J 言語の PLOT 入門
ビジネス・サイエンスグラフの作成

SHIMURA Masato
JCD02773@nifty.ne.jp

2009 年 4 月 28 日

目次

1 入門編

- J の plot は世界で有名な巨大ビジネスソフトに組み込まれているらしい。頻繁に細かい点が改良されており、最先端の機能が利用できる。ビジネスグラフで *EXCEL* を持ち出すことはない。
- J602 レベルで記述する。J602 での変更点は

  ディレクトリ・フォルダ　J602 はレジストリを切断して、USB や CDROM から起動できるようにした。インストール先を選ばなくなった代償として plot の出力先を明示して指定しなければならなくなった。

  WIKI　*plot* の User-manual が WIKI になった。一括 DL も可能。

  HELP → *USR* → *PLOT* で WIKI に繋がる
- plot を用いる場合に最初に次の一行を *ijx* か *ijs* に書いておく。

  `require 'plot numeric trig'`

  trig は円関数、numeric は便利な util.
- 詳細なチュートリアルは LAB にある。

- http://www.jsoftware.com/にいろいろな Example がある

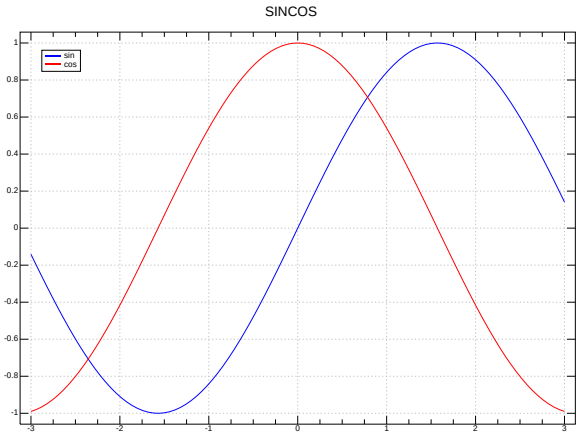
1.1 簡易用法/Function plot

A　いつ頃かそっとサポートされていた関数と区間の簡易表示法

Z　X 軸が簡単に指定できて便利だ。

```
plot _5 5 ;'sin,cos'
```

; で区間と関数を区切り、関数を'' で囲む。




```
'title SINCOS;key sin cos'
  plot  _3 3;'sin,cos'

pd 'eps /temp/ara_01.eps'
```

A　関数は tacit（関数型）は当然で explicit（明示）型も用いることができる。

Z　ちょっとした図はこれで足りる。関数を 2 個続ける時は *sin*, *cos* のように (,) で区切る

```
   test=: 3 : '3 * y ^ 4'

  test i.10
0 3 48 243 768 1875 3888 7203 12288 19683

 plot  test i:5
 plot _5 5 ; 'test'
```

1.2 SAVE

A　SAVE も入門レベルでレビューしよう。次の 3 通りが簡潔で便利である。

clip	クリップボードに入る。	*pd 'clip'*
eps	*TEX*	*pd 'eps /temp/foo.eps'*
emf	*HTML etc*	*pd 'save emf /temp/foo.emf'*
print	*printer*	*pd 'print'*

Z　この例ではファイルは */temp* に入る。

A　他の形式は C.Reiter のアドオンを組み込まなければならない。

1.3　plot の種類

A　type で指定する。, で複数を同時に用いることもできる。*line*, *stick*

Z　`'key sin cos;type marker,stick'  plot  _3 3;'sin,cos'`

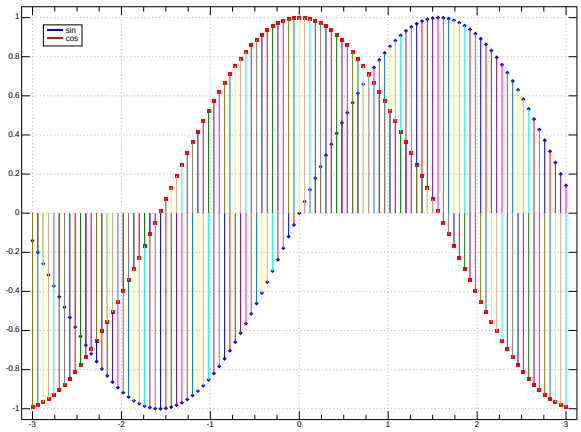
A　2D の種類は 16 もある。マニュアル参照

デフォルト　*line*

bar

marker　大きな点

point　細かい点　*CAOS* の図



1.4　2D グラフ

A　正式な *XY* の書式は次の通り。; で *XY* を連結する。Y は何行あっても良い。

```
   (i.10); 2 10 $ 20?20
   X                       Y1,Y2
+------------------+-------------------------+
|0 1 2 3 4 5 6 7 8 9| 3 13 1 15  4 12 9 2  7 18|
|                   |14 16 8  6 19 10 0 5 11 17|
+------------------+-------------------------+
```

A　X は省略できる。（J が X を 0 からの順序数を付ける）

Z　X と Y の横の長さは必ず会わせる。大きなグラフでエラーになるときはここが原因のこと

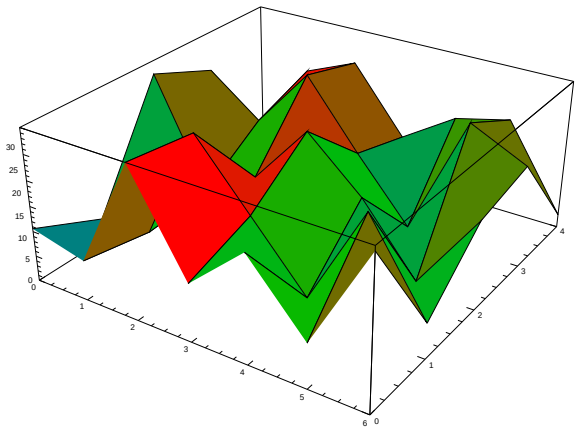
が多い。

1.5 3D

A　*X*; *Y*; *X* の構成は次の通り。YZ は交換しても良いが XYZ は必須

Z　3D は *line wire surface stick* とグラフの種類は制限される。3DBAR はない。

```
   (i.7);(i.5) ;5 7 $ 35?35
      X           Y           Z
+-------------+---------+--------------------+
|0 1 2 3 4 5 6|0 1 2 3 4|25 11 22 33 21  1 24|
|             |         |19 15 30  3 14  2 32|
|             |         |28 29  8 26 23 34  0|
|             |         | 9  7  4 18 17 20 10|
|             |         |13  6 27 31 16  5 12|
+-------------+---------+--------------------+
```



1.6 いろいろと

1.6.1 color

次の 16 色はデフォルトで入っている。(packages/color/color16.ijs) colortab.ijs をロードすると多くの色が使用できる。

'color blue' plot i.12 の様に指定できる。

```
Aqua=:      0 255 255
Black=:     0   0   0
Blue=:      0   0 255
```

```
Fuchsia=: 255   0 255
Gray=:    128 128 128
Green=:     0 128   0
Lime=:      0 255   0
Maroon=:  128   0   0
Navy=:      0   0 128
Olive=:   128 128   0
Purple=:  128   0 128
Red=:     255   0   0
Silver=:  192 192 192
Teal=:      0 128 128
White=:   255 255 255
Yellow=:  255 255   0
```

色の指定は直接 **RGB** でもできる。

```
 'color 0 255 255' plot i.10

pd 'color 0 255 255'
```

1.6.2 steps

numeric に入っている *steps* の用法。X 軸を細かく指定し、滑らかな曲線を描く。100 は刻みの数。

```
 'key sin' plot tmp; sin  tmp=. steps _5 5 100
```

1.6.3 Key

Key はよく使うので使う部分をメモしておこう。**o***pen* とは *keystyle o* でも *keystyle open* でも良い。

```
 'keypos bottom;key sin' plot tmp; sin  tmp=. steps _5 5 100
 'keypos ob;keystyle ho;key sin' plot tmp; sin  tmp=. steps _5 5 100
```

keypos :	**l***eft*	**c***enter*	**r***ight*
	t*op*	**m***iddle*	**b***ottom*
	i*nside*	**o***utside*	
keystyle	**b***oxed*	**o***pen*	
	v*ertical*	**h***orizontal*	

1.7 日本語

J のバージョン 5 以降は UNICODE をサポートしている。WIN は XP 以降。[*1]

J の IJS の組み込みエディタは UNICODE の入力が出来ないが認識はできる。*Notepad* 等の外部エディタで書いて *Cut&Paste* で ijs に張り付けると plot で日本語や多国籍文字が表示できる。

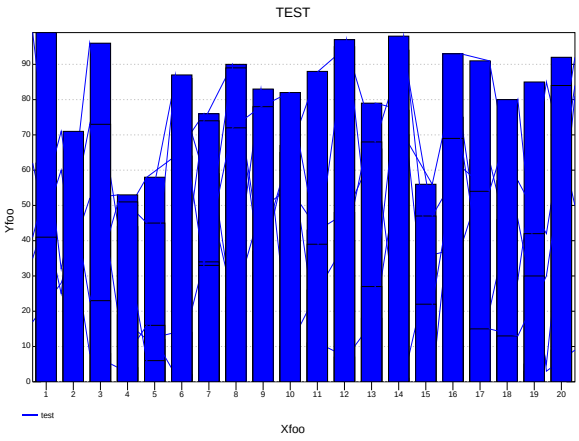
```
'title 日本語テスト' plot i.12
```

pd でも可能で細かな記述ができる。

[*2]

1.8 正調で

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



A　本格的に書き上げると次のようだ。次の 3 行は必須

pd 'reset' または pd 'new'

pd data

pd 'show'

[*1] WIN2000 は UNICODE の取り扱いが異なるようだ。

[*2] eps 出力では崩れる。画面キャプッチャしなければならない

Z　erase　は隠しコマンド。eps を連続して出力すると「書きました」とダイアログボックスを出してその都度停止する。この表示を消し連続出力するもの。(Oleg Kobchenko から教わった）

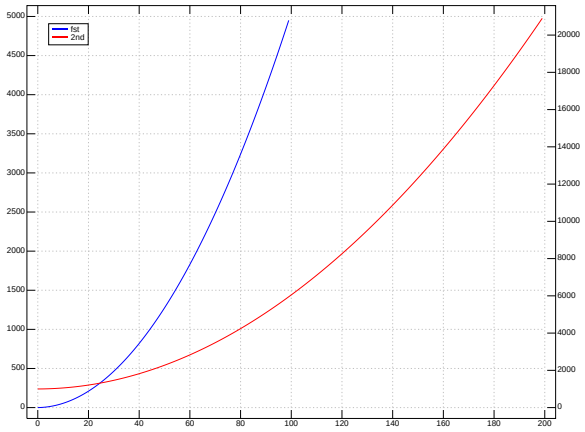
A　ループで連続出力できる。このときカウンターを文字化してファイル名に付け加える。

2　応用編

2.1　y2axis

A　　Y 軸に単位の異なる 2 軸が使える

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



2.2　オブジェクト

A　　複数の plot 画面を使いたい時に用いる。
Z　　確かに 2 枚の画面が現れる

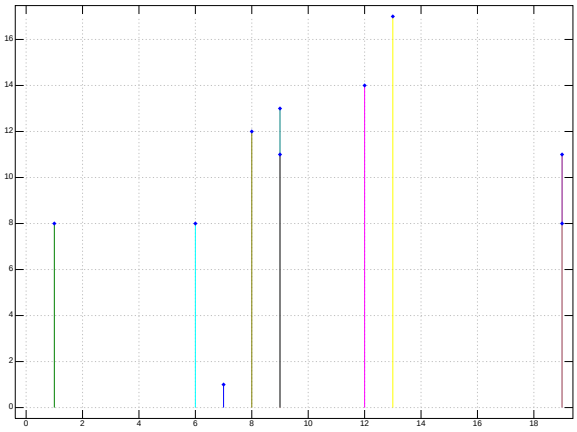
```
require 'plot jzplot'

plot_test4=: 3 : 0
a=. conew 'jzplot'
b=. conew 'jzplot'
plot__a +/\ i.100
plot__b 100?100
)
```

2.3　複素数と plot

A　　1800 年頃に 3 人（ガウス、アルガン、ブエッセル）により独立に提案された。
Z　　これも何時かそっとサポートされていた。
A　　`plot 0 3j4` のように 2 ポイントにすると描画できる

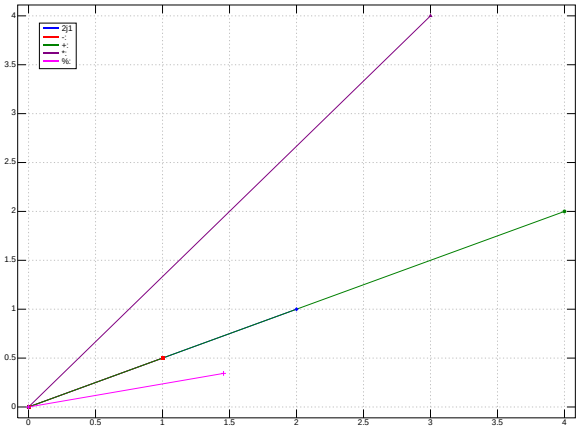
x *real* y *complex*

```
 'stick,marker' plot
 a=. j./("1) |: ? 2 10$ 20
```



```
   (i.10),.a
0   7j1
1  9j11
2   1j8
3 19j11
4 12j14
5  8j12
6  9j13
7 13j17
8  19j8
9   6j8
```

```
 0,.  ]` -: `+:`*:` %:`:(0)  2j1
 NB. evoke gerund
```



```
0              2j1
0            1j0.5
0              4j2
0              3j4
0 1.45535j0.343561
```

$-:$ *half*

$+:$ *double*

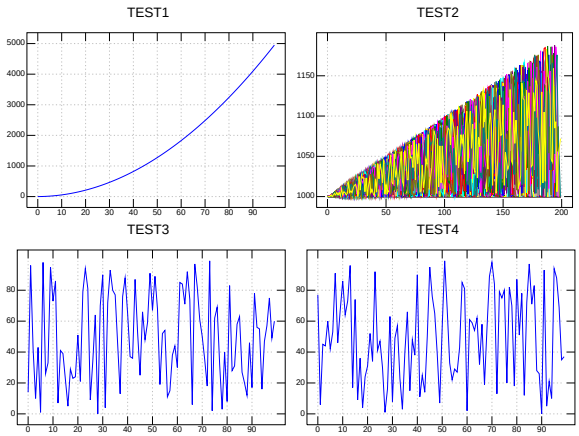
$*:$ ^{2} *length* $\times 2$ *angle*

$\%:$ $\sqrt{}$ *length* $1/2$ *angle*

2.4 マルチ画面

A　　sub を用いた画面分割。一個毎に *new* か *reset* をしなければならない。

```
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'
)
```



3 グラフィックスと plot

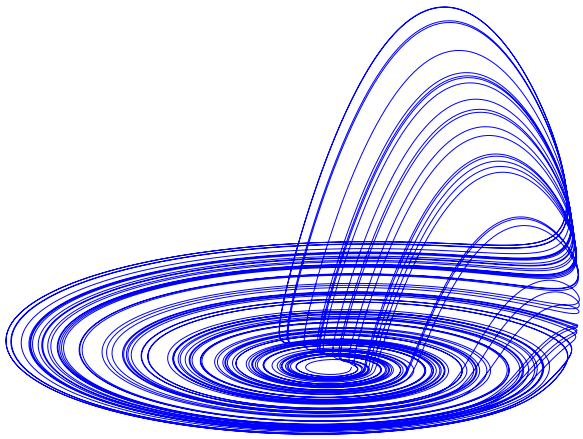
3.1 rossler アトラクタ

A　　plot の高度機能をグラフィックスに用いる。

Z　　*plot_lorenz* は左にいろいろな関数（図形の数学表現）をとれように副詞型（1：0）を用いている。副詞型は u で左の関数を引く。*noaxis,viewpoint* でグラフィックスらしくしている。（line 専用になっており type の指定には改良が必要）

```
 calc_rossler plot_lorenz 1 1 1
```

```
NB. rossler attractor
dt=: 0.005
rsr=:4 : 0
NB. Rossler attracta
```



```
'a b c'=. x  NB. 0.2 0.2 14
'xx yy zz'=: y
X=: xx + dt*-(yy+zz)
Y=: yy + dt* xx + a*yy
Z=: zz + dt* (b+ (xx*zz)-c*zz)
X,Y,Z
)
 calc_rossler=:3 : ' <"1 |: (0.2 0.2 14)&rsr ^:(i.100000) y'
NB. 0.2 0.2 14&calc_rossler plot_lorenz 1 1 1


  plot_lorenz=:1 : 0
NB.  10 50 8r3&calc_lorenz plot_lorenz init
pd 'reset'
pd 'noaxes'
pd 'viewpoint 1.6 _2.4 _1.3'
pd u y
pd 'show'
)
```

3.2 フォームエディタに張り付ける

A　*Form editor* は ijs の画面で EDIT に入ると下の方にある。画面設計が簡単に出来て便利だ。グラフィックスには *isigraph* を用いる。*plot* を載せると plot の画面が単独でポップアップするのでオブジェクトを用いよう。

Z　WIKI のユーザーマニュアルに従って作ってみよう。isigraph のキーワードは *g*0 ととする。

図を選択できるように *combobox* を付け加えよう。こちらのキーワードは *sel* とする。

```
xywh 11 10 182 162;cc g0 isigraph;
xywh 213 22 46 37;cc sel combobox;
```

A　オブジェクトの簡易な作成方法が上手く掴めていない。取り敢えず *PForm* からの 3 行は WIKI の EXAMPLE の通りとしないと plot が popup してしまう。

```
myplot_run=: 3 : 0
wd MYPLOT
NB. initialize form here
loc=: conew 'jzplot'  NB. set object
PForm__loc=: 'myplot'
PFormhwnd__loc=: wd 'qhwndp'
PId__loc=: 'g0'
setcombobox_sel ''
wd 'pshow;'
)
```

A　取り敢えず図は *density lorenz rossler* の 3 とし、combobox に登録する。後で幾らでも追加できる。文字列にシーケンシャルマシーン (;:) を使用している。

```
setcombobox_sel=: 3 : 0
NB. multiple select listbox with 4 items, initial selection=1
boxList=. ;:'Density Lorenz Rossler' NB. add any
wd 'set sel ',;boxList ,each LF
wd 'setselect sel 1;'     NB. top set Lorenz
)
```

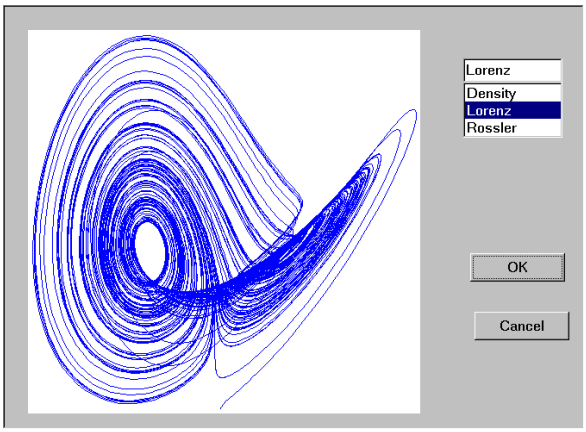
Z　マウスで選択すると *sel* に選んだ文字列が入る。ここから先が難解である。 文字列を選択した時の選択の認識は最近（いつ頃かよく分からないが）*sel_select* の様にキーワードに select を付けて、数値化すればよくなった。
ここから一気に書き上げてしまおう。

```
myplot_sel_button=:myplot_ok_button=: 3 : 0
NB. IND=. I. tmp e. >./ tmp=. ; +/ L:0 sel e. L:0 boxList
IND=. ". sel_select
if. 0 = IND do. 'density' plot__loc 7|i.25 25 end.
if. 1 = IND do. 'noaxes' plot__loc calc_lorenz 1 1 1 end.
if. 2=IND do.  'noaxes' plot__loc calc_rossler  1 1 1 end.
```

```
)
```

A　`myplot_run''`

Z　最初に OK ボタンを 2 度クリックしなければならないのはご愛敬。*rossler* は 10 万回リピートしているので少々 k 時間がかかる。



4　拡張

4.1　経済月次データの X 軸の自動生成

A　経済データは月次が多い。X 軸に年を簡単に示したい。

Z　面倒な方法を避けて自作する。[*3] X 軸の表示は煩雑にならないように J が適当に選んでくれるし、長さも自動。XY の個数（長さ）には厳格で異なるとエラーになる。

```
 xaxis_sub=: 4 : 0
NB. Usage: plot (1994 xaxis_sub tmp );tmp=. 200?200
NR=.>. (#y)%12
XAXIS=.(#y){. ;({@>x+i.NR)+ L:0 ,. NR # < (>:i.12)%12
)
```

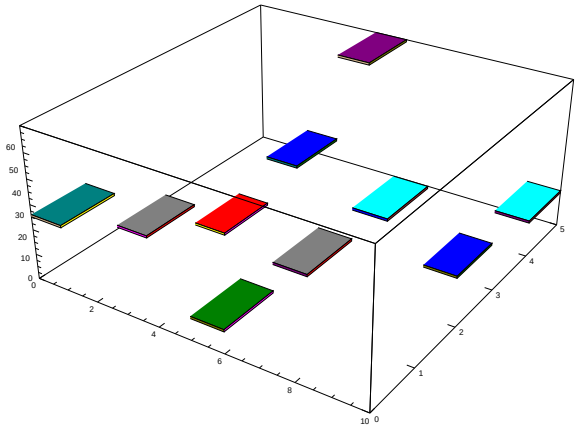
`plot (1994 xaxis_sub data);data=. 200?200` NB. start at 1994/JAN

4.2　3DFbar

A　Oleg Kobchenko が数年前に簡易 3DFbar を発表していた。Oleg のページは
`http://www.jsoftware.com`
からも辿れるが改良されてはいないようだ。

[*3] 最初の年が 4 月から始まるときは先頭の 3 個を落とす。`(3}.)`

Z　　fbar は隠れる部分が少ないので多量のデータには向いているが、少ないときは分かりづらい。



Z　　Oleg のソースコードはたったのこれだけだ。

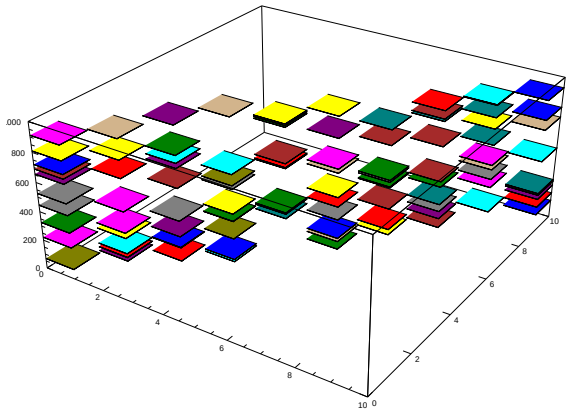
```
cut=: <"2 @ (0 1&|:)  NB. 3-array to list of boxed matrices
cubic=: 3 : 0         NB. unit 3-cube
  v   =. 2 2 2#: i.8
  ndx =. 0 2 6 4, 4 6 7 5, 1 5 7 3, 0 1 3 2, 2 3 7 6,: 0 4 5 1
  ndx { v
)
cubes=: [: cut  cubic ,/@:(+"1"_ 1) ] NB.  unit cubes shited by list of 3-coords
plotcubes=: 'poly'"_ plot cubes       NB. 3d plot of list of 3-coords
view3d=: [: plotcubes ($ #: I.@,)     NB. 3d boolean viewmat
```

A　　箱を浮かせる発想のようだ。

```
  plotcubes a
```

```
    a=. (i.10),.((i.5),i.5),.10?100
   a
x y z
0 0 29
1 1 14
2 2  5
3 3 28
4 4 68
5 0 10
6 1 23
```

```
7 2 37
8 3  0
9 4 16
```



Z　　xy を敷き詰めるようにしたら 3DFbar らしくなる。

```
a1=. (;10# <i.10)
plotcubes a1,.a1,.100?1000
```

Reference

Author: SHIMURA Masato

J Language: http://www.jsoftware.com　Download Available

SRC: http://homepage3.nifty.com/asagaya_avenue → APL → workshop