去設定這些參數及用 show 命令去察看這些參數。show all 則是察看所有的參數設定。本章各節依序說明繪圖參數的設定：

4.1 Axis

繪圖參數在設定座標軸方面的參數可分為變數名稱、標點、網格、顯示範圍、座標軸顯示方式與顯示與否等六方面的設定：

功能	繪圖參數名稱
變數名稱設定	dummy
標點設定	format, tics, xdtics, xmtics, xtics, ydtics, ymtics, ytics, zdtics, zmtics, ztics, ticlevel
網格設定	grid
座標顯示方式	logscale
顯示範圍設定	autoscale, rrange, trange, urange, vrange, xrange, yrange, zrange
座標軸顯示與否	xzeroaxis, yzeroaxis

1. 變數名稱設定 ：一般以 x 為橫軸上的變數。可用 dummy 設定為其它的名稱，所繪函數的變數名稱亦隨之改變。如 `set dummy t` 將自變數改為 t，圖8、圖17、圖20 均改變自變數名稱。

2. 標點設定 ：設定標點的顯示方式與格式。由 format 此項參數設定顯示格式，其語法為 ：

```
    set format {<axes>} {"<format-string>"}
    show format
# 顯示各軸標點顯示的型式
```

[11] GNUPLOT 繪圖參數之初始值設定可參考附錄 A

其中 axis 為 x、y、z、xy 或預設為xy。format-string 為描述每一標點的格式，可接受如 C 語言中 printf 對數字的 f、e、g 三種格式化描述，亦可加入文字 (必須少於 100 字)。以下舉一些例子：

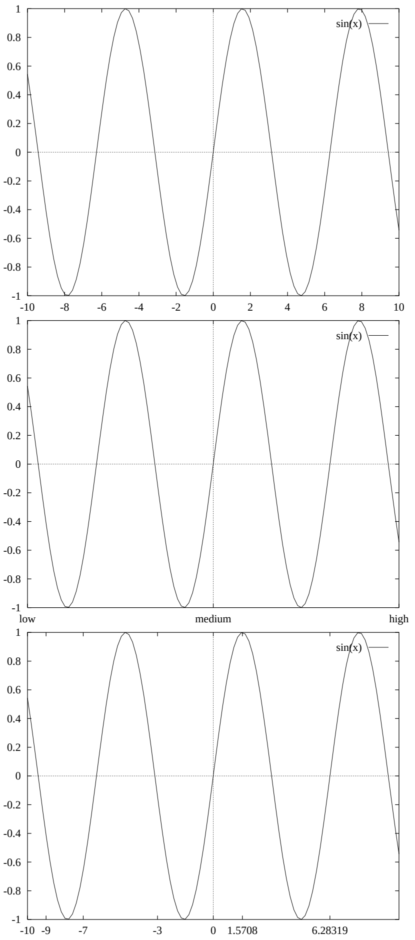
```
set format xy "%.2e"
set format x  "%3.0f cm"
```

顯示方式由 tics、xtics、. . . 等設定。

xtics 是對 X 座標軸上的標點做設定。如起始點、結束點、間隔或在軸上特定點放特定的名稱。其語法為:

```
set xtics { {<start>, <incr>{, <end>}} |
            {({"<label>"} <pos> {, {"<label>"} <pos>}...)} }
set noxtics   # 不標示任何 X 軸上的標點。
show xtics    # 顯示 X 軸標點的狀況。
```

圖5 是三個改變標點的例子。



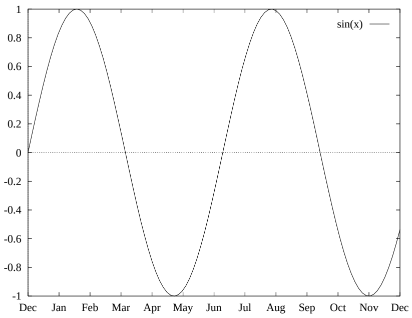

```
# 每隔 2 格一個標點
set xtics -10,2,10
plot sin(x)
```

```
# 以文字作為標點
set xtics ("low" -10, "medium" 0, "high" 10)
plot sin(x)
```

```
# 在特定位置放上標點
set xtics (-10,-9,-7,-3,0,pi/2,2*pi)
plot sin(x)
```

Figure 5: Demonstration of set xtics

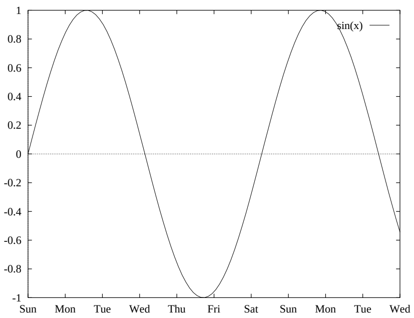
xmtics 將 X 座標軸上標點名稱依 0、1、. . . 改為 Jan、Feb、. . . Dec 等。大於 12 的數目除以 12 取其餘數。圖6 即是一例。




```
# 將標點名稱改為 Jan, Feb, ... Dec 等
set xmtics
plot [0 : 12] sin(x)
```

Figure 6: Demonstration of set xmtics

xdtics 將 X 座標軸上標點名稱依 0, 1, . . . 改為 Sun, Mon, . . . Sat 等。大於 7 的數目除以 7 取其餘數。圖7 即是一例。




```
# 將標點名稱改為 Sun, Mon, ... Sat 等
set xdtics
plot [0 : 10] sin(x)
```

Figure 7: Demonstration of set xdtics

ytics, ymtics, ydtics, ztics, zmtics, zdtics 與 xtics, xmtics, xdtics 相似，不同點是作用在不同的軸上。

ticslevel 是在畫 3D 圖形時，將標點移到距離 Z 軸上方一段距離的地方，此時圖形亦隨移動。語法為:

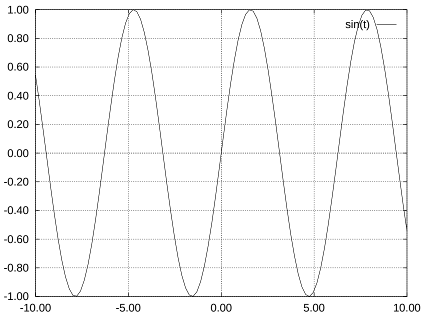
```
set ticslevel {<level>}
show tics
```

其中 `level` 為正數。

3. 網格設定：在 XY 座標平面上依刻度畫上方格子。如圖8。
4. 座標顯示方式：分為線性與對數兩種。一般為前者，若要改為對數方式，其語法為：

```
set logscale <axes> <base>
set nologscale <axes>
show logscale
```

其中 axes 為 X 軸、Y 軸、Z 軸的任意組合。base 預設為 10。




```
# 設定變數為 t
set dummy t
# 設定 X 軸 Y 軸標點的格式
set format xy "%3.2f"
# 產生網格
set grid
plot  sin(x)
```

Figure 8: Demonstration of set xdtics

5. 顯示範圍設定：改變各軸的顯示範圍。autoscale 參數設定後 GNUPLOT 自動調整顯示範圍。其餘的如 rrange, trange, xrange, yrange, zrange 則是由使用者設定該軸的範圍，以 xrange 為例，其語法為:

    ```
    set xrange [{<xmin> : <xmax>}]
    ```

    其中參數 <xmin> 與 <xmax> 代表 X 軸的起點與終點，可以是數字或數學式子。如圖 7 中 set [0:10] sin(x) 設定 X 軸顯示範圍為 0 與 10 之間。此時可用

    ```
    set xrange [0:10]
    plot sin(x)
    ```

    代替之。

    使用 autoscale 參數調整顯示範圍，其語法為:

    ```
    set autoscale <axes>
    set noautoscal <axes>
    show autoscale
    ```

    其中 <axes> 為 GNUPLOT 欲調整的軸，可以是 x, y, z 或 xy，預設為所有的軸。

6. 座標軸顯示與否設定：設定是否要畫出座標軸，以 X 軸為例：

    ```
    set xzeroaxis      # 設定顯示 X 座標軸
    set noxzeroaxis    # 設定不顯示 X 座標軸
    show xzeroaxis     # 檢查 X 座標軸顯示與否
    ```

4.2 Label

GNUPLOT 除了繪出圖形外，尚可加入註解做為輔助說明。這註解包括文字與線條兩方面，其提供的設定有:

功能	繪圖參數名稱
線條	arrow
文字註解	key, label, time, title, xlabel, ylabel, zlabel

1. 線條 : 在圖上畫一線段 可以選擇有無箭頭。其語法為:

```
set arrow {<tag>} {from <sx>,<sy>{,<sz>}}
          {to <ex>,<ey>{,<ez>}} {{no}head}
set noarrow {<tag>}     # 刪除一線條
show arrow              # 顯示線條使用情況
```

其中參數 <tag> 是給該條線條一個整數名稱，若不設定則為最小可用整數。此線條由座標 (sx, sy, sz) 到 (ex, ey, ez)(在 2D 中為(sx, sy)到(ex, ey))。參數 nohead 為畫沒有箭頭的線段，參數 head 或沒有 nohead 為畫有箭頭的線段。圖24 中使用沒有箭頭的線段作為輔助說明。以下為一些例子:
[12]

```
# 畫一帶有箭頭的線條 由原點到 (1,2)。
set arrow to 1,2
# 畫一名為 3 的帶箭頭線條 由 (-10,4,2) 到 (-5,5,3)。
set arrow 3 from -10,4,2 to -5,5,3
# 改變名為 3 的線條起始點至 (1,1,1)。
set arrow 3 from 1,1,1
# 刪除名為 2 的線條。
set noarrow 2
# 刪除所有線條。
set noarrow
# 顯示線條的使用情形。
show arrow
```

2. 文字註解 : 分為設定標頭 (title), 標示 (label) 與時間 (time) 三部份。標頭設定為在圖的正上方加上說明本圖的文字。其語法為:

```
set title {"<title-text>"} {<xoff>}{,<yoff>}
show title
```

設定參數 <xoff> 或 <yoff> 為微調標頭放置的位址。xlabel, ylabel, zlabel 的語法與 title 相同，其各自描述一座標軸。可見圖10 的描述。
標示 (label) 為在圖上任一位置加上文字說明，一般與線條一併使用。其語法為:

```
set label {<tag>} {"<label_text>"} {at <x>,<y>{,<z>}}
          {<justification>}
set nolabel {<tag>}       # 刪除一標示
show label                # 顯示標示使用情況
```

其中參數 <tag> 與 "線條" (arrow) 中 <tag> 意義相同，用以區別不同的 label。參數 <justification> 是調整文字放置的位置，可以是 **left**, **right** 或 **center**。圖24 中使用 label 作為文字輔助說明。舉一些例子: [13]

```
# 將 y=x 放在座標 (1,2) 之處。
set label "y=x" at 1,2
```

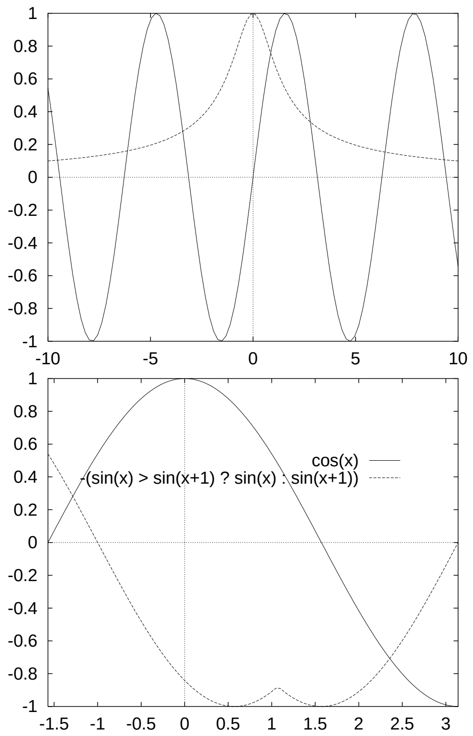
[12]例子由 GNUPLOT manual 中引用
[13]例子由 GNUPLOT manual 中引用

```
# 將 y=x^2 放在座標 (2,3,4) 之處，並命名為 3。
set label 3 "y=x^2" at 2,3,4 right
# 將名為 3 的標示居中放置。
set label 3 center
# 刪除名為 2 的標示。
set nolabel 2
# 刪除所有標示。
set nolabel
# 顯示標示使用情形。
show label
```

一般繪一圖形後，GNUPLOT 將函數名稱或圖形檔案名稱置於右上角。key 參數設定可改變名稱放置位置。其語法為:

```
set key
set key <x>,<y>{,<z>}
set nokey
show key
```

其中參數 <x>, <y>, <z> 設定名稱放置位置 。set nokey 為不顯示名稱，若使用 set key 則再度顯示名稱。若使用 set key 0.2, 0.5 則顯示函數名稱於座標 (0.2, 0.5) 之處。圖 9 表達上述說明。




```
set nokey
plot sin(x), cos(atan(x))
```

```
set key 2, 0.5
plot [-pi/2:pi] cos(x),\
      -( sin(x) > sin(x+1) ? sin(x) : sin(x+1))
```

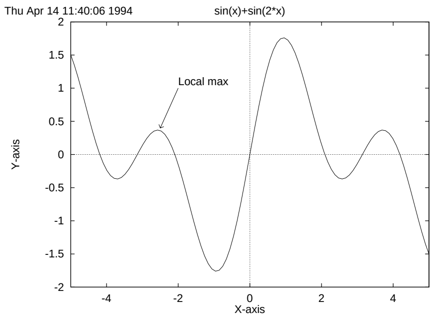
Figure 9: Put key on different place

時間參數設定是將圖產生的時間標在圖上。其語法為

```
set time {<xoff>}{,<yoff>}
```

```
set notime
show time
```

設定參數 <xoff> 或 <yoff> 為微調時間放置的位址，正數表示向上或向右，負數為反方向，以字的長寬作為單位。圖10 為一例子。




```
set title "sin(x)+sin(2*x)"
set xlabel "X-axis"
set ylabel "Y-axis"
set arrow from  -2,1 to -2.5,0.4
set label "Local max" at -2,1.1
set nokey
set time
plot [-5:5] sin(x)+sin(2*x)
```

Figure 10: Demonstration of set label

4.3 Coordinate system

GNUPLOT 除提供繪圖在 Cartesian coordinate 外，尚提供 Polar coordinate 的繪圖環境。並具有將平面圖形投影到非平面圖形的功能。

繪圖參數名稱	說明
angles	設定角度單位為徑度量 (radian) 或度度量 (degree)。
mapping	畫資料時，設定其採用的座標系統。此有三種選擇 cartesian, spherical, cylindrical。
polar	設定為極座標系統。
parametric	所繪圖形由一般函數改成參數函數描述。

1. angles：設定角度單位為徑度量或度度量。其語法為:

```
set angles degrees  # 採用徑度量
set angles radians  # 採用度度量
show angles         # 顯示目前使用狀況
```

2. mapping : GNUPLOT 讀入數據而繪出平面圖形，可藉由 set mapping spherical 或 set mapping cylindrical 而將圖案投影至球面或圓柱上，使圖形在不同曲面中展現出不同特色。其投影方式是將資料數據經由一組函數轉換後再描繪於 3D 空間上。此函數為

```
x = cos( theta ) * cos( phi )
y = sin( theta ) * cos( phi )
z = sin( phi )
```

此時每一筆數據為 (theta,phi)。使用 set angle degrees 改成度度量，設定 3D 顯示範圍在 [-pi/2:pi/2] 和 [0:2*pi] 之間。第三部份是將地圖繪於圓柱上。其轉換函數為

```
x = cos( theta )
y = sin( theta )
z = z
```

此時每一筆數據為 (phi,z)。此投影僅對描繪在 3D 平面上的數據資料圖形有用。圖23 為一典型例子，其語法為:

```
set mapping { cartesian | spherical | cylindrical }
```

3. polar : 設定為極座標系統。此時變數為 x 代表角度，採用徑度量或度度量，由 angles 此項參數決定。圖22 為一典型例子。語法為：

```
set polar     # 採用極座標系統繪圖
set nopolar   # 不採用極座標系統繪圖 (恢復成 Cartesian coordinate)
show polar    # 顯示目前狀況
```

4. parametric : 輸入 `set parametric` 後，以參數式描述圖形。畫平面圖形時以 t 為變數名稱，以 $x = f(t)$, $y = g(t)$ 描述圖形。畫立體圖形時以 u,v 為變數名稱，以 $x = f(u,v)$, $y = g(u,v)$, $z = h(u,v)$ 描述圖形。圖21、圖30、圖31、圖32為典型例子。

4.4 Graphics process

GNUPLOT 使用下列繪圖參數改變繪出圖形的外貌:

功能	繪圖參數名稱
圖形修正設定	clip
圖樣設定	data style, function style, boxwidth
取樣點數設定	samples
圖形大小設定	size

1. 圖形修正設定 : GNUPLOT 提供 clip 作為修正圖形設定 其語法為 :

```
set clip <clip-type>
set noclip <clip-type>
show clip
```

提供三種 cliptype 分別為 **points**, **one**, **two** 其意義如下:

```
# 將太接近的點，只用一點代替。
set clip points
# 當線段的端點出現在顯示區域內時，才畫此線段，否則不畫。
set clip one
# 將出現在顯示區域內的任何線段或部份線段繪出。
set clip two
```

`set noclip <clip-type>`是取消由 `set clip <clip-type>`設定的功能。`show clip` 顯示目前狀況。

2. 圖樣設定：GNUPLOT 提供設定圖樣的參數。其語法為：

```
show data style                     # 顯示目前繪數值資料的圖樣
set data style <style-choice>       # 設定繪數值資料的圖樣
show function style                 # 顯示目前繪函數的圖樣
set function style <style-choice>   # 設定繪函數的圖樣
```

`<style-choice>` 可選擇的圖樣有九種，依序為 lines, points, linespoints, dots, steps, impulses, errorbars, boxes, boxerrorbars。圖14 為採用不同圖樣所繪出的圖形。boxwidth 參數設定以 boxes 為圖樣時，box 之寬度。其語法為:

```
set boxwidth {<width>}
show boxwidth
```

參數 `<width>` 為設定 box 的寬度。若鍵入 `set boxwidth` 時，GNUPLOT 自動調整寬度。

3. 取樣點數設定(sampling)：gnuplot 計算若干點的函數值後，以某種圖樣將函數值繪上。其語法為:

```
set samples <samples_1>,{<samples_2>}
show samples
```

`<samples_1>,<samples_2>` 是分別設定 3D 中 X Y 的取樣點數。一般是取樣 100 點及採取線條作為圖樣。可提高取樣點數而得到失真較低的圖形，圖11 可觀察出在圖形轉角處有明顯不同。

4. 圖形大小設定：GNUPLOT 提供改變圖形長與寬的比例，是可放大或縮小圖形的參數。其語法為:

```
set size {<xscale>,<yscale>}
show size
```

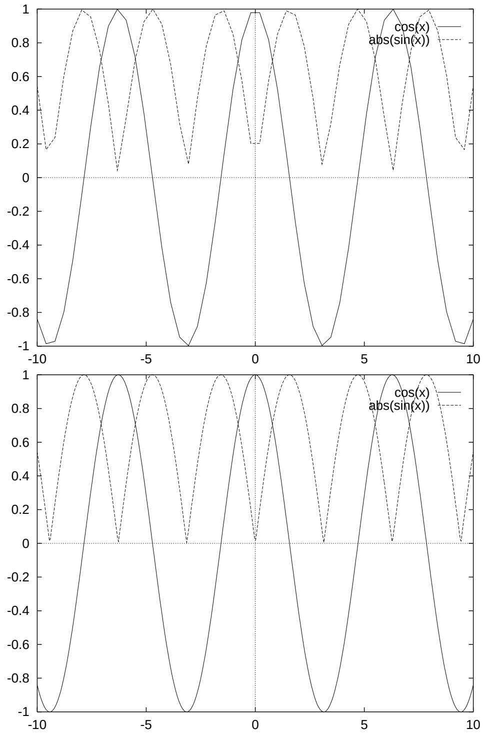
參數 `<xscale>,<yscale>` 為圖長與寬的比例，預設值為 1:1。舉一些例子：

```
# 將圖縮成四分之一大小。
set size 0.5 0.5
# 調整後在 landscape 中有 1:1 的比例。
set size 0.721,1.0
```

4.5 3D

GNUPLOT 在畫立體圖形方面，提供 contour、消去隱藏線、改變視角等功能。

功能	繪圖參數名稱
contour	clabel, cntrparam, contour, surface
消隱藏線	hidden3d
取樣線數	isosamples
改變視角	view
畫格狀圖	dgrid3d




```
set samples 50
plot abs(sin(x)), cos(x)
```

```
set samples 500
plot abs(sin(x)), cos(x)
```

Figure 11: Demonstration of set samples

1. contour : GNUPLOT 提供畫 contour 的參數。

```
set contour { base | surface | both }
set nocontour
```

其中參數 **base** 將等高線繪於 XY 平面。參數 **surface** 將等高線繪於圖形曲面上。參數 **both** 將等高線繪於 XY 平面與圖形曲面上。

畫等高線時，由 cntrparam 設定 contour 的參數。語法為:

```
set cntrparam { { linear | cubicspline | bspline } |
   points <n> |
   order <n>  |
   levels { [ auto ] <n> |
   discrete <z1>,<z2>, ... |
   incremental {<start>, <incr>{, <end>} } }
```

參數說明如下:

linear, cubicspline, bspline - 設定 contour 的線條形態。**linear** 是以直線連接。**cubicspline** 以片段的圓滑曲線連接。**bspline** 以圓滑曲線逼近等高線。
points - 在 **cubicspline, bspline** 決定採用的點數，數目愈大則愈平滑。
order - 只用在 **bspline**，此數值可選在 2 至 10 之間，數字愈大，線條愈平滑。
levels - 設定 contour 分成 n 階(level)，選擇 levels 的方法有三種：'auto'，'discrete' 和 'incremental'。其中 'auto' 是將 z 的最大值與最小值之間平均分成 n 等份。'discrete'

是由其後跟隨的數字設定在 Z=? 時畫上 levels。'incremental' 是由 user 設定起始點與每一階段所欲增加的值。

`set clabel` 將 contour 所畫每一條等高線以不同顏色表示。

有時我們僅對等高線有興趣 可用 surface 參數設定 語法為:

```
set surface     # 畫出圖形曲面與等高線。
set nosurface   # 僅畫等高線。
show surface    # 顯示狀態。
```

2. 消隱藏線 : GNUPLOT 提供消隱藏線的參數。[14]

```
set hidden3d     # 不畫出隱藏的線段。
set nohidden3d   # 畫出隱藏的線段。
show hidden3d    # 顯示狀態。
```

3. 取樣線數 : GNUPLOT 提供畫 3D 圖形時，決定以多少線條繪出曲面的參數。語法為:

```
set isosamples <iso_1> {,<iso_2>}
show isosamples
```

參數 <iso_1> 為繪出曲面的線條數。若設定 <iso_1>, <iso_2> 則分別是縱與橫方向的線條數，預設值為 10。

4. 改變視角 : GNUPLOT 提供 3D 圖形投影到 2D 平面的視角設定。語法為:

```
set view <rot_x> {,{<rot_z>}{,{<scale>}{,<scale_z>}}}
show view
```

參數 <scale>, <scale_z> 是控制圖的大小比例。參數 <rot_x>, <rot_z> 是旋轉 X軸、Z 軸的角度。預設為 `set view 60,30,1,1`。

5. 畫格狀圖 : 我們很難觀察雜亂無序的資料所繪的圖形。GNUPLOT 提供將 3D 中數值資料以格狀方式畫出的方法。語法為:

```
set dgrid3d {,{<row_size>}{,{<col_size>}{,<norm>}}}
set nodgrid3d
```

參數 <row_size>, <col_size> 設定以縱橫各幾格組成此網格。它以距離為反比地加權所有的數值，再求其平均值。圖28 為典型的例子。

4.6 Output

GNUPLOT 的輸出參數設定有 `terminal`, `output`, 在第 6 章有詳細的討論。

1. terminal : 設定 GNUPLOT 的輸出裝置型態。GNUPLOT 提供許多輸出裝置。其語法為:

[14] 目前 DOS 版本尚不提供此項設定。

```
set terminal {<terminal-type>}
show terminal
```

參數 `<terminal-type>` 可為表7、表8、表9 中所列出的裝置名稱。

2. output：GNUPLOT 將輸出送至標準輸出。此設定可將輸出送至一檔案或另一程式的標準輸入。語法為:

   ```
   set output {"<filename>"}
   show output
   ```

   舉例如下:

   ```
   # 將輸出送至檔案中
   set output "filename"
   ```

   我們可由鍵入下列命令將 $sin(x)$ 函數圖形由 PostScript 印表機印出。

   ```
   # 將輸出送至 PostScript 印表機列印
   set terminal postscript
   set output "sin.ps"
   plot sin(x)
   set output "|lpr -P postscript sin.ps"    # BSD 系統
   set output "|lp  -d postscript sin.ps"    # System-V 系統
   ```

   因此我們可由鍵入下列命令將 $sin(x)$ 函數圖形由接在 PC 上的 HP 繪圖機繪出圖形。

   ```
   # hpgl -- HP 繪圖機的描述語言
   set terminal hpgl
   # 印表機接在 lpt1
   set output "lpt1"
   plot sin(x)
   ```

4.7 Miscellaneous

在此討論不屬於前幾類的參數設定

1. functions：鍵入 `show functions` 顯示所有使用者定義的函數。
2. variables：鍵入 `show variables` 顯示所有使用者定義的常數。
3. border：鍵入 `set border` 將圖形外圍加框。鍵入 `set noborder` 除去圖形外框。鍵入 `show border` 顯示目前是否加框。
4. offsets：在圖形四週留下一些空白。其語法為:

   ```
   set offsets <left>, <right>, <top>, <bottom>
   show offsets
   ```

   其中參數 <left>, <right> 以橫軸單位長度為單位長度，參數 <top>, <bottom> 以縱軸單位長度為單位長度。
5. zero： `set zero <expression>` 設定一數值，當小於此值時 gnuplot 視為零。

5 Plotting

GNUPLOT 以函數或數據資料為繪製圖形的來源。此兩者皆使用 plot(splot) 指令繪圖，plot 用於繪製 2 維的函數或數據資料；而 splot 則用以繪製 3 維空間中的曲面。其基本語法為：

```
plot {ranges} {<function> | {"<datafile>" {using ...}}}
              {title} {style} {, <function> {title} {style}...}

splot {ranges} {<function> | {"<datafile>" {index i} {using ...}}}
               {title} {style} {, <function> {title} {style}...}
```

由修改各參數的設定，可產生許多種表達圖形的方式。其參數的設定大致分為以下四類：

1. range：設定圖形所顯示的區域。
2. function or datafile：設定所繪的函數或存有數據資料的檔案。
3. title：設定標題內容。
4. style：設定顯示圖形的顏色、圖案等。

以下分別描述之。

5.1 Ranges

設定此次 plot(splot) 所顯示的範圍。在設定繪圖環境中亦有 xrange、yrange、zrange、rrange、trange、urange、vrange 等相似功能的參數可加以設定，但其影響設定後所有的繪圖狀態。設定 range 的語法為：

```
[{<dummy-var> =} {<xmin> : <xmax>}] { [{<ymin> : <ymax>}] }
```

其中參數 <dummy-var> 為變數名稱 (預設為 X Y，可由 set dummy 而更改)，而 min 和 max 則是設定顯示範圍，均為常數表示式。並非 min 和 max 要同時存在，不設定表示不改變此項數值。一但設定了 range 之後，會自動將 autoscaling 關閉。以下是一些範例：

```
plot cos(x)                           # 使用目前的 range。
plot [-10:30] sin(pi*x)/(pi*x)        # 設定 X range。
plot [t = -10 :30]  sin(pi*t)/(pi*t) # dummy-variable 改成 t，設定 t range。
plot [-pi:pi] [-3:3]  tan(x), 1/x     # 設定 X 和 Y range。
# 僅設定 Y range，但關閉 X 軸 、Y 軸 的 autoscaling。
plot [ ] [-2:sin(5)*-8] sin(x)**besj0(x)
plot [:200] [-pi:]  exp(sin(x))       # 僅設定 xmax 和 ymin。
splot [0:3] [1:4] [-1:1] x*y          # 設定 X、Y、Z range。
```

5.2 Function

函數為一數學表示式。可以是 GNUPLOT 提供的函數或是自行定義或合成的函數。若使用參數式 (parametric function)，則可能是一對 (使用 plot) 或 3 個 (使用 splot) 數學表示式。要注意的是 dummy variable 必須符合當時所設定的，一般為 X、Y、Z，若為參數式則為 t (2 維) 或 u v (3 維)。以下是一些範例：

```
plot sin(x)    # 描繪數學三角函數 $sin(x)$。
splot x * y    # 描繪函數 $x*y$。
```

5.3 Datafile

```
#
#  datafile.example
#
0       11.062  9.757   12.667  2.345
2       9.212   8.0908  10.932  1.725
5       8.55    7.6612  9.0388  2.846
10      8.122   7.327   8.617   0.957
15      8.156   7.6101  8.9019  1.577
20      8.122   7.5431  8.8009  3.141
25      7.988   6.3007  9.2753  1.57
27      8.096   6.665   10.555  2.172
29      7.936   6.9395  8.325   0.582
30      7.854   6.4909  9.8171  0.603
35      8.01    7.1556  8.644   1.063
40      8.02    7.0301  9.7099  2.152
45      8.092   6.9164  9.7676  1.44
50      8.072   7.2993  8.4447  1.96
60      8.286   7.7265  8.755   1.0
```

Table 4: An example of datafile.(I)

GNUPLOT 依照檔案中的數據，逐點描出圖形。表 4 即是一組數據資料。數字之間以空白或 tab 區別。每筆數字為一欄位，每行則由幾個欄位構成，儲存組數字，描繪數據圖形的基本語法為

```
plot "datafile" { using { <ycol> |
                          <xcol>:<ycol> |
                          <xcol>:<ycol>:<ydelta> |
                          <xcol>:<ycol>:<ylow>:<yhigh> |
                          <xcol>:<ycol>:<ylow>:<yhigh>:<boxwidth> }
                        {"<scanf string>"} } ...

splot "datafile" { using { <xcol>:<ycol>:<zcol> | <zcol> }
                         {"<scanf string>"} } ...
```

由 `plot "datafile" with lines` 即能產生如圖 23 的結果。此例中 GNUPLOT 依序用線條連接檔案 "datafile" 內各點數據所定出的座標值。將第一欄位視為 X 座標值，將第

二欄位視為 Y 座標值。表 4 為一組資料檔案。參數 <xcol> <ycol> 和 <zcol> 均為在檔案中選擇適當的欄位分別做為 X、Y、Z 的座標值。如在 plot 語法中第一行僅用 <ycol> 則以檔案中的行數代替 X 座標值繪出圖形。第二行則為同時標出 X、Y 座標值。第三、四行為畫 errorbars 時使用 <ydelta>、<ylow> 及 <yhigh>。第五行為畫 boxes 或 boxerrorbars 時使用，可設定 box 的寬度；此時若以 boxes 為圖樣 (即不必使用 <ylow> 及 <yhigh>)，亦要設定 <ylow> 及 <yhigh>。圖 13 即是使用表 4 為 datafile 運用不同敘述所繪的結果。plot 語法中第六行是採用 C 語言中 scanf 讀入浮點數字的語法 %f 來表達使用數字的情形。

```
plot "datafile" using "%*f%f%*20[^\n]%f" with lines
```

是忽略第一行欄位的數字，採用第二欄位的數字為 X 座標值，再忽略 20 個非換行字元的文字，再將接者的數字做為 Y 座標值。這樣就可忽略數字之間的文字說明了。如 6.2.3 節中由設定輸出為 `table` 所產生的資料檔案 "datafile" 。可用

```
plot "datafile" using "%*4[^\n]%f%3[^\n]%f" with lines
```

得到 "datafile" 所描述的圖形。使用 `%*4[^\n]` 忽略前四個字 "i x=" `%*3[^\n]` 忽略 "y=" 三個字。

儲存檔案中的數據或許未按大小順序排列者，此時使用 "points" 為圖樣畫出數據尚無影響。但若使用 "lines" 為圖樣畫出數據，則圖形顯得雜亂。如圖 12 左半部，此時可用 unix 指令

```
sort -n data.old > data.new
```

將數據依順序，由小而大排列。則將圖12 左半部改成圖12 右半部。

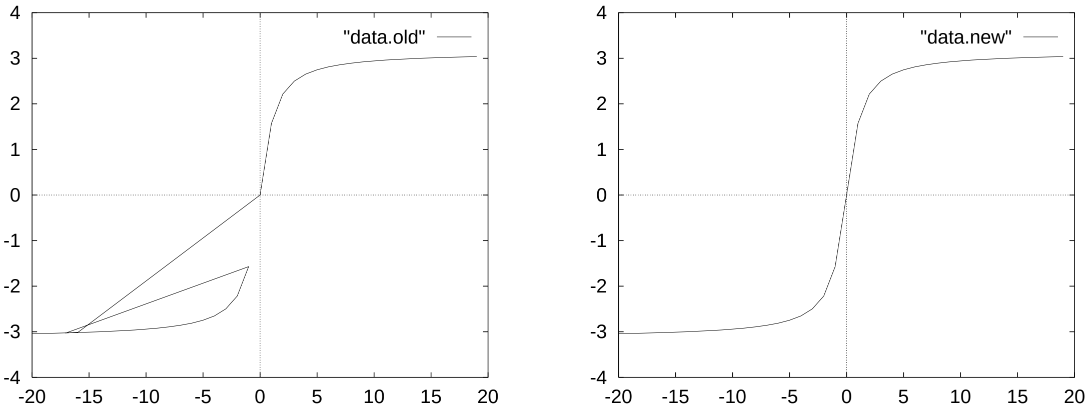
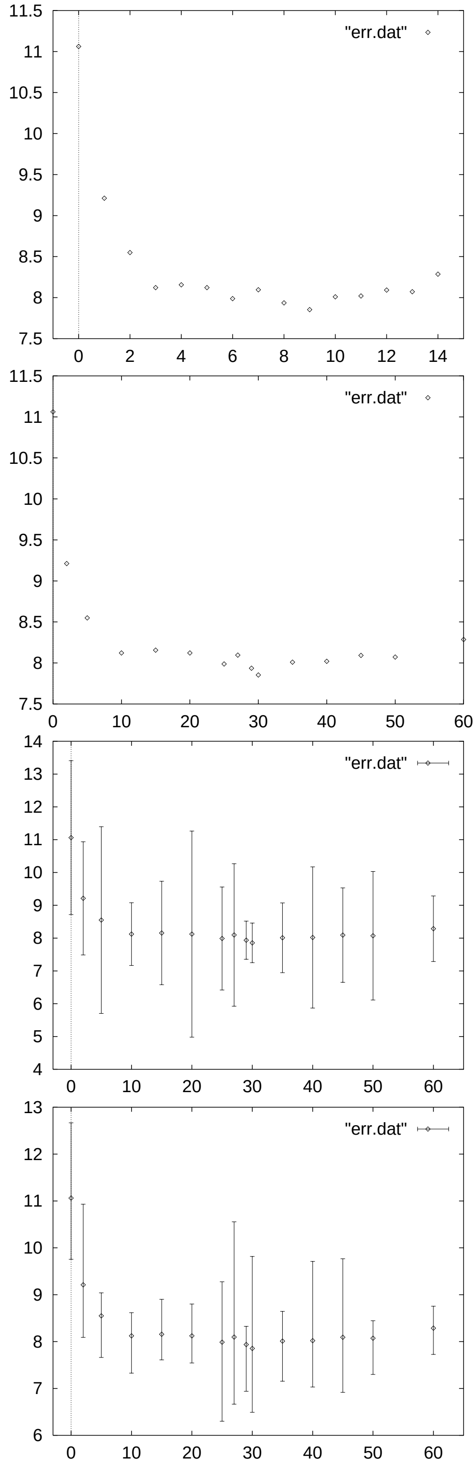


Figure 12: Plotting unsorted and sorted data

這方法適用於 2 維上的繪圖，至於 3 維上繪圖 則須設定 dgrid3d 此項參數。




```
set xrange [-1 : 15]
plot "err.dat" using 2
```

```
plot "err.dat" using 1:2
```

```
set xrange [-3 : 65]
plot "err.dat" using 1:2:5 with errorbars
```

```
set xrange [-3 : 65]
plot "err.dat" using 1:2:3:4 with errorbars
```

Figure 13: Plotting datafile with different style

在一檔案中亦可包含幾個數值資料圖形，只要在不同數值資料圖形之間以兩行空白隔開，就可僅畫其中一個數值資料圖形(僅能運用於畫 3D 圖形)。範例如下：

```
splot "datafile" index 0
```

datafile 內容如表5 ，index 0 表示畫第一個數值資料圖形。此時若要畫第三個圖形，可使用 index 2。

```
# index 0
 0   11.062   9.757   ]
 2   9.212    8.0908  ]  index 0
     ⋮                ]

# index 1
 10  8.122    7.327   ]
 15  8.156    7.6101  ]  index 1
     ⋮                ]

# index 2
 25  7.988    6.3007  ]
 27  8.096    6.665   ]  index 2
     ⋮                ]
```

Table 5: An example of datafile.(II)

5.4 Style

GNUPLOT 描繪數據資料圖形是以讀入檔案中的座標值後，以圖樣繪上。而描繪函數圖形是計算若干點的函數值後，以某種圖樣將函數值繪上。一般是取樣 100 點及採取線條作為圖樣。GNUPLOT 可提供 9 種圖樣，分別是

- lines：將相鄰的點以線條連接。如 `plot sin(x) with lines`。
- points：將每一點以一符號繪上。如 `plot sin(x) with points`。
- linespoints：同時具有 lines 及 points 的功能。如 `plot sin(x) with linespoints`。
- impulses：將每一點畫一垂直線至 X 軸。如 `plot sin(x) with impulses`。
- dots：將每一點繪一細點。如 `plot sin(x) with dots`。
- steps：以垂直線及水平線各一條來連接兩點，形成梯形。如連接 (x1,y1)，(x2,y2)兩點，以 (x1,y1) 到 (x2,y1) 和 (x2,y1) 到 (x2,y2) 兩線段連接。如 `plot sin(x) with steps`。
- errorbars：對每一點座標值 (x,y)，畫一由 (x,ylow) 至 (x,yhigh) 的線段。並在線段兩端做上 tic mark。如 `plot sin(x) with errorbars`。
- boxes：The boxes style draws a box centred about the given x coordinate from the yaxis to the given y coordinate.如 `plot sin(x) with boxes`。
- boxerrorbars：組合 errorbars 與 boxes 兩者功能。如 `plot sin(x) with boxerrorbars`。

若改變圖樣，則可得到不同形式的輸出。圖14 以九種圖樣為範例分別繪出數值資料圖形。其中 **errorbars**, **steps**, **boxes** 和 **boxerrorbars** 僅能在平面 (2D) 上使用。在設定繪圖環境之中，亦可由 set function style 和 set data style 分別設定描繪函數與數據資料的圖形，得到相同的結果。

設定圖樣的語法為

```
with <style> {<linetyle> {<pointtype>}}
```

其中參數 <style> 為 **lines**, **points**, **linespoints**, **impulses**, **dots**, **steps**, **errorbars** 之中一種，參數 <linetype> 和 <pointtype> 均為一正數，分別選定繪圖顏色及畫 points 的符號。GNUPLOT 提供 <linetype> 和 <pointtype> 各六種選擇，供繪圖循環使用。以下舉出一些使用範例：

- 使用 impulses 畫 $sin(x)$ 函數圖形。

```
plot sin(x) with impulses
```

- 使用 points 畫 $x * y$ 函數圖形，使用預設圖樣畫 $x^2 + y^2$ 函數圖形。

```
splot x*y with points, x**2 + y**2
```

- 使用相同的線條顏色畫 $x^2 + y^2$ 及 $x^2 - y^2$ 函數圖形。

```
splot x**2 + y**2 with lines 1, x**2 - y**2 with lines 1
```

- 使用相同的線條顏色但不同的符號畫 $sin(x)$, $cos(x)$。

```
plot sin(x) with linespoints 1 3, cos(x) with linespoints 1 4
```

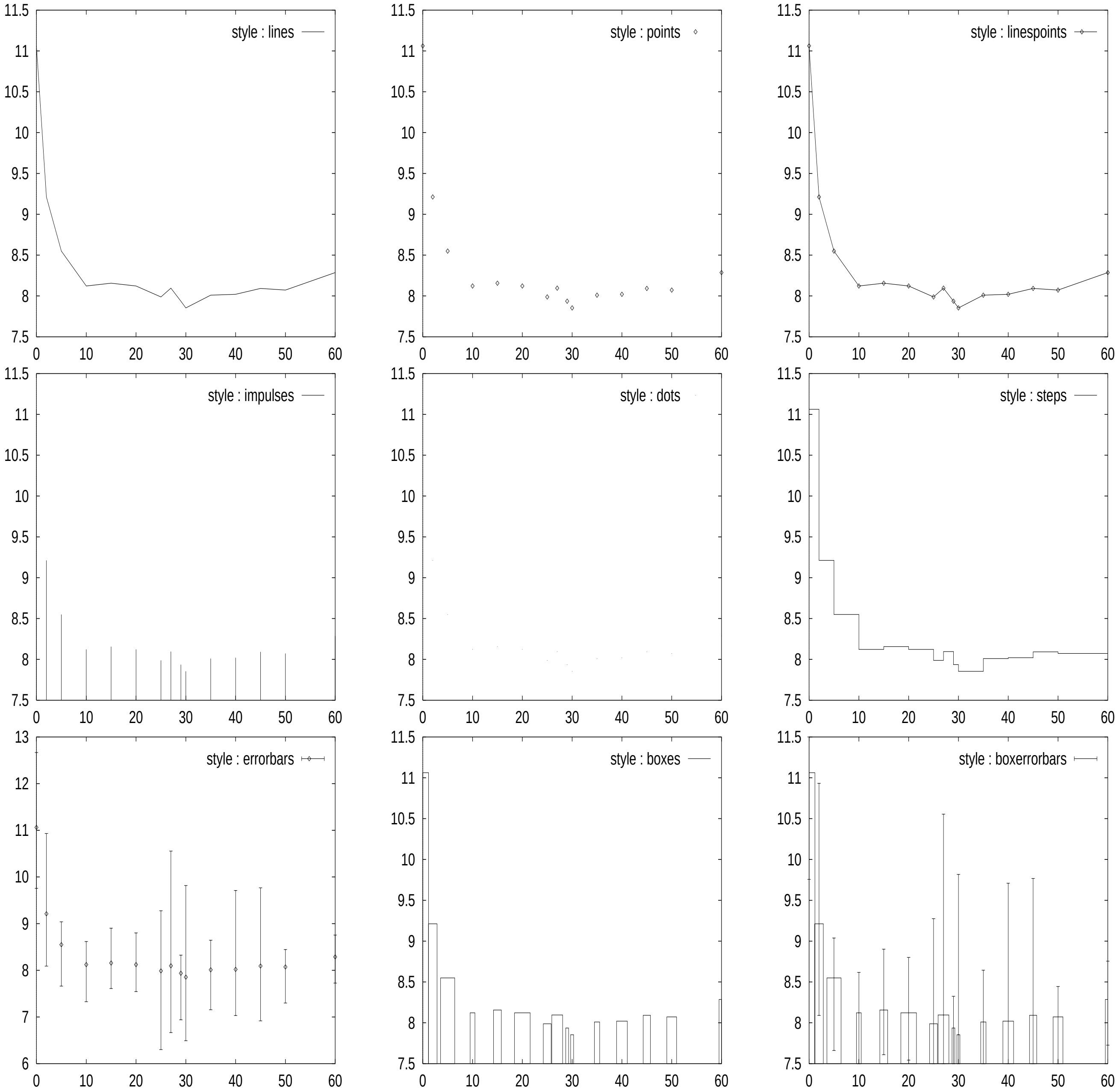


Figure 14: Various plotting style

5.5 Title

一般使用 GNUPLOT 繪圖時，會將函數名稱或存有數據資料的檔案名稱設定為 title。我們可用

```
title "<title>"
```

改變 title。如圖14 中即是運用

```
plot "err.dat" title "style : lines" with lines
```

將名稱由描述檔案名稱改為描述圖案名稱。我們亦可使用 "notitle" 使 title 不出現。如

```
plot my_function(x), 1 notitle
```

不將函數名稱 "1" 顯示在圖形上，而得到 y=1 直線，而這條直線可作為輔助線。

6 Input, Output

GNUPLOT 描繪數學函數或數值資料圖形，所以 GNUPLOT 除接受繪圖、定義常數函數等命令外，尚要能讀入數值資料。數值資料通常由程式產生，如表6。GNUPLOT 須用指令 plot 或 splot 由檔案內讀入數值資料。在輸出方面 GNUPLOT 除在終端機上顯示繪圖結果外，尚可輸出至許多種輸出裝置上。在此介紹 GNUPLOT 讀入、寫出方面的指令。

6.1 Input

在 GNUPLOT 中輸入命令的方式有兩種：

1. 執行 gnuplot 後，在 gnuplot> 提示符號下逐行輸入命令或是執行 `load "work.gnu"` 命令。此時，GNUPLOT 執行檔案 `work.gnu` 中的命令。
2. 在系統提示符號下 執行

   ```
   % gnuplot work.gnu
   ```

   GNUPLOT 執行 work.gnu 檔案內的命令。

GNUPLOT 不論繪數學函數或數值資料皆使用 plot(splot) 指令，此指令在第 5 章有詳細的介紹。在此介紹 GNUPLOT 讀入的數值資料格式及 "load" 與 "reread"— 可讀入含有 GNUPLOT 命令的檔案。

6.1.1 Read data from file

GNUPLOT 使用 plot(2D) 或 splot(3D) 指令讀入數值資料檔及繪出圖形，詳細的描述見 5.3 節。數值資料的格式為每行一組資料，若該行開頭為 # 表示該行為註解；若開頭為空白，則忽略這些空白。一組資料中的數字以空格或 tab 分開，數字以整數、浮點數字或科學記號來表示。下表即是數值資料檔 "data.gnuplot" 的範例。

```
# data.gnuplot
0 2 3 1
1 4 2 3
3 10 8 4
4 8 12 1
10 2 3 2
```

Table 6: An example of datafile.(III)

輸入

```
plot "data.gnuplot"  # 畫出 2D 的圖形
plot "data.gnuplot" using 1:3:4 with errorbars   # 畫出 2D errorbar 圖形
```

6.1.2 Read commands from file

當使用 GNUPLOT 進行繁雜的繪圖工作時，需要許多命令來完成工作。這些命令可儲存在檔案內，再經由 ”load” 指令將命令由檔案內讀出及執行。這些包含常數、函數的定義及繪圖環境的設定。load 可讀由 save 命令所儲存的檔案。load 語法為：

```
load "<input-file>"   # 使用 "(雙引號) 或 '(單引號) 皆可
```

例子如下：

```
load "work.gnu" # 讀入 work.gnu 檔案中的指令。
```

work.gnu 檔案內容為

```
damp(t) = exp(-s*wn*t)/sqrt(1.0-s*s)
per(t) = sin(wn*sqrt(1.0-s**2)*t - atan(-sqrt(1.0-s**2)/s))
c(t) = 1-damp(t)*per(t)
wn = 1.0
set xrange [0:13]
set samples 50
set dummy t
plot s=.1,c(t),s=.3,c(t),s=.5,c(t),s=.7,c(t)
replot s=.9,c(t),s=1.0,c(t),s=1.5,c(t),s=2.0,c(t)
```

在 GNUPLOT 中執行 `load "work.gnu"` 之後，繪出圖15 的結果。使用 load 指令執行 gnuplot 指令可不用每次皆逐一設定繪圖環境與函數常數設定等，亦可將此次結果儲存起來留待下次使用。

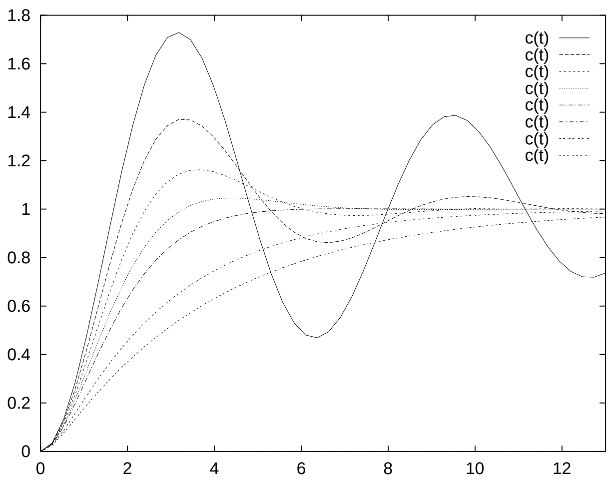


Figure 15: Demonstration of read commands from file

通常 GNUPLOT 完成一件繪圖工作的過程中，須設定許多繪圖環境。若緊接著畫另一張圖時，往往需調整許多繪圖環境參數。此時可讀一 GNUPLOT 原始設定的檔案，使 GNUPLOT 的設定恢復成起始狀態，就可不必逐一調整繪圖環境。例如將 GNUPLOT 的起始狀

態存於 environment.gnu [15] 檔案中，只要執行 load "environment.gnu"，即刻恢復 GNUPLOT 的繪圖環境成初始狀態。這是 load 指令的另一項應用。

6.1.3 Command : reread

GNUPLOT 提供 reread 指令，用來重複執行檔案中的指令。例如動態展示圖形或週期性改變部份常數而觀察函數的變化情形。以動態展示圖形為例，如圖16，reread 指令使 GNUPLOT 反覆地執行一組指令改變 3D 圖形的視角 (view point) 達到旋轉圖形的視覺效果。如此週而復始地重複執行命令 — 達到連續輸出的執行結果。下述指令：[16]

```
set parametric
set hidden3d
set nokey
set xrange [0:8]
set yrange [-4:4]
set zrange [-2:2]
set data style line
set title "Demo of animation capability - Digitized Blue Whale"
xrot=60
zrot=0
splot "whale.dat"     # 藍鯨的圖形資料檔
load "gnuplot.rot"    # 讀入 rotate 指令
```

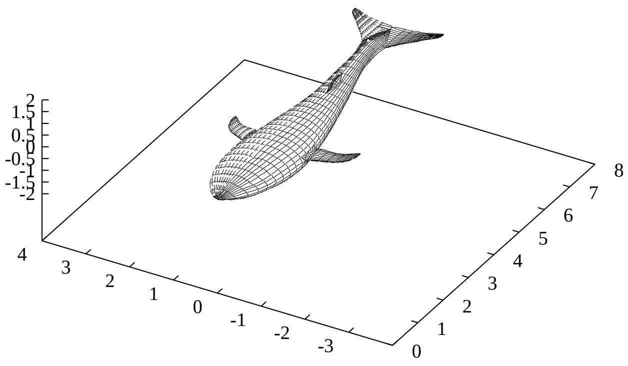


Figure 16: Demo of animation capability – Digitized Blue Whale

繪出如圖16 而不斷旋轉的藍鯨。其 gnuplot.rot 檔案中命令為不斷改變 3D 圖形的視角；其執行改變視角、重繪圖形後，因 reread 命令而重新執行 gnuplot.rot 檔案中命令。因而得到不斷移動的藍鯨。gnuplot.rot 檔案內容為

[15]可在執行 gnuplot 時，即刻執行 save "environment.gnu" 指令而得到 environment.gnu 檔案。
[16]此例引自 gnuplot demo

```
zrot=(zrot+10)%360
xrot=(xrot+17)%180
set view xrot,zrot
pause 3 # wait three seconds
replot
reread
```

此檔案內命令將 X 軸旋轉 17 度、Z 軸旋轉 10 度後，再重繪一次藍鯨。 reread 是重複執行本檔案命令。如此重複執行，直至按 Ctrl-c 才結束。

6.2 Output

在此可分為寫出數據資料與寫出鍵入的 GNUPLOT 命令、函數、常數 ...等兩部分

6.2.1 Write resuls to output device

GNUPLOT 提供許多種裝置的輸出語言，使得 GNUPLOT 的結果可輸出至不同的輸出裝置。此為 GNUPLOT 的特色。這許多種輸出裝置約略可分為以下三類 (見表7、表8、表9)：

- terminals : 輸出圖形於不同種類的終端機。
- printers or plotters : 產生列印裝置所能讀懂的語言。
- other descriptive languages : 產生其他程式描述圖形的語言。

name	description
dumb	printer or glass dumb terminal
aed512	AED 512 Terminal
aed767	AED 767 Terminal
bitgraph	BBN Bitgraph Terminal
kc_tek40xx	MS-DOS Kermit Tek4010 terminal emulator - color
km_tek40xx	MS-DOS Kermit Tek4010 terminal emulator - monochrome
tek410x	Tektronix 4106, 4107, 4109 and 420X terminals
tek40xx	Tektronix 4010 and others; most TEK emulators
vttek	VT-like tek40xx terminal emulator
x11	X11 Window System
X11	X11 Window System (identical to x11)
xlib	X11 Window System (gnulib_x11 dump)

Table 7: Output device : terminals

GNUPLOT 使用參數 terminal 設定輸出裝置形態，可設成任一種輸出裝置；使用參數 output 設定輸出的目的地，可送至標準輸出(standard output)、或檔案或另一程式的輸入。[17] 以下介紹幾種典型的應用：

[17]關於指令的描述，請參考4.6 節

name	description
dxy800a	Roland DXY800A plotter
epson_180dpi	Epson LQ-style 180-dot per inch (24 pin) printers
epson_60dpi	Epson-style 60-dot per inch printers
epson_lx800	Epson LX-800, Star NL-10, NX-1000, PROPRINTER ...
excl	Talaris EXCL Laser printer (also Talaris 1590 and others)
hp2623A	HP2623A and maybe others
hp2648	HP2648 and HP2647
hp7580B	HP7580, and probably other HPs (4 pens)
hp500c	HP DeskJet 500c, [75 100 150 300] [rle tiff]
hpgl	HP7475 and (hopefully) lots of others (6 pens)
hpljii	HP Laserjet series II, [75 100 150 300]
hpdj	HP DeskJet 500, [75 100 150 300]
hppj	HP PaintJet and HP3630 [FNT5X9 FNT9X17 FNT13X25]
imagen	Imagen laser printer
nec_cp6	NEC printer CP6, Epson LQ-800 [monocrome color draft]
okidata	OKIDATA 320/321 Standard
pcl5	HP LaserJet III [mode] [font] [point]
postscript	PostScript graphics language [mode "fontname" font_size]
prescribe	Prescribe - for the Kyocera Laser Printer
kyo	Kyocera Laser Printer with Courier font
qms	QMS/QUIC Laser printer (also Talaris 1200 and others)
starc	Star Color Printer
tandy_60dpi	Tandy DMP-130 series 60-dot per inch graphics
vx384	Vectrix 384 and Tandy color printer

Table 8: Output device : printers or plotters

6.2.2 Output format : postscript

介紹產生以 PostScript 語言描述的圖形。語法如下:

```
set terminal postscript {<mode>} {<color>} {<dashed>}
                        {"<fontname>"} {<fontsize>}
```

其中 <mode> 可為 **landscape**, **portrait**, **eps** 或 **default**。若選擇 **default** 則一切為預設值 — **landscape**, **monochrome**, **dashed**, 14點 "Helvetica" 的字形。預設為 10 吋寬 7 吋高的紙張。<color> 可為 **color** 或 **monochrome** 設定輸出為彩色或是單色。<dashed> 可為 **solid** 或 **dashed**。"<fontname>" 為 PostScript 認識的字形名稱。<fontsize> 為字形尺寸。

下面舉一些例子:

```
set terminal postscript # 設定輸出為 PostScript
# 設定為紙直放，彩色輸出，14 點 Times-Roman 字形。
set terminal postscript portrait color "Times-Roman" 14
```

此時搭配

name	description
aifm	Adobe Illustrator 3.0 Format
corel	EPS format for CorelDRAW
dxf	dxf-file for AutoCad (default size 120x80)
eepic	EEPIC – extended LaTeX picture environment
emtex	LaTeX picture environment with emTeX specials
gpic	GPIC – Produce graphs in groff using the gpic preprocessor
latex	LaTeX picture environment
mf	Metafont plotting standard
mif	Frame maker MIF 3.00 format
pbm	Portable bitmap [small medium large] [monochrome gray color]
pslatex	LaTeX picture environment with PostScript (specials)
pstricks	LaTeX picture environment with PSTricks macros
regis	REGIS graphics language
rgip	RGIP metafile (Uniplex). Option: fontsize (1-8)
uniplex	RGIP metafile (Uniplex). Option: fontsize (1-8)
table	Dump ASCII table of X Y [Z] values to output
texdraw	LaTeX texdraw environment
tgif	TGIF X-Window draw tool (file version 10)
tpic	TPIC – LaTeX picture environment with tpic (specials)

Table 9: Output : other descriptive languages

```
set output "result.ps"
plot sin(x)
```

命令，即將數學三角函數 $sin(x)$ 以 PostScript 語言儲存於 "result.ps" 檔案中。若執行

```
set output
set terminal x11
```

恢復成將結果畫在終端機上的狀態。

6.2.3 Output format : table

以數值精確描述取樣後圖形的各點座標值。舉例如下：

```
set terminal table
set output "datafile"
plot sin(x)
```

檔案 datafile 內容為

```
Curve 0, 100 points
i x=-10 y=0.544021
i x=-9.79798 y=0.364599
i x=-9.59596 y=0.170347
i x=-9.39394 y=-0.0308337
```

.
.
.

其每一行為描述數學三角函數 $sin(x)$ 取樣後的座標值。

6.2.4 Saving commands

當我們在 GNUPLOT 中辛苦地逐行鍵入命令，且經一番細心的修改而得到希望的結果時。我們期望能重複使用這組命令或留待下回參考。在 GNUPLOT 中提供 SAVE 命令，它以文字方式儲存使用者定義的函數、常數、繪圖參數或最後一個 plot (splot) 命令於檔案中。其語法如下：

```
save  {<option>} "<filename>"
```

此處 <option> 是設定儲存的項目，其項目可為 **functions**, **variables** 或 **set**。若不設定 option 則 GNUPLOT 儲存函數、常數、繪圖參數及最後一個 plot (splot) 命令於檔案中。下面是一些例子：

```
save "gnuplot.ini"           # 儲存所有的命令、函數、常數、options 等。
save functions 'func.dat'  # 儲存所有的函數。
save var 'var.dat'           # 儲存所有的常數。
save set "options.dat"       # 儲存所有的繪圖參數。
```

我們可用 save 儲存所鍵入的命令，待下回使用或儲存函數、常數、options 做為檢查使用。如附錄 A 中即是執行 GNUPLOT 後，立即儲存所有的設定，而得到 GNUPLOT 在 X Window System 環境下執行的原始狀態。

7 Miscellaneous commands

GNUPLOT 除了提供繪製圖形所需的指令外，尚提供許多有用的指令。如

- 與系統有關的指令 : cd, pwd, shell, start-up。
- 註解 : comment。
- 與編輯有關的指令 : clear, help, print, pause, command line-editing。
- 離開 GNUPLOT : quit, exit。

7.1 Instructions

1. cd : 改變目前的工作目錄。其語法為

   ```
   cd "<directory-name>"
   ```

   例如

   ```
   cd 'subdir'  # 改變至子目錄。
   cd ".."      # 改變至上一層目錄。
   ```

2. pwd : 執行 `pwd` 顯示目前的工作目錄於螢幕上。
3. shell : 執行 shell 命令。執行 `shell` 後，回到 shell 的提示符號下。欲回到 GNUPLOT 時，輸入 exit 或 Ctl-d 即可。如果只是執一個 shell 命令，只要在 shell 命令之前加 ! 即可。如 執行 `!pwd` 顯示目前的工作目錄。
4. start-up : GNUPLOT 執行時，先撿查是否有 .gnuplot 檔案。若有則執行檔案內的 GNUPLOT 命令。因此可適當加入一些命令於 .gnuplot 內，成為個人化的 GNUPLOT。

7.2 Comment

在撰寫 GNUPLOT 命令時，可適當加入註解以幫助我們瞭解。方法是在欲加註解之處，先寫 `#` 則 GNUPLOT 忽略其後的文字。但若 `#` 是在引號之內，此時為一單純字元。

7.3 Working window

GNUPLOT 提供方便使用者操作的指令。

1. clear : 輸入 `clear` 時，將螢幕及輸出清除。
2. help : GNUPLOT 提供線上輔助說明的功能。由此可得較多的說明。其語法為
   `help {<topic>}`。

3. print : `print <expression>` 將表示式的值印出。
4. pause : GNUPLOT 提供暫停螢幕一段時間並顯示一段文字的命令。其語法為

   ```
   pause <time> {"<string>"}
   ```

   其中 <time> 可以是一整數或表示式，表示 GNUPLOT 停留的時間。若為 -1 則 GNUPLOT 停留至按下 ”carriage return” 為止。若為 0 則 GNUPLOT 不停留。以下是一些例子：

   ```
   pause -1    # Wait until a carriage return is hit
   pause 3     # Wait three seconds
   pause -1  "Hit return to continue"
   pause 10  "Isn't this pretty?  It's a cubic-spline."
   ```

5. command line-editing : GNUPLOT 儲存以往輸入的命令，讓使用者可重複執行，或經修改後再使用。且不論游標在何處皆可按下 ”return key” 而輸入整行命令。這些編輯命令為

Character	Function
	Line Editing
`^B`	move back a single character.
`^F`	move forward a single character.
`^A`	move to the beginning of the line.
`^E`	move to the end of the line.
`^H, DEL`	delete the previous character.
`^D`	delete the current character.
`^K`	delete from current position to the end of line.
`^L, ^R`	redraw line in case it gets trashed.
`^U`	delete the entire line.
`^W`	delete from the current word to the end of line.
	History
`^P`	move back through history.
`^N`	move forward through history.

7.4 Quit

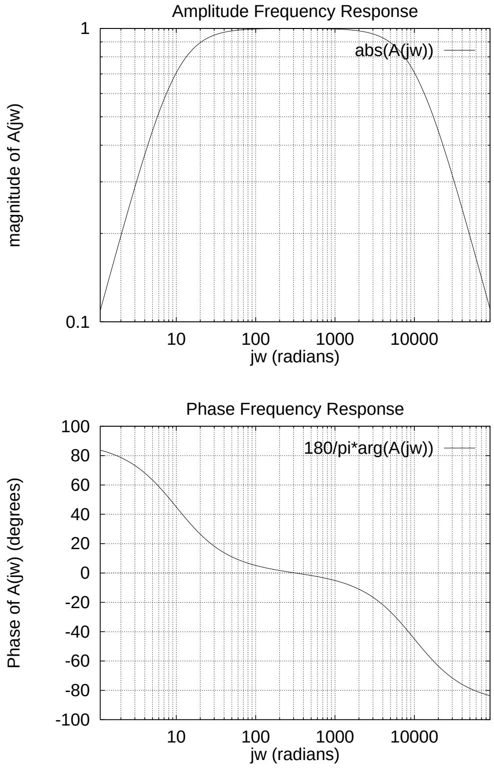
不論是 **exit**、**quit** 或 END-OF-FILE 字元，均可使 GNUPLOT 結束執行。

8 Examples

在前面的章節中曾提出 GNUPLOT 繪製圖形的步驟，也針對各步驟做詳細的說明。在此介紹自 GNUPLOT demo 中整理的範例，這些範例均在數行間展現 GNUPLOT 具體的功能。這些範例亦提供設計 GNUPLOT 程式時良好的參考，可酌量修改部份程式以符合個別需求。下表為此章中各個範例所描述的主題。

節次	說明
8.1	將座標軸改成 log scale 顯示圖形
8.2	用許多圖樣繪出數值資料圖形
8.3	以 error bar 繪製圖形
8.4	改變函數中的參數而繪出函數的變化情況
8.5	用參數式繪平面圖形
8.6	在極座標系統中繪圖
8.7	將平面圖形投影至曲面上
8.8	繪製立體圖形
8.9	繪製 contour 圖形
8.10	繪製亂數圖形
8.11	在 3D 中以格子狀的網線繪沒有秩序的資料圖形
8.12	繪製消去隱藏線的 3D 圖形
8.13	用參數式繪立體圖形
8.14	在 3D 中的繪數值資料圖形

8.1 Amplify frequency response



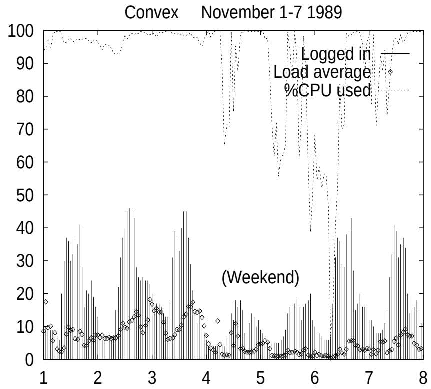

```
# amplitude frequency response
set samples 160
A(jw) = ({0,1}*jw/({0,1}*jw+p1))*(1/(1+{0,1}*jw/p2))
p1 = 10
p2 = 10000
set dummy jw
set grid
set key
set logscale xy
set title "Amplitude Frequency Response"
set xlabel "jw (radians)"
set xrange [1.1 : 90000.0]
set ylabel "magnitude of A(jw)"
set autoscale  y
plot abs(A(jw))
```

```
# phase frequency response
set nolog y
set logscale x
set title "Phase Frequency Response"
set ylabel "Phase of A(jw) (degrees)"
plot 180/pi*arg(A(jw))
```

Figure 17: Example : amplifier frequency response

此例為某一放大器的頻率響應圖，先繪增益部份再繪相位部份。其將座標軸改為對數方式顯示。因 $\mathrm{A(jw)}$ 為一複數方程式，故增益部份取其函數實數部份，而相位部份取其函數的相角的變化。

8.2 2D data plot



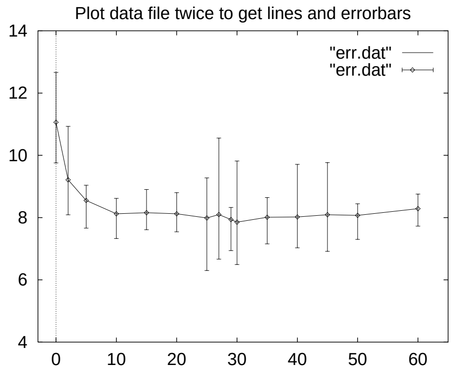

```
set title "Convex     November 1-7 1989"
set xrange [1:8]
set label "(Weekend)" at 5,25 center
plot 'using.dat' using 3:4 title "Logged in" with impulses,\
     'using.dat' using 3:5 t "Load average" with points,\
     'using.dat' using 3:6 t "%CPU used" with lines

# data file: using.dat $
891101 00   1.00  14   8.6  94.0
891101 01   1.04  10  17.5  94.4
891101 02   1.08   9   9.7  97.1
                .
                .
                .
891107 23   7.96  11   3.3  99.6
```

Figure 18: Example : 2D data plot

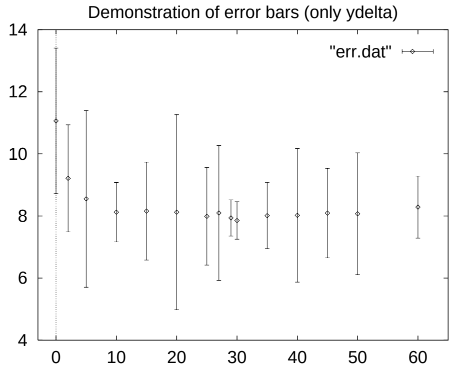
此例是將一資料檔案中的數據圖形化的例子。不同 column 的資料表示不同的資料，故以不同的圖形表示。在 plot 指令中的 using 後的數字就是指資料來源是第幾 column。

8.3 Error bar

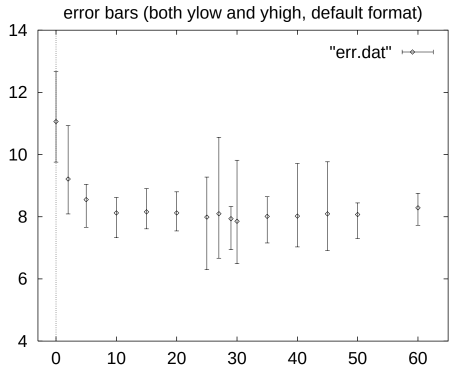
GNUPLOT 提供繪製 error bar 的方式有兩種：一種是設定 y delta，即標出一段誤差的範圍。另一種是設定 y low 與 y high，分別設定上下兩邊的誤差量。
一般繪製 error bar 是採用後者，故每一點需用四筆數字來描述。如第二個例子，採用 plot "err.dat" using 1:2:5 with errorbars 時，只採用第一、第二及第五行的數據。此時使用 y delta 方式描繪 error bar。




```
set xrange [-3:65]
set yrange [ 4:14]
set title "Plot data file twice to get lines and errorbars"
plot "err.dat" with lines, "err.dat" using 1:2:3:4 with errorbars

# data file: err.dat
0       11.062  9.757   12.667  2.345
2       9.212   8.0908  10.932  1.725
5       8.55    7.6612  9.0388  2.846
                  .
                  .
                  .
60      8.286   7.7265  8.755   1.0
```



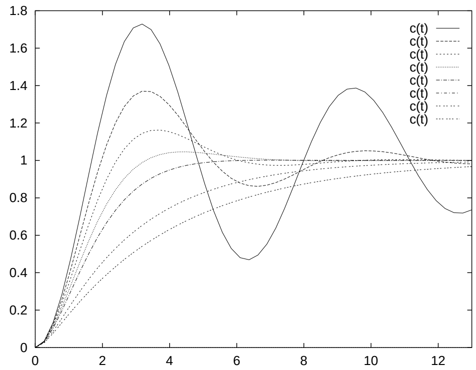

```
set xrange [-3:65]
set yrange [ 4:14]
set title "Demonstration of error bars (only ydelta)"
plot "err.dat" using 1:2:5 with errorbars

# data file: err.dat
0       11.062  9.757   12.667  2.345
2       9.212   8.0908  10.932  1.725
5       8.55    7.6612  9.0388  2.846
                  .
                  .
                  .
60      8.286   7.7265  8.755   1.0
```




```
set xrange [-3:65]
set yrange [ 4:14]
set title "error bars (both ylow and yhigh, default format)"
plot "err.dat" with errorbars

# data file: err.dat
0       11.062  9.757   12.667  2.345
2       9.212   8.0908  10.932  1.725
5       8.55    7.6612  9.0388  2.846
                  .
                  .
                  .
60      8.286   7.7265  8.755   1.0
```

Figure 19: Example : error bar

8.4 User-defined variable

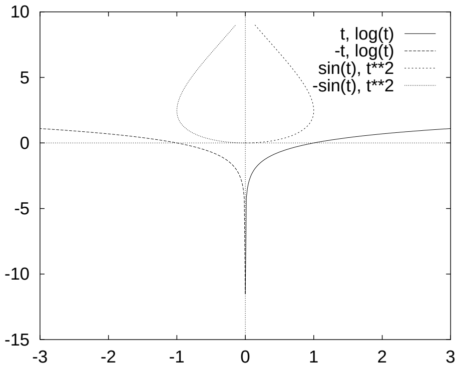



```
#
# From Automatic Control Systems, fourth ed.,
# figure 6-14
# transient response of a second-order system
# to a unit step input function
#
damp(t) = exp(-s*wn*t)/sqrt(1.0-s*s)
per(t) = sin(wn*sqrt(1.0-s**2)*t-atan(-sqrt(1.0-s**2)/s))
c(t) = 1-damp(t)*per(t)
#
#       wn is natural undamped frequency
#       s is damping factor
#
wn = 1.0
set xrange [0:13]
set samples 50
set dummy t
#
# plot c(t) for several different damping factors s
#
plot s=.1,c(t),s=.3,c(t),s=.5,c(t),s=.7,c(t),s=.9,c(t),\
     s=1.0,c(t),s=1.5,c(t),s=2.0,c(t)
```

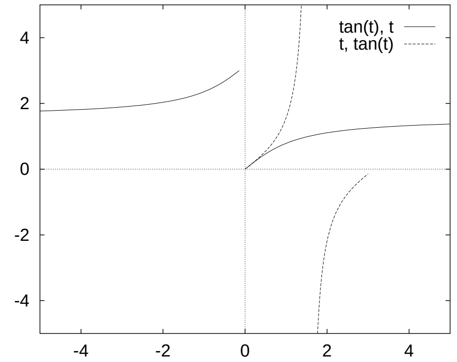
Figure 20: Example : unit step response

GNUPLOT 提供使用者設定參數，讓使用者可繪出不同參數值下函數圖形的變化。

8.5 2D parametric function




```
set parametric
set samples 200
set trange [0.00001:3]
plot t,log(t),-t,log(t),sin(t),t**2,-sin(t),t**2
```

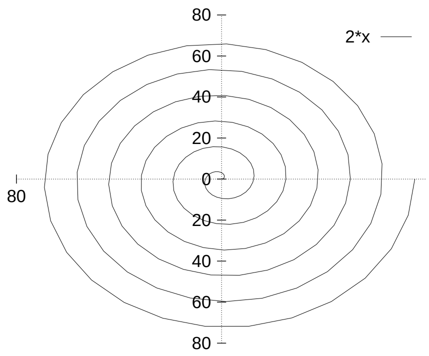



```
set parametric
set samples 200
set xrange [-5:5]
set yrange [-5:5]
plot tan(t),t,t,tan(t)
```

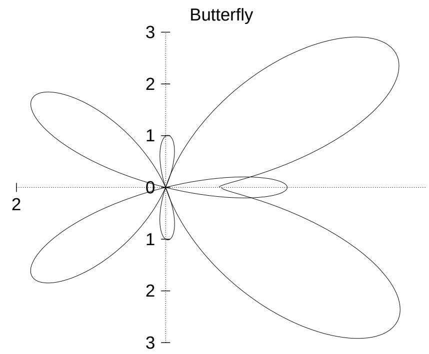
Figure 21: Example : 2D parametric functions

GNUPLOT 提供以參數式作為描述圖形的方式。在 2D 中，是由一對參數式決定所繪的圖形 (即 $x = f(t)$, $y = g(t)$)，此時以 t 作為變數。此圖第二部份則以設定 x 與 y 的範圍來控制顯示範圍。

8.6 Polar coordinates




```
set noborder
set clip
set polar
set samples 160
set xrange [0:12*pi]
plot 2*x
```




```
set noborder
set clip
set polar
butterfly(x)=exp(cos(x))-2*cos(4*x)+sin(x/12)**5
set samples 800
set title "Butterfly"
set nokey
plot butterfly(x)
```

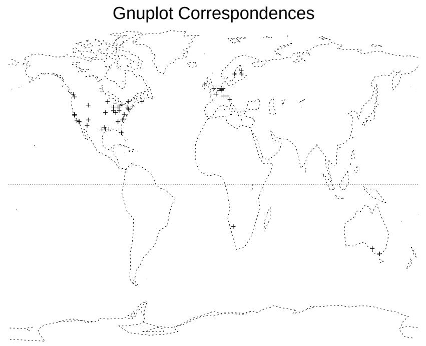
Figure 22: Example : plotting on polar coordinate

GNUPLOT 提供在極座標中繪圖的環境。此時以 x 做為變數代表 angle，以徑度量做為單位 (2π 為一圓周)，若輸入 `set angle degrees` ，則改為度度量 (360° 為一圓周)。輸入 `set noborder` 為不將四周的方框繪出。

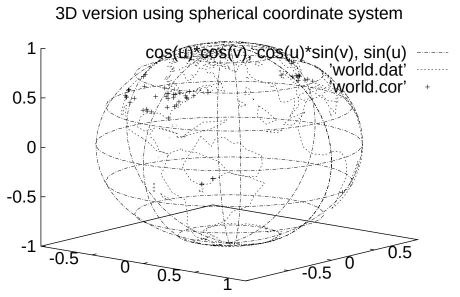
8.7 Mapping

一般將 2D 數值資料圖形繪於平面，我們可使用 mapping 此參數將資料投影於球面或圓柱面上。圖23 中世界地圖的地形曲線資料是儲存在 world.dat 檔案內，各大城市的座標位置資料儲存在 world.cor 檔案內。所有資料數據局限在 (180,90) 和 (-180,-90) 之間。

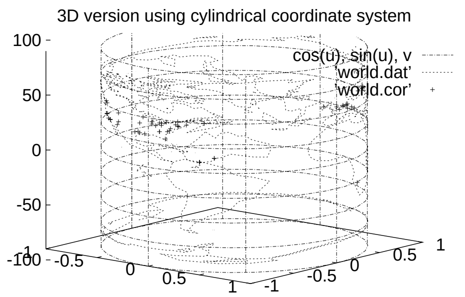
第一部份是將地圖繪於平面上。第二部份是將地圖繪於球面上，是將資料數據經由一組函數轉換後再描繪於 3D 空間上。第三部份是將地圖繪於圓柱上。




```
set title "Gnuplot Correspondences"
set nokey
set noborder
set noyzeroaxis
set noxtics
set noytics
#
# plot world map and correspondent locations as a +
plot 'world.dat' with lines 3 4, 'world.cor' with points 1 2
```



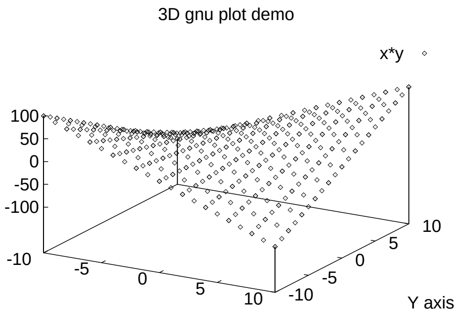

```
# 3D version using spherical coordinate system
set angles degrees
set title "3D version using spherical coordinate system"
set view 70,40,,2.0
set mapping spherical
set parametric
set samples 32
set isosamples 9
set urange [-pi/2:pi/2]
set vrange [0:2*pi]
splot cos(u)*cos(v),cos(u)*sin(v),sin(u) with lines 5 6,\
      'world.dat' with lines 3 4, 'world.cor' with points 1 2
```



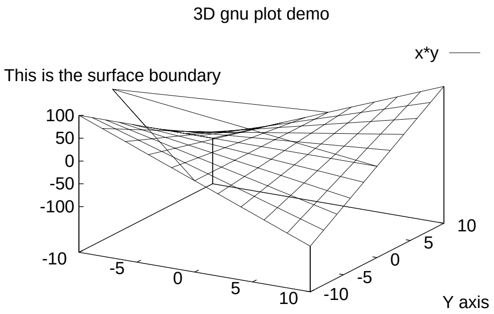

```
# 3d version using cylindrical coordinate system
set title "3d version using cylindrical coordinate system"
set view 70,40,,2.0
set mapping cylindrical
set parametric
set samples 32
set isosamples 9
set urange [-pi:pi]
set vrange [-90:90]
splot cos(u),sin(u),v with lines 5 6,\
      'world.dat' with lines 3 4, 'world.cor' with points 1 2
```

Figure 23: example : the world map

8.8 3d plotting




```
set samples 21
set isosample 11
set xlabel "x axis" -3,-2
set ylabel "y axis" 3,-2
set zlabel "z axis" -5
set title "3d gnu plot demo"
splot x*y with points
```



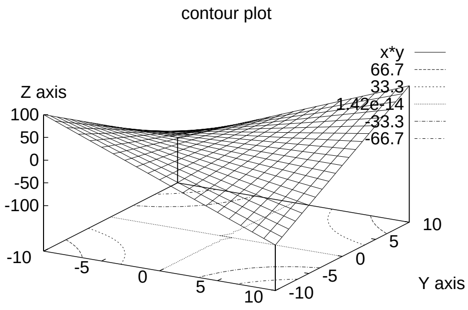

```
set samples 21
set isosample 11
set xlabel "x axis" -3,-2
set ylabel "y axis" 3,-2
set zlabel "z axis" -5
set title "3d gnu plot demo"
set label 1 "this is the surface boundary" at -10,-5,150 center
set arrow 1 from -10,-5,120 to -10,0,0 nohead
set arrow 2 from -10,-5,120 to 10,0,0 nohead
set arrow 3 from -10,-5,120 to 0,10,0 nohead
set arrow 4 from -10,-5,120 to 0,-10,0 nohead
set xrange [-10:10]
set yrange [-10:10]
splot x*y
```

Figure 24: example : 3d plotting

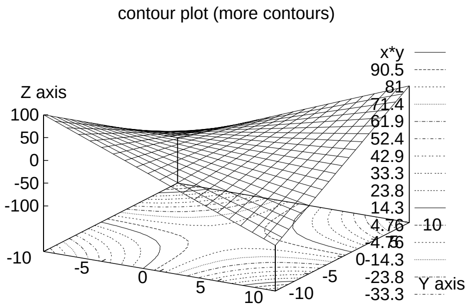
此例說明以 gnuplot 畫 3d 圖形。gnuplot 使用 xlable, ylable, zlable 設定座標軸名稱及放在特定的位置。圖中第二部份舉出加入線條及文字說明的方法。

8.9 contour

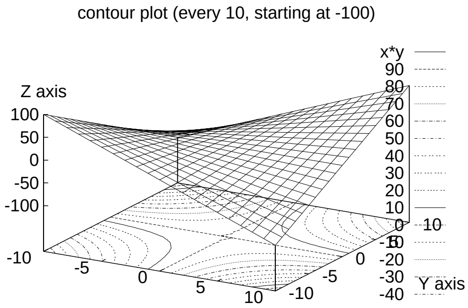
gnuplot 提供 contour 的功能。contour 畫在 x、y 平面上，或曲面本身。只要輸入指令 set contour 後，即可繪出 contour。contour 可由設定參數 cntrparam 而改變 contour 外貌。




```
set samples 20
set isosamples 21
set xlabel "x axis" -5,-2
set ylabel "y axis" 4,-1
set zlabel "z axis"
set title "contour plot"
set contour
splot x*y
```

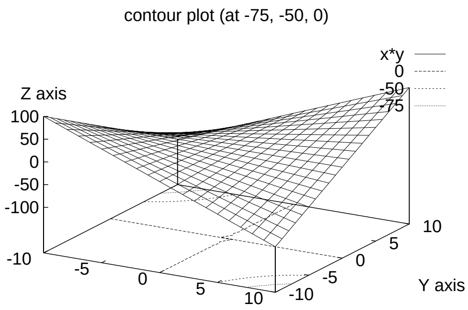



```
set samples 20
set isosamples 21
set xlabel "x axis" -5,-2
set ylabel "y axis" 4,-1
set zlabel "z axis"
set cntrparam levels 20
set title "contour plot (more contours)"
set contour
splot x*y
```

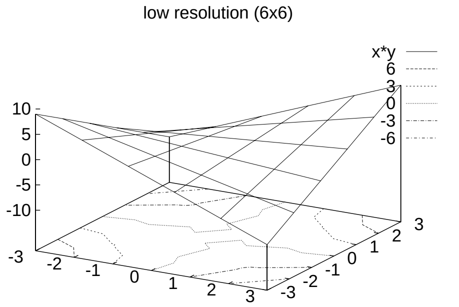



```
set samples 20
set isosamples 21
set xlabel "x axis" -5,-2
set ylabel "y axis" 4,-1
set zlabel "z axis"
set title "contour plot (every 10, starting at -100)"
set contour
set cntrparam levels incr -100,10,100
splot x*y
```

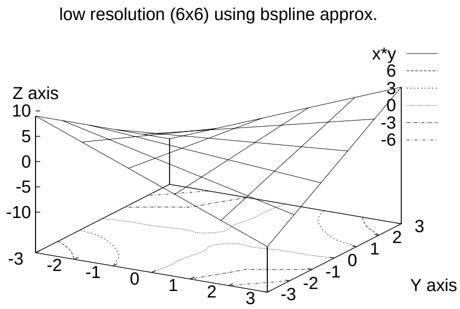
Figure 25: example : contour (i)




```
set samples 20
set isosamples 21
set xlabel "x axis" -5,-2
set ylabel "y axis" 4,-1
set zlabel "z axis"
set contour
set cntrparam levels disc -75,-50,0
set title "contour plot (at -75, -50, 0)"
splot x*y
```




```
set xrange [-3:3]
set yrange [-3:3]
set samples 6
set isosamples 6
set cntrparam levels 5
set title "low resolution (6x6)"
set contour
splot x*y
```

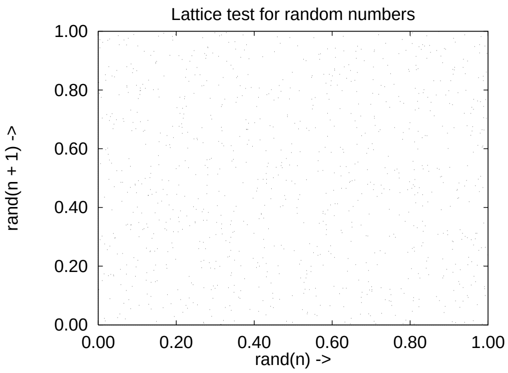



```
set xrange [-3:3]
set yrange [-3:3]
set xlabel "x axis" -5,-2
set ylabel "y axis" 4,-1
set zlabel "z axis"
set samples 6
set isosamples 6
set contour
set cntrparam order 4
set cntrparam bspline
set title "low resolution (6x6) using bspline approx."
splot x*y
```

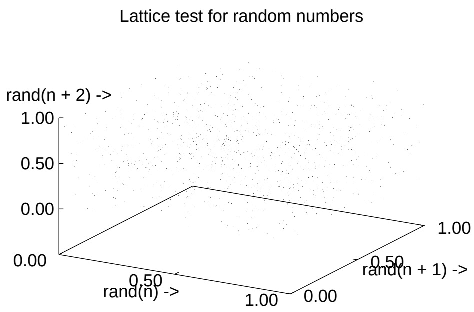
Figure 26: example : contour (ii)

8.10 random number

gnuplot 利用一簡單的亂數產生器產生一組亂數，並繪於 2d 平面或 3d 立體空間中。由參數 sample 決定產生多少點 (若在 3d 中，參數 isosample 亦有影響)，在圖27 中 2d 之中是 set sample 1000，因此產生 1000 點。而在 3d 之中 set sample 50，產生 $50 * 20 = 1000$ 點。(因 isosamples 為 20，以 20 條線畫出圖形，故為 20 倍。) 此實際點數可將圖形裝置設定為 table 後，而得到數值的表示方法。




```
seed = 1317
seed = rand(seed)
set nokey
set xrange [0: 1]
set yrange [0: 1]
set title "lattice test for random numbers"
set xlabel "rand(n) ->"
set ylabel "rand(n + 1) ->"
set zlabel "rand(n + 2) ->"
set format x "%3.2f"
set format y "%3.2f"
set tics
set sample 1000
set function style dots
set parametric
plot rand(1), rand(1)
```

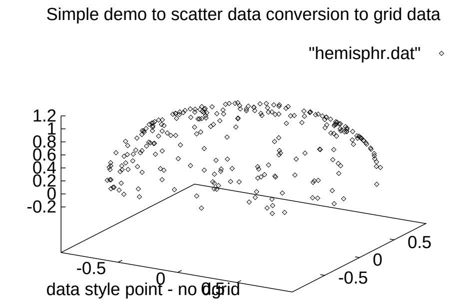



```
seed = 1317
seed = rand(seed)
set nokey
set xrange [0: 1]
set yrange [0: 1]
set zrange [0: 1]
set title "lattice test for random numbers"
set xlabel "rand(n) ->"
set ylabel "rand(n + 1) ->"
set zlabel "rand(n + 2) ->"
set format x "%3.2f"
set format y "%3.2f"
set format z "%3.2f"
set tics
set function style dots
set parametric
pause 0 "3d plot ahead, one moment please ..."
set sample 50
splot rand(1), rand(1), rand(1)
```

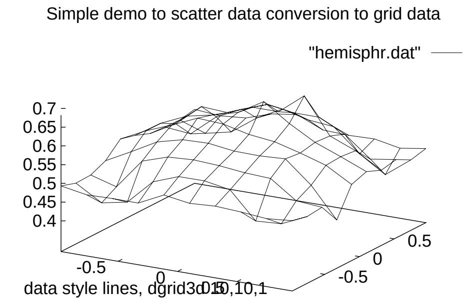
Figure 27: example : random number

8.11 scatter data -> grid data

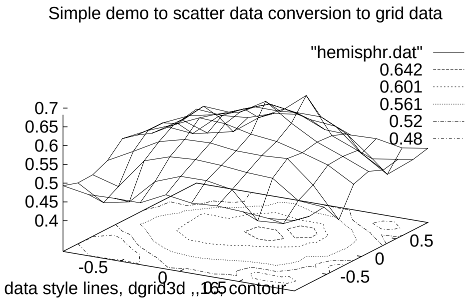
我們很難觀察雜亂無序的資料所繪的圖形。在 2d 上的數據資料可經由 sort 將數據依大小順序排列。在 3d 上的數據，無法使用 sort 的技巧；而可在點出現的方形區域內，以格子狀的網線代替原來的點。而格子上交點之 z 值以距離為反比地加權所有點後再求平均值而求得。由此法可觀察到數據分佈的情形。圖中 set dgrid3d 10,10,1 為以 10x10 的格子狀網線代替原來的數據，而以 l1 norn 做為計算距離，此值為 2 的乘冪。
在 gnuplot 中，僅能以排成格狀的資料畫 contour 圖形。故將整組資料畫成格狀圖形後，可畫 contour 圖形。




```
set title "simple demo to scatter data conversion to grid data"
set nohidden3d
set ticslevel 0.5
set view 60,30
set autoscale
set parametric
set data style points
set xlabel "data style point - no dgrid"
splot "hemisphr.dat"
```



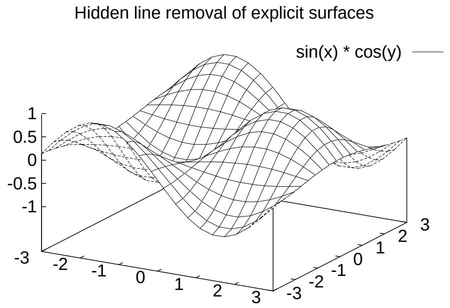

```
set title "simple demo to scatter data conversion to grid data"
set nohidden3d
set ticslevel 0.5
set view 60,30
set autoscale
set parametric
set data style points
set dgrid3d 10,10,1
set xlabel " data style lines, dgrid3d 10,10,1"
set data style lines
splot "hemisphr.dat"
```



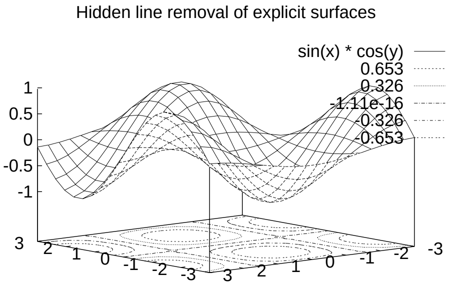

```
set title "simple demo to scatter data conversion to grid data"
set nohidden3d
set ticslevel 0.5
set view 60,30
set autoscale
set parametric
set dgrid3d 10,10,1
set data style lines
set contour
set xlabel "data style lines, dgrid3d ,,16, contour"
splot "hemisphr.dat"
```

Figure 28: example : scatter data -> grid data

8.12 3d hidden




```
set samples 20
set isosamples 20
set hidden3d
set title "hidden line removal of explicit surfaces"
set view 60,30
set xrange [-3:3]
set yrange [-3:3]
splot sin(x) * cos(y)
```



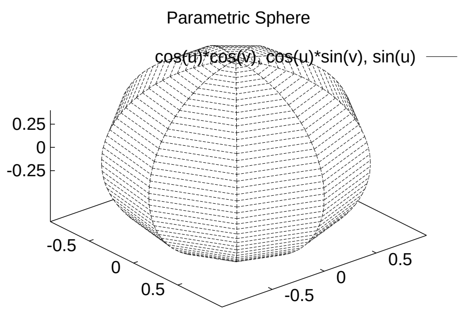

```
set samples 20
set isosamples 20
set hidden3d
set title "hidden line removal of explicit surfaces"
set xrange [-3:3]
set yrange [-3:3]
set view 75,230
set contour
splot sin(x) * cos(y)
```

Figure 29: example : 3d hidden

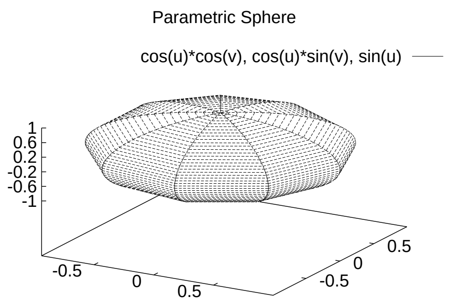
gnuplot 提供 3d 中消去隱藏線的功能，使圖形更具立體感。

8.13 3d parametric function

此為 gnuplot 在 3d 中使用參數式的例子。gnuplot 設定變數為 u 和 v (即 $x = f(u,v)$, $y = g(u,v)$, $z = h(u,v)$)。set view 45, 50,,2.5 為改變 3d 圖形投影到平面上的視角及圖形放大的比例。



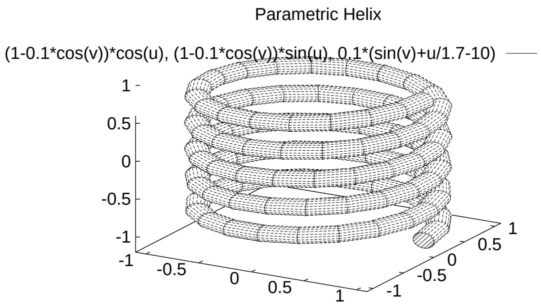

```
set parametric
set isosamples 50,10
set hidden
set title "parametric sphere"
set urange [-pi/2:pi/2]
set vrange [0:2*pi]
set zrange [-1:1]
set ztics -1.0,0.25,1.0
set view 45,50,,2.5
splot cos(u)*cos(v),cos(u)*sin(v),sin(u)
```




```
set parametric
set isosamples 50,10
set hidden
set title "parametric sphere"
set urange [-pi/2:pi/2]
set vrange [0:2*pi]
set zrange [-1:1]
set ztics -1.0,0.25,1.0
set view ,,,0.8
splot cos(u)*cos(v),cos(u)*sin(v),sin(u)
```

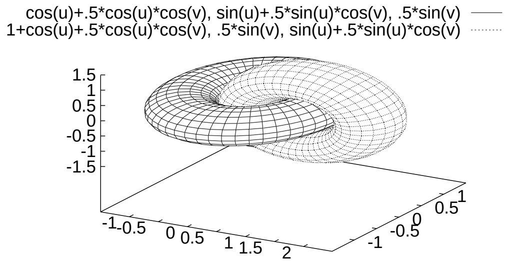
Figure 30: example : 3d parametric function

此為使用 3d 參數式描繪一些典型的函數。




```
set parametric
set hidden
set title "Parametric Helix"
set isosamples 100,20
set urange [0:10*pi]
set vrange [0:2*pi]
set zrange [-0.6:0.6]
splot (1-0.1*cos(v))*cos(u),(1-0.1*cos(v))*sin(u),\
      0.1*(sin(v)+u/1.7-10)
```

Figure 31: Example : Parametrix Helix

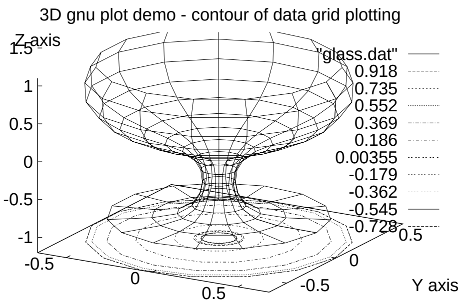



```
set parametric
set hidden
set urange [-pi:pi]
set vrange [-pi:pi]
set isosamples 50,20
splot cos(u)+.5*cos(u)*cos(v),sin(u)+.5*sin(u)*cos(v),\
      .5*sin(v) with lines,1+cos(u)+.5*cos(u)*cos(v),\
      .5*sin(v),sin(u)+.5*sin(u)*cos(v) with lines
```

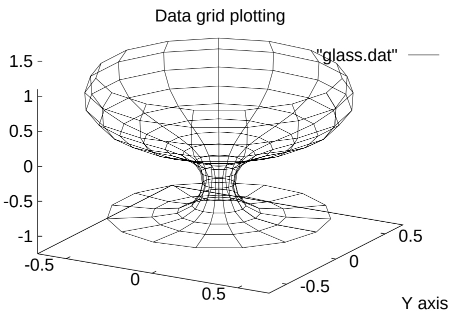
Figure 32: Example : Interlocking Tori

8.14 3D data plotting

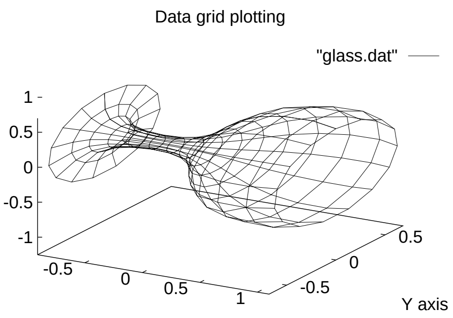
此為 GNUPLOT 畫 3D 數值資料圖形的例子。




```
set samples 20
set isosamples 21
set xlabel "X axis" -5,-2
set ylabel "Y axis" 4,-1
set zlabel "Z axis"
set contour
set cntrparam levels 10
set xrange [0:15]
set yrange [0:15]
set auto
set zrange [-0.6:0.6]
set data style lines
set title "3D gnu plot demo - contour of data grid plotting"
set parametric
splot "glass.dat"
```




```
set samples 21
set isosample 11
set xlabel "X axis" -3,-2
set ylabel "Y axis" 3,-2
set zlabel "Z axis" -5
set nolog
set auto
set zrange [-0.6:0.7]
set data style lines
set title "Data grid plotting"
set parametric
splot "glass.dat"
```




```
set samples 21
set isosample 11
set xlabel "X axis" -3,-2
set ylabel "Y axis" 3,-2
set zlabel "Z axis" -5
set nolog
set auto
set zrange [-0.6:0.7]
set data style lines
set title "Data grid plotting"
set parametric
splot "glass.dat" using 3:2:1
```

Figure 33: Example : 3D data plotting

9 Conclusion

在前面章節中，先介紹 GNUPLOT 繪圖的方法與步驟。接者由許多例子展現 GNUPLOT 不同方面的特性及程式的整體性。

GNUPLOT 是一個資料型態轉換的處理程式，它將數學函數或數值資料轉換成平面或立體的圖形資料，以圖形方式對資料做整體性的觀察及探討，是一非常有用的工具。

A The initial setting of GNUPLOT

此附錄為 GNUPLOT 在 UNIX 和 X 環境下執行的初始狀態。

```
set terminal x11        # 設定輸出裝置為 X 終端機
set output              # 將輸出送至標準輸出
set noclip points       # 將接近的點依然畫出
set clip one            # 當線段的端點出現在顯示區域內時，才畫此線段，否則不畫
set noclip two          # 不將出現在顯示區域內的部份線段繪出
set border              # 將圖形外圍加框
set boxwidth
set dummy x,y           # 設定自變數名稱為 x,y
set format x "%g"       # 設定 X 軸標點顯示方式為 C 語言中 %g 格式
set format y "%g"       # 設定 Y 軸標點顯示方式為 C 語言中 %g 格式
set format z "%g"       # 設定 Z 軸標點顯示方式為 C 語言中 %g 格式
set nogrid              # 不產生網格
set key                 # 將函數名稱或檔案名稱置於右上角
set nolabel             # 取消圖上任何文字說明
set noarrow             # 取消圖上任何線段
set nologscale          # 採取線性方式顯示圖形
set offsets 0, 0, 0, 0  # 不在圖四周留下空白
set nopolar             # 採用 Cartesian coordinate
set angles radians      # 以徑度量表示角度，範圍[0:2*pi]
set noparametric        # 不採用參數式
set view 60, 30, 1, 1   # 旋轉 X 軸 60 度 Z 軸 30 度
set samples 100, 100    # 設定 X 軸 (Y 軸) 的取樣點數為 100 點
set isosamples 10, 10   # 在 3D 中，採用縱橫各 10 條線構成圖形曲面
set surface             # 在 3D 中，畫出曲面
set nocontour           # 不畫出 contour
set clabel              # 設定將 contour 所畫每一條等高線以不同顏色表示
set nohidden3d          # 不用消去隱藏線的功能
set cntrparam order 4
set cntrparam linear
set cntrparam levels auto 5
set cntrparam points 5
set size 1,1            # 設定圖形長寬比例為 1:1
set data style points   # 以 points 作為繪數值資料的圖案
set function style lines  # 以 lines 作為繪函數的圖案
set xzeroaxis           # 設定顯示 X 軸
set yzeroaxis           # 設定顯示 Y 軸
set tics in             # 設定標點在軸的內面
set ticslevel 0.5
set xtics               # X 軸標點以數字標示
set ytics               # Y 軸標點以數字標示
set ztics               # Z 軸標點以數字標示
set title "" 0,0        # 圖的標頭不顯示
set notime              # 不顯示產生圖形的時間
set rrange [-0 : 10]
set trange [-5 : 5]     # 在 2D 參數式中以 t 為變數，此為設定顯示區間
set urange [-5 : 5]     # 在 3D 參數式中以 u,v 為變數，此為設定顯示區間
set vrange [-5 : 5]     # 在 3D 參數式中以 u,v 為變數，此為設定顯示區間
set xlabel "" 0,0       # X 軸標頭不顯示
```

```
set xrange [-10 : 10]   # 設定 X 軸顯示區間 -10,10 之間
set ylabel "" 0,0       # Y 軸標頭不顯示
set yrange [-10 : 10]   # 設定 Y 軸顯示區間 -10,10 之間
set zlabel "" 0,0       # Z 軸標頭不顯示
set zrange [-10 : 10]   # 設定 Z 軸顯示區間 -10,10 之間
set autoscale r
set autoscale t         # 自動調整 t 的顯示區間使圖可全部顯示
set autoscale xy        # 自動調整 X 軸 Y 軸顯示區間使圖可全部顯示
set autoscale z         # 自動調整 Z 軸顯示區間使圖可全部顯示
set zero 1e-08          # 設定比 1e-08 小的正數為零
```

B Call gnuplot from C program

我們可在 C 程式中呼叫 gnuplot 替我們繪圖。方法為產生 process 執行 gnuplot，建立對 gnuplot 下達命令的管道，使用檔案傳遞資料。以下是在 UNIX 中呼叫 gnuplot 的例子：[18]

```
#include <sys/types.h>
#include <sys/stat.h>
#include <stdlib.h>
#include <stdio.h>
#include <math.h>
#include <unistd.h>

#define PANIC(a) do { \
                perror(a); \
                if (temp_name) unlink(temp_name);\
                exit(1);\
        } while(0)

int main() {
    FILE *command,*data;
    char *temp_name;
    double a,b;
    int i;

    if ((temp_name = tmpnam((char *) 0)) == 0) PANIC("tmpnam failed");
    if(mkfifo(temp_name, S_IRUSR | S_IWUSR) != 0) PANIC("mkfifo failed");
    command = popen("gnuplot","w");
    fprintf(command,"plot \"%s\" with lines\n",temp_name); fflush(command);
    data = fopen(temp_name,"w");
    for (i=0; i<20; i++) {
        a = i/10.0;
        b = sin(a);
        fprintf(data,"%f %f\n",a,b);
    }
    fclose(data);
    fprintf(stderr,"press enter to continue..."); fflush(stderr);
    getchar();

    fprintf(command,"plot \"%s\" with lines\n",temp_name); fflush(command);
    data = fopen(temp_name,"w");
    for (i=0; i<20; i++) {
        a = i/10.0;
        b = cos(a);
        fprintf(data,"%f %f\n",a,b);
    }
    fclose(data);
    fprintf(stderr,"press enter to continue..."); fflush(stderr);
    getchar();
    fclose(command);
    unlink(temp_name);
    return 0;
}
```

[18] 此範例取自 newsgroup – comp.graphics.gnuplot 中的 FAQ

程式在第 22 行使用函數 popen() 執行 gnuplot 並建立對 gnuplot 下命令的通道。[19] 程式在 23 行使用

```
fprintf(command,"plot \"%s\" with lines\n",temp_name);
fflush(command);
```

下達命令給 gnuplot。在 21 行使用函數 mkfifo() 建立儲存資料的檔案。[20] 由第 35 至 40 行程式

```
data = fopen(temp_name,"w");
for (i=0; i<20; i++) {
    a = i/10.0;
    b = cos(a);
    fprintf(data,"%f %f\n",a,b);
}
```

可在檔案中產生一組描述數學三角函數 $cos(x)$ 的數值資料。因此程式可對 gnuplot 下命令並將數值資料準備在檔案內，再由 gnuplot 讀取。

下面是 OS/2 中 C 程式呼叫 gnuplot 的例子 (使用 GCC compilier)：

```
#include <stdio.h>
#define INCL_DOS
#define INCL_DOSPROCESS
#define INCL_DOSNMPIPES
#include <os2.h>

main()
    {
    HPIPE hpipe ;
    FILE *hfile, *hgnu ;
        /* create a named pipe. Use NP_WAIT so that DosConnect...
           blocks until client (gnuplot) opens, and client reads
           are blocked until data is available */
    DosCreateNPipe( "\\pipe\\gtemp",
                    &hpipe,
                    NP_ACCESS_OUTBOUND,
                    NP_WAIT|NP_TYPE_BYTE|1,
                    256,
                    256,
                    -1 ) ;
        /* use stream i/o */
    hfile = fdopen( hpipe, "w" ) ;

        /* start gnuplot; use unbuffered writes so we don't need to
           flush buffer after a command */
    hgnu = popen( "gnuplot", "w" ) ;
    setvbuf( hgnu, NULL, _IONBF, 0 ) ;

```

[19] 關於函數 popen() 的說明，可參考 W. Richard Stevens, Advanced Programming in the UNIX Environment. Addison-Wesley, pp 435-441, 1992.

[20] 關於函數 mkfifo() 的說明，可參考 W. Richard Stevens, Advanced Programming in the UNIX Environment. Addison-Wesley, pp 445-449, 1992.

```
        /* plot a set of data */

    fprintf( hgnu, "plot '/pipe/gtemp'\n" ) ;  /* issue plot command */
    DosConnectNPipe( hpipe ) ;              /* wait until 'file' opened */
    fprintf( hfile, "1 1\n" ) ;             /* write data to 'file' */
    fprintf( hfile, "2 2\n" ) ;
    fprintf( hfile, "3 3\n" ) ;
    fprintf( hfile, "4 4\n" ) ;
    fflush( hfile ) ;                       /* flush buffer forces read */
    DosSleep( 500 ) ;
    DosDisConnectNPipe( hpipe ) ;           /* disconnect this session */
    fprintf( hgnu, "pause -1\n" ) ;         /* admire plot */

        /* plot another set of data */

    fprintf( hgnu, "plot '/pipe/gtemp'\n" ) ;
    DosConnectNPipe( hpipe ) ;
    fprintf( hfile, "1 4\n" ) ;
    fprintf( hfile, "2 3\n" ) ;
    fprintf( hfile, "3 2\n" ) ;
    fprintf( hfile, "4 1\n" ) ;
    fflush( hfile ) ;
    DosSleep( 500 ) ;
    DosDisConnectNPipe( hpipe ) ;
    fprintf( hgnu, "pause -1\n" ) ;

    DosClose( hpipe ) ;
    pclose( hgnu ) ;
    }
```

C Using gnuplot commands in shell script

我們也可將 gnuplot 命令放入 shell script 中執行。舉例如下：

```
#! /bin/csh
        .
        .
        .
set datafile = Mydata
gnuplot << EOF
set terminal tex40xx
plot "$datafile" with lines
quit
EOF
        .
        .
        .
```

執行此 shell script 時，gnuplot 用線條畫出 Mydata 檔案中數值資料的圖形。在第三行中的 `gnuplot << EOF` 為執行 gnuplot 並將其下的數行當做 gnuplot 的輸入。`EOF` 表示輸入資料的起始及結束點，所以第四至第六行為 gnuplot 的輸入資料。gnuplot 先設定輸出裝置為 `tex40xx`，再畫出變數 `data1` 所指定檔案的內容，然後離開 gnuplot。