# gnuplot.pyの使い方

高エネルギー加速器研究機構
加速器第二研究系
山本昇

October 7, 1998

**Abstract**

gnuplot.py は手軽に図を作成するためのプログラムgnuplotをプログラミング言語 Python から利用するためのパッケージである。Python と gnuplot を組み合わせることで、さまざまなデータ解析とその結果のグラフ表示が簡単に行える。この文書ではこの gnuplot.py の利用法を説明する。

## 1 始めに

gnuplot.pyはfreeなグラフ作成ソフトであるgnuplotの機能をPython言語から使えるようにするPythonのモジュールである。gnuplot.pyはKonrad Hinsen氏作成のGnuplot.pyを拡張したモジュールである。オリジナルのGnuplot.pyとはUpward compatibleになるように拡張されている。特にgnuplotプログラムとPythonのデータの受け渡しの部分はGnuplot.pyから受け継がれている。

gnuplot.pyでは、Gnuplot.pyに対しgnuplot オブジェクトの追加、一つの画面への複数グラフの描画（multiplot, Show), 横軸を時間としたグラフの描画(tplot,tPlot)、三次元グラフの描画、Tcl/TkのCanvasへの描画等が追加されている。また、gnuplotオブジェクトの導入により、gnuplot のオプションをPythonから自由に設定できるようになっている。 gnuplot.pyは現在のところHP-UX, Digital-Unix( formerly known as DEC-OSF1), NeXTOS 3.0等のUnixOS+X-window環境で動作が確認されている。pythonおよびgnuplotが動作するUnixOS環境では多分動作するとおもう。gnuplot 3.5(pre3.6) patch level 347以降のgnuplotが必要である。gnuplotおよびPythonそれぞれはMacintoshやwindowsなどのPCでも動作することが知られているが、gnuplot.py はgnuplotとpythonとのデータ交換の一部にpipe機能をつかっているためまだこれらのPC上では動作しない。

## 2 簡単な例

gnuplot.pyはPythonのモジュールとなっている。他のPythonモジュールと同じようにgnuplot.pyを使用するに先立ちこのモジュールをimportする必要がある。

```
import gnuplot
```

あるいは

```
from gnuplot import *
```

を実行する必要がある。以下の説明では、第一の形を使ったものとして説明を続ける。これはgnuplotのモジュールをどこで使っているかを明確にするためで、実際の使用に際しては後者の形式で問題はない。pythonの起動毎にgnuplotをインポートする事が面倒であれば、環境変数PYTHONSTARTUPに設定されたファイルに上記の行を追加しておくとよい。この環境変数によるスタートアップ・ファイルの読み込みはPythonを対話的に使っただけ有効となるので、Pythonを非対話的に使うスクリプトの中で、このスタートアップ・ファイルを読み込ませるには、スクリプトの最初に、

```
import os; execfile(os.environ['PYTHONSTARTUP'])
```

を追加しておく。

## 2.1 一次元プロット：plot()

一次元のグラフを表示するにはplot関数を使う。例えば、サイン関数のグラフを表示するには、

```
from Numeric import *
x=arange(256)*pi/128
y=sin(x)

gnuplot.plot(y)
```

とする。これによって図 1のグラフが X の画面に表示される。
この画面をファイルに書き出すためには、

```
gnuplot.plot(y,file='gpsample1.eps')
```

とファイル名を指定すればよい。
これらの例では横軸はデータのindexとなっていた。横軸を$x$にするためには、与えるデータを$(x, y)$の組のリストにする［図 2］。

```
gnuplot.plot(map(None,x,y))
```

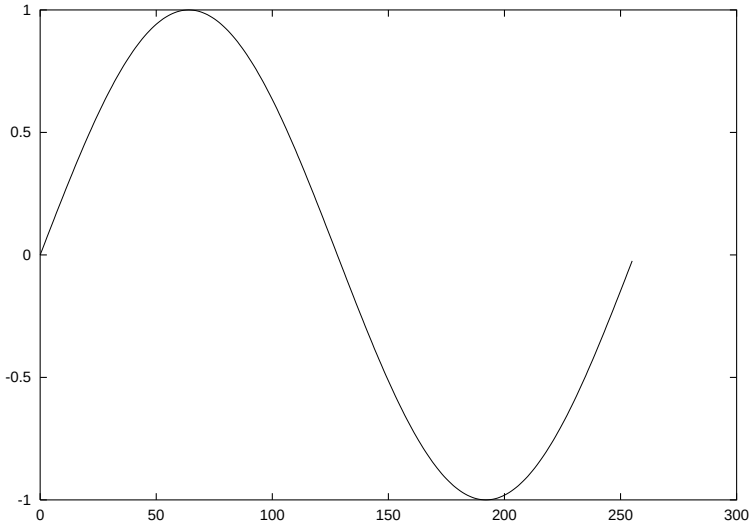


図 1: gnuplot.pyによるsin 関数の描画例

## 2.2 gnuplotオプションの変更

gnuplotにはさまざまなオプションが用意されている。これらのオプションをpythonからのplotコマンドで指定するには例えば、

```
gnuplot.plot(map(None,x,y),xrange='[0:6.3]',yrange='[-1.2:1.2]',
title='"This is a Title"')
```

とする［図 3］。キーワード付きの引数$xrange$,$yrange$等を与えることで、gnuplotのオプション $xrange$,$yrange$ を設定出来る。キーワードのオプションに与える値は文字列であることを注意しておく。keyword = 'val'をplot関数の引数に与えることはgnuplotで、

```
set keyword val
```

を実行するのと同じである。valが空である場合には右辺として空の文字列（”または””）を指定する必要がある。

## 2.3 複数のデータを一つのグラフに表示する。

複数のデータを一つのグラフに表示するには、単にデータを並べて、

```
y2=cos(x)
gnuplot.plot(y,y2,yrange='[-1.2:1.2]')
```

とすればよい。

x－軸の座標を指定するためには、

```
gnuplot.plot(map(None,x,y,y2),xrange='[0:6.3]',yrange='[-1.2:1.2]')
```

あるいは、

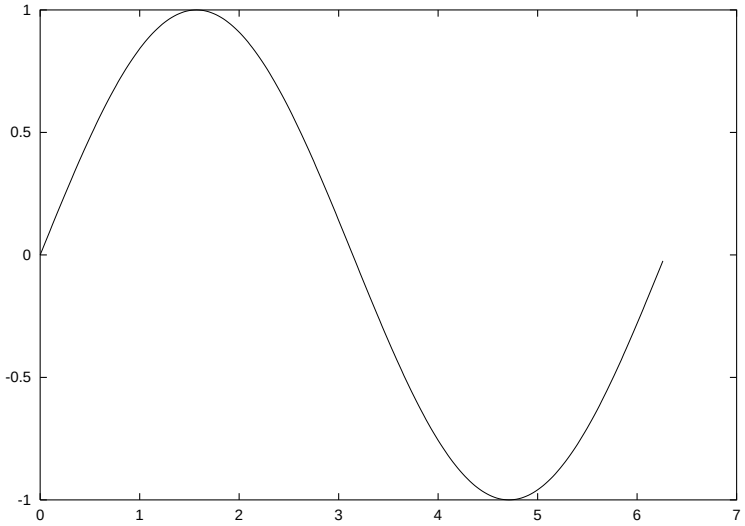


図 2: X座標軸をデータとして与えて、sin関数を描画

```
gnuplot.plot(map(None,x,y),map(None,x,y2),xrange='[0:6.3]',
yrange='[-1.2:1.2]')
```

とすればよい。どちらも図 5に示されたグラフを生成する。

## 2.4 複数のグラフを一つの画面に表示:multiplot()

複数のグラフを一つ画面に表示するために関数multiplot()が用意されている。

```
gnuplot.multiplot(y,y2,yrange='[-1.2:1.2]')
```

によって、図 6が画面に表示される。 画面の分割を指定しない場合には、multiplot()関数がグラフの分割数を適当に判断して分割する。分割を指定するには、multiplot()関数にblock引数を与える。block引数の値は長さ2のtupleで、第一成分が横軸方向の分割数、第二成分が縦軸の分割数である。multiplotは夫々の方向で等分割して出来るグリッドにグラフを順に入れていく。

```
gnuplot.multiplot(y,y2,yrange='[-1.2:1.2]',block=(1,2))
```

multiplotに複数のデータを表示するには、

```
gnuplot.multiplot(map(None,x,y,y2),map(None,sqrt(x),y2),
yrange='[-1.2:1.2]',block=(1,2))
```

のように複数のデータを一つのTupleに詰め込んでプロットさせる。

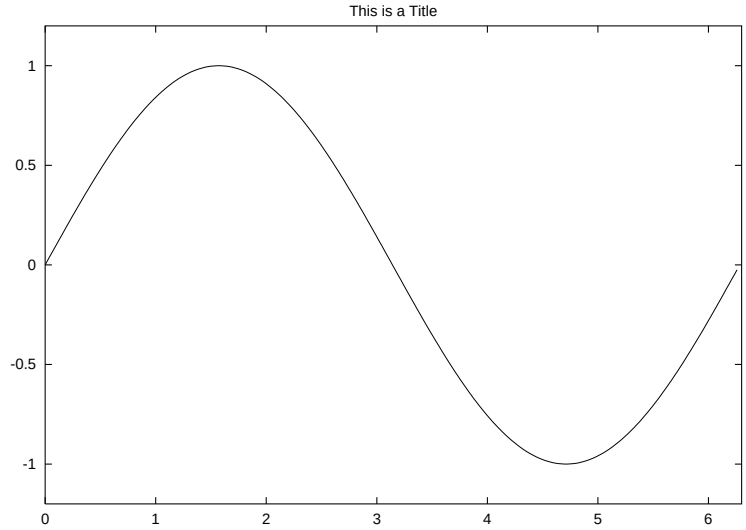


図 3: gnuplotオプション設定の例:xrange='[0:6.3]',yrange='[-1.2:1.2]',title='"This is a Title"'を設定した。

## 2.5 時系列グラフの表示:tplot()

tplot関数はplot関数の特別なバージョンと考えてよい。時系列データがあったとき、これを横軸を時間として表示するグラフを作成する。データは第一成分がUnix標準のepochからの秒数で表わされた時刻、後に其の時刻の時系列データが並ぶsequenceのsequencを受け付ける。epochからの秒数で表わされた時刻への変換にはtime moduleのmktime()関数等を使うことができる。

```
gnuplot.tplot(map(None,1000*x,y,y2),yrange='[-1.2:1.2]',block=(1,2))
```

## 2.6 三次元グラフ:splot()およびsplot_matrix()

gnuplotは図 10の様に三次元のグラフを生成することもできる。 このグラフは、次のPythonプログラムで生成された。

```
t=arange(64)*(6.0/64)-3.0
u=t+0.0 # force to create copy of t
t.shape=(len(t),1)
u.shape=(1,len(u))
gnuplot.splot_matrix(sin(u**2+t**2+0.001**2)/(u**2+t**2+0.001**2))
gnuplot.splot_matrix(sin(u**2+t**2+0.001**2)/(u**2+t**2+0.001**2),
hidden3d="" )
```

splot()も同様に三次元グラフを生成する。データの形式はsplot_matrix()とは異なり、三次元座標値の組のリストのリストを一組のデータとする。

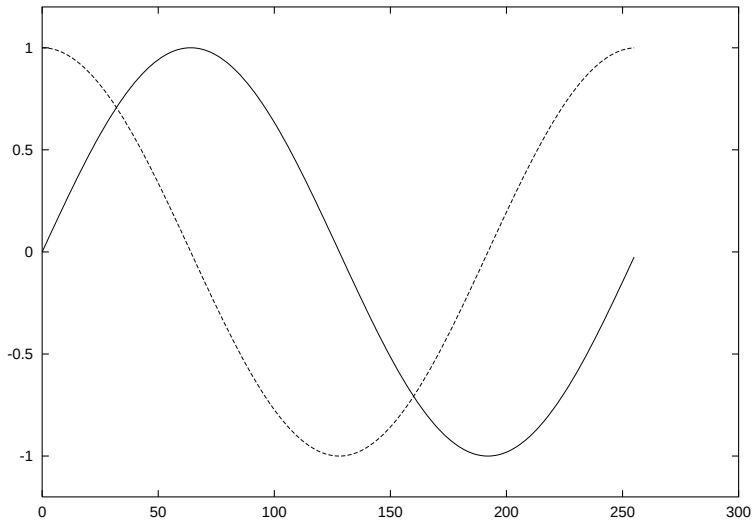


図 4: 複数データの一つのグラフへの表示

### 2.7 Tk/Canvasへの描画

これまで説明したグラフ描画関数にCanvasオプションを付けることで、TkinterのCanvasオブジェクト上にグラフを表示させることができる。

```
import Tkinter
r=Tkinter.Toplevel()
c=Tkinter.Canvas(r)
c.pack()
gnuplot.splot_matrix(sin(u**2+t**2+0.001**2)/(u**2+t**2+0.001**2),
hidden3d="",canvas=c)
c.mainloop()
```

## 3 gnuplot オブジェクト

前節では、gnuplotモジュールの関数について説明した。夫々の関数を実行すると新たにXのウィンドウが作られ、その中にグラフが表示された。これらのウィンドウには Pythonのクラス gnuplot.gnuplot のインスタンスが一つずつ生成されている。前節で説明した関数はこのインスタンスの同名のメソッドを呼ぶことで実現されている。前節の関数はいずれもgnuplot.gnuplotオブジェクトをその値として返すので、このインスタンスのメソッドを呼ぶことにより同じウィンドウにことなるグラフを表示できる。

gnuplot.gnuplotのインスタンスを生成するには、もちろん

```
gnuplot_instance=gnuplot.gnuplot()
```

を使ってもよい。

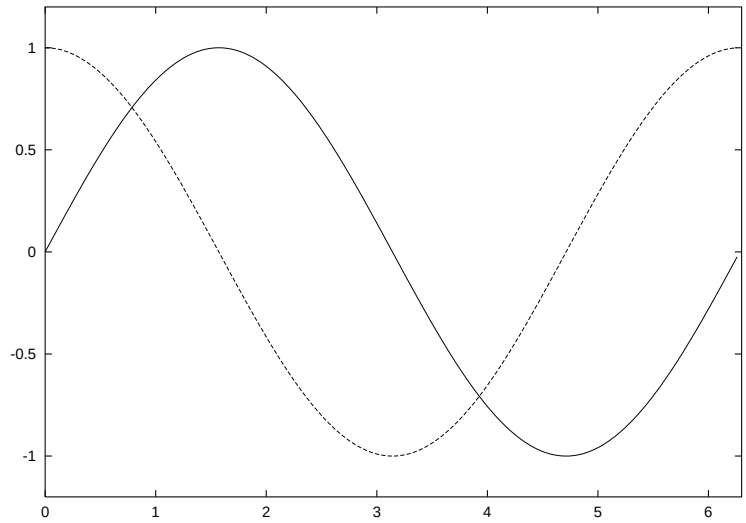


図 5: 複数データの表示：x－座標付きの場合。

## 3.1 Show()メソッド

gnuplot.gnuplotオブジェクトのメソッドには、plot,splot,tplot,multiplotなどのgnuplotモジュールの関数群と同じ名前を持つメソッドの他に、Plot,sPlot,tPlot等のメソッドおよびShowメソッドがある。Plot,tPlot, sPlotなどのメソッドはgnuplot.gnuplotオブジェクトに対応するwindowにグラフを描画するとともに、Show コマンドの引数として使うことのできる3成分からなるtupleを値として返す。Show()メソッドは複数のこれらのtupleを引数として渡されると、multiplot()と同じくウィンドウを分割してその中にそれぞれのtupleに対応するグラフを描画する。multiplot()同様に、画面の分割数はblockキーワードを引数の中で指定することで制御できる[図 11]。

```
gp=gnuplot.gnuplot()
plt1=gp.plot(y,y2)
plt2=gp.splot_matrix(sin(u**2+t**2+0.001**2)/(u**2+t**2+0.001**2),
hidden3d="")
gp.Show(plt1,plt2)
```

# 参考文献

[1] N. Yamamoto “arr.py の使い方”, in preparation

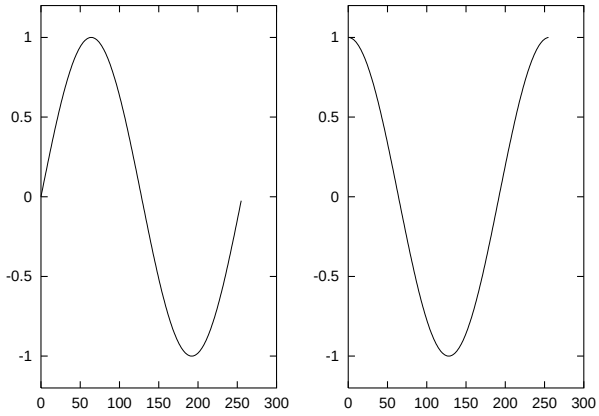


図 6: 複数グラフの同一画面への表示例

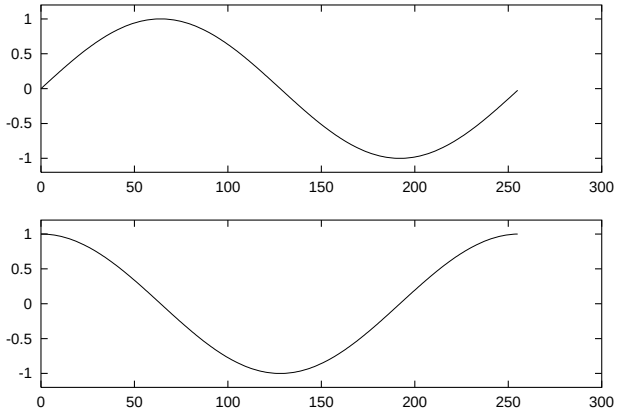


図 7: multiplot()関数でのblock引数の使用例

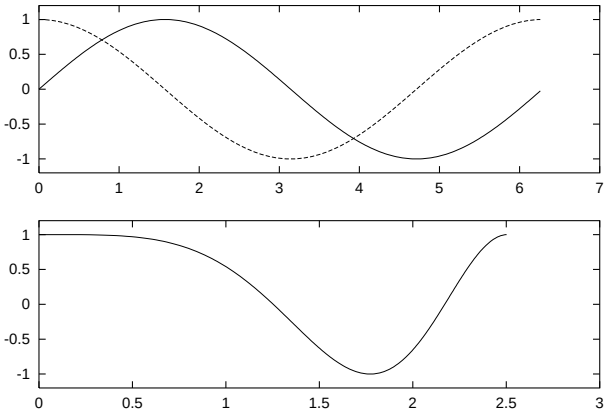


図 8: multiplot関数で一つのグラフに複数データの表示

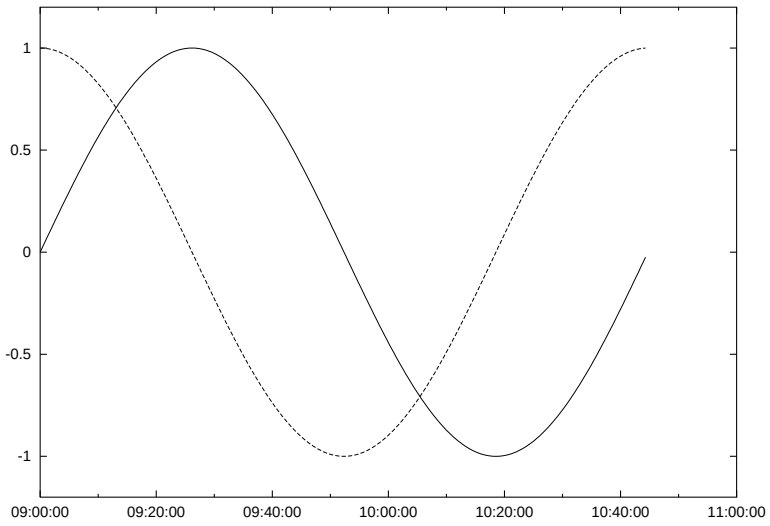


図 9: tplot()によるプロット例。横軸が時間フォーマットになっている。

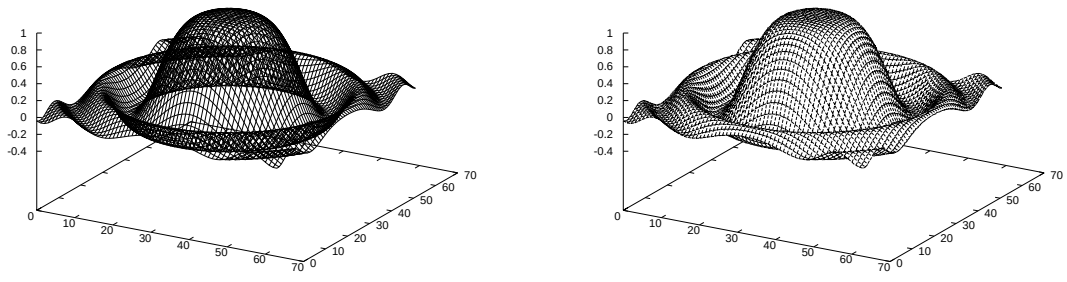


図 10: splot_matrix()による三次元グラフィックス。hidden3dオプションなし（左）およびhiden3dオプション付き（右）

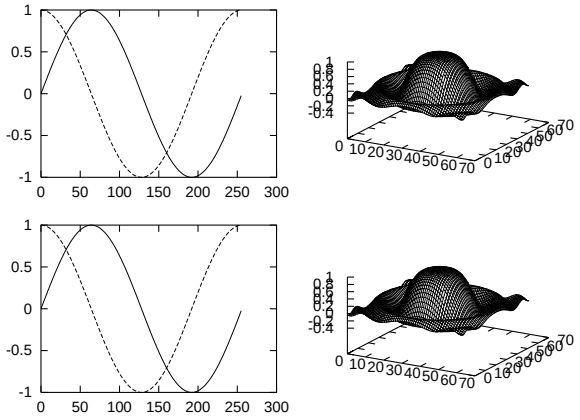


図 11: gnuplot.gnuplotオブジェクトとShow()メソッド