GNUPLOT 使用手册

中央研究院计算中心
ASPAC 计划
aspac@beta.wsl.sinica.edu.tw
技术报告： 94002
1994 年 5 月 20 日
Version：2.1

Chapter 1. Introduction

GNUPLOT 是在1986年由 Colin kelley 和 Thomas Williams 所发展的绘图公用程式，可将数学函数或数值资料以平面或立体的图形画在不同种类终端机或绘图输出装置上。目前已广泛地被人们使用，在 Internet 网路上尚有 newsgroup --- comp.graphics.gnuplot 作为大家交换心得或学习的园地。此 newsgroup 将许多问题集成 FAQ (Frequently Answered Questions) 并定期刊载。

GNUPLOT 是一个命令导向的交谈式绘图程式（command-driven interactive function plotting program）。GNUPLOT 执行使用者输入的每一项命令，可逐步设定或修改绘图环境。它以图形表达数据或函数，使我们藉由图形 做更进一步的分析。

在此以 GNUPLOT 3.5版为讨论主体。而本篇文章所引用范例皆整理自 GNUPLOT demo 中范例。在本篇文章中，而本篇文章中的图例亦引用自子目录 demo 中的范例。在本篇文章中，第二章、介绍 GNUPLOT 的基本观念。第三章、介绍 GNUPLOT 的计算环境。 第四章、设定绘图环境的设定。第五章、介绍绘图指令。第六章、介绍输入及输出部分。第七章、介绍杂项指令。第八章、介绍由 GNUPLOT 所绘出不同类别的例子。第九章、结论。我们期望读者能由此使用手册中了解 GNUPLOT，进一步绘出期望的结果。

Chapter 2. Getting start

2.1 How to get GNUPLOT

GNUPLOT 可在 UNIX、OS/2、MSWINDOW、MSDOS 等作业系统上执行，若你所使用的计算机内没有 gnuplot 此项软体。可透过 anonymous FTP 由下述地方取得：

1 GNUPLOT Source code：
   phi.sinica.edu.tw：/pub/math/gnuplot35/src.zip
   NCTUCCCA.NCTU.edu.tw：/PC/simtel/plot/gpt35exe.zip
   ftp.dartmouth.edu：/pub/gnuplot/gnuplot3.5.tar.Z
2 MS-Windows binaries code：
   phi.sinica.edu.tw：/pub/math/gnuplot35/win.zip
   NCTUCCCA.NCTU.edu.tw：/PC/simtel/plot/gpt35exe.zip
   oak.oakland.edu：/pub/msdos/plot/gpt35exe.zip
3 OS/2 2.x binaries code：
   phi.sinica.edu.tw：/pub/math/gnuplot35/os2.zip
   NCTUCCCA.NCTU.edu.tw：/PC/os2/2_x/unix/gnuplt35.zip
   ftp-os2.nmsu.edu：/os2/2_x/unix/gnuplt35.zip
4 MS-DOS 32bit binariey code：
   phi.sinica.edu.tw：/pub/math/gnuplot35/dos.zip

或者透过邮寄服务(ftp-mail server)取得。方法为送一封信 ftpmail@ftp.dartmouth.edu，信的内容为：

```
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 绘出三角函数 sin(x)与数值资料图形及 sin(x)*cos(y)函数3D 图形的命令。

```
% gnuplot

GNUPLOT
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)
gnuplot> quit
%
```

档案 data.demo 为一数据资料档案，每一行描述一点座标位置。 内容如下：

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

执行 GNUPLOT 程式时，GNUPLOT 首先检查是否设定环境参数 DISPLAY，若有则依其设定。当其确定为 X 环境时，将输出模式设定为 X11。输入

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

产生图1 结果 --- 以线段绘出三角函数 sin(x)；以点绘出档案 data.demo 内各点座标。

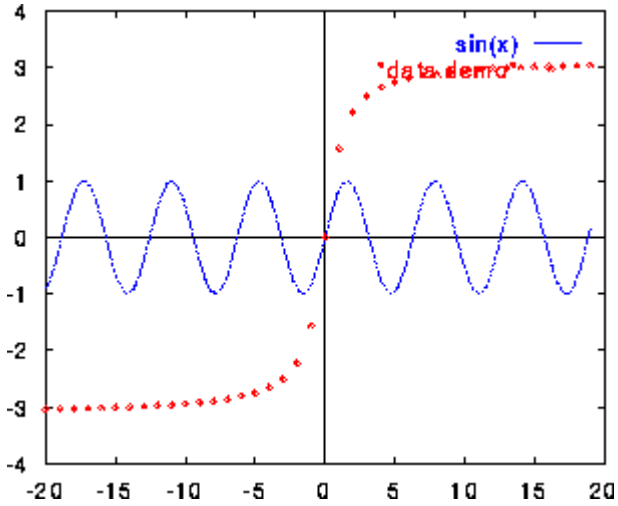


图1： Plotting sin(x), data file - data.demo

接者输入

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

产生图 2 结果 --- 以纵横各 10 条线组成的网格画出 sin(x)*cos(y)的图形。而 quit 指令结束 gnuplot 程式。

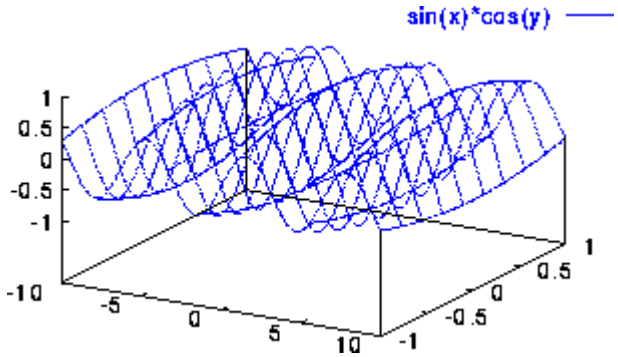


图2： Plotting sin(x)*cos(y)

由此例中，可知 GNUPLOT 具有下列特色 ：

- 不需复杂的设定即可画出平面或立体的数学函数或数值资料图形。
- 自动调整或可由使用者调整 X 轴、Y 轴及 Z 轴的范围和各轴的标点，显示完整地结果。
- 将函数名称自动标示在右上角。
- 以不同的线条颜色或不同的图样明显地区别不同的函数图形。
- GNUPLOT 有许多内建函数，亦可自建函数。

在 X 环境下，GNUPLOT 如同其他 X client 程式一般，可接受许多参数。如

```
gnuplot -font 8x13bold          # 设定字形为 8x13bold。
gnuplot -geometry 900x700       # 设定视窗的长为 900 pixel 宽为 700 pixel。
gnuplot -bg black               # 设定背景颜色为黑色。
```

等。

2.3 Programming style

我们依据一般绘制数学函数或数值资料可能发生的步骤。 提出一份 GNUPLOT 程式撰写程序，如图3。

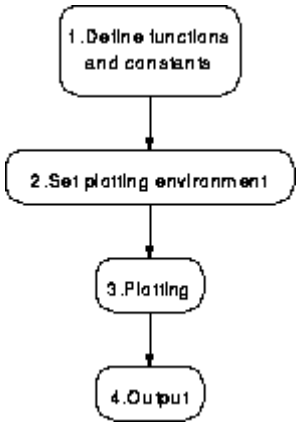


图3： Programming flow chart

- 定义常数及函数：定义常数及函数，使程式模组化及易于了解。
- 设定绘图环境：GNUPLOT 绘图之前已预先设定许多绘图参数(见附录 A)。

这些设定做为 GNUPLOT 绘图时的参考，包括座标轴、文字说明、图形显示范围、3D 图形投影到平面的投影角度、取样点数、绘制图形所用的图案等等的设定。我们可根据个人所需逐一改变绘图参数。GNUPLOT 的参数设定可见附录 A，或在 GNUPLOT 程式内，键入 show all 命令得知目前的绘图环境设定情况。

```
gnuplot> show all
        .
        .
autoscaling is  x:  ON, y:  ON, z:  ON
border is drawn
boxwidth is auto
point clip is OFF
drawing and clipping lines between inrange and outrange points
                .
                .
                .
```

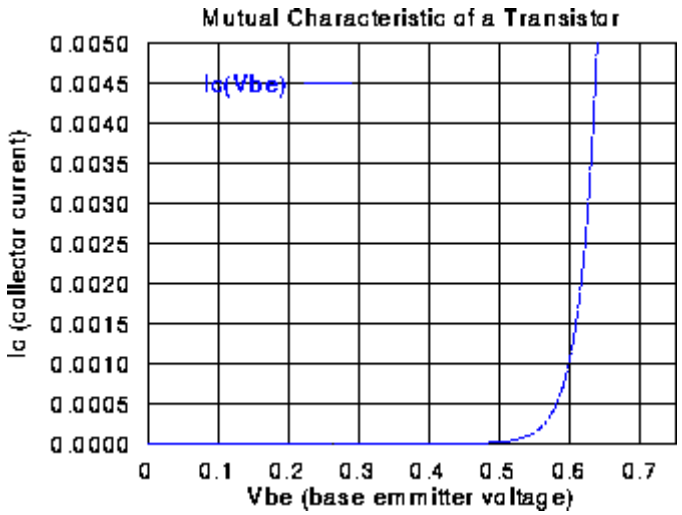
- 绘图：在定义数学函数或设定绘图环境之后，接著是绘出数学函数或数值资 料图形。GNUPLOT 提供操作方便的绘图指令 --- plot(2D) 或 splot(3D)， 具有丰富的参数设定，可由参数的设定做个别调整。
- 产生输出结果：在终端机上绘出结果后，可经由一些步骤而产生不同输出装 置所需的输出语言，得到输出结果。

依照上述过程，可快速地画出图形。剩下的步骤就是细心的调整绘图环境参 数或修改函数方程式、常数等，即可得到满意的结果。

在图4中第1行由 # 符号开始为注解，第2至第7行定义方程式及设定参数，第8行之后是设定绘图的状态，第18行将图画出，其馀是将图以 PostScript 格式存于'npn.ps'档案中。其中设定绘图参数为：

- set dummy Vbe ：gnuplot 一般以 x 为变数，代表横轴上的变数。GNUPLOT 可设定成其它名称以增加可读性。此例设定为 Vbe (表示 base emmitter 之间的电位差)。
- set grid ： 将座标平面绘上格子。
- set title ： 改变 title 名称，将 title 改成 "Mutual Characteristic of a Transistor"。
- set xlable, set ylable ：设定横轴与纵轴名称，将横轴名称定为 "Vbe (base emitter voltage)"，将纵轴名称定为 "Ic(collector current)"。
- set xrange, set yrange ：设定横轴与纵轴显示范围，此调整显示范围至最 想观察的区域。

- set key ： 设定所绘的函数名称所放置的座标位置。




```
# Bipolar Transistor (NPN) Mutual Characteristic
```
comment

```
Ic(Vbe)=Icw*exp(Vbe/kT_q)
Ic(Vbe)=alpha*Ic(Vbe)+Ico
alpha = 0.98
Icw = 4e-14
Ico = 1e-08
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

图4： Example ： NPN Mutual Characteristic

GNUPLOT 指令可分成五部份 ：

- 使用者自定函数、常数 ：在绘图前定义函数或常数，产生绘图所需的函数。
- 设定绘图环境：GNUPLOT 的绘图环境由一组参数构成，由设定参数可改变绘图环境。
- 设定输出环境：GNUPLOT 提供许多种输出装置的输出语言，可由命令设定输出装置。
- 绘图 ：GNUPLOT 提供操作方便的绘图指令 --- plot(splot)。
- 杂项 ：不属于前四类范围的指令，如系统相关指令。

接下来的几章就针对 GNUPLOT 指令加以描述。

Chapter 3. Constants, Operators and Functions

我们在此讨论 GNUPLOT 中数字、常数、运算子及函数。

3.1 Number

GNUPLOT 表示数字可分成整数、实数及复数三类：

- 整数：GNUPLOT 与 C 语言相同，采用4 byte 储存整数。故能表示-2147483647至+2147483647之间的整数。
- 实数：能表示约6或7位的有效位数，指数部份为不大于308的数字。亦即

正数 ： 约 4.19e-307 至 1.67e+307
零 ： 0
负数 ： 约 -1.67e+308 至 -4.19e-307

- 复数 ： 以{<real>,<imag>}表示 --- 复数。其中为此复数的实数部份，<imag>为虚数部份，此两部份均以实数型态表示。 如 3 + 2i 即以{3,2}表示。

GNUPLOT 储存数字的原则为，若能以整数方式储存则以整数储存数字，不然以实数方式储存，其次以复数方式储存。例如在 GNUPLOT 执行

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

分别得到 0 和 1.0 的结果。这是因前者使用整数计算，而后者采用实数计算的结果。执行

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

分别得到 1234.57、12345.6、2.46e+304 和 undefined value 的结果。这些例子是受到实数的有效位数和所能表现最大数字的限制。这是我们要注意的。

3.2 Operators

GNUPLOT 所提供如 C 语言般的运算子(operator)。所有的运算子均可做用在整数、实数或复数上。运算子可分作 Binary Operator 和 Unary Operator，就是分别作用在两个或一个数学式子上，数学式子可为数字或方程式。

Unary Operators

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

表 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

表2： Binary Operators

3.3 Functions

在 GNUPLOT 中函数的参数可以是整数，实数或是复数。表3是 GNUPLOT 所提供的函数。GNUPLOT 画条件函数的方法可由图9中观察， 其采用类似 C 语言的语法，画出函数

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

Function	Auguments	Returns
abs(x)	any	absolute value of x, |x|; same type
abs(x)	complex	length of x, sqrt(real(x)^2 + imag(x)^2)
acos(x)	any	1/cos(x) (inverse cosine) in radians
arg(x)	complex	the phase of x in radians
asin(x)	any	1/sin(x) (inverse sin) in radians
atan(x)	any	1/tan(x) (inverse tangent) in radians
besj0(x)	radians	J0 Bessel function of x
besj1(x)	radians	J1 Bessel function of x
besy0(x)	radians	Y0 Bessel function of x
besy1(x)	radians	Y1 Bessel function of x
ceil(x)	any	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	exponential function of x
floor(x)	any	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	ln(x), natural logarithm (base e) of x
log10(x)	any	log(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

表3：GNUPLOT functions

3.4 User-defined functions and Constants

在 GNUPLOT 中，使用者可自定函数。函数可有1至5个变数。其定义函数的语法如下:

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

而使用者定义常数的语法如下：

<variable-name> = <constant-expression>

下面举一些例子：

```
# 常数 w 为 2。
w = 2
# 常数 q 为小于但最接近 tan(pi/2 - 0.1) 的整数。
q = floor(tan(pi/2 - 0.1))
# 函数 f(x) 为 sin(w*x)，其中 w 为常数。
f(x) = sin(w*x)
# 函数 sinc(x) 为 sin(pi*x)/(pi*x)。
sinc(x) = sin(pi*x)/(pi*x)
# 函数 delta(t) 为脉冲函数。
delta(t) = (t == 0)
# 函数 ramp(t) 当其小于零为零，当其大于零为斜率等于 1 的直线。
ramp(t) = (t > 0) ? t : 0
# 函数 min(a,b) 取两者中较小的数。
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 已定义的常数仅有 **pi** (pi = 3.14159)。

Chapter 4. Plotting enviroment

如第二章所述，只要键入 plot sin(x), 'data.demo' 即可得到图1的结果。GNUPLOT 自动调整 X 轴、Y 轴的显示范围，使图形显示在适当的位置；选择不同的颜色、图形，用以区别不同的函数或数值资料……也就是 GNUPLOT 自动调整其所需的绘图环境。若我们需要一些特别的，如在3D 中加入 contour、设定消去隐藏线、改变 X 轴、Y 轴的座标点名称……等，可由改变绘图参数而改变之。本章说明这些绘图参数设定的方法与功能。

GNUPLOT 的绘图参数可分作下列几项：

名称	说明
座标轴(axis)	改变座标轴的外观，如 linear 或 logscale 等。
标题(label)	设定文字说明或线条，产生时间等。
座标系统	改变座标系统。
图形处理	处理欲绘出的图形，使其易于观察。
3D	调整3D 图形的参数。
输出	设定输出的选。
杂项	不属于上述几项的设定。

我们可用 set 命令去设定这些参数及用 show 命令去察看这些参数。show all 则是察看所有的参数设定。本章各节依序说明绘图参数的设定：

4.1 Axis

绘图参数在设定座标轴方面的参数可分为变数名称、标点、网格、显示范围、 座标轴显示方式与显示与否等六方面的设定：

功能	绘图参数名称
变数名称设定	dummy
标点设定	format, tics, xdtics, xmtics, xtics, ydtics, ymtics, ytics, zdtics, zmtics, ztics, ticlevel
网格设定	grid
座标显示方式	logscale
显示范围设定	autoscale, rrange, trange, urange, vrange, xrange, yrange, zrange
座标轴显示与否	xzeroaxis, yzeroaxis

1　变数名称设定 ：一般以 x 为横轴上的变数。可用 dummy 设定为其它的名称，所绘函数的变数名称亦随之改变。如 set dummy t 将自变数改为 t，图8、图17、图20均改变自变数名称。

2　标点设定 ：设定标点的显示方式与格式。由 format 此项参数设定显示格式，其语法为 ：

```
set format {<axes>} {"<format-string>"}
show format
# 显示各轴标点显示的型式
```

3　其中 axis 为 **x**、**y**、**z**、**xy** 或预设为 **xy**。format-string 为描述每一标点的格式，可接受如 C 语言中 printf 对数字的 f、e、g 三种格式化描述，亦可加入文字(必须少于100字)。以

下举一些例子：

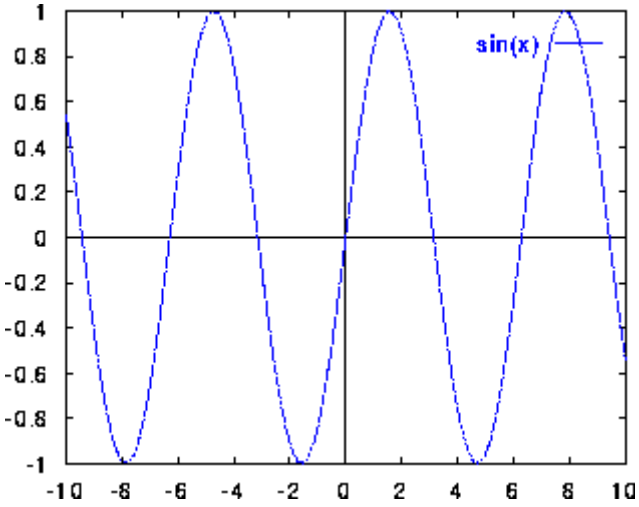
```
set format xy "%.2e"
set format x    "%3.0f cm"
```

4　显示方式由 tics、xtics、... 等设定。

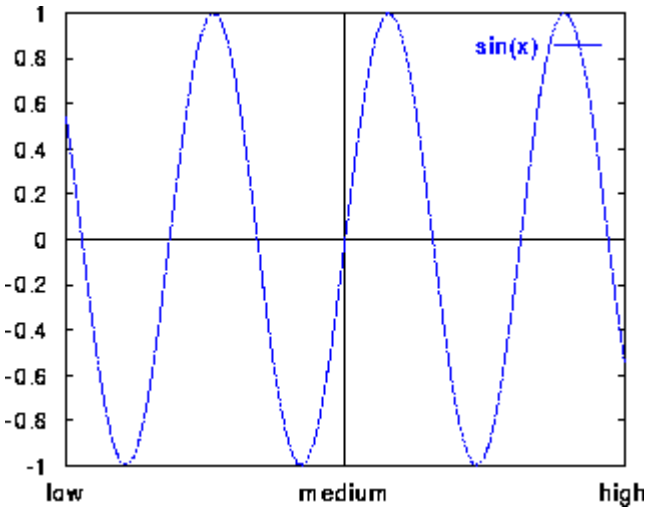
xtics 是对 X 座标轴上的标点做设定。如起始点、结束点、间隔或在轴上特定点放特定的名称。其语法为：

```
set xtics { {<start>, <incr>{, <end>}} |
        {({"<label>"} <pos> {, {"<label>"} <pos>}...)} }
set noxtics     # 不标示任何 X 轴上的标点。
show xtics      # 显示 X 轴标点的状况。
```

图5是三个改变标点的例子。

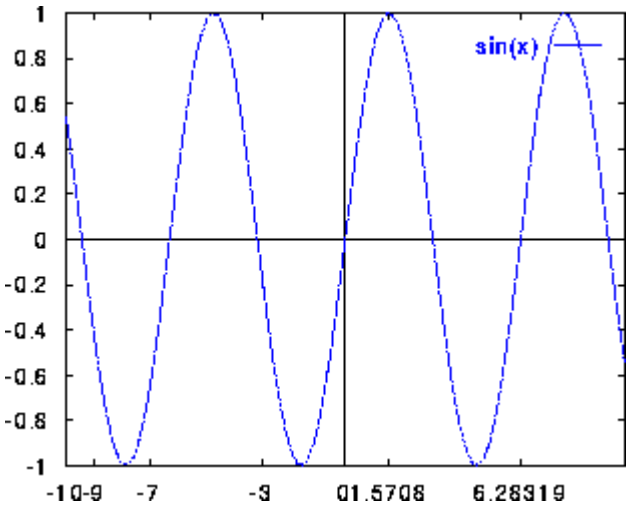



```
# 每隔2格一个标点
set xtics -10,2,10
plot sin(x)
```




```
# 以文字作为标点
set xtics ("low" -10, "medium" 0, "high" 10)
```

```
plot sin(x)
```




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

xmtics 将 X 座标轴上标点名称依0、1、... 改为 Jan、Feb、... Dec 等。大于12的数目除以12取其馀数。图6即是一例。

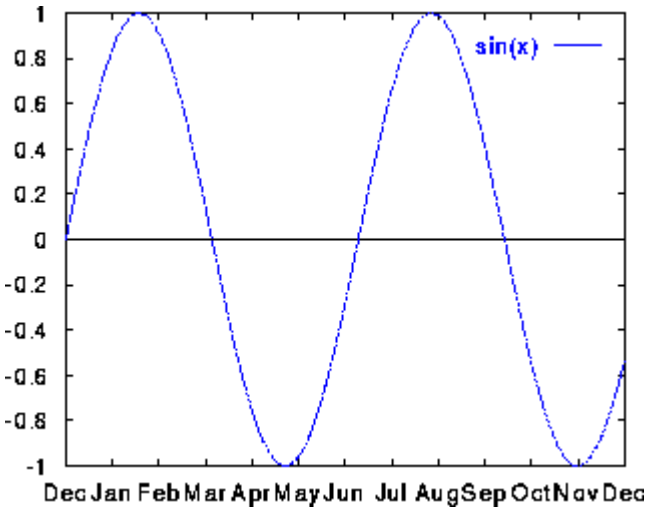


图6： Demonstration of set xmtics

```
# 将标点名称改为 Jan, Feb, ... Dec 等
set xmtics
plot [0 :12] sin(x)
```

xdtics 将 X 座标轴上标点名称依0，1，... 改为 Sun，Mon，... Sat 等。大于7的数目除以7取其馀数。图7即是一例。

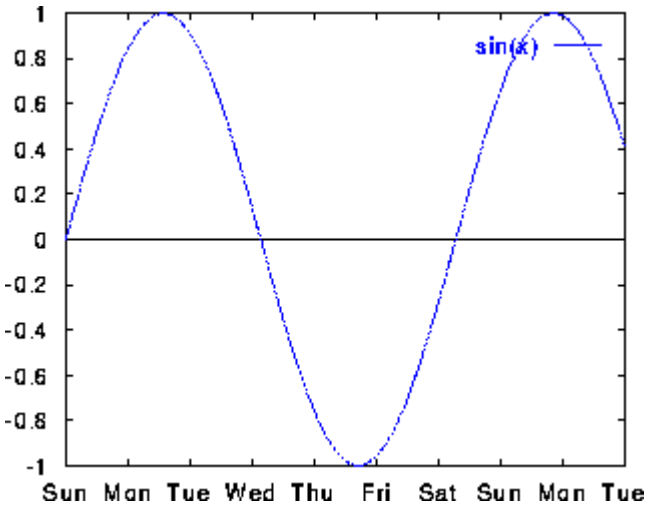


图7： Demonstration of set xdtics

```
# 将标点名称改为 Sun, Mon, ... Sat 等
set xdtics
plot [0 ： 10] sin(x)
```

ytics, ymtics, ydtics, ztics, zmtics, zdtics 与 xtics, xmtics, xdtics 相似，不同点是作用在不同的轴上。

ticslevel 是在画3D 图形时，将标点移到距离 Z 轴上方一段距离的地方，此时图形亦随移动。语法为：

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

1 其中 level 为正数。

2 网格设定 ：在 XY 座标平面上依刻度画上方格子。如图8。

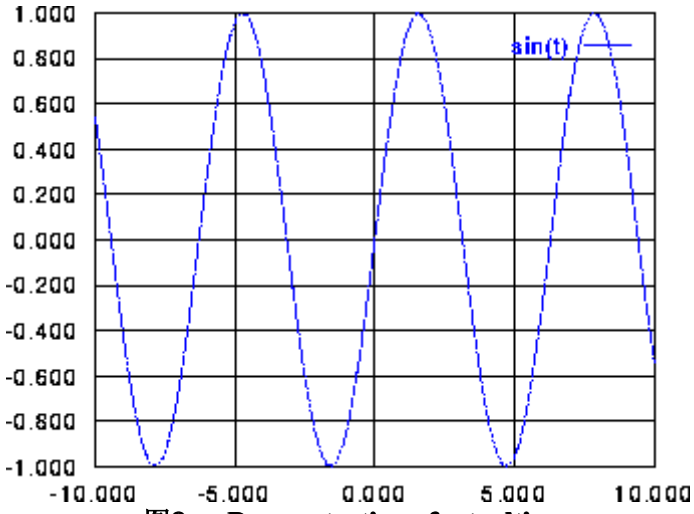


图8： Demonstration of set xdtics

```
# 设定变数为 t
set dummy t
# 设定 X 轴 Y 轴标点的格式
set format xy "%3.2f"
# 产生网格
```

```
set grid
plot   sin(x)
```

座标显示方式：分为线性与对数两种。一般为前者，若要改为对数方式，其语法为：

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

其中 axes 为 X 轴、Y 轴、Z 轴的任意组合。base 预设为10。

5　显示范围设定 ：改变各轴的显示范围。autoscale 参数设定后 GNUPLOT 自动调整显示范围。其馀的如 rrange, trange, xrange, yrange, zrange 则是由使用者设定该轴的范围，以 xrange 为例，其语法为：

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

其中参数<xmin>与<xmax>代表 X 轴的起点与终点，可以是数字或数学式子。如图7中 set [0:10] sin(x)设定 X 轴显示范围为0与10之间。此时可用

```
set xrange [0：10]
plot sin(x)
```

1　代替之。使用 autoscale 参数调整显示范围，其语法为：

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

2　其中<axes>为 GNUPLOT 欲调整的轴，可以是 **x**, **y**, **z** 或 **xy**，预设为所有的轴。

3　座标轴显示与否设定：设定是否要画出座标轴，以 X 轴为例：

```
set xzeroaxis       # 设定显示 X 座标轴
set noxzeroaxis     # 设定不显示 X 座标轴
show xzeroaxis       # 检查 X 座标轴显示与否
```

4.2 Label

GNUPLOT 除了绘出图形外，尚可加入注解做为辅助说明。这注解包括文字与线条两方面，其提供的设定有：

功能	绘图参数名称
线条	arrow
文字注解	key, label, time, title, xlabel, ylabel, zlabel

4　线条：在图上画一线段可以选择有无箭头。其语法为：

```
set arrow {<tag>} {from <sx>,<sy>{,<sz>}}
                  {to <ex>,<ey>{,<ez>}} {{no}head}
set noarrow {<tag>}       # 删除一线条
show arrow                  # 显示线条使用情况
```

其中参数<tag>是给该条线条一个整数名称，若不设定则为最小可用整数。此线条由座

标(sx, sy, sz)到(ex, ey, ez)(在2D 中为 (sx, sy)到(ex, ey))。参数 nohead 为画没有箭头的线段，参数 head 或没有 nohead 为画有箭头的线段。图24中使用没有箭头的线段作为辅助说明。以下为一些例子：

```
# 画一带有箭头的线条 由原点到 (1,2)。
set arrow to 1,2
# 画一名为 3 的带箭头线条 由 (-10,4,2) 到 (-5,5,3)。
set arrow 3 from -10,4,2 to -5,5,3
# 改变名为 3 的线条起始点至 (1,1,1)。
set arrow 3 from 1,1,1
# 删除名为 2 的线条。
set noarrow 2
# 删除所有线条。
set noarrow
# 显示线条的使用情形。
show arrow
```

5 文字注解：分为设定标头(title)，标示(label)与时间(time)三部份。标头设定为在图的正上方加上说明本图的文字。其语法为：

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

设定参数<xoff>或<yoff>为微调标头放置的位址。xlabel, ylabel, zlabel 的语法与 title 相同，其各自描述一座标轴。可见图10的描述。

标示 (label)为在图上任一位置加上文字说明，一般与线条一并使用。其语法为：

```
set label {<tag>} {"<label_text>"}
{at <x>,<y>{,<z>}}{<justification>}
set nolabel {<tag>}          # 删除一标示
show label                   # 显示标示使用情况
```

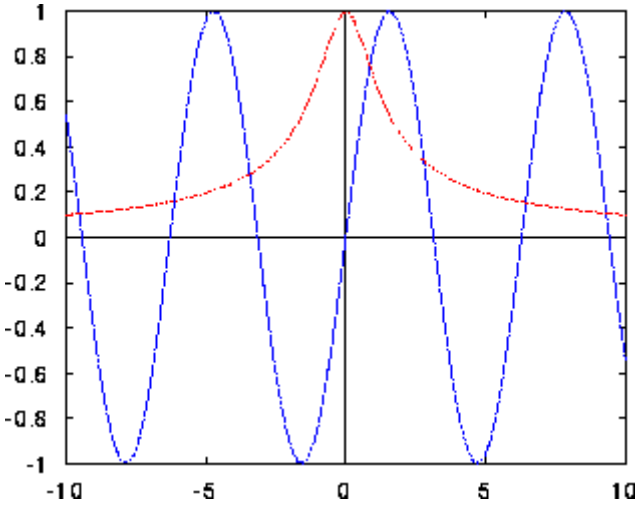
其中参数<tag>与"线条" (arrow)中<tag>意义相同，用以区别不同的 label。参数<justification>是调整文字放置的位置，可以是 **left**, **right** 或 **center**。图24 中使用 label 作为文字辅助说明。举一些例子：

```
# 将 y=x 放在座标 (1,2) 之处。
set label "y=x" at 1,2
# 将 y=x^2 放在座标 (2,3,4) 之处，并命名为 3。
set label 3 "y=x^2" at 2,3,4 right
# 将名为 3 的标示居中放置。
set label 3 center
# 删除名为 2 的标示。
set nolabel 2
# 删除所有标示。
set nolabel
# 显示标示使用情形。
show label
```

一般绘一图形后，GNUPLOT 将函数名称或图形档案名称置于右上角。key 参数设定可改变名称放置位置。其语法为：

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

其中参数<x>, <y>, <z>设定名称放置位置 。set nokey 为不显示名称，若使用 set key 则再度显示名称。若使用 set key 0.2, 0.5则显示函数名称于座标 (0.2, 0.5) 之处。图9表达上述说明。




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

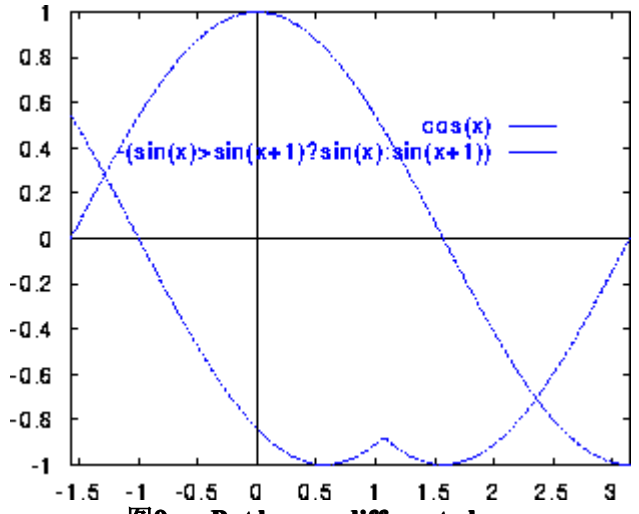


图9： Put key on different place

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

时间参数设定是将图产生的时间标在图上。其语法为

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

show time

设定参数<xoff>或<yoff>为微调时间放置的位址，正数表示向上或向右，负数为反方向，以字的长宽作为单位。图10 为一例子。

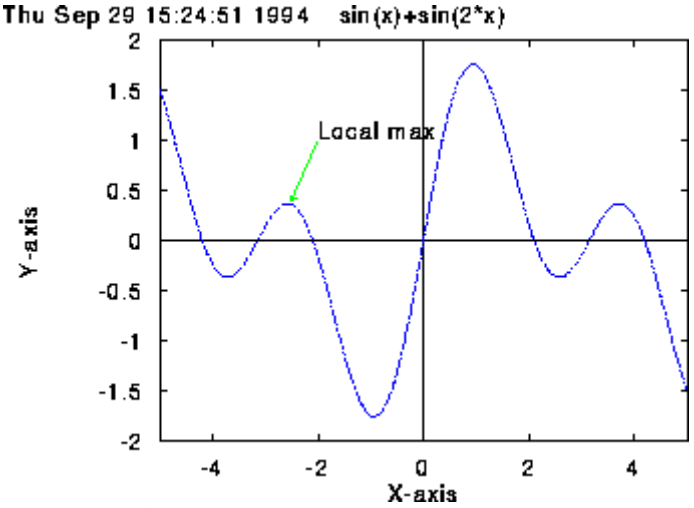


图10： Demonstration of set label

```
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)
```

4.3 Coordinate system

GNUPLOT 除提供绘图在 Cartesian coordinate 外，尚提供 Polar coordinate 的绘图环境。并具有将平面图形投影到非平面图形的功能。

绘图参数名称	说明
angles	设定角度单位为径度量 (radian) 或度度量 (degree)。
mapping	画资料时，设定其采用的座标系统。此有三种选择 cartesian，spherical，cylindrical。
polar	设定为极座标系统。
parametric	所绘图形由一般函数改成参数函数描述。

1 angles ： 设定角度单位为径度量或度度量。其语法为：

```
set angles degrees   # 采用径度量
set angles radians   # 采用度度量
show angles          # 显示目前使用状况
```

2 mapping ：GNUPLOT 读入数据而绘出平面图形，可藉由 set mapping spherical 或 set mapping cylindrical 而将图案投影至球面或圆柱上，使图形在不同曲面中展现出不同特色。其投影方式是将资料数据经由一组函数转换后再描绘于3D 空间上。此函数为

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

此时每一笔数据为(theta,phi)。使用 set angle degrees 改成度度量，设定3D 显示范围在[-pi/2：pi/2]和[0：2*pi]之间。第三部份是将地图绘于圆柱上。其转换函数为

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

此时每一笔数据为(phi,z)。此投影仅对描绘在3D 平面上的数据资料图形有用。图23为一典型例子，其语法为：

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

3　polar ： 设定为极座标系统。此时变数为 x 代表角度，采用径度量或度度量，由 angles 此项参数决定。图22为一典型例子。语法为：

```
set polar       # 采用极座标系统绘图
set nopolar     # 不采用极座标系统绘图 (恢复成 Cartesian coordinate)
show polar      # 显示目前状况
```

parametric ： 输入 set parametric 后，以参数式描述图形。画平面图形时以 t 为变数名称，以 x=f(t), y=g(t)描述图形。画立体图形时以 u,v 为变数名称，以 x=f(u,v), y=g(u,v), z=h(u,v)描述图形。图 21、图 30、图 31、图 32 为典型例子。

4.4 Graphics process

GNUPLOT 使用下列绘图参数改变绘出图形的外貌：

功能	绘图参数名称
图形修正设定	clip
图样设定	data style, function style, boxwidth
取样点数设定	samples
图形大小设定	size

1. 图形修正设定 ：GNUPLOT 提供 clip 作为修正图形设定其语法为 ：

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

提供三种 clip-type 分别为 points, one, two 其意义如下：

```
# 将太接近的点，只用一点代替。
set clip points
# 当线段的端点出现在显示区域内时，才画此线段，否则不画。
set clip one
# 将出现在显示区域内的任何线段或部份线段绘出。
set clip two
```

set noclip<clip-type>是取消由 set clip <clip-type>设定的功能。show clip 显示目前状况。

2. 图样设定 ：GNUPLOT 提供设定图样的参数。其语法为：

```
show data style                          # 显示目前绘数值资料的图样
set data style                           # 设定绘数值资料的图样
show function style                      # 显示目前绘函数的图样
set function style <style-choice>   # 设定绘函数的图样
```

<style-choice>可选择的图样有九种，依序为 **lines**, **points**, **linespoints**, **dots**, **steps**, **impulses**, **errorbars**, **boxes**, **boxerrorbars**。图 14 为采用不同图样所绘出的图形。

boxwidth 参数设定以 boxes 为图样时，box 之宽度。其语法为：

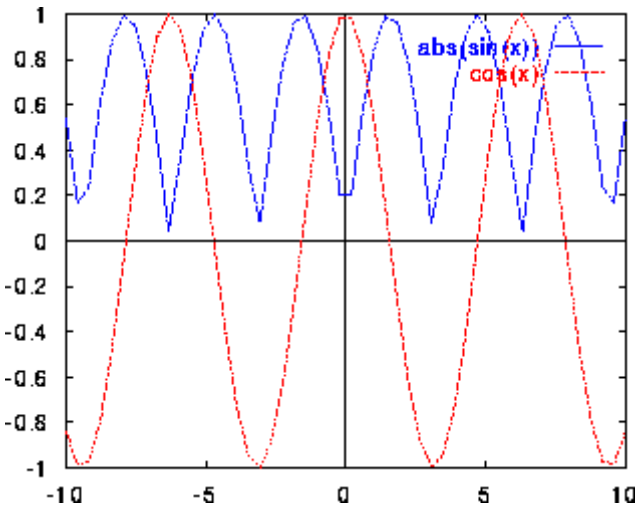
```
set boxwidth {<width>}
show boxwidth
```

参数<width>为设定 box 的宽度。若键入 set boxwidth 时，GNUPLOT 自动调整宽度。

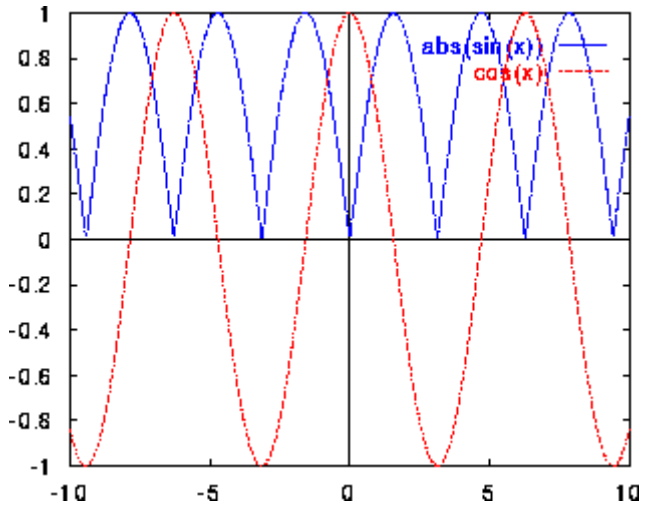
3. 取样点数设定(sampling) ：gnuplot 计算若干点的函数值后，以某种图样将函数值绘上。其语法为：

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

<samples_1>,<samples_2>></samples_1>,<samples_2>> 是分别设定 3D 中 X Y 的取样点数。一般是取样 100 点及采取线条作为图样。可提高取样点数而得到失真较低的图形，图 11 可观察出在图形转角处有明显不同。




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




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

图11： Demonstration of set samples

4. 图形大小设定 ：GNUPLOT 提供改变图形长与宽的比例，是可放大或缩小图形的参数 。其语法为：

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

参数<xscale>,<yscale>为图长与宽的比例，预设值为 1：1。举一些例子：

```
# 将图缩成四分之一大小。
set size 0.5 0.5
# 调整后在 landscape 中有 1：1 的比例。
set size 0.721,1.0
```

4.5 3D

GNUPLOT 在画立体图形方面，提供 contour、消去隐藏线、改变视角等功能。

功能	绘图参数名称
contour	clabel, cntrparam, contour, surface
消隐藏线	hidden3d
取样线数	isosamples
改变视角	view
画格状图	dgrid3d

1. contour ：GNUPLOT 提供画 contour 的参数。

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

其中参数 base 将等高线绘于 XY 平面。参数 surface 将等高线绘于图形曲面上。参数 both 将等高线绘于 XY 平面与图形曲面上。

画等高线时，由 cntrparam 设定 contour 的参数。语法为：

```
set cntrparam { { linear | cubicspline | bspline } |
```

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

参数说明如下：

linear, cubicspline, bspline - 设定 contour 的线条形态。linear 是以直线连接。cubicspline 以片段的圆滑曲线连接。 bspline 以圆滑曲线逼近等高线。

points - 在 cubicspline, bspline 决定采用的点数，数目愈大则愈平滑。

order - 只用在 bspline，此数值可选在 2 至 10 之间，数字愈大，线条愈平滑。

levels - 设定 contour 分成 n 阶(level)，选择 levels 的方法有三种：'auto'，'discrete' 和 'incremental'。其中'auto'是将 z 的最大值与最小值之间平均分成 n 等份。'discrete' 是由其后跟 随的数字设定在 Z=? 时画上 levels。'incremental' 是由 user 设定起始点与每一阶段所欲增加的值。

set clabel 将 contour 所画每一条等高线以不同颜色表示。

有时我们仅对等高线有兴趣可用 surface 参数设定语法为：

```
set surface       # 画出图形曲面与等高线。
set nosurface     # 仅画等高线。
show surface      # 显示状态。
```

2. 消隐藏线 ：GNUPLOT 提供消隐藏线的参数。

```
set hidden3d      # 不画出隐藏的线段。
set nohidden3d    # 画出隐藏的线段。
show hidden3d     # 显示状态。
```

3. 取样线数 ：GNUPLOT 提供画 3D 图形时，决定以多少线条绘出曲面的参数。语法为:

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

参数<iso_1>为绘出曲面的线条数。若设定<iso_1>, <iso_2>则分别是纵与横方向的线条数, 预设值为 10。

4. 改变视角 ：GNUPLOT 提供 3D 图形投影到 2D 平面的视角设定。语法为：

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

参数 <scale>, <scale_z>是控制图的大小比例。参数<rot_x>, <rot_z>是旋转 X 轴、Z 轴的角度。预设为 set view 60,30,1,1。

5. 画格状图 ：我们很难观察杂乱无序的资料所绘的图形。 GNUPLOT 提供将 3D 中数值资料以格状方式画出的方法。语法为：

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

参数 <row\size>, <col\size>设定以纵横各几格组成此网格。它以距离为反比地加权所有的数值，再求其平均值。图\ref{p18}为典型的例子。

4.6 Output

GNUPLOT 的输出参数设定有 terminal, output，在第 6 章有详细 的讨论。

1. terminal ：设定 GNUPLOT 的输出装置型态。GNUPLOT 提供许多输出装置。其语法为:

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

参数 <terminal-type> 可为 7、表 8、表 9 中所列出的装置名称。

2. output ：GNUPLOT 将输出送至标准输出。此设定可将输出送至一档案或另一程式的标准输入。语法为：

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

举例如下：

```
# 将输出送至档案中
set output "filename"
```

我们可由键入下列命令将 sin(x)函数图形由 PostScript 印表机印出。

```
# 将输出送至 PostScript 印表机列印
set terminal postscript
set output "sin.ps"
plot sin(x)
set output "|lpr -P postscript sin.ps"      # BSD 系统
set output "|lp   -d postscript sin.ps"      # System-V 系统
```

因此我们可由键入下列命令将 sin(x)函数图形由接在 PC 上的 HP 绘图机绘出图形。

```
# hpgl -- HP 绘图机的描述语言

set terminal hpgl

# 印表机接在 lpt1

set output "lpt1"
plot sin(x)
```

4.7 Miscellaneous

在此讨论不属于前几类的参数设定

1. functions ： 键入 show functions 显示所有使用者定义的函数。
2. variables ： 键入 show variables 显示所有使用者定义的常数。
3. border ： 键入 set border 将图形外围加框。键入 set noborder 除去图形外框。键入 show border 显示目前是否加框。
4. offsets ：在图形四周留下一些空白。其语法为：

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

其中参数<left>, <right>以横轴单位长度为单位长度,参数 <top>, <bottom> 以纵轴单位长度为单位长度。

5. zero ： ; set zero <expression> 设定一数值， 当小于此值时 gnuplot 视为零。

Chapter 5. Plotting

GNUPLOT 以函数或数据资料为绘制图形的来源。此两者皆使用 plot(splot)指令绘图，plot 用于绘制 2 维的函数或数据资料；而 splot 则用以绘制 3 维空间中的曲面。其基本语法为：

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

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

由修改各参数的设定,可产生许多种表达图形的方式。其参数的设定大致分为以下四类：

1. range ： 设定图形所显示的区域。
2. function or datafile ： 设定所绘的函数或存有 数据资料的档案。
3. title ： 设定标题内容。
4. style ： 设定显示图形的颜色、图案等。

以下分别描述之。

5.1 Ranges

设定此次 plot(splot)所显示的范围。在设定绘图环境中亦有 xrange、yrange、 zrange、rrange、trange、urange、vrange 等相似功能的参数可加以设定，但其影响设定后所有的绘图状态。设定 range 的语法为：

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

其中参数<dummy-var>为变数名称(预设为 X Y，可由 set dummy 而更改)，而 min 和 max 则是设定显示范围，均为常数表示式。并非 min 和 max 要同时存在，不设定表示不改变此项数值。一但设定了 range 之后，会自动将 autoscaling 关闭。以下是一些范例：

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

5.2 Function

函数为一数学表示式。可以是 GNUPLOT 提供的函数或是自行定义或合成的函数。若使用参数式 (parametric function)，则可能是一对(使用 plot)或 3 个(使用 splot)数学表示式。要注意的是 dummy variable 必须符合当时所设定的，一般为 X、Y、Z，若为参数式则为 t(2 维) 或 u v(3 维)。以下是一些范例：

```
plot sin(x)     # 描绘数学三角函数 sin(x)。
splot x * y     # 描绘函数 x*y。
```

5.3 Datafile

GNUPLOT 依照档案中的数据，逐点描出图形。4 即是一组数据资料。

```
#
#   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
```

表4： An example of datafile.(I)

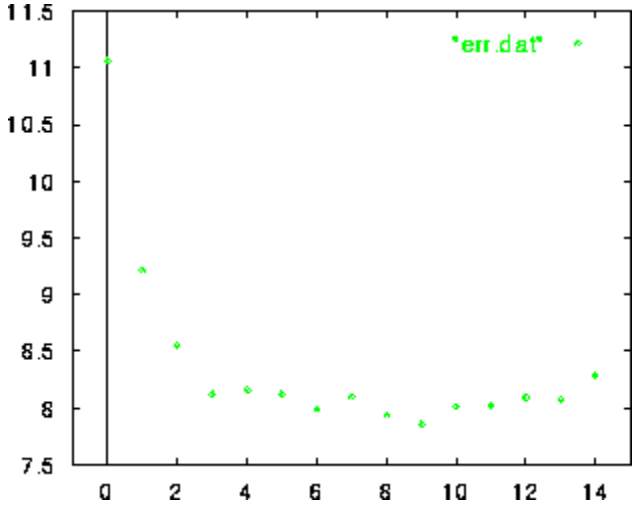
数字之间以空白或 tab 区别。每笔数字为一栏位，每行则由几个栏位构成，储存组数字，描绘数据图形的基本语法为：

```
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>"} } ...
```

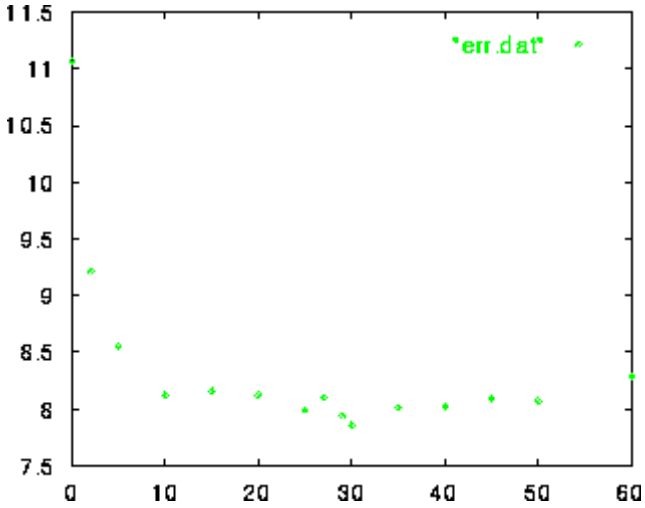
由 plot ″datafile″ with lines 即能产生如图 23 的结果。此例中 GNUPLOT 依序用线条连接档案 ″datafile″内各点数据所定出的座标值。将第一栏位视为 X 座标值，将第二栏位视为 Y 座标值。表 4 为一组资料档案。参数<xcol><ycol>和<zcol>均为在档案中选择适当的栏位分别做为 X、Y、Z 的座标值。如在 plot 语法中第一行仅用<ycol>则以档案中的行数代替 X 座标值绘出图形。第二行则为同时标出 X、Y 座标值。第三、四行为画 errorbars 时使用<ydelta>、<ylow>及<yhigh>。第五行为画 boxes 或 boxerrorbars 时使用，可设定 box

的宽度;此时若以 boxes 为图样(即不必使用<ylow>及<yhigh>),亦要设定<ylow>及<yhigh>。图 13 即是使用表 4 为 datafile 运用不同叙述所绘的结果。

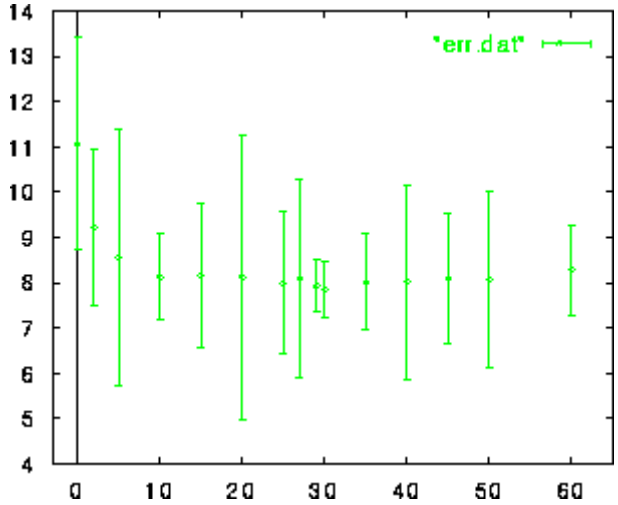


set xrange [-1 ： 15]

plot "datafile" using 2

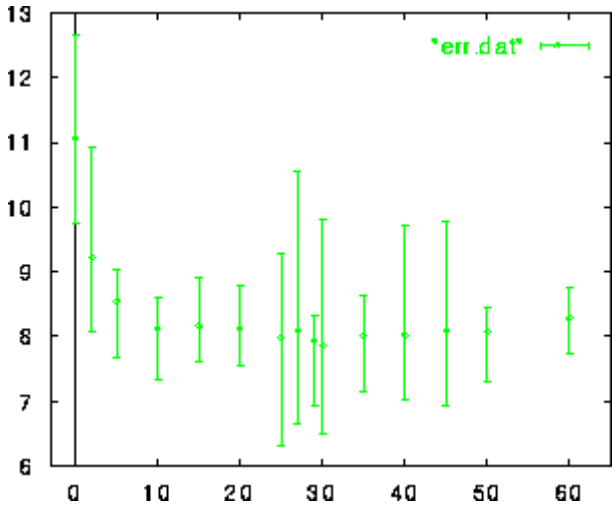


plot "datafile" using 1：2



set xrange [-3 ： 65]

plot "datafile" using 1：2：5 with errorbars



set xrange [-3 ： 65]
plot "datafile" using 1：2：3：4 with errorbars

图 12： Plotting unsorted and sorted data

plot 语法中第六行是采用 C 语言中 scanf 读入浮点数字的语法%f 来表达使用数字的情形。

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

是忽略第一行栏位的数字，采用第二栏位的数字为 X 座标值，再忽略 20 个非换行字元的文字，再将接者的数字做为 Y 座标值。这样就可忽略数字之间的文字说明了。如 6.2.3 节中由设定输出为 table 所产生的资料档案 "datafile"。可用

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

得到"datafile"所描述的图形。使用 %*4[^\n] 忽略前四个字"i x=" %*3[^\n] 忽略 " y="三个字。

储存档案中的数据或许未按大小顺序排列者，此时使用"points"为图样画出数据尚无影响。但若使用"lines"为图样画出数据，则图形显得杂乱。如图 12 左半部，此时可用 unix 指令

sort -n data.old > data.new

将数据依顺序，由小而大排列。则将图 12 左半部改成图 12 右半部。

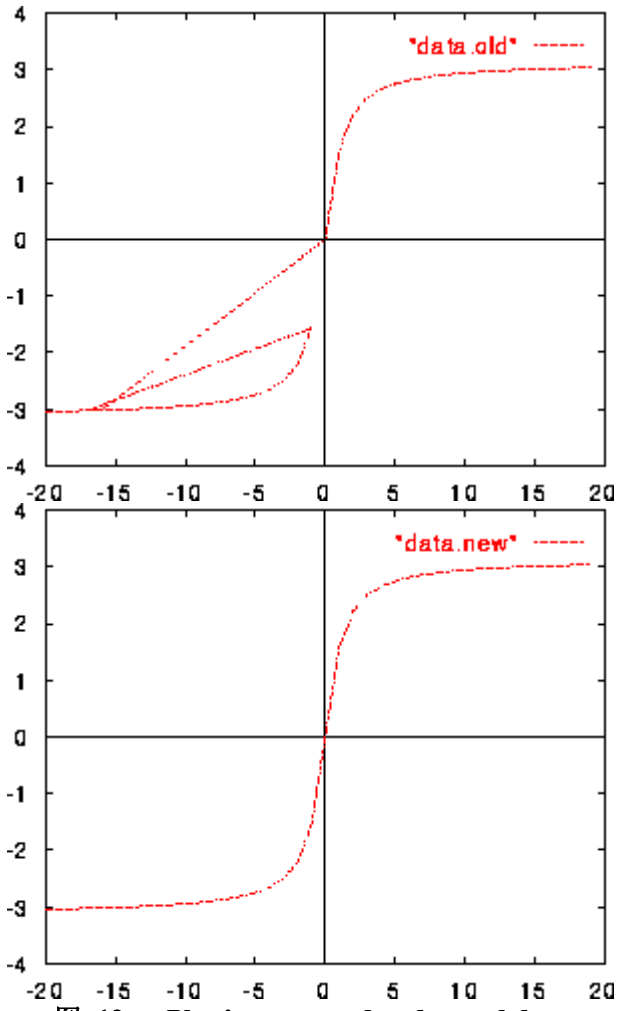


图 12: Plotting unsorted and sorted data

这方法适用于 2 维上的绘图，至于 3 维上绘图 则须设定 dgrid3d 此项参数。

在一档案中亦可包含几个数值资料图形，只要在不同数值资料图形之间以两行空白隔开，就可仅画其中一个数值资料图形(仅能运用于画 3D 图形)。范例如下：

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

datafile 内容如表 5 ，index 0 表示画第一个数值资料图形。此时若要画第三个图形，可使用 index 2。

```
#   index 0
0   11.062    9.757
2    9.212    8.0908
      .
      .
#   index 1
10    8.122   7.327
15    8.156   7.6101
       .
       .
#   index 2
25    7.988   6.3007
```

27 8.096 6.665

.

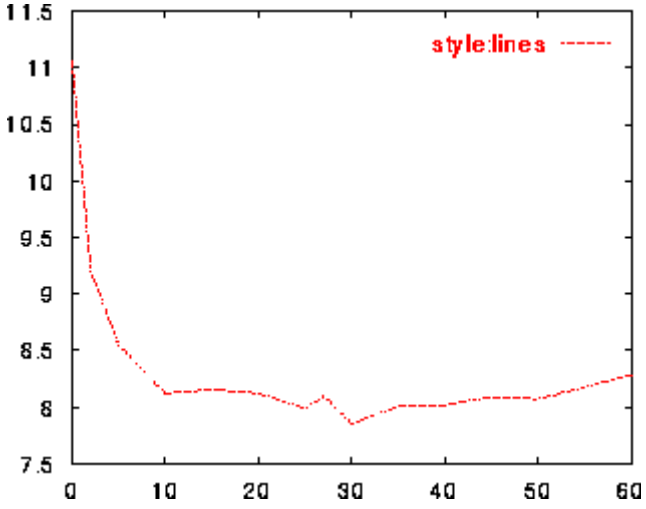
.

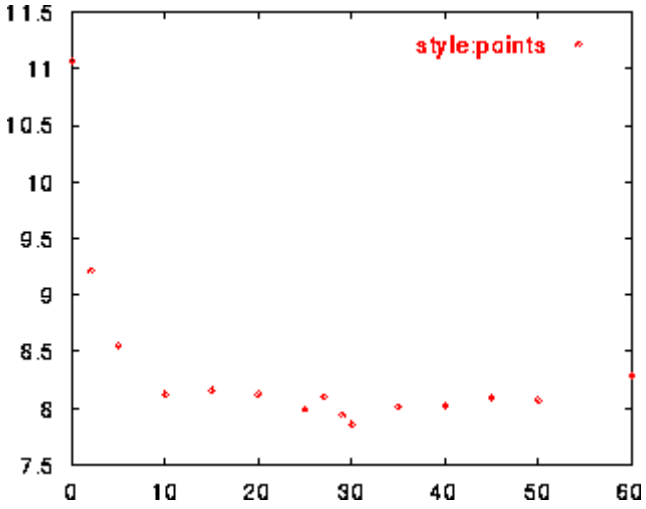
表5： An example of datafile.(II)

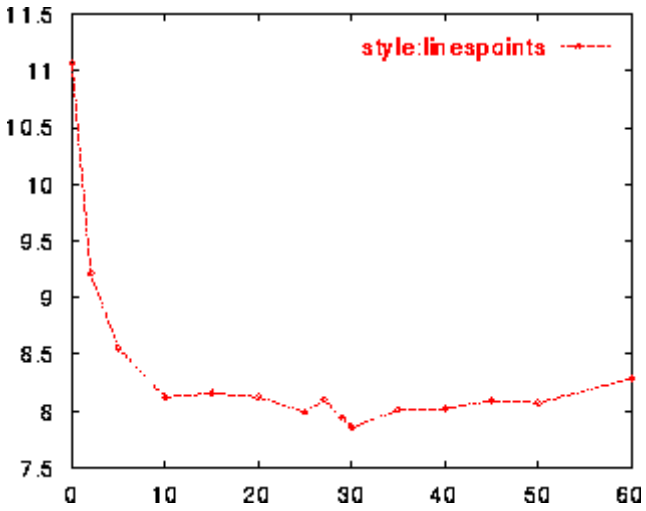
5.4 Style

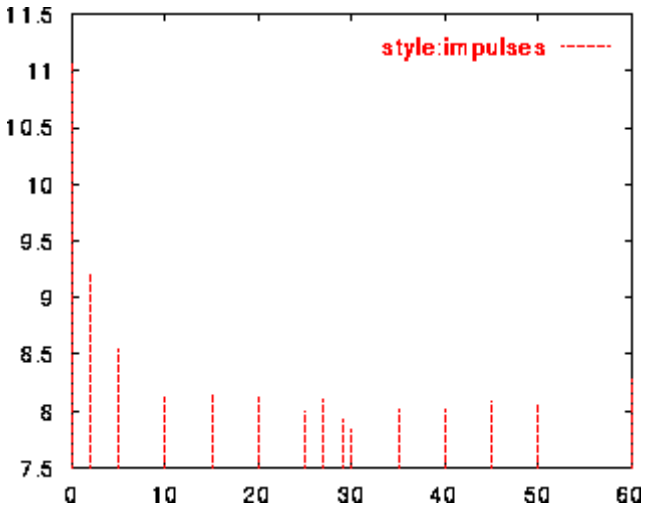
GNUPLOT 描绘数据资料图形是以读入档案中的座标值后，以图样绘上。而描绘函数图形是计算若干点的函数值后，以某种图样将函数值绘上。一般是取样 100 点及采取线条作为图样。GNUPLOT 可提供 9 种图样，分别是

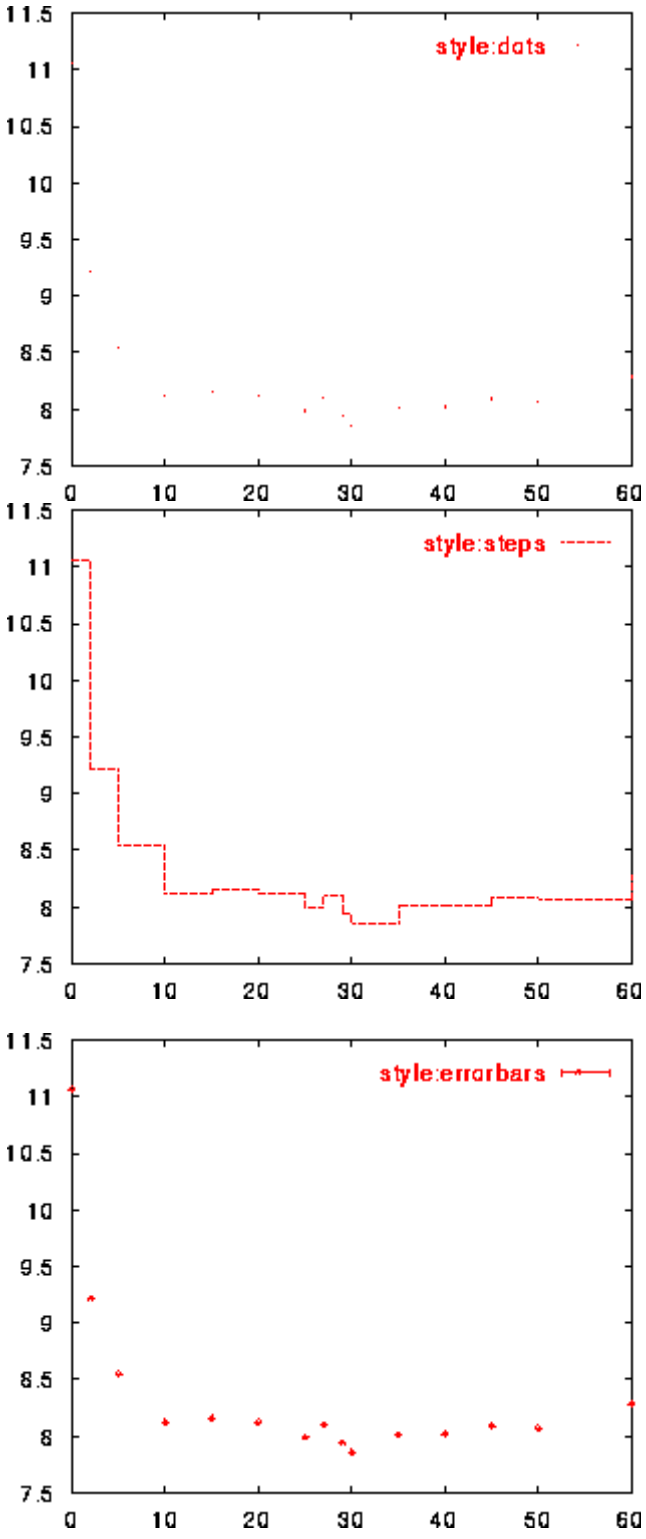
- lines ： 将相邻的点以线条连接。如 plot sin(x) with lines 。
- points ： 将每一点以一符号绘上。如 plot sin(x) with points。
- linespoints ： 同时具有 lines 及 points 的功能。如 plot sin(x) with linespoints。
- impulses ： 将每一点画一垂直线至 X 轴。如 plot sin(x) with impulses 。
- dots ： 将每一点绘一细点。如 plot sin(x) with dots 。
- steps ： 以垂直线及水平线各一条来连接两点，形成梯形。如连接 (x1,y1)，(x2,y2) 两点，以 (x1,y1) 到 (x2,y1) 和 (x2,y1) 到 (x2,y2) 两线段连接。如 plot sin(x) with steps 。
- errorbars ： 对每一点座标值 (x,y)，画一由 (x,ylow) 至 (x,yhigh) 的线段。并在线段两端做上 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 ： 组合 errorbars 与 boxes 两者功能。如 plot sin(x) with boxerrorbars 。




style:points
11.5
11
10.5
10
9.5
9
8.5
8
7.5
0
10
20
30
40
50
60


style:linespoints
11.5
11
10.5
10
9.5
9
8.5
8
7.5
0
10
20
30
40
50
60


style:impulses
11.5
11
10.5
10
9.5
9
8.5
8
7.5
0
10
20
30
40
50
60


style:dots
style:steps
style:errorbars
11.5
11
10.5
10
9.5
9
8.5
8
7.5
0
10
20
30
40
50
60

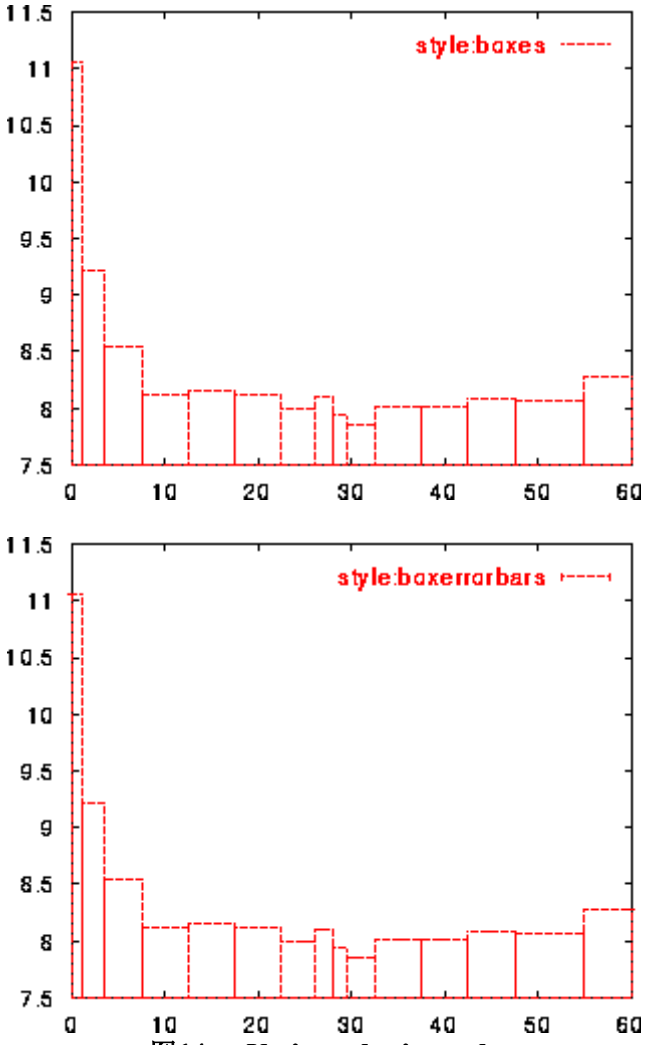


图14： Various plotting style

若改变图样，则可得到不同形式的输出。图 14 以九种图样为范例分别绘出数值资料图形。其中 errorbars， steps， boxes 和 boxerrorbars 仅能在平面(2D)上使用。在设定绘图环境之中，亦可由 set function style 和 set data style 分别设定描绘函数与数据资料的图形，得到相同的结果。

设定图样的语法为

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

其中参数<style>为 lines， points， linespoints， impulses， dots， steps， errorbars 之中一种，参数<linetype>和<pointtype>均为一正数，分别选定绘图颜色及画 points 的符号。GNUPLOT 提供<linetype>和<pointtype>各六种选择，供绘图循环使用。以下举出一些使用范例：

- 使用 impulses 画 sin(x)函数图形。
- plot sin(x) with impulses
- 使用 points 画 x*y 函数图形，使用预设图样画 x^2 +y^2 函数图形。
- splot x*y with points, x**2 + y**2
- 使用相同的线条颜色画 x^2+y^2 及 x^2-y^2 函数图形。
- splot x**2 + y**2 with lines 1, x**2 - y**2 with lines 1
- 使用相同的线条颜色但不同的符号画 sin(x), cos(x)。
- plot sin(x) with linespoints 1 3, cos(x) with linespoints 1 4

5.5 Title

一般使用 GNUPLOT 绘图时，会将函数名称或存有数据资料的档案名称设定为 title。我们可用

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

改变 title。如图 14 中即是运用

```
plot "datafile" title "style ： lines" with lines
```

将名称由描述档案名称改为描述图案名称。我们亦可使用"notitle"使 title 不出现。如

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

不将函数名称"1"显示在图形上，而得到 y=1 直线，而这条直线可作为辅助线。

Chapter 6. Input, Output

GNUPLOT 描绘数学函数或数值资料图形，所以 GNUPLOT 除接受绘图、定义常数函数等命令外，尚要能读入数值资料。数值资料通常由程式产生，如表 4。GNUPLOT 须用指令 plot 或 splot 由档案内读入数值资料。在输出方面 GNUPLOT 除在终端机上显示绘图结果外，尚可输出至许多种输出装置上。在此介绍 GNUPLOT 读入、写出方面的指令。

6.1 Input

在 GNUPLOT 中输入命令的方式有两种：

1. 执行 gnuplot 后，在 gnuplot>提示符号下逐行输入命令或是执行 load "work.gnu" 命令。此时，GNUPLOT 执行档案 work.gnu 中的命令。
2. 在系统提示符号下执行

```
% gnuplot work.gnu
```

GNUPLOT 执行 work.gnu 档案内的命令。

GNUPLOT 不论绘数学函数或数值资料皆使用 plot(splot)指令，此指令在第 5 章有详细的介绍。在此介绍 GNUPLOT 读入的数值资料格式及"load"与 "reread" --- 可读入含有 GNUPLOT 命令的档案。

6.1.1 Read data from file

GNUPLOT 使用 plot(2D)或 splot(3D)指令读入数值资料档及绘出图形，详细的描述见 5.3 节。数值资料的格式为每行一组资料，若该行开头为#表示该行为注解；若开头为空白，则忽略这些空白。一组资料中的数字以空格或 tab 分开，数字以整数、浮点数字或科学记号来表示。下表即是数值资料档"data.gnuplot"的范例。

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

An example of datafile.(III)

输入

```
plot "data.gnuplot"   #画出 2D 的图形
```

```
plot "data.gnuplot" using 1:3:4 with errorbars #画出 2Derrorbar 图形
```

6.1.2 Read commands from file

当使用 GNUPLOT 进行繁杂的绘图工作时，需要许多命令来完成工作。这些命令可储存在档案内，再经由"load"指令将命令由档案内读出及执行。这些包含常数、函数的定义及绘图环境的设定。load 可读由 save 命令所储存的档案。load 语法为：

```
load "<input-file>"    # 使用 "(双引号) 或 '(单引号) 皆可
```

例子如下：

```
load "work.gnu"        # 读入 work.gnu 档案中的指令。
```

work.gnu 档案内容为

```
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 中执行 load "work.gnu"之后，绘出图 15 的结果。使用 load 指令执行 gnuplot 指令可不用每次皆逐一设定绘图环境与函数常数设定等，亦可将此次结果储存起来留待下次使用。

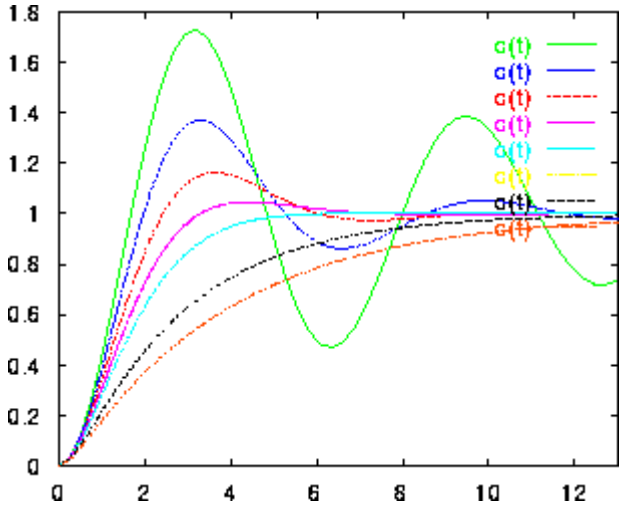


图 15： Demonstration of read commands from file

通常 GNUPLOT 完成一件绘图工作的过程中，须设定许多绘图环境。若紧接著画另一张图时，往往需调整许多绘图环境参数。此时可读一 GNUPLOT 原始设定的档案，使 GNUPLOT 的设定恢复成起始状态，就可不必逐一调整绘图环境。例如将 GNUPLOT 的起始状态存于 environment.gnu

Chapter 7. Miscellaneous commands

GNUPLOT 除了提供绘制图形所需的指令外，尚提供许多有用的指令。如

- 与系统有关的指令 ： cd, pwd, shell, start-up。
- 注解 ： comment。
- 与编辑有关的指令 ： clear, help, print, pause, command line-editing。
- 离开 GNUPLOT ： quit, exit。

7.1 Instructions

1. cd ：改变目前的工作目录。其语法为

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

例如

```
cd 'subdir'   # 改变至子目录。
cd ".."       # 改变至上一层目录。
```

2. pwd：执行 pwd 显示目前的工作目录于萤幕上。

3. shell：执行 shell 命令。执行 shell 后，回到 shell 的提示符号下。欲回到 GNUPLOT 时，输入 exit 或 Ctl-d 即可。如果只是执一个 shell 命令，只要在 shell 命令之前加!即可。 如执行!pwd 显示目前的工作目录。

4. start-up： GNUPLOT 执行时，先捡查是否有.gnuplot 档案。若有则执行档案内的 GNUPLOT 命令。因此可适当加入一些命令于.gnuplot 内，成为个人化的 GNUPLOT。

7.2 Comment

在撰写 GNUPLOT 命令时，可适当加入注解以帮助我们了解。方法是在欲加注解之处，先写#则 GNUPLOT 忽略其后的文字。但若#是在引号之内，此时为一单纯字元。

7.3 Working window

GNUPLOT 提供方便使用者操作的指令。

1. clear ： 输入 clear 时，将萤幕及输出清除。

2. help ： GNUPLOT 提供线上辅助说明的功能。由此可得较多的说明。其语法为 help {<topic>} 。

3. print ： print <expression>将数学式的值印出。

4. pause ： GNUPLOT 提供暂停萤幕一段时间并显示一段文字的命令。其语法为

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

其中<time>可以是一整数或表示式，表示 GNUPLOT 停留的时间。 若为 -1 则 GNUPLOT 停留至按下"carriage return"为止。若为 0 则 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 储存以往输入的命令，让使用者可重复执行，或经修改后再使用。且不论游标在何处皆可按下"return key"而输入整行命令。这些编辑命令为：

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

不论 exit、quit 或 END-OF-FILE 字元，均可使 GNUPLOT 结束执行。

Chapter 8. Examples

在前面的章节中曾提出 GNUPLOT 绘制图形的步骤，也针对各步骤做详细的说明。 在此介绍自 GNUPLOT demo 中整理的范例，这些范例均在数行间展现 GNUPLOT 具体的功能。这些范例亦提供设计 GNUPLOT 程式时良好的参考，可酌量修改部份程式以符合个别需求。下表为此章中各个范例所描述的主题。

8.1 Amplify frequency response 将座标轴改成 log scale 显示图形
8.2 2D data plot 用许多图样绘出数值资料图形
8.3 Error bar 以 error bar 绘制图形
8.4 User-defined variable 改变函数中的参数而绘出函数的变化情况
8.5 2D parametric function 用参数式绘平面图形
8.6 Polar coordinates 在极座标系统中绘图
8.7 Mapping 将平面图形投影至曲面上
8.8 3d plotting 绘制立体图形
8.9 contour 绘制 contour 图形
8.10 random number 绘制乱数图形
8.11 scatter data -> grid data 在 3D 中以格子状的网线绘没有秩序的资料图形
8.12 3d hidden 绘制消去隐藏线的 3D 图形
8.13 3d parametric function 用参数式绘立体图形
8.14 3D data plotting 在 3D 中的绘数值资料图形

Chapter 9. Conclusion

在前面章节中，先介绍 GNUPLOT 绘图的方法与步骤。接者由许多例子展现 GNUPLOT 不同方面的特性及程式的整体性。

GNUPLOT 是一个资料型态转换的处理程式，它将数学函数或数值资料转换成平面或立体的图形资料，以图形方式对资料做整体性的观察及探讨，是一非常有用的工具。

Appendix A. The initial setting of GNUPLOT

此附录为 GNUPLOT 在 UNIX 和 X 环境下执行的初始状态。

```
set terminal x11        # 设定输出装置为 X 终端机
set output              # 将输出送至标准输出
set noclip points       # 将接近的点依然画出
set clip one        # 当线段的端点出现在显示区域内时，才画此线段，否则不画
set noclip two          # 不将出现在显示区域内的部份线段绘出
set border                  # 将图形外围加框
set boxwidth
set dummy x,y               # 设定自变数名称为 x,y
set format x "%g"          # 设定 X 轴标点显示方式为 C 语言中%g 格式
set format y "%g"          # 设定 Y 轴标点显示方式为 C 语言中%g 格式
set format z "%g"          # 设定 Z 轴标点显示方式为 C 语言中%g 格式
set nogrid                  # 不产生网格
set key                     # 将函数名称或档案名称置于右上角
set nolabel                # 取消图上任何文字说明
set noarrow                 # 取消图上任何线段
set nologscale             # 采取线性方式显示图形
set offsets 0, 0, 0, 0   # 不在图四周留下空白
set nopolar                 # 采用 Cartesian coordinate
set angles radians         # 以径度量表示角度，范围[0：2*pi]
set noparametric           # 不采用参数式
set view 60, 30, 1, 1    # 旋转 X 轴 60 度 Z 轴 30 度
set samples 100, 100      # 设定 X 轴 (Y 轴) 的取样点数为 100 点
set isosamples 10, 10    # 在 3D 中，采用纵横各 10 条线构成图形曲面
set surface                # 在 3D 中，画出曲面
set nocontour              # 不画出 contour
set clabel                 # 设定将 contour 所画每一条等高线以不同颜色表示
set nohidden3d              # 不用消去隐藏线的功能
set cntrparam order 4
set cntrparam linear
set cntrparam levels auto 5
set cntrparam points 5
set size 1,1               # 设定图形长宽比例为 1：1
set data style points    # 以 points 作为绘数值资料的图案
set function style lines   # 以 lines 作为绘函数的图案
set xzeroaxis              # 设定显示 X 轴
set yzeroaxis              # 设定显示 Y 轴
set tics in               # 设定标点在轴的内面
set ticslevel 0.5
set xtics                  # X 轴标点以数字标示
set ytics                  # Y 轴标点以数字标示
set ztics                  # Z 轴标点以数字标示
set title "" 0,0          # 图的标头不显示
set notime                  # 不显示产生图形的时间
set rrange [-0 ：  10]
set trange [-5 ：  5]      # 在 2D 参数式中以 t 为变数，此为设定显示区间
set urange [-5 ：  5]      # 在 3D 参数式中以 u,v 为变数，此为设定显示区间
set vrange [-5 ：  5]      # 在 3D 参数式中以 u,v 为变数，此为设定显示区间
set xlabel "" 0,0         # X 轴标头不显示
set xrange [-10 ：  10]    # 设定 X 轴显示区间-10,10 之间
set ylabel "" 0,0         # Y 轴标头不显示
set yrange [-10 ：  10]    # 设定 Y 轴显示区间-10,10 之间
set zlabel "" 0,0         # Z 轴标头不显示
set zrange [-10 ：  10]    # 设定 Z 轴显示区间-10,10 之间
set autoscale r
set autoscale t            # 自动调整 t 的显示区间使图可全部显示
set autoscale xy           # 自动调整 X 轴 Y 轴显示区间使图可全部显示
set autoscale z            # 自动调整 Z 轴显示区间使图可全部显示
set zero 1e-08             # 设定比 1e-08 小的正数为零
```

Appendix B. Call gnuplot from C program

我们可在 C 程式中呼叫 gnuplot 替我们绘图。方法为产生 process 执行 gnuplot，建立对 gnuplot 下达命令的管道，使用档案传递资料。以下是在 UNIX 中呼叫 gnuplot 的例子：(程式码可由此获得)

```
 1  #include <sys/types.h>
 2  #include <sys/stat.h>
 3  #include <stdlib.h>
 4  #include <stdio.h>
 5  #include <math.h>
 6  #include <unistd.h>
 7
 8  #define PANIC(a) do { \
 9                       perror(a); \
10                       if (temp_name) unlink(temp_name);\
11                       exit(1);\
12              } while(0)
13
14  int main() {
15      FILE *command,*data;
16      char *temp_name;
17      double a,b;
18      int i;
19
20      if ((temp_name = tmpnam((char *) 0)) == 0) PANIC("tmpnam
failed");
21      if(mkfifo(temp_name, S_IRUSR | S_IWUSR) != 0) PANIC("mkfifo
failed");
22      command = popen("gnuplot","w");
23      fprintf(command,"plot \"%s\" with lines\n",temp_name);
fflush(command);
24      data = fopen(temp_name,"w");
25      for (i=0; i<20; i++) {
26          a = i/10.0;
27          b = sin(a);
28          fprintf(data,"%f %f\n",a,b);
29      }
30      fclose(data);
31      fprintf(stderr,"press enter to continue..."); fflush(stderr);
32      getchar();
33
34      fprintf(command,"plot \"%s\" with lines\n",temp_name);
fflush(command);
35      data = fopen(temp_name,"w");
36      for (i=0; i<20; i++) {
37          a = i/10.0;
38          b = cos(a);
39          fprintf(data,"%f %f\n",a,b);
40      }
41      fclose(data);
42      fprintf(stderr,"press enter to continue..."); fflush(stderr);
43      getchar();
44      fclose(command);
45      unlink(temp_name);
46      return 0;
```

```
47    }
```

程式在第 22 行使用函数 popen()执行 gnuplot 并建立对 gnuplot 下命令的通道。(关于函数 popen()的说明，可参考 W.Richard Stevens,Advanced Programming in the UNIX Environment. Addison-Wesley, pp 435-441, 1992)程式在 23 行使用

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

下达命令给 gnuplot。在 21 行使用函数 mkfifo()建立储存资料的档案。(关于函数 mkfifo()的说明，可参考 W. Richard Stevens, Advanced Programming in the UNIX Environment. Addison-Wesley, pp 445-449, 1992.)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);
}
```

可在档案中产生一组描述数学三角函数 cos(x)的数值资料。因此程式可对 gnuplot 下命令并将数值资料准备在档案内，再由 gnuplot 读取。
下面是 OS/2 中 C 程式呼叫 gnuplot 的例子(使用 GCC compilier)：(程式码可由此获得)

```
1   #include <stdio.h>
2   #define INCL_DOS
3   #define INCL_DOSPROCESS
4   #define INCL_DOSNMPIPES
5   #include <os2.h>
6
7   main()
8         {
9         HPIPE hpipe ;
10        FILE *hfile, *hgnu ;
11              /* create a named pipe. Use NP_WAIT so that DosConnect...
12                 blocks until client (gnuplot) opens, and client reads
13                 are blocked until data is available */
14        DosCreateNPipe( "\\pipe\\gtemp",
15                              &hpipe,
16                              NP_ACCESS_OUTBOUND,
17                              NP_WAIT|NP_TYPE_BYTE|1,
18                              256,
19                              256,
20                              -1 ) ;
21              /* use stream i/o */
22        hfile = fdopen( hpipe, "w" ) ;
23
24              /* start gnuplot; use unbuffered writes so we don't need to
25                 flush buffer after a command */
26        hgnu = popen( "gnuplot", "w" ) ;
27        setvbuf( hgnu, NULL, _IONBF, 0 ) ;
28
29              /* plot a set of data */
30
31        fprintf( hgnu, "plot '/pipe/gtemp'\n" ) ;    /* issue plot command */
32        DosConnectNPipe( hpipe ) ;                        /* wait until 'file' opened */
33        fprintf( hfile, "1 1\n" ) ;                  /* write data to 'file' */
34        fprintf( hfile, "2 2\n" ) ;
```

```
35          fprintf( hfile, "3 3\n" ) ;
36          fprintf( hfile, "4 4\n" ) ;
37          fflush( hfile ) ;                          /* flush buffer forces read */
38          DosSleep( 500 ) ;
39          DosDisConnectNPipe( hpipe ) ;              /* disconnect this session */
40          fprintf( hgnu, "pause -1\n" ) ;          /* admire plot */
41
42               /* plot another set of data */
43
44          fprintf( hgnu, "plot '/pipe/gtemp'\n" ) ;
45          DosConnectNPipe( hpipe ) ;
46          fprintf( hfile, "1 4\n" ) ;
47          fprintf( hfile, "2 3\n" ) ;
48          fprintf( hfile, "3 2\n" ) ;
49          fprintf( hfile, "4 1\n" ) ;
50          fflush( hfile ) ;
51          DosSleep( 500 ) ;
52          DosDisConnectNPipe( hpipe ) ;
53          fprintf( hgnu, "pause -1\n" ) ;
54
55          DosClose( hpipe ) ;
56          pclose( hgnu ) ;
57          }
```

Appendix C. Using gnuplot commands in shell script

我们也可 gnuplot 命令放入 shell script 中执行。举例如下：

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

执行此 shell script 时，gnuplot 用线条画出 Mydata 档案中数值资料的图形。在第三行中的 gnuplot << EOF 为执行 gnuplot 并将其下的数行当做 gnuplot 的输入。EOF 表示输入资料的起始及结束点，所以第四至第六行为 gnuplot 的输入资料。gnuplot 先设定输出装置为 tex40xx，再画出变数 data1 所指定档案的内容，然后离开 gnuplot。
