J805 と PLOT

SHIMURA Masato
JCD02773@nifty.ne.jp

2016 年 8 月 3 日

J の plot パッケージ

J の plot パッゲージは世界中の大企業が用いる有名な基幹会計パッケージに採用されており、完成度は高い。
J8 の QT 版への対応も早く、J6 との相違も極少ない。
QT 版の plot の機能を確認しながら J8 版での入門編を編んでいくこととする。

1　準備

1.1　plot パッケージ

最初に plot パッケージをロードする

```
require 'plot numeric trig'
require 'jpeg png'
```

plot パッケージのロード時に次のパッケージもロードしておくと便利である

numeric　数値計算のユーティリティー
trig　円関数
jpeg png　jpeg や png でのセーブ

1.2　Plot のマニュアル

J804 のリリース（2015/12 末）に合わせて *jsoftware.com* の *WIKI* ページが大幅に更新され、大幅な *HTML* 化が図られたが、戸惑うこともある。
Plot のマニュアルは *PDF* 版がなくなり、*HTML* 版が *WIKI* の *MainPage* に組み込まれ、探しづらくなっている。

1. 次のようにたどると *plot* に辿り着く
   WIKI→MainPage→Guides→FrameWorks →Plot Packages

2. ここの *plot* のマニュアルは辞書で、学習には不向きである。次のタグをクリックすると項目ごとに出てくる

PLOT	*Verbs*	*Class*	*Commands*	*DATA*	···

1.3 plot のデモ

plot のデモは
Help → *Studio* → *Labs* → *PlotPackage*
に入っている。ロードして *Ctrl* + *J* でページ送りしながら概略を体験できる。
このデモの ***ijt*** をプリントすると有用なスクリプトが多く得られる。

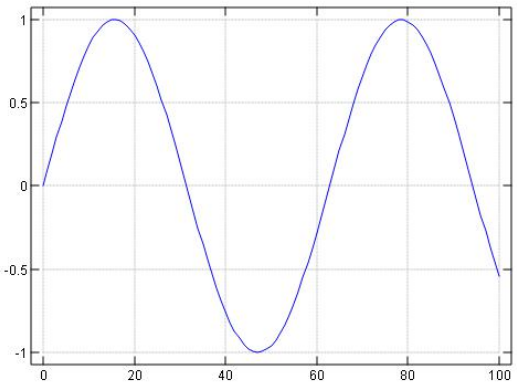
2 plot 入門

タイプ	用途
plot	簡易なグラフ出力 ファンクションプロット
pd	本格的なグラフ出力

2.1 先ず、描いてみる

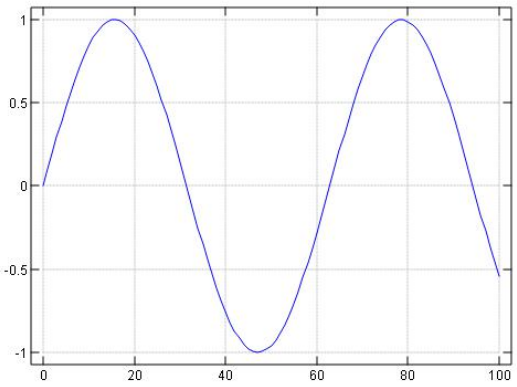
```
plot sin steps 0 10 100
```

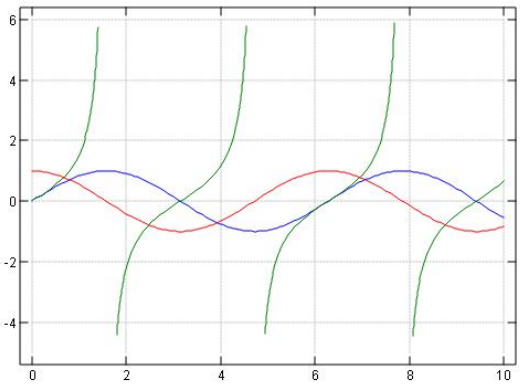
steps で *0,10* の間を *100* に分割してスムーズ化している



ファンクションプロットと書式例

- ファンクションプロットは関数を描く場合に便利な簡易書式である。
  `plot 0 10;'sin'`
- ; セミコロンで区切る
- 書式は `plot 区間;' 関数'` であり、関数は'(クオート）で囲む。
- 関数を並べるときは `'sin ‘ cos ‘ tan'` のように`‘`(チルド) で区切ると便利である





３つ関数を描く
`plot 0 10;'sin ‘ cos ‘ tan '`
色はデフォルトでは青 *(sin)*、赤 *(cos)*、緑 *(tan)*

2.2 データの書式

1. データには次の形がよく用いられる

$$\begin{array}{cc} x_0 & y_0 \\ x_1 & y_1 \\ x_2 & y_2 \\ x_3 & y_3 \\ \vdots & \vdots \\ x_n & y_n \end{array}$$

2. *J* の *plot* に渡すデータは次の型である

x_0	x_1	x_2	x_4	$\cdots$	x_n	y_0	y_1	y_2	y_4	$\cdots$	y_n

3. *3D* では次のようになる

$$\begin{matrix} x_0 & y_0 & z_0 \\ x_1 & y_1 & z_1 \\ x_2 & y_2 & z_2 \\ x_3 & y_3 & z_4 \\ \vdots & \vdots & \vdots \\ x_n & y_n & z_n \end{matrix}$$

4. J の *plot* に渡すデータは次の型である

x_0	x_1	x_2	x_3	$\cdots$	x_n	y_0	y_1	y_2	y_3	$\cdots$	y_n	z_0	z_1	y_2	z_3	$\cdots$	z_n

5. 縦型のデータを横型にして *x,y* をボックスに入れるには次が便利である

```
   (i.6),. 6?6
0 1
1 5
2 4
3 3
4 2
5 0
```

```
 {@|:  (i.6),. 6?6
+-----------+-----------+
|0 1 2 3 4 5|4 0 2 1 3 5|
+-----------+-----------+
```

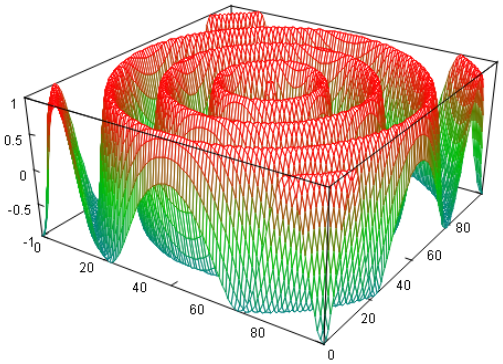
6. 横型のデータは単項の{（カタログ）でボックスに入れる。*2D* でも *3D* でも用いることができる。

```
  { (i.6),  6 2?12
0 1 2 3  4  5
9 7 0 3 11 10
4 2 0 0  0  0
```

```
   { (i.6),  6 2?12
+-----------+------------+------------+
|0 1 2 3 4 5|6 10 0 5 7 8|10 8 0 0 0 0|
+-----------+------------+------------+
```

7. 次のような簡潔なスクリプトで *3D* が描ける

```
'wire'  plot (sin % tan) a=. %: +/~ *: i:20j99
```



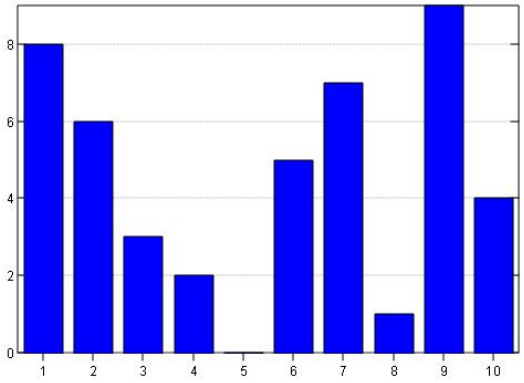
- *i:*は *i.* を両側に打ち出す。`steps _5 5 10` と同じ

```
      i:5
_5 _4 _3 _2 _1 0 1 2 3 4 5
```

- *i:20j99*
  $-20 \leftrightarrow 20$　の間を *99* 分割する
- `%: *:` は $\sqrt{n}, n^2$
- +/ 両項型は足し算の外積
- *3D* のタイプには *wire, surface* がある

8. *2D* で *x* がなく *y* のみの一行の場合は自動で `i.# y` が挿入される

' bar' plot 10 ? 10



2.3　plot の書式

type　*3D* 柱状図と流行りのバルーンはないが、柱状より精密な *stick* はある。[*1]

20 種の *type* は Mainpage → Guide → FrameWorks → plotPackage の Explore → Type で紹介されている。

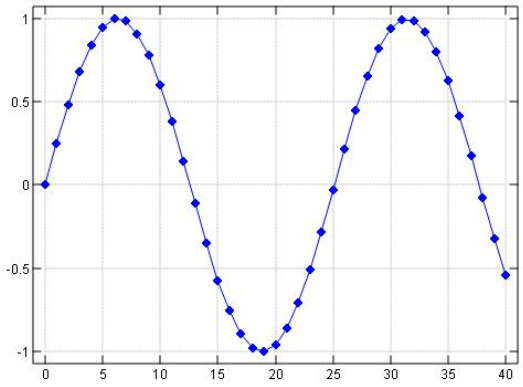
[*1] plot のマニュアルの Extention に様々な大きさの円とピラミッドプロットが紹介されている

点	dot, point, marker(星)
線	line
柱	bar, sbar, fbar, stick, hilo
面	area, polygon
円	pie
3D	wire, surface と line, stick
3D → 2D	contour, density

type　*type* の例

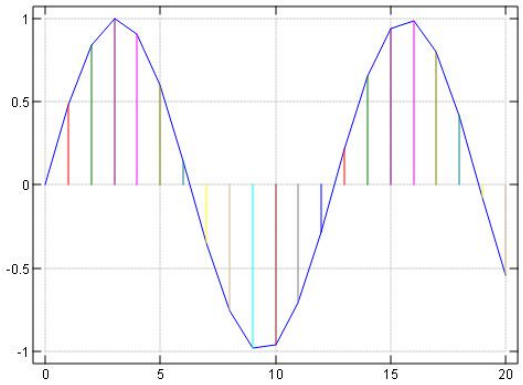
- *line,marker*

'line ,marker' plot sin steps 0 10 40



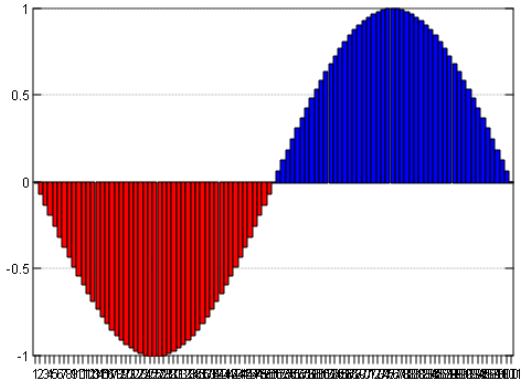
- *line, stick*

'line ,stick' plot sin steps 0 10 40



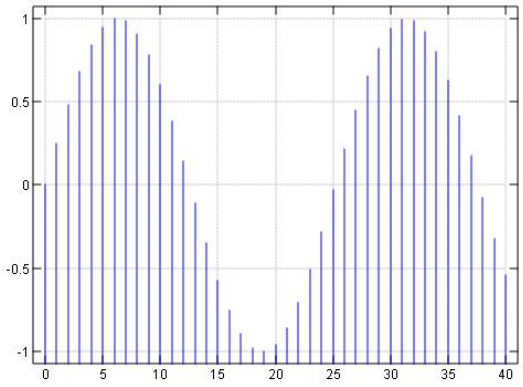
- *bar*

```
'bar' plot _1p1 1p1 ; 'sin'
```



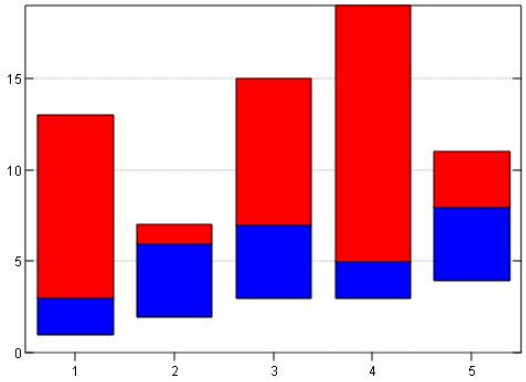
- *hilo*

’ hilo’ plot sin steps 0 10 40



- *fbar*

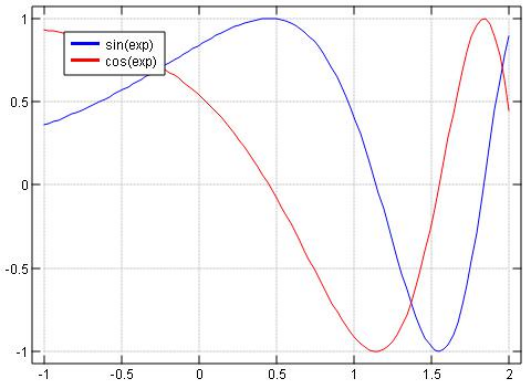
```
   ] a=. >: ?>: i. 3 5

 1 2 3  3 4
 2 4 4  2 4
10 1 8 14 3

’fbar’ plot a
```



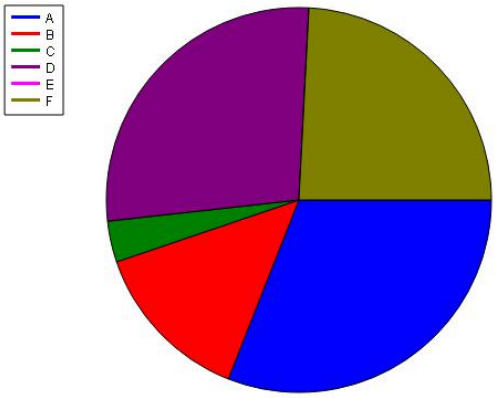
fbar では テーブルの上段は白抜き、中段は青、下段は赤で表示。*sbar* では下段も表示される

key *key* の例

```
’key sin(exp),cos(exp)’
plot (;(sin,:cos)@:^) -: >: i:3j10
```



'pie; key A B C D E F ' plot 6?10



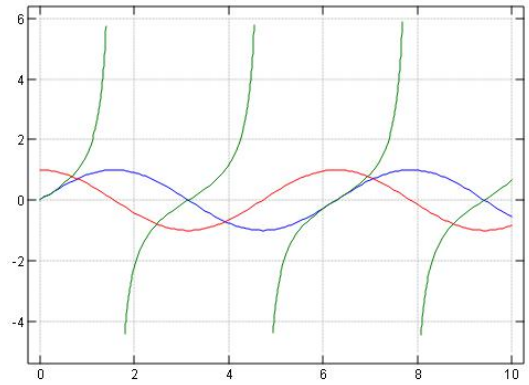
2.4 画像のセーブ

1. 画像のセーブには *pd* を用いる [*2]
2. *eps* は簡単に使える
3. *jpeg png* は先に次のファイルを読み込んでおく
   require 'jpeg png'
4. 線の多い図は *jpg* よりも *png* の方がきれいに出るようで、*Tex* でも *jpg* と同じ書式で通る。
5. *jpeg* や *png* は *Tex* ではバウンディングボックスでサイズを指定すれば用いることができる。
   `\includegraphics[width=7cm , bb=0 0 480 360]{plot_primar05.png}`
   [*3]

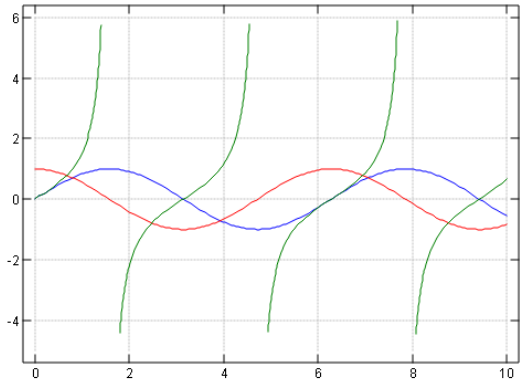
```
plot 0 10;'sin ‘cos‘ tan'

pd 'eps  c:/temp/plot_primar02.eps'

pd 'save jpg c:/temp/plot_primar02.jpg'

pd 'save png c:/temp/plot_primar02.png'
```

[*2] plot driver
[*3] pd 'clip' は QT 版ではサポートされなくなったようだ

jpg



png



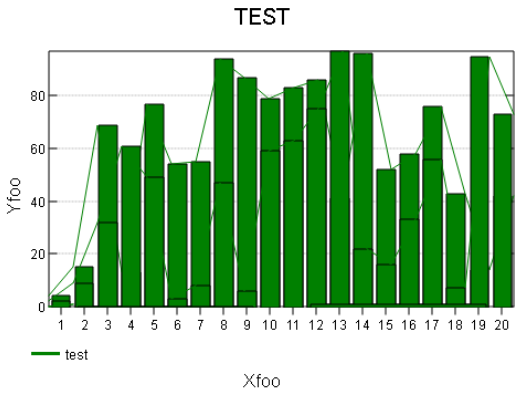
3　本格的なグラフィックス

論文やプレゼンテーション用に表題や説明要素も入れたグラフを作成するためには ***pd*** *(plot driver)* で記述する

3.1　pd

pd による本格的な *plot* の例。*pd 'reset'* または *pd 'new'*　から *pd 'show'* までに詳細を書き込む。この例では 画面は行数分（*5* 枚）連続してセーブされる。

```
  plot_test1=: 3 : 0
DAT=: 5 20 $ 100?100
pd 'reset'
pd 'type bar,line'
pd 'keypos bo'    NB. botom out
pd 'keystyle ho'  NB. horizontal ope
pd 'xcaption Xfoo'
pd 'color green'
pd 'title TEST'
for_ctr. i. # DAT do.
pd ctr{DAT
pd erase '<EPSREADER_j_'
pd 'eps /temp/test_',(": ctr),'.eps'
 end.
pd 'show'
)
   test_plot ''
```



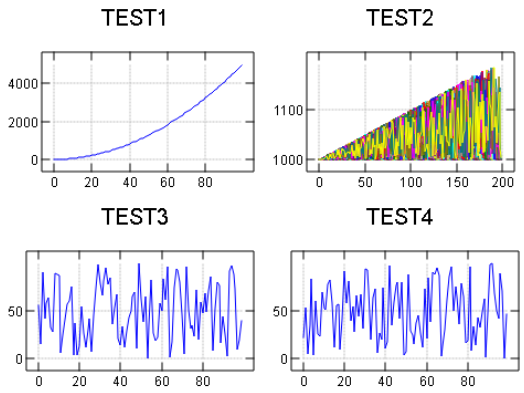
J6 では連続描画と連続ファイルセーブをするときに次を入れて中断を防いだ。*J8* ではなくとも連続でできるようだが、このコマンドも引き続きサポートされているようだ。

```
pd erase '<EPSREADER_j_'
```

3.2 マルチ画面（１）

プレゼンテーションなどでは一枚の図にいくつかのグラフを入れたいことがある。

```
   plot_test3=: 3 : 0
pd 'sub 2 2'
pd 'new'
pd 'title TEST1'
pd +/\i.100
pd 'new'
pd 'title TEST2'
pd 1000++?\i.200
pd 'new'
pd 'title TEST3'
pd 100?100
pd 'new'
pd 'title TEST4'
pd 100?100
pd 'endsub'
pd 'show'
)
```



4 枚の画面を *sub endsub* で定義し、画面一個
ずつ *new* から始める

3.3 マルチ画面（２）-オブジェクト

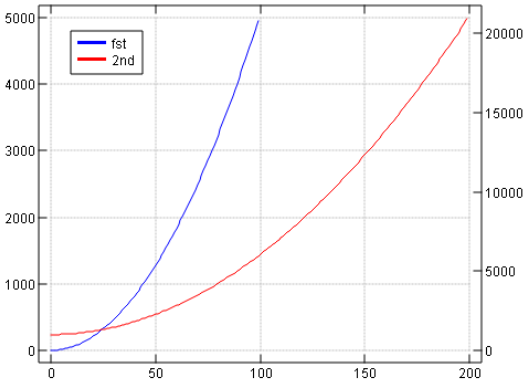
こちらは *PC* の画面上に複数のグラフを並べる方法。
オブジェクトを用いると数画面を同時に表示できて見比べるのに便利である

```
load 'jzplot'          NB. load plot class
a=: conew 'jzplot'     NB. create plot object a
b=: conew 'jzplot'     NB. create plot object b
c=: conew 'jzplot'     NB. create plot object b

plot__a  ? >:i.10      NB. draw plot in a
plot__b  ? >:i.10      NB. draw plot in b
```

3.4　Y 軸の単位を複数表示する

pd 'y2axis' で可能

```
plot_test2=: 3 : 0
pd 'reset'
pd 'key fst 2nd'
pd +/\i.100
pd 'y2axis'
pd 1000++/\i.200
pd 'show'
)
```



4　いろいろな座標で

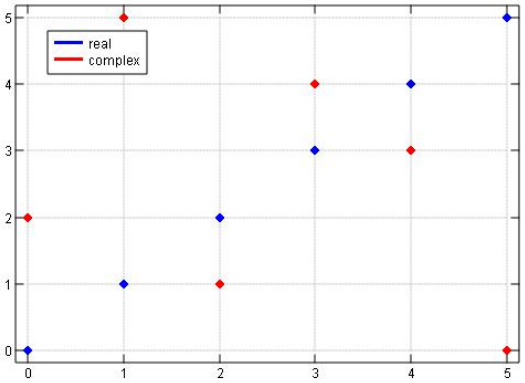
4.1　複素数

数学では複素数の図ははアルガン座標やガウス座標で説明される。この複素数を 2 元数と考えれば、複素数を実部と虚部に分離して標準のデカルト座標に x,y として表示できる。数学と異なり、x,y データを一つの数として取り扱い、同一の座標に表示できる。

J のタートルグラフィックスは x,y データを複素数の一つの数として保持しこの方法で描画している。

```
   ] a=.  (6?6)j. 6?6
4j1 5 0j3 1j4 3j2 2j5

   +. a
4 1
5 0
0 3
1 4
3 2
2 5
```

```
pd 'reset'
pd 'key real complex'
pd 'type marker'
pd i.6
pd (6?6)j.6?6
pd 'show'
```



もう一つ、グラフィックツール　*viewmat* でも複素数を扱うことができる

'require viewmat'

viewmat は複素数のテーブルで、複素数の方向を矢印で表示する。(*Studio　LAB* の *viewmat* を参照)