本稿は Linux Japan 誌 2002 年 02 月号に掲載された記事に補筆修正したものです.

高機能グラフ化ツール Paw

いつも同じことばかり題材にしているようで少し気がひけますが，今回は（も？）理工学系のユーザーに向けたグラフ化のツールを紹介しようと思います.「えー，また Gnuplot+Tgif の話?」と思われた方，連載を良ーく読んでいただきありがとうございます．常日ごろ私が教材を作成しているのはこの 2 つのツールで，概ね事足りています．が，もちろんもっと高度な画像処理が要求されるユーザーの方もいらっしゃるでしょう．そこで，にわか仕込みで筆者がどこまで解説できるか判りませんが，高度なグラフ化ツール Paw のほんのさわりを（もう弱気）絵物語してみようと思います.

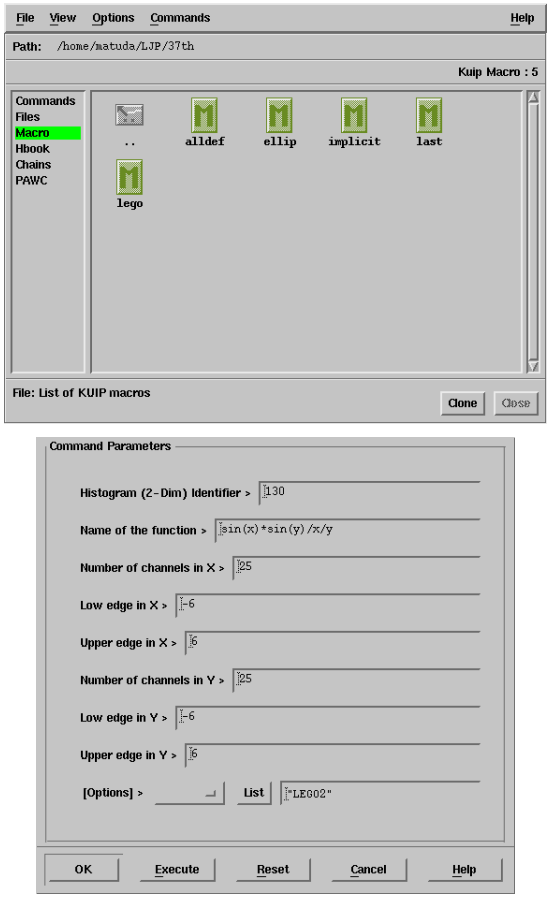

図 1　paw++ のメイン画面（左）と関数プロット fun2 のダイアログ画面 (右)

Paw とは

Paw（Physical Analysis Workstation）は 1980 年代後半から主として，原子物理学分野のデータ処理とグラフ化のために，Netscape の始祖である MOSAIC 発祥の地 CERN で開発されたツールです．さすがに世界の頭脳集団が開発しているだけあって，PAW 自身はコマンドを駆使してサクッと図面を仕上げる目的に適したツールですが，誰でも迷わず使えるように PAW++ という Motif 上の GUI まで用意されています（図 1）．最新バージョンは 2.12 で，CERN の program library から RedHat6.1 と RedHat7.2 用が入手できます [1]W[3]，筆者愛用の Plamo2.1 では RH6.2 用のバイナリがそのまま動きました.

元々 Paw は原子物理学の実験データを処理するための専用ツールでして，入力データとしては実験の性格上非常に大きなヒストグラム（2 次元も含みます）がバイナリで収まったファイルを扱い，その視覚化が容易となるようにできています．Paw 本来の用途ならばこのヒストグラムから話をしなければならないのでしょうが，それは省略します．なぜならデータ数が少なければ，一般的に言えば数値のテキストファイルの方が扱い易いはずで，Paw も一般データ向けに vector（配列）形式を持っており，グラフを描くといった用途に限ればそれを利用する方がずっと判りやすいからです.

実行

起動：対話モード

では，さっそく動かしてみましょう．pawX11 を起動すると，グラフ描画窓が現れ，ターミナルに Workstation のタイプを尋ねるメッセージが表示されます．尋ねられても判りませんから，素直に Enter を押します．すると Paw は次のようにプロンプトを出してコマンド入力待ちになります.

```
$ pawX11
 ************************************************
 *                                              *
 *          W E L C O M E   to   P A W          *
 *                                              *
 *      Version 2.12/22     13 June 2001        *
 *                                              *
 ************************************************
 Workstation type (?=HELP) <CR>=1 :
 Version 1.27/03 of HIGZ started
 *** Using default PAWLOGON file "/home/matuda/.pawlogon.kumac"

PAW >
```

以下対話的に実行をすることになるわけです．さて，データファイルの入力は後回しにして，3 次元の俯瞰図を関数を与えて描いてみましょう．次のコマンドを次々に入力してみてください（行頭のプロンプト “PAW

>” は省略しています）.

```
title_global 'Plot3d'
fun2 100 sin(x)*sin(y)/x/y 25 -6 6 25 -6 6 "LEGO"
fun2 110 sin(x)*sin(y)/x/y 25 -6 6 25 -6 6 "LEGO1"
fun2 120 sin(x)*sin(y)/x/y 25 -6 6 25 -6 6 "LEGO2"
set ncol 32
palette 1
fun2 130  sin(x)*sin(y)/x/y 25 -6 6 25 -6 6 "LEGO2"
```

すると図 2のような 2 元のヒストグラムが描かれます．ただし，タイトルの大きさや書体が異なっているはずです．Paw は初期設定マクロファイル ~/.pawlogon.kumac を読み込みますから（オプション -n で抑止することも可能です），好みにあわせて図の大きさやタイトルや軸のフォントの設定などを設定しておくことができます（**List 1** 参照）.

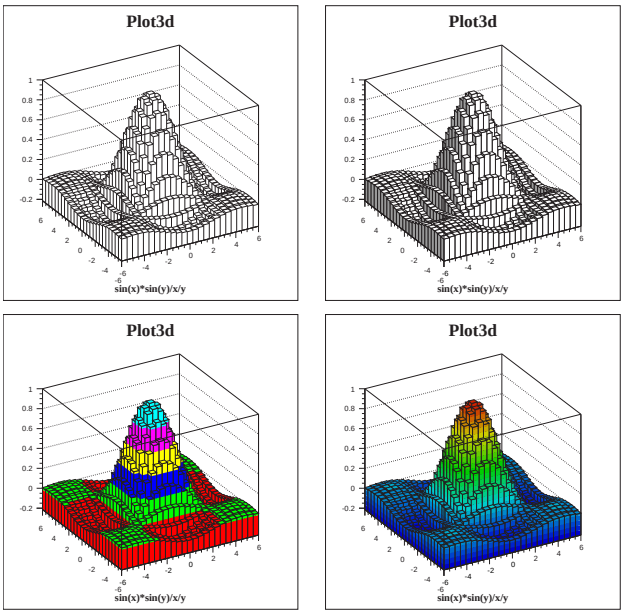

図 2　関数プロット fun2 による 2 元ヒストグラム：LEGO, LEGO1, LEGO2, LEGO2（色数 32）

List 1　~/.pawlogon.kumac の例

```
* defaults ./
* option fit
* option grid
* option UTIT=0
option ZFL1
set gsiz 0.6
set ygti 0.5
set asiz 0.4
set vsiz 0.28
set gfon -20
set tfon -20
set lfon -60
set vfon -40
set cfon -20
graphics/viewing/size 15. 15.
histogram/delete *
vector/delete *
```

終了は quit です．fun2(function/fun2 の省略形) が 2 変数の関数 $z = f(x, y)$ を描画するコマンドで，書式は

```
FUN2 ID UFUNC NCX XMIN XMAX NCY YMIN YMAX [ CHOPT ]
```

となっています．識別番号 ID を使っているところが大型計算機らしい雰囲気出してます．NCX は Number of Channel in X の略で，Channel（検出器の番号）という言葉が原子物理学の実験データを色濃く反映しています．最後の CHOPT で種類などを指定することができます．しかし，LEGO とは楽しい命名ですね．最後の例はカラーマップの色数を増やす方法を示しています．詳しくはオンラインヘルプを読んでください．

```
help fun2
help palette
```

なお，ビューアとして **vi** が呼ばれますから，ヘルプの終了は “:q!” となります．これら一連のマクロを記述したファイル（拡張子は kumac に指定されています）を exec コマンドでまとめて実行することもできます．

```
exec MACRO（.kumac は省略可）
```

バッチ処理

もちろん，上記のマクロを以下のようにバッチ処理させることもできます．

```
paw -b ***.kumac
```

こうしてグラフが次々に現れては消えていくのを眺めるわけですが，もちろん図をファイルに残したくなるでしょう．

図の保存

ファイルに保存するには

```
PICTURE/PRINT FILE WIDTH HEIGHT
```

を実行するのが良いでしょう．FILE の拡張子は出力形式に応じて .ps .eps .tex .gif を選択します．ただし，ヘルプの記述が判りにくいのですが，“option ZFL” を立てる必要があるようです．例として，次のマクロ surf.kumac を作成して exec で実行させてみてください．

```
option ZFL
title_global 'Plot3d'
fun2 100 sin(x)*sin(y)/x/y 25 -6 6 25 -6 6 "SURF"
pic/print surf.eps
fun2 110 sin(x)*sin(y)/x/y 25 -6 6 25 -6 6 "SURF1"
pic/print surf1.eps
fun2 120 sin(x)*sin(y)/x/y 25 -6 6 25 -6 6 "SURF4"
pic/print surf4.eps
fun2 130 sin(x)*sin(y)/x/y 25 -6 6 25 -6 6 "BOX"
pic/print box.tex
option UTIT=0
fun2 140 100*sin(x)*sin(y)/x/y 25 -6 6 25 -6 6 "Z"
atitle X-axis Y-Axis ! 220
pic/print z.ps
set ncol 48
palette 1
fun2 150 sin(x)*sin(y)/x/y 25 -6 6 25 -6 6 "SURF1"
pic/print surf1n.gif
```

つづいて，バッチ処理させようとして

```
pawX11 -b surf.kumac
```

とするだけでは “No picture in memory” が表示され失敗します．実際に図を描く workstation を指定しなければならないのです．対話モードの時にならって 1 番を指定しましょう（1〜10 までどれを使っても良いようです）．

```
pawX11 -w 1 -b surf.kumac
```

とします．これで図が画面に現れファイルもできます（図 3）．

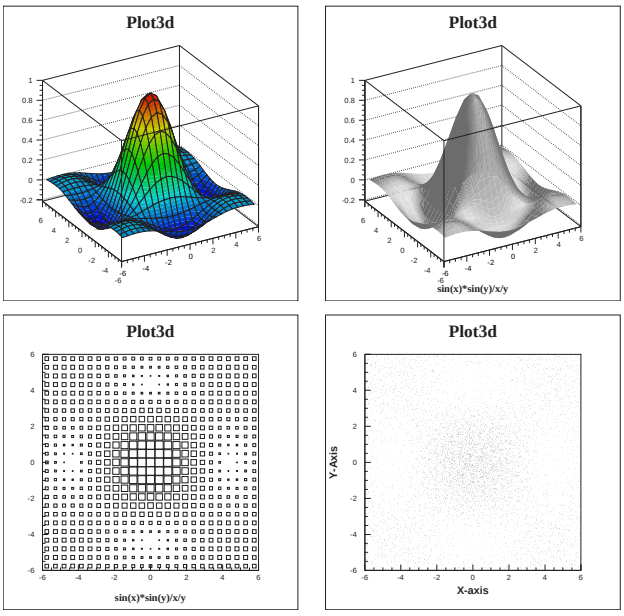

図 3　関数プロット fun2 による俯瞰図と密度分布図：左から SURF1，SURF4，BOX，Z

データファイルの読み込み

データファイルを読み込んで表示させましょう．まずは簡単な xy プロットの例です．

```
option UTIT=0
title_global 'Symbol'
color 1 0 0 0
vec/create x(100) R
vec/create y(100) R
vec/read x,y cos.dat
color 1 0 1 0
null -7 7 -1.1 1.1
symbols x y 100 24 0.3
color 1 0 0 0.5
atitle X-Axis Y-axis ! 220
set hcol 2
set hwid 8
fun/plot sin(x) -7 7 S
pic/print plot2d.eps
```

予め vector（配列）x(100)，y(100) を用意しておきます．データファイル cos.dat のレコードの形式は

```
x(1)  y(1)
x(2)  y(2)
...
```

であるのが普通ですから，それぞれの列を x(100)，y(100) に代入します．

```
NULL XMIN XMAX YMIN YMAX
```

でグラフの枠を描いておき，

```
SYMBOLS X Y N ISYMB SSIZE
```

で散布図を描きます．ISYMB は記号番号で 20〜31 に割り当てがあります．SSIZE は記号の大きさです．また

```
COLOR ICOL RED GREEN BLUE
```

として，色番号 ICOL の RGB 値を指定し，色の現在値を設定します．atitle XTIT YTIT ZTIT IALGN は軸のラベルを付加するコマンドです．おまけの fun/plot は最も簡単に関数 $y = f(x)$ を描くコマンドです．前のグラフを上書きしないように，オプション S（Superimpose）をつけて重ね描きしています．histogram を用いればもっと簡単になるらしいのですが，一般のデータファイルから作成しようとすると，結構手間がかかります．これに比べると gnuplot の

```
plot 'cos.dat'
```

はずいぶんと手軽であることが判ります．

つづいて，2 次元ベクトル上のスカラー量 $z = f(x, y)$ の観測データを表現する方法です．もちろん 3 次元の俯瞰図あるいはカラーマップが一般的です．gnuplot では，

```
x(1,1) y(1,1) z(1,1)
x(1,2) y(1,2) z(1,2)
...
x(1,m) y(1,m) z(1,m)
                    <--1 つの空行
x(2,1) y(2,1) z(2,1)
...
```

```
x(2,m) y(2,m) z(2,m)
                    <--1 つの空行
...
...
x(n,m) y(n,m) z(n,m)
```

1 つの空行で区切られた 1 つの線を表すブロックデータの集合を与えます．これは，必要な情報が全て内在しています．このようなデータファイル形式はむしろ珍しく，他のグラフィックツールでは，領域 x,y に関する情報は別にして，z(n,m) の行列データを与えるのが普通です．すなわち

```
z(1,1) z(1,2)... z(1,m)
z(2,1) z(2,2)... z(2,m)
...
z(n,1) ...    ... z(n,m)
```

のような形式です．しかしこの方法ですと，要素 m が大きい場合に，ラインバッファからあふれてしまうという実際上の問題が生じます．PAW ではラインバファの大きさに依存しない

```
z(1,1)
z(2,1)
...
z(n,1)
z(1,2)
z(2,2)
...
z(n,2)
...
...
z(n,m)
```

という並びを期待します（添え字の増加の順序は，FORTRAN のメモリ配置に従うので C とは逆になっています．これ常識ですね）．具体的な手順は次のようになります．

```
title_global '2D Vector'
vec/create data(25,25)
vec/read data h2.dat
2dhist 100 ' ' 25 0 1 25 0 1
hist/put_vect/contents 100 data
angle theta=30 phi=-150
set ncol 50
palette 1
hist/plot 100 "SURF1"
pic/print 2dvector.eps
```

vector（配列）から histogram へのデータの代入 `/hist/put_vect/contents` が大切です．

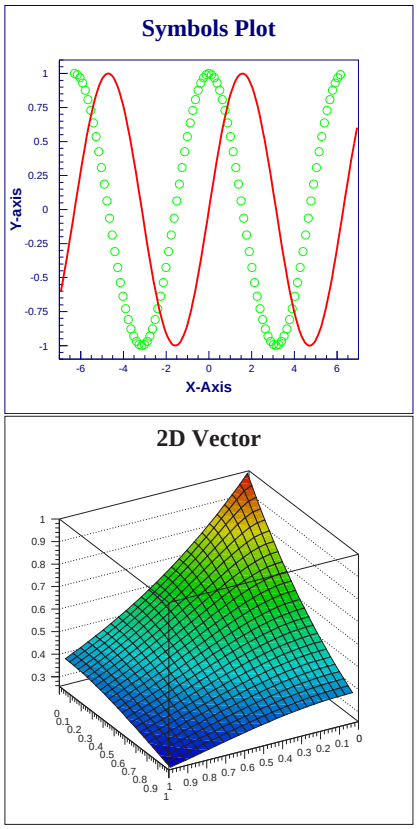

図 4　Vector を用いたデータファイルのグラフ化

Gnuplot にない特徴

以上のように Paw は LEGO や BOX のような，gnuplot では描くことのできないグラフをサポートしています．さらに高度な機能もあります．

陰関数のグラフ化

陰関数 $f(x,y,z) = C$ で与えられた図を描くことができます（図 5）．

```
title_global 'Ellipsoidal'
zone 2 2
range -1 1 -1 1 -1 1
fun/draw x**2+y**2+z**2=1
fun/draw (x**0.5)**4+y**2+z**2=1
fun/draw x**2+y**2-z**2=0.5
points 100 100 2
angle theta=90 phi=0
range -3 3 -3 3
fun/draw x**2+2*x*y-y**3=1
pic/print ellip.eps
```

`zone 2 2` とすると．1 ページを 2×2 の領域に分割して図を順番に配置します．最後の図は，視点を変えるコマンド `angle theta=deg1 phi=deg2` を使って，xy

平面図形（つまり z 軸 の正方向に視点をおく設定です）を描きました．

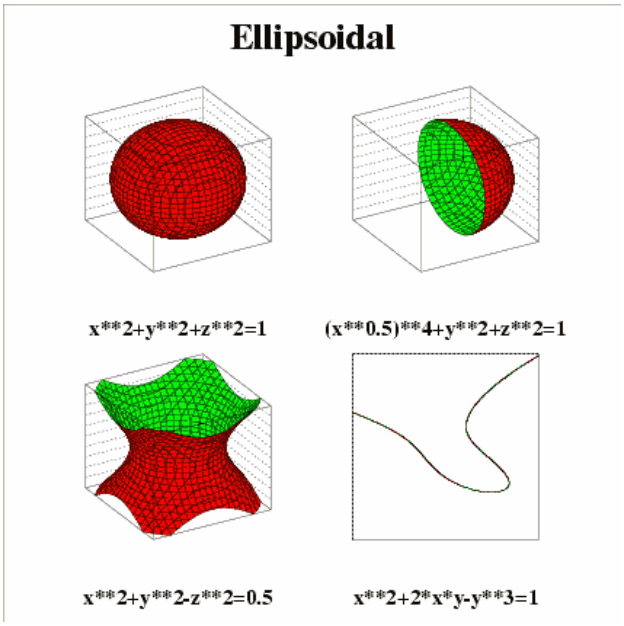

図 5　陰関数 $f(x,y,z)=C$ のグラフ：右下図は視点を設定して xy 平面図形を描いています．

ユーザー関数

FORTRAN（自由書式）で別ファイルに書かれたユーザー関数を実行時に呼び出すことができます．例えば次のような内容のファイル userf.f を用意しておき，

```
function userf(t,r)
if(r.gt.1) userf = -1/(r**2)
if(r.le.1) userf = -r
if(t.ge.0.5) userf = -1
end
```

PAW からは以下のようにユーザー関数 userf を呼び出すことができるのです．

```
fun2 100 userf.f 36 0 1 30 0 3  SURF,POL
```

POL を指定しているので極座標系における関数 $z=f(\theta,r)$ と解釈されます（図 6）．

他にも盛り沢山の機能があるのですが，誌面も尽きましたのでこの辺で筆をおきます．おっと最後に，もし日本語を入れたいなら pstoedit[2]W[3] で tgif のデータに変換できますから，tgif 上で編集・加工してください．

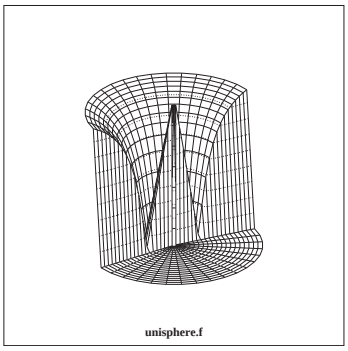

図 6　ユーザー関数を極座標系で与えた場合の描画例

参考文献

[1] CERN にある Paw の公式サイト W[3]
http://wwwinfo.cern.ch/asd/cernlib/version.html

[2] PostScript からドローツールへのデータ変換ユティリティ pstoedit の公式サイト W[3]
http://www.pstoedit.net/pstoedit