

gnuplot の使い方

gnuplot は汎用的で、しかも手軽に使えるプロット・プログラムです。計算結果をグラフにするときに便利なので、ぜひ覚えてください。

1 gnuplot の始めかた、終わりかた

gnuplot の始めるには、ターミナル上のプロンプトの後ろで gnuplot と打ちます。すると、

```
/home/snaoki> gnuplot
```

```
G N U P L O T
```

```
Version 4.0 patchlevel 0
```

```
last modified Thu Apr 15 14:44:22 CEST 2004
```

```
System: CYGWIN_NT-5.1 1.5.13(0.122/4/2)
```

```
Copyright (C) 1986 - 1993, 1998, 2004
```

```
Thomas Williams, Colin Kelley and many others
```

```
This is gnuplot version 4.0. Please refer to the documentation  
for command syntax changes. The old syntax will be accepted  
throughout the 4.0 series, but all save files use the new syntax.
```

```
Type `help` to access the on-line reference manual.
```

```
The gnuplot FAQ is available from
```

```
http://www.gnuplot.info/faq/
```

```
Send comments and requests for help to
```

```
<gnuplot-info@lists.sourceforge.net>
```

```
Send bugs, suggestions and mods to
```

```
<gnuplot-bugs@lists.sourceforge.net>
```

```
Terminal type set to 'x11'
```

```
gnuplot>
```

というメッセージが出てきます。これまで

```
/home/snaoki>
```

というプロンプトだったのが、gnuplot>というプロンプトに変わっていることに注意してください。

gnuplot を終了させるときには、gnuplot>というプロンプトの後ろで、quit と打ちこみます。すると、/home/snaoki>のようなプロンプトに戻るはずですが。

2 グラフを描いてみよう

2.1 関数をグラフにしてみよう

まず適当な関数をグラフにすることから始めましょう。ここでは sin 関数で試してみます。

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

と打ち込んでください。するとウィンドウの枠が出てきますので、適当なところにマウスカーソルを持っていて、左ボタンを押してください。図1のような絵が得られます。x、y座標の範囲は今は指定していないので、自動的に適当な値が設定されています。それではx方向の範囲を1周期分に見てみましょう。

```
gnuplot> set xrange [-pi:pi]
```

```
gnuplot> replot
```

と打ち込むと、図2のようになります。

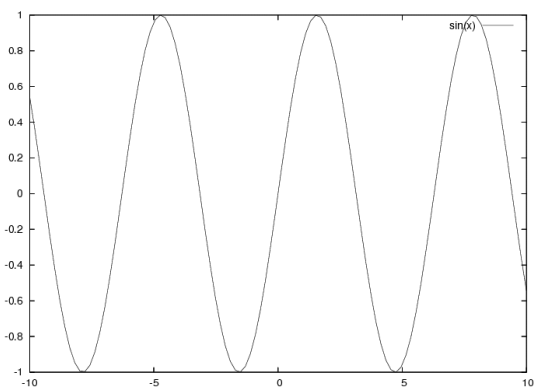


図1 : sin(x) その1

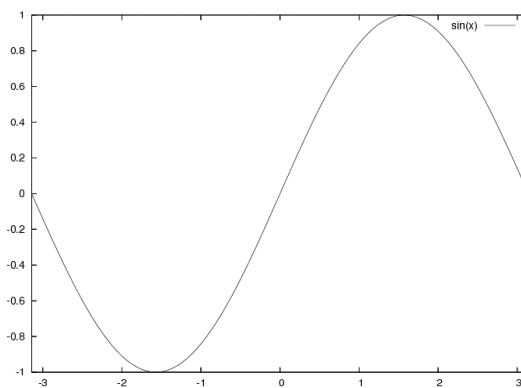


図2 : sin(x) その2 (1周期分の表示)

次に、範囲指定を少し変えて、2つの関数を同時に表示してみましょう。

```
gnuplot> set xrange [-2*pi:2*pi]
```

```
gnuplot> plot sin(2*x), sin(x)
```

とすると、図3のようになります。右上にどの線が何を表しているか書いてあります。この名前を変えてみましょう。

```
gnuplot> plot sin(2*x) title 'mouse', sin(x) title 'cat'
```

と打ちこむと、図4のようになります。 $\sin(2*x)$ の線（実線）には mouse と、 $\sin(x)$ の線（点線）には cat という名前がつけました。

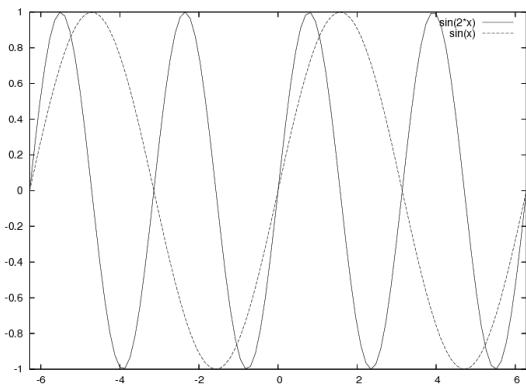


図3： $\sin(2*x)$ と $\sin(x)$ その1

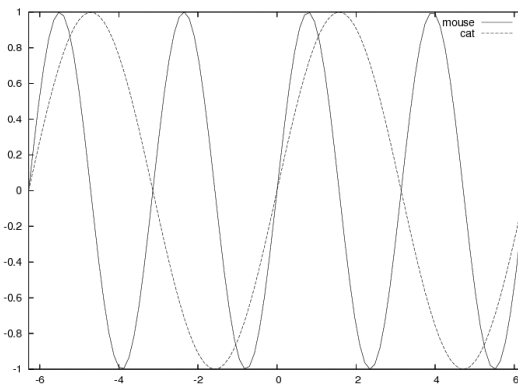


図4： $\sin(2*x)$ と $\sin(x)$ その2（凡例の変更）

それでは、次に、このグラフに名前をつけてみましょう。

```
gnuplot> set title 'sin(2*x) and sin(x)'  
gnuplot> set xlabel 'x-axis'  
gnuplot> set ylabel 'y-axis'  
gnuplot> replot
```

のように指定します。title はグラフの上に表示されます。xlabel は x 座標の下に、ylabel はグラフの左上にそれぞれ表示されます（ただしファイルに書き出すと ylabel の位置は変更されます）。

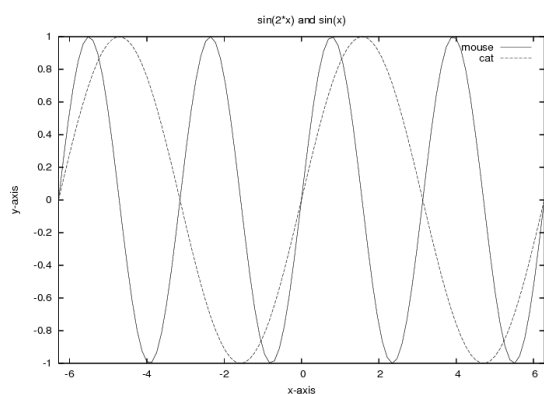


図5： $\sin(2*x)$ と $\sin(x)$ その3（タイトルをつけた）

次に線種をいろいろ変えてみましょう。今度は $\sin(x)+\cos(x)$ を表示してみます。

```
gnuplot> set title
```

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

これは今まで通り実線で描かれています (図は示しません)。次に、ポイントで表示してみましょう。

```
gnuplot> set style function points
gnuplot> replot
```

すると、図6のようになると思います。これを、

```
gnuplot> set style function impulses
gnuplot> replot
```

と変更すると図7のように変わります。

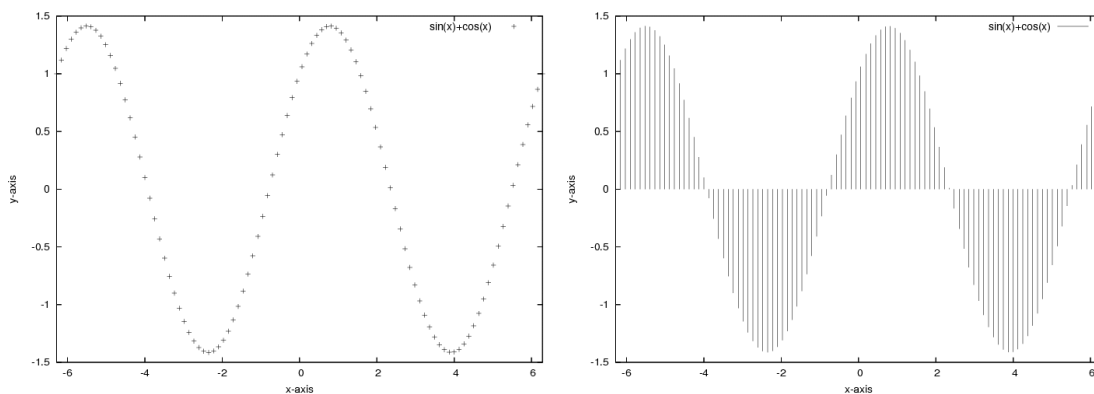


図6 : $\cos(x)+\sin(x)$ その1 (ポイントグラフ) 図7 : $\cos(x)+\sin(x)$ その2 (針グラフ)

再び点で描画すると、

```
gnuplot> set style function points
gnuplot> replot
```

図6に戻ります。今表示している点の数は自動的に決められています。そこで、描画する点の数を変更してみましょう。

```
gnuplot> set samples 20
gnuplot> replot
```

図8のようになります。

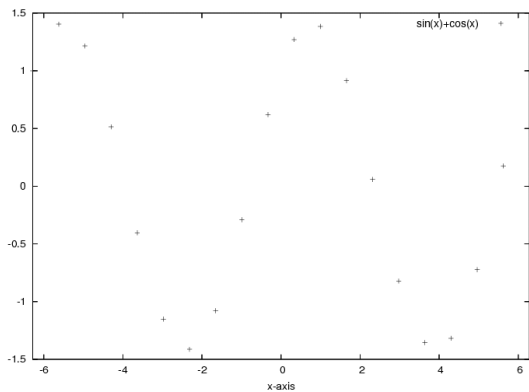


図 8 : $\cos(x)+\sin(x)$ その 3 (点の数を変更)

ここで階段状のグラフに変えてみましょう。

```
gnuplot> set style function steps
gnuplot> replot
```

としてください。図 9 のように変化すると思います。これを今度は

```
gnuplot> set style function boxes
gnuplot> replot
```

とすると、図 10 のように棒グラフに変わります。これでいろいろな線種のグラフを描けるようになりました。

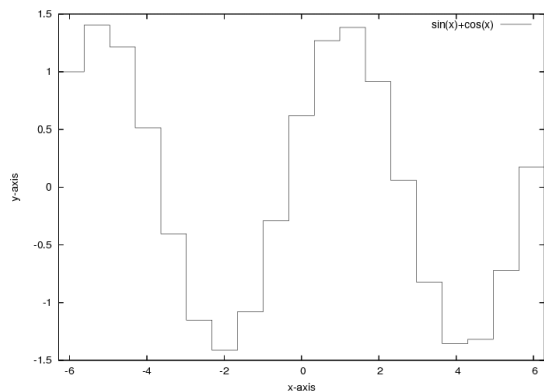


図 9 : $\cos(x)+\sin(x)$ その 5 (階段状)

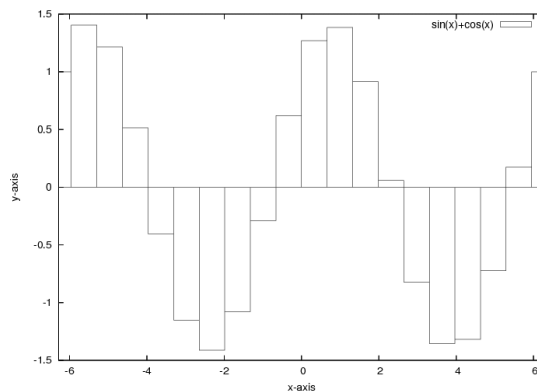


図 10 : $\cos(x)+\sin(x)$ その 6 (棒グラフ)

次は、少し特殊なグラフに挑戦します。

```
gnuplot> unset sample
gnuplot> set style function lines
gnuplot> plot exp(x)
```

これは $y=\exp(x)$ を実線で表示したもので、今までと何ら変わりはありません (図 1 1)。

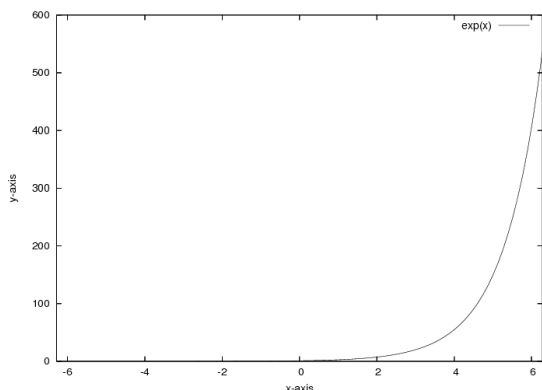


図 1 1 : $y=\exp(x)$ その 1

このグラフの y 軸を対数軸にしてみましょう。

```
gnuplot> set logscale y  
gnuplot> replot
```

図 1 2 のようになります。これにグリッド (格子) を引いてみましょう。

```
gnuplot> set grid  
gnuplot> replot
```

図 1 3 のようになると思います。

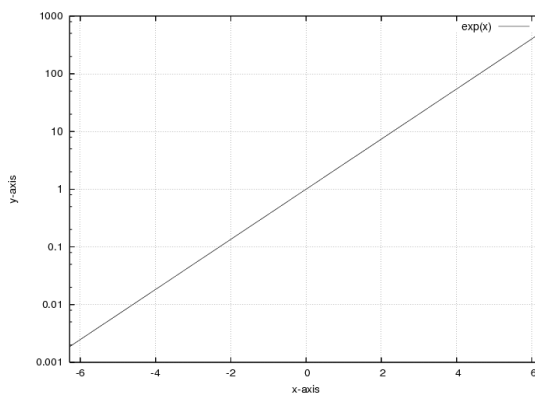
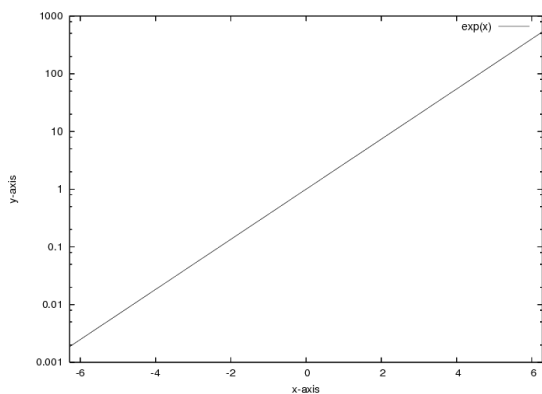


図 1 2 : $y=\exp(x)$ その 2 (片対数グラフ) 図 1 3 : $y=\exp(x)$ その 3 (グリッドあり)

さて、たくさんのパラメータを設定して、いろいろなグラフを書いてきました。パラメータが多いので、どのように設定されているか忘れてしまうこともあるでしょう。そこで、`show` というコマンドが用意されていて、`show parameter` という使い方で確認することができます。また、`show all` ですべてのパラメータを見ることができますので、一度試してみてください。

2. 2 データファイルを読みこんでグラフを描こう

ここでは、 (x, y) のデータが格納されているファイルを読みこんでグラフを描く方法を学びましょう。まず、ファイルには、

```
x1 y1  
x2 y2  
...  
xn yn
```

のようにデータをしまってください。値 x と値 y の間には必ず 1 つ以上のスペースが必要です。このファイルの名前を `data.txt` とします。

```
gnuplot> plot 'data.txt'
```

と入力すると、図 1 4 のようにポイントでプロットされます。これを

```
gnuplot> set style data lines  
gnuplot> replot
```

とすると、図 1 5 のように実線表示に変更できます。ここで、先の 2. 1 節で学んだ方法とは少し違うことに注意してください。先は線種を変更するときに、`set style function linetype` を用いたのに対し、ここでは `set style data linetype` を用いているからです。*linetype* の種類は 2. 1 節と同じですので、いろいろと試してみてください。

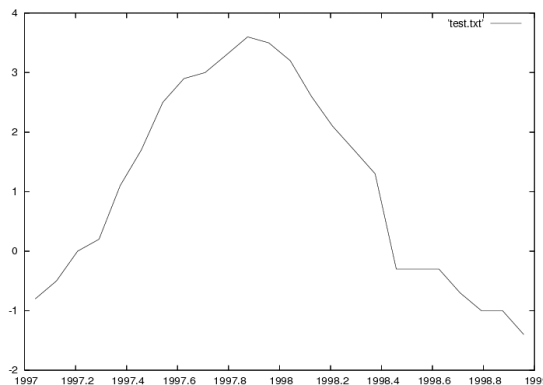
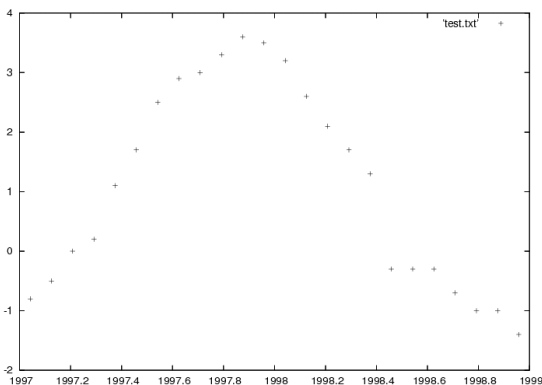


図 1 4 : データのプロット (ポイント表示) 図 1 5 : データのプロット (線表示)

3 絵を印刷しよう

これまではすべて画面に表示する方法を述べてきました。演習や研究を進めていくと、計算結果を画面に出してみるだけでなく、レポートのために紙に印刷する必要があると思います。ここでは、簡単に紙に出す方法を見ていくことにします。画面に出してある絵を印刷するためには、

```
gnuplot> set output "fig.ps"
gnuplot> set term postscript
gnuplot> replot
```

として、まず **PostScript** ファイルにします。それから **gnuplot** を終了するか、別のターミナル上から、

```
/home/snaoki> convert -rotate 90 fig.ps fig.gif
```

とすれば、**GIF** 形式の画像ファイルに変換することができます (注)。**PostScript** 形式のファイルは **evince**、**GIF** 形式のファイルは **display** というコマンドを用いて、

```
/home/snaoki> evince fig.ps
/home/snaoki> display fig.gif
```

のようにすれば見ることができます。あとは、このファイルをプリンタで印刷するだけです。**evince** や **display** のようなアプリケーションのメニューから印刷できる場合もありますが、**PostScript** 形式のファイルについては、ターミナルから印刷することもできます。

```
/home/snaoki> lpr -Pps2 fig.ps
```

-P のあとでプリンタ名を指定しています。この場合はプリンタ名は **ps2** です (**-P** の後に空白を空けない)。**gnuplot** で作成された絵は基本的に横長です。これを **landscape** といいます。縦長に使うときは **portrait** といいます。

```
gnuplot> set term postscript portrait
```

で設定できます。

(注) **GIF** 形式のファイルの解像度が低すぎる場合や、背景色が白ではなく透明になってしまう場合は、**PostScript** 形式のファイルを一度 **PDF** 形式に変換してから、**GIF** 形式に変換することによって解決できることがあります。

```
/home/snaoki> convert fig.ps fig.pdf
/home/snaoki> convert -rotate 90 fig.pdf fig.gif
```

4 最後に

ここでは、2次元描画に限った、簡単な **gnuplot** の使い方の説明しかしませんでした。**gnuplot** を使うと3次元描画、簡単な等値線図も描くことができます。また、今までいちいち入力していたコマンド、パラメータの

設定等は1つのファイルに書きこんで置いて、それを読みこんで実行することもできます。このような少し高度な使い方は、インターネットなどを使って調べてみてください。

課題1 : gnuplot を用いて、関数 $f(x)=\sin(0.2x)\cos(x)$ と $g(x)=\sin(4x)+\sin(5x)$ を作図し、印刷して提出せよ。2つの関数について別々に作図すること。作図の範囲、グラフや座標軸のタイトルなどは必要に応じて適切に設定せよ。

→横軸の範囲は、最低でも1周期分を表示できるように設定する。

→グラフが曲線ではなく折れ線になってしまう場合は、たとえば、`set samples 400` のようにして、サンプリングを細かくするよい。

余裕のある人は、次の課題にも挑戦してみてください。

課題1 (追加) : 中央のドの音の周波数は261 Hzである*。1オクターブ高くなると、音の周波数は2倍になる。1オクターブを均等な比で分割し、半音高くなるごとに周波数が $2^{1/12}$ 倍になるように定めた音律のことを平均律という。平均律には、自由に転調、移調できる利点がある。一方、純正律は、周波数の比が単純な整数比になるように定義した音律であり、たとえば、ド、ミ、ソの音の周波数の比は、4 : 5 : 6である。純正律には、和音を構成したときに、うなりが生じないという利点がある。

表. 音名と周波数比

音名	純正律	平均律
ド	1	1.000
レ	9/8	1.122
ミ	5/4	1.260
ファ	4/3	1.335
ソ	3/2	1.498
ラ	5/3	1.682
シ	15/8	1.888
ド	2	2.000

中央のドを根音（三和音のうちの最も低い音）とする長三和音（ド、ミ、ソ）の波形を、純正律、平均律のそれぞれについて作図せよ。和音を構成する3つの音は振幅の等しい単振動とする。両者の違いがわかるように作図の範囲を設定すること。

*厳密には、ラを440Hzと定義することが多いですが、簡単のためドを261Hzとします。