gunplotによるグラフ作成

山本昌志*

2006年2月13日

1　本日の学習内容

本日は教科書を離れて，グラフの作成方法を学習する．技術者や科学技術の研究者は，情報を目に見える形で示すためにグラフを多用する．現象の分析や，その現象が物語っていることを伝えるためにグラフが使う．また，諸君は今後の学習で大量のグラフを書かなくてはならない．実験のデータ整理もあるが，数学や専門科目の理解を深めるためにはグラフを書き，現象を視覚でとらえることが重要となる．そうしないと，なかなか自然現象の理解は深まらない．そこで，本日はコンピューターを使ったグラフの作成方法を学習する．

ここでのグラフの作成には，世界中で最も使われている **gnuplot** というソフトウェアーを使う．本日の学習内容は，以下の通りである．

- gnuplot を使ったグラフ作成方法
- C 言語を通しての gnuplot の操作方法

他にも，グラフ作成ソフトウェアーはたくさんあるし，グラフィックライブラリーを使うこともできる．興味のある者は調べると面白いだろう．

2　グラフ作成

2.1　gnuplot とは

gnuplot は簡単に 2D，3D のグラフが作成できるフリーのソフトウェーである．単純なグラフから，学術論文用の高品質なグラフまで作成可能で，広く普及している．名前に gnu とついているが，まぎらわしいことに，Free Software Foundation (FSF) が進めている GNU プロジェクト[1]とは関係が無い．gunplot の読み方は「ニュープロット (あるいは，ヌープロット)」ではあるが,「グニュープロット」と呼ばれることも多い．

gnuplot は UNIX に限らず Windows や Machintosh でも動作し，Excel とくらべものにならないくらい美しいグラフを作成することができる．しかも，フリーである．諸君は実験実習のグラフ作成に使うのが良

*独立行政法人　秋田工業高等専門学校　電気情報工学科

[1]Unix に似た フリーソフトウェアの完全なオペレーティングシステムの作成を目指す.

いだろう．実験実習程度のグラフなら Excel でも作成可能ではあるが，そのグラフは醜く，とても科学技術の報告書や論文に載せるクオリティに達していない．gnuplot は，C 言語などプログラミング言語と組み合わせて使うと，様々な処理が自動的にできる．そのようなことから，私はコンピュータープログラムによる数値計算の結果の表示に使っている．

マニュアル類は web にたくさんある．情報が必要になれば，以下のサイトを調べるのが良いだろう．

http://t16web.lanl.gov/Kawano/gnuplot/

http://lagendra.s.kanazawa-u.ac.jp/ogurisu/manuals/gnuplot-intro/

2.2 起動と終了

簡単な操作方法を述べるが，本当は，先ほど示した web ページを見て各自学習する方が良い．まずは，gunplot を立ち上げてみよう．以下のコマンドを端末に入力する．

```
$ gnuplot
```

すると，gnuplot が立ち上がり，コマンド入力画面になる．終了したい場合は，

```
gnuplot> exit
```

とする．

2.3 2 次元グラフ

まずは，三角関数のグラフを書いてみよう．以下のコマンドを入力する．

```
gnuplot> plot sin(x)
```

sin 関数のグラフが描けただろう．描画範囲を変えたい場合は，

```
gnuplot> plot [0:6.28] [-1.5:1.5] sin(x)
```

とする．同時に複数のグラフを各場合は，次のように関数を並べればよい．

```
gnuplot> plot [-6.28:6.28] [-1.5:1.5] sin(x),cos(x),tan(x)
```

次のようにすると様々なグラフが描ける．

`gnuplot> plot x**3+x+1`	$x^3 + x + 1$
`gnuplot> plot x**0.5`	$x^{0.5}$
`gnuplot> plot log(x)`	$\log_e(x)$
`gnuplot> plot log10(x)`	$\log_{10}(x)$
`gnuplot> plot real(exp({0,1}*x))`	$\Re(e^{ix})$
`gnuplot> plot sqrt(x)`	$\sqrt{x}$

さらに，定義した関数のグラフを書くこともできる．

```
gnuplot> f(x)=sin(x)
gnuplot> g(x)=cos(x)
gnuplot> plot f(x)+g(x), f(x)*g(x)
```

練習問題

[練習 1] 以下の関数のグラフを作成せよ.

$$\sin(x)\cos(x) \qquad \sin^2(x) \qquad \sin(x)+\cos(x)$$
$$xe^{-x} \qquad x-\frac{x^3}{6}+\frac{x^5}{120}-\frac{x^7}{5040} \qquad \cos(x),\ \sin(x)$$

gunplot には付録の表 2 のような関数が用意されている．普通に使う関数はほとんど用意されているし，複素数もサポートされている．複素数は，実数部，虚数部のように記述する．すなわち，

`gnuplot> {1,0}` 実数の 1 を表す.
`gnuplot> {0,1}` 虚数単位 i を表す.
`gnuplot> {5.3,6.8}` $5.3+6.8i$ を表す.

である.

2.4 3 次元グラフ

3 次元グラフも簡単にかける．3 次元グラフの場合，右マウスでドラッグすると視点を変えることができるのでおもしろい.

`gnuplot> splot x**2+y**2` x^2+y^2
`gnuplot> splot x*sin(x+y)` $x\sin(x+y)$

3 次元グラフで隠線処理が必要であれば，`set hidden3d` とする．また，表示するデータ点は，`set isosample` で設定する．たとえば，以下のようにすれば，隠線処理し，x 方向と y 方向とも 40 点のデータを出力する.

```
gnuplot> set hidden3d
gnuplot> set isosample 40,40
gnuplot> splot 1/(x*x+y*y+5)*cos(0.1*(x*x+y*y))
```

2.5 媒介変数表示

媒介変数を使ったグラフの作成もできる.

```
gnuplot> set parametric
gnuplot> plot sin(5*t), cos(2*t+pi/2)
```

媒介変数表示を止めるためには,

```
gnuplot> unset parametric
```

とする.

2.6 ファイルのデータの描画

ファイルに格納されたデータをグラフ表示することもできる．講義では，三角関数の値が格納されたデータをダウンロードして，それをグラフにする．表の各行には，θ と $\sin\theta$，$\cos\theta$，$\tan\theta$ の値が書き込まれている．このファイルのデータをグラフ化するときには，plot コマンドを使う．引き続いて，ダブルクォーテーションでファイル名を囲む．最後に，using を使って x 座標と y 座標が書かれている列を示す．具体的には，

```
gnuplot> plot "trifunc.txt" using 1:2
```

である．各データ点を線で結びたければ，

```
gnuplot> plot "trifunc.txt" using 1:2 with line
```

とする．複数のデータを一度に描くためには，

```
gnuplot> plot "trifunc.txt" using 1:2 with line,
                          "trifunc.txt" using 1:3 with line,
                          "trifunc.txt" using 1:4 with line
```

とする．ただし，改行しないで (Enter キーを押さない) 記述する必要がある．これでは，$\tan(x)$ の値が大きすぎるので，プロットするレンジを変える．以下のように set xrange[ymin:ymax] でレンジを変えて，replot コマンドをつかう．

```
gnuplot> set yrange[-1.5:1.5]
gnuplot> replot
```

2.7 グラフのファイル出力

2.7.1 出力先の変更

gnuplot はディスプレイのみならず，グラフの図形ファイルを作成することができる．デフォルトの出力先はディスプレイとなっているので，これまでのグラフは画面に描画された．ディスプレイに表示されるだけでは不便なので，図形ファイルを作成する方法を示す．

諸君が使う代表的な出力先やファイルフォーマットを表 1 に示す．ディスプレイからファイルに出力先を変更するには，次のようにする．

```
gnuplot> set terminal emf
gnuplot> set output "hogehoge.emf"
```

これは，出直先を emf フォーマットのファイルに変更して，ファイル名を hogehoge.emf としている．

表 1: 代表的な出力デバイスとフォーマットを表す `set terminal` オプション

出力先	説明
`x11`	UNIX のディスプレイ
`windows`	Windows のディスプレイ
`emf`	Windows でよく使われるデータのフォーマット
`postscript`	UNIX で使われるデータフォーマット
`gif`	web などでおなじみ
`png`	これも web などでおなじみ
`epslatex`	LaTeX を使う場合便利

2.7.2 MS word に貼り付ける

gnuplot で作成したグラフを MS Word に貼り付ける時は，Windows 標準の emf フォーマットが便利である．emf フォーマットのグラフは次のようにして作成する．

```
gnuplot> set terminal emf
gnuplot> set output "hoge.emf"
gnuplot> plot sin(x)
```

最初の行で，出力先を emf ファイルに指定している．次の行で，出力ファイル名を `hoge.emf` としている．拡張子は重要で，`emf` としなくてはならない．プログラムは拡張子でファイルの種類を判断しているからである．最後の行で，三角関数のグラフを作成している．これで，正弦関数が書かれたファイル `hoge.emf` ができあがる．

これを MS Word に貼り付けるのは簡単である．

- Word を立ち上げて，メニューの挿入 → 図 → ファイルからを選択する．
- ファイルを選択する．

もし，emf フォーマットで張り付けができなければ，他のフォーマットを試してみる．

2.7.3 Starsuite に貼り付ける

学校の Linux では，office 環境として Sun microsystems の Starsuite が使えるようになっている．Starsuite へのグラフの張り付けもほとんど，MS Word と同じである．ここではあえて説明しないので，各自，実施して見よ．

2.7.4 LaTeX に貼り付ける

LaTeX というすばらしい文書作成ソフトウェアーがある．使い方はちょっと難しいが，長い文書を作成する時は楽だし，なによりも出来上がりがすばらしい．本年度，諸君に配布した講義ノートは全て LaTeX を使っている．以下のようにしてグラフのファイルを作成する．

```
gnuplot> set terminal epslatex
gnuplot> set output "hoge.eps"
gnuplot> plot sin(x)
```

そうすると，hoge.eps と hoge.tex というファイルができあがるので，それを LaTeX のソースに以下のように組み込めばよい．

```
\documentclass[10pt,a4paper]{jarticle}
\usepackage{graphicx}
\begin{document}

\begin{figure}[hbtp]
  \input{hoge}
  \caption{三角関数のグラフ}
\end{figure}

\end{document}
```

3 gnuplot のコマンド

3.1 ヘルプ

gnuplot にはかなり詳しいヘルプがある．英文であることと，コマンド入力のため初心者には使いにくいと思うが，使い込むとかなり便利である．ヘルプモードに入るためには，

```
gnuplot> help
```

とする．また，コマンドについて調べたければ，

```
gnuplot> help plot
```

のように，help の後にコマンド名を書く．

web ページの方が圧倒的に分かり易いが，ネットに接続されていない環境の場合やコマンドの使い方を忘れた場合には，かなり重宝する．

3.2 各種コマンド

gnuplot はターミナルにコマンドを打ち込んで，動作させる．使えるコマンドを，付録の表 3 に示しておく．

3.3 便利な機能

カレントディレクトリーの表示　シェルと同じ pwd コマンドが使える．

```
gnuplot> pwd
```

ディレクトリーの移動　シェルと同じ cd コマンドが使える．

```
gnuplot> cd "/home/yamamoto/hoge"
gnuplot> cd ".."
```

ヒストリー　キーボードの上矢印 (↑) や下矢印 (↓) でヒストリー機能が使える．

```
gnuplot> 矢印
```

シェルコマンド　!を付ければシェルコマンドが使える．

```
gnuplot> !ls
```

4　C 言語から gnuplot を操作する

4.1　パイプラインとは

gnuplot を C 言語のプログラムから制御するには，パイプを使うのが最も簡単である．C 言語のプログラムで，パイプを開いて，それを gnuplot に接続するのである．接続方法を説明する前に，UNIX のパイプと言う機能を簡単に説明しておく．

UNIX のコマンドの大部分は，標準入力 (キーボード) からデータを受け取り，標準出力 (ディスプレイ) に処理した結果を出力するようになっている．例えば，

```
ls -l
```

などである．このように，コマンドをフィルターと呼ぶ．

複数のコマンドを使って，処理したいデータがある場合，UNIX ではコマンドを接続することができる．標準出力から出てくるデータを次のコマンドの標準入力に渡すのである．例えば，

```
ls -l | sort -n +4
```

のようにするのである．最初のコマンドで，カレントディレクトリーのファイルとディレクトリーの情報を調べ，次のコマンドでファイル容量の順に並べている．

このようにコマンドを連結する機能をパイプラインという．そして，連結する | をパイプという．あたかも，パイプにデータが流れているかのようである．もちろん，2 個以上のコマンドの連結が可能である．このようにパイプを使ってコマンドをつなぐことにより，UNIX ではかなり複雑な動作も簡単に記述できる．

4.2　パイプを使う方法

パイプを使うことにより，C 言語のプログラムを通して gnuplot を制御することができる．パイプを通して C 言語のプログラムから，gnuplot にコマンドを流して，プログラマーの思い通りに動作させることが可能である．これを実現するためには，

- パイプを開く
- パイプを通してコマンドを送る

- パイプを閉じる

の操作が必要である．

パイプを開くためには，ファイルポインターをつかう．そのためファイルポインターを格納すの変数を用意しなくてはならない．パイプの先もファイルとして扱われるのである．

```
FILE *hoge;
```

次に gnuplot を立ち上げて，そこにパイプを接続する必要がある．パイプの情報のファイルポインターで示される．

```
hoge = popen("gnuplot -persist","w");
```

`popen()` 関数がパイプを開く命令である．これで，gnuplot が立ち上がり，パイプを通して，コマンドを送ることができる．オプションの `persist` で，gnuplot が終了してもグラフが残るようにしている．そうしないと，コンピューターの動作は高速なので，gunplot は一瞬にして終了し，グラフが消えてしまい，ほとんど動作内容が分からなくなる．

パイプを通して，gnuplot にコマンドを送るのは `fprintf()` 関数を使う．

```
fprintf(hoge, "plot sin(x)\n");
```

この `fprintf` を使って，gnuplot にいくらでもコマンドを送ることができる．あたかも，C 言語の向こう側で gnuplot が立ち上がって，それから命令を送っているかのように動作する．このようなことができるのが，コマンドを打ち込む Character-based User Interface(CUI) の良いところである．

すべての動作が終了したならば，パイプを閉じなくてはならない．これは，ファイルの操作と全く同じである．

```
pclose(hoge);
```

4.3 プログラム例

4.3.1 グラフ作成

先に示した方法で，C 言語から gnuplot を制御して，三角関数のグラフを作成するプログラムをリスト 1 に示す．

リスト 1: パイプを使い C 言語から gnuplot を制御している．

```
1  #include <stdio.h>
2
3  int main(void){
4    FILE *hoge;
5
6    hoge = popen("gnuplot -persist","w");
7    fprintf(hoge, "plot sin(x)\n");
8
9    pclose(hoge);
10
11   return 0;
12 }
```

4.3.2 データファイルとグラフ作成

複雑な数値計算を行う場合，データを作成してから，それをグラフ化することがしばしばある．次に，インパルスのデータを作成した後，グラフ化する例をリスト 2 に示す．

リスト 2: パイプを使い C 言語から gnuplot を制御している．

```c
#include <stdio.h>

int main(void){
  FILE *data, *gp;
  int i, imax=20;
  double x=-5.0, dx, y;
  char *data_file;

  /*------- データファイル作成 ---------- */

  data_file="out.dat";
  data = fopen(data_file,"w");

  dx=10.0/imax;

  for(i=0; i<=imax; i++){
    if(x<-1){
      y=-1.0;
    }else{
      y=1.0;
    }

    fprintf(data,"%f\t%f\n", x, y);
    x+=dx;
  }

  fclose(data);

  /*------- グラフの作成 ---------- */

  gp = popen("gnuplot -persist","w");
  fprintf(gp, "set xrange [-5:5]\n");
  fprintf(gp, "set yrange [-1.5:1.5]\n");
  fprintf(gp, "set pointsize 2\n");
  fprintf(gp, "plot \"%s\" using 1:2 with linespoints 1 4 \n", data_file);
  pclose(gp);

  return 0;
}
```

4.3.3 set コマンドを使った例

もう少し複雑な，set コマンドを駆使した例をリスト 3 に示す．

リスト 3: パイプを使った gnuplot の制御．データファイルを作成して，それをプロットしている．軸なども細かく制御している．

```c
#include <stdio.h>
#include <math.h>
void mk_triangle_data(char *a, double x1, double x2, int n);
```

```
4  void mk_graph(char *f, char *xlb, double x1, double x2,
5                char *ylb, double y1, double y2);
6
7  /*=========================================================*/
8  /*    main function                                        */
9  /*=========================================================*/
10 int main(void){
11
12   double pi = 4*atan(1);
13
14   mk_triangle_data("out.txt", -2*pi, 2*pi, 1000);
15   mk_graph("out.txt", "x", -2*pi, 2*pi, "y", -3, 3);
16
17   return 0;
18 }
19
20 /*=========================================================*/
21 /*    make a data file                                     */
22 /*=========================================================*/
23 void mk_triangle_data(char *a, double x1, double x2, int n){
24   double x, dx;
25   double y1, y2, y3;
26   int i;
27   FILE *out;
28
29   dx = (x2-x1)/n;
30
31   out = fopen(a, "w");
32
33   for(i=0; i<=n; i++){
34     x = x1+dx*i;
35     y1 = sin(x);
36     y2 = cos(x);
37     y3 = tan(x);
38
39     fprintf(out, "%e\t%e\t%e\t%e\n", x, y1, y2, y3);
40   }
41
42   fclose(out);
43 }
44
45 /*=========================================================*/
46 /*    make a graph                                         */
47 /*=========================================================*/
48 void mk_graph(char *f, char *xlb, double x1, double x2,
49               char *ylb, double y1, double y2)
50 {
51
52   FILE *gp;
53
54   gp = popen("gnuplot -persist","w");
55
56   fprintf(gp, "reset\n");
57
58   /* --------  set x grid ----------*/
59
60   fprintf(gp, "set grid\n");
61
62   /* --------  set x axis ----------*/
63
64   fprintf(gp, "set xtics 1\n");
65   fprintf(gp, "set mxtics 10\n");
```

```
66    fprintf(gp, "set xlabel \"%s\"\n", xlb);
67    fprintf(gp, "set nologscale x\n");
68    fprintf(gp, "set xrange[%e:%e]\n", x1, x2);
69
70    /* --------  set y axis ----------*/
71
72    fprintf(gp, "set ytics 1\n");
73    fprintf(gp, "set mytics 10\n");
74    fprintf(gp, "set ylabel \"%s\"\n", ylb);
75    fprintf(gp, "set nologscale y\n");
76    fprintf(gp, "set yrange[%e:%e]\n", y1, y2);
77
78          /* --------  plat graphs ----------*/
79
80    fprintf(gp, "set terminal x11\n");
81
82    fprintf(gp, "plot \"%s\" using 1:2 with line,\
83                      \"%s\" using 1:3 with line,\
84                      \"%s\" using 1:4 with line\n", f, f, f);
85
86    fprintf(gp, "set terminal emf\n");
87    fprintf(gp, "set output \"tri.emf\"\n");
88
89    fprintf(gp, "replot\n");
90
91    pclose(gp);
92  }
```

5 付録

5.1 数学関数

表 2: gunplot の組み込み数学関数．x は実数，z は複素数，rz は実数もしくは複素数の実部を表す．文献 [1] を参考に作成．

関数	動作
abs(z)	絶対値 $\|z\|$
acos(z)	$\arccos(z)$
acosh(z)	$\mathrm{arccosh}(z)$
asin(z)	$\arcsin(z)$
asinh(z)	arcshinh(z)
atan(z)	$\arctan(z)$
atan2(z1,z2)	逆正接関数 $(-\pi\sim\pi)$
atanh(z)	$\mathrm{arctanh}(z)$
besj0(x)	0 次ベッセル関数 $J_0(x)$
besj1(x)	1 次ベッセル関数 $J_1(x)$
besy0(x)	0 次ノイマン関数 $Y_0(x)$
eesy1(x)	1 次ノイマン関数 $Y_1(x)$
ceil(rz)	z 以上の最小の整数を求める関数
cos(z)	$\cos(z)$
cosh(z)	$\cosh(z)$
erf(rz)	(正規化) 誤差関数 $\mathrm{erf}(z)$
erfc(rz)	$1-\mathrm{erf}(z)$ を示す関数
exp(z)	e^z
floor(rz)	z 以下の最大整数を求める関数
gamma(rz)	ガンマ関数 $\Gamma(z_r)$

関数	動作
ibeta(p,q,rz)	不完全ベータ関数
igamma(a,rz)	不完全ガンマ関数
imag(z)	z の虚部 $\Im(z)$
int(rz)	rz の整数部を求める関数
inverf(rz)	erf(rz) の逆関数
invnorm(rz)	norm(rz) の逆関数
lgamma(rz)	対数ガンマ関数
log(z)	$\log_e(z)$
log10(z)	$\log_{10}(z)$
norm(rz)	正規分布関数の累積を求める関数
rand(rz)	擬似乱数を発生させる関数
real(z)	z の実部 $\Re(z)$
sgn(rz)	$\Re(z)$ の符号を求める関数
sin(z)	$\sin(z)$
sinh(z)	$\sinh(z)$
sqrt(z)	$\sqrt{z}$
tan(z)	$\tan(z)$
tanh(z)	$\tanh(z)$

5.2 コマンド

表 3: gunplot のコマンドの一覧．文献 [1] から引用.

コマンド	動作
cd	作業ディレクトリを移動
call	スクリプトファイル内の変数 ($n) に値を渡す
clear	現在スクリーンに表示されている内容を全消去
exit	GNUPLOT を終了
fit	最小二乗法によるデータの補間
help	ヘルプビューアを軌道
if	条件分岐命令
load	スクリプトファイルの読み込み
pause	GNUPLOT の動作を一時停止
plot	グラフの描画
print	変数の内容などの表示
pwd	現在の作業ディレクトリ名を表示
quit	GNUPLOT の終了
replot	前に実行した plot コマンドを再実行
reread	load コマンドで読み込んだスクリプトファイルの再読込
reset	set コマンドで設定したオプションをすべてデフォルトの値に戻す
save	ユーザ定義関数やグラフの数値，使用オプションをファイルに保存
set	各種オプションの設定
show	現在の設定内容の表示
shell	シェルの起動
splot	3 次元グラフの描画
test	使用可能な線種や点種などをすべて表示
update	fit コマンドで用いるパラメータファイルの更新

5.3 詳細設定

gunplot では設定のコマンド (set) に引き続き設定項目を記述すると，きめ細かな設定が可能である．設定は set コマンドを使い，現在の設定を見るためには，show all を使う．デフォルトの設定に戻すには，rest コマンドをつかう．表 4 に設定可能なものを示すが，詳細は help コマンド，あるいは web を調べよ．

表 4: gunplot のコマンドの一覧．文献 [2] を参考に作成.

angles	角度の単位	missing	欠けているデータ	x2label	軸のラベル
arrow	矢印	mouse	マウス	x2mtics	軸目盛りを月
autoscale	scale の自動設定軸	multiplot	複数のグラフ描画	x2range	軸の最大値/最小値
bars	誤差棒の先	mx2tics	軸の小目盛	x2tics	軸の大目盛
bmargin	下の余白	mxtics	軸の小目盛	x2zeroaxis	ゼロ軸の表示/非表示
border	境界 (軸，枠)	my2tics	軸の小目盛	xdata	データ型が日時
boxwidth	箱の幅	mytics	軸の小目盛	xdtics	軸目盛りを曜日
cbdata		mztics	軸の小目盛	xlabel	軸のラベル
cbdtics		offsets	描画素領域	xmtics	軸目盛りを月
cblabel		origin	原点の位置	xrange	軸の最大値/最小値
cbmtics		output	ファイル/デバイス	xtics	軸の大目盛
cbrange		palette		xzeroaxis	ゼロ軸の表示/非表示
cbtics		parametric	媒介変数の使用	y2data	データ型が日時
clabel	等高線の凡例	pm3d	3 次元カラー	y2dtics	軸目盛りを曜日
clip	枠付近のデータ制御	pointsize	記号の大きさ	y2label	軸のラベル
cntrparam	等高線の制御	polar	極座標表示	y2mtics	軸目盛りを月
colorbox		print	リダイレクト先	y2range	軸の最大値/最小値
contour	等高線の表示	rmargin	右の余白	y2tics	軸の大目盛
datafile	データファイル設定	rrange	軸の最大値/最小値	y2zeroaxis	ゼロ軸の表示/非表示
date_specifiers	日時のフォーマット	samples	サンプル数	ydata	データ型が日時
decimalsign		size	図の大きさ	ydtics	軸目盛りを曜日
dgrid3d	3 次元データを格子	style	プロットスタイル	ylabel	軸のラベル
dummy	独立変数の変更	surface	3 次元表示の面	ymtics	軸目盛りを月
encoding	文字コード	term	ターミナルの設定	yrange	軸の最大値/最小値
fit	フィッティング	terminal	ターミナルの設定	ytics	軸の大目盛
fontpath	フォントのパス	tics	目盛の向き	yzeroaxis	ゼロ軸の表示/非表示
format	軸の数字の書式	ticscale	目盛の長さ	zdata	データ型が日時
grid	格子を描く	ticslevel	splot の面間隔	zdtics	軸目盛りを曜日
hidden3d	隠線処理	time	プロットの日時	zero	ゼロの敷居値
historysize		time_specifiers	日時のフォーマット	zeroaxis	ゼロ軸の表示/非表示
isosamples	3 次元の線の数	timefmt	日時のフォーマット	zlabel	軸のラベル
key	凡例	timestamp	プロットの日時	zmtics	軸目盛りを月
label	任意のラベル	title	図のタイトル	zrange	軸の最大値/最小値
lmargin	左の余白	tmargin	上の余白	ztics	軸の大目盛
loadpath	パス	trange	軸の最大値/最小値		
locale	ロケール設定	urange	軸の最大値/最小値		
log	log プロット	view	3 次元表示の視点		
logscale	対数軸	vrange	軸の最大値/最小値		
mapping	3 次元の座標系	x2data	データ型が日時		
margin	外側の余白	x2dtics	軸目盛りを曜日		

参考文献

[1] Gnuplot reference. http://plum.nak.nw.kanagawa-it.ac.jp/docs/LaTeX/Gnuplot-Reference/.

[2] http://t16web.lanl.gov/Kawano/gnuplot/set.html.