JAPLA 研究会資料 2021/4/15

# Jによるフラクタルグラフィックス
# H-フラクタルとTree-フラクタル

西川　利男

JAPLA 先月の研究会で、Jによる正方形フラクタルと円フラクタルについて報告した。フラクタルの基本的な図形生成の原理を見直し、Jに適したプログラミングスタイルの方式を確立できた、と思っている。[1]

[1] 西川利男「Jによるフラクタルのグラフィックス」
JAPLA 研究会資料　2021/3/25

今回は、この方法をH-フラクタルとTree-フラクタルとに適用してみた。
なお、H-フラクタルとTree-フラクタルとは、私の訳書「初めてのフラクタル」[2]の訳書に以下のように記載されている。前回にも述べたが、同書にはBASICのプログラムがついているが、これを利用することなく、私の方式でJプログラムを作成した。

[2] H. ラウヴェリエール、西川利男訳「初めてのフラクタル
－数学とプログラミング」、丸善(1996).

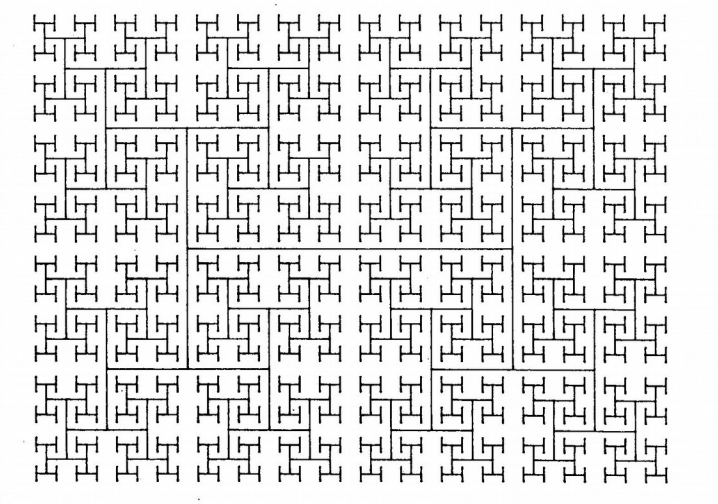


図 1.1　H-フラクタル

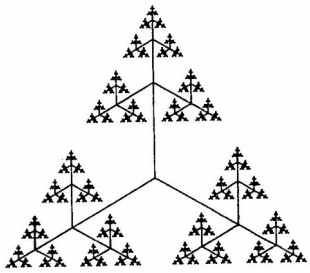


図 1.4　3進の木

## １．Ｊによるフラクタル図形の描き方－段階を追って整理されたプログラムを

フラクタル図形を描くプログラムはいろいろな本に載っているが、ふつうはプログラムを記述して、複雑なフラクタル図形を一度で描くものが多い。しかし、これではフラクタル構造がどう生成されるかは分からない。

私は一段階ずつその都度、図形を描いて、次第に細かく描いて、その過程がわかるようにした。そしてそれに合うようＪプログラムを構造化して記述した。

一方、Ｊプログラミング言語の特徴として、値として配列構造を用いて、それを操作する動詞は、副詞、接続詞も用いて、一瞬で行うことが常である。私はあえてＪのループ構文を用いて、繰り返し構造が分かるようにした。

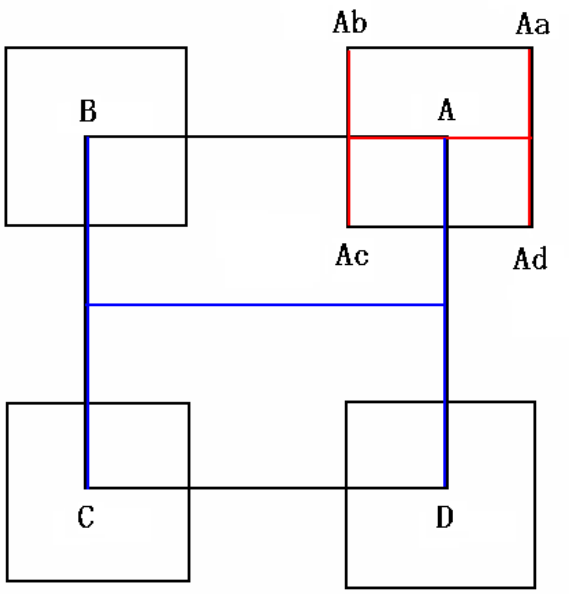
このためにＪの for.－ do. － end. 構文を次のように書式化して用いた。
前回の報告からこの使い方を再記する。

```
loop =: 3 : 0
'N M' =. y.
for_I. i. N do.
  for_J. i. M do.
    wr I, J
  end.
end.
'*** end ***'
)

   loop 2 4
0 0
0 1
0 2
0 3
1 0
1 1
1 2
1 3
*** end ***
```

## ２．H-フラクタルのＪプログラムと実行

H-フラクタルの描き方の基本的な考え方はつぎのとおりである。




ある次数 n での正方形 ABCD をとりあえず頂点を A, B, C, D とする。これに対して次の次数 n+1 での正方形は、中心を A とし、いまの正方形を適当に縮小して Aa, Ab, Ac, Ad としたものとしてつくられる。これは前回の報告で正方形フラクタルとしてを行った方法である。なお、正方形を求めるには前回に報告した動詞 square を用いた。

さらに H-フラクタルでは、正方形の中心と４つの頂点をもとに、タテ線２本、ヨコ線１本の座標を求めてこれらを結んで描けばよい。この座標値の計算には動詞 hfract で行った。

実際に描くには、計算した数学座標値から、動詞 adj により画面のピクセル値に変換して、gl2 コマンド gllines により描く。

以上の操作をまとめてみると次のようになる。

- 正方形の座標値の計算
- H 字図形の座標値の計算
- 数学座標値から画面ピクセル値への変換計算
- gl2 コマンドにより実際の描画

このとき、J の for. － do. － end. 構文を用いて、整理したプログラムとすることが、極めて大切である。

そして、ウィンドウズプログラムとして、ボタンを押すごとに１ステップずつ描くようにした。

## ２．０　０次のH-フラクタルのプログラム

まず最初に０次のH-フラクタルとしてH字の図形を大きく描く。

```
NB. 0 order H ========================================
XY0DA =: square 0, 0, 2
HHH =. hfract XY0DA
HH0DA =: ''
 for_J. (i.3) do.
  HH0DA =: HH0DA, J{HHH
  gllines adj , ; J{HHH
 end.
glshow ''
)
```

## ２．１　１次のH-フラクタルのプログラム

１次のH-フラクタル図形のＪプログラムはつぎのようになる。

```
NB. 1 order H-fractal ================================
fract_RunH1_button=: 3 : 0
XY1DA =: ''
SQ1 =. square(0, 0, 2)
for_J. (i.4) do.
  XYXY =. square (;J{XY0DA), 1
  XY1DA =: XY1DA, XYXY
  for_G. (i.3) do.
    gllines adj , ; G{hfract (square (; J{XY0DA), 1)
  end. NB. G
end. NB. J
glshow ''
)
```

ここでＪの２重のfor.－ do. － end. 構文を用いている。

さらに大切なことは、この次数でのをつぎのフラクタル図形のためにグローバル値として準備している。

## ２．２　２次のH-フラクタルのプログラム

```
NB. 2 order H-fractal ====================================
fract_RunH2_button=: 3 : 0
XY2DA =: ''
for_K. (i.16) do.
  XYXY =. square (; K{XY1DA), 0.5
  XY2DA =: XY2DA, XYXY
     for_G. (i.3) do.
       gllines adj , ; G{hfract (square (; K{XY1DA), 0.5)
     end. NB. G
end. NB. K
glshow ''
```

ここでは、２次H-フラクタルとして、16個のH図形を描いている。
同様に、次の次数３次のために、正方形の頂点座標値XY2DAを準備している。

## ２．３　３次のH-フラクタルのプログラム

```
NB. 3 order H-fractal ====================================
fract_RunH3_button=: 3 : 0
XY3DA =: ''
for_L. (i.64) do.
  XYXY =. square (; L{XY2DA), 0.25
  XY3DA =: XY2DA, XYXY
     for_G. (i.3) do.
       gllines adj , ; G{hfract (square (; L{XY2DA), 0.25)
     end. NB. G
end. NB. K
glshow ''
)
```

今度は、３次H-フラクタルとして、64個のH図形を描いている。

Jプログラムのリスティングの全体は最後にあげてある。

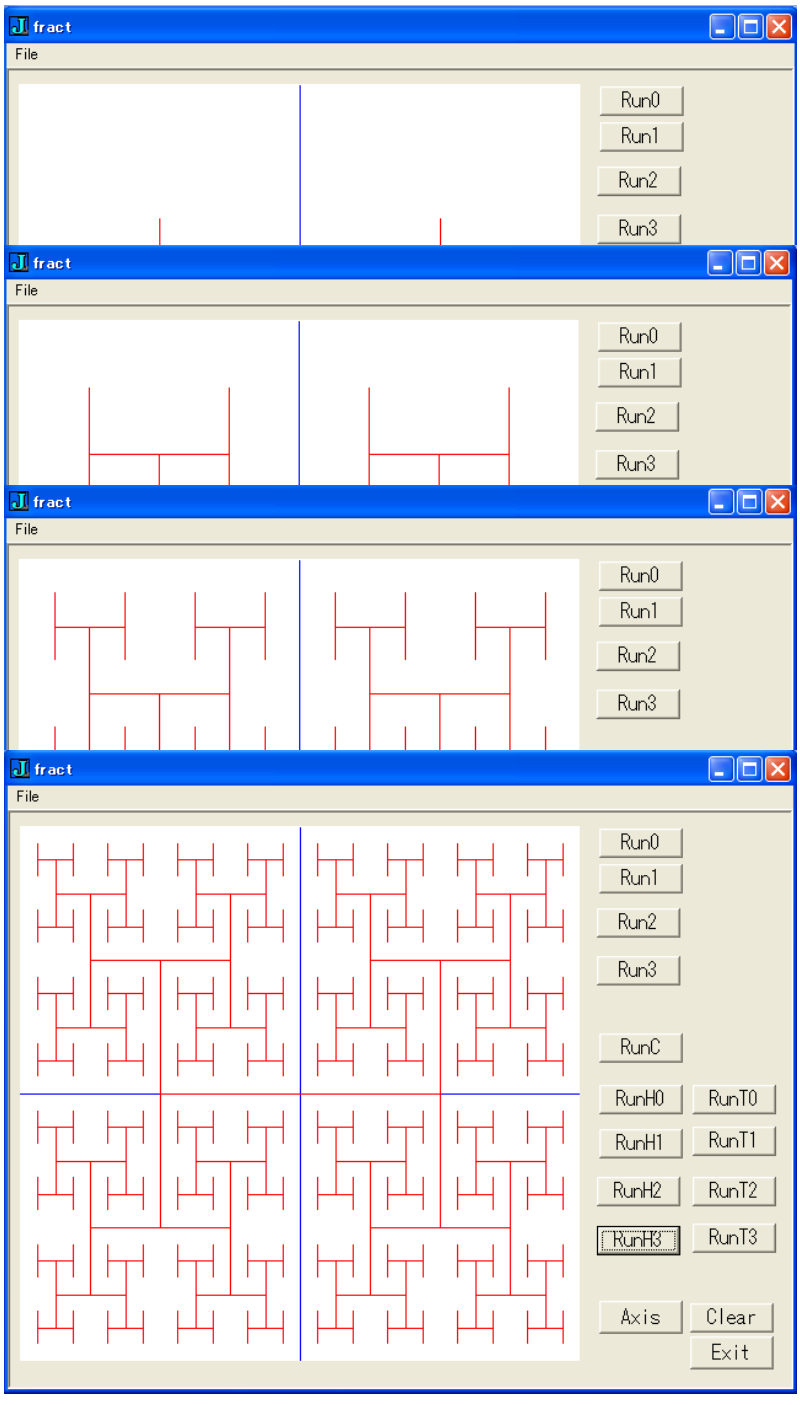
## ２．４　H-フラクタルグラフィックスの実際

ウィンドウズプログラムを

run　‘’

により起動し、ボタンRunH0を押すと0次のH-フラクタルとして画面の中央にH字が大きく描かれる。

続けて、ボタンRunH1を押すと1次のH-フラクタルが追加されて描かれる。




さらに、ボタンRunH2を押すと2次のH-フラクタルが追加されて描かれる。

なお続けて、ボタン3次のRunH3を押すとH-フラクタルが追加されて描かれる。

ようやく、フラクタル図形らしくなってきた。

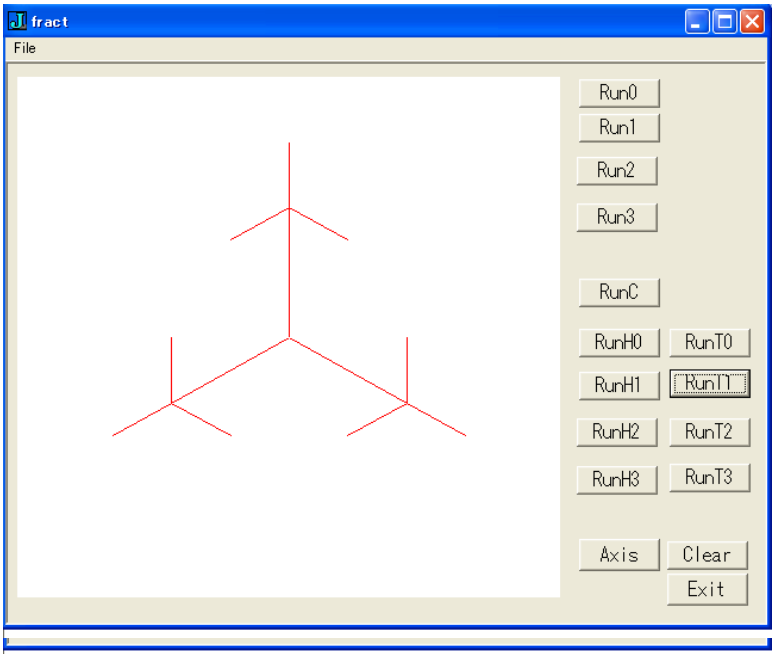
## ３．Tree フラクタルの Ｊプログラムと実際

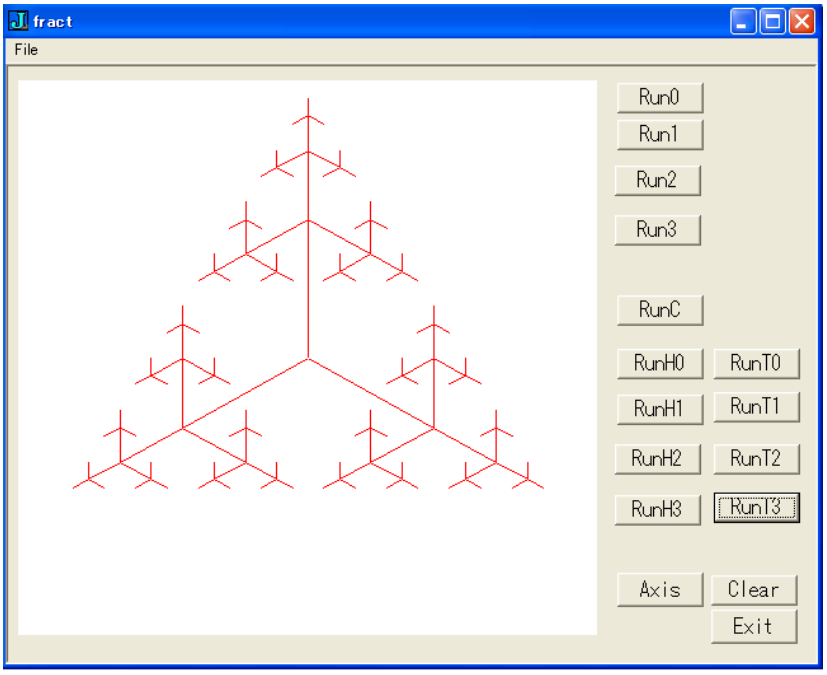
Tree フラクタルでは、まず正三角形の頂点座標を求めてから、中心と各頂点とを結んで、Tree 図形を作り、0 次の Tree フラクタルとする。

次に、今出来上がった３つの頂点を中心として、縮小した Tree 図形を描き、1 次の Tree フラクタルとする。

続いて、同じようにして、2 次の Tree フラクタルとして 9 個の Tree 図形を描く。さらに、同様にして 3 次の Tree フラクタルでは 27 個の Tree 図形を描く。

実行は、ボタン RunH0, RunH1, RunH2, RunH3 を順次押すと、追加表示される。




fract
File
Run0
Run1
Run2
Run3
RunC
RunH0
RunT0
RunH1
RunT1
RunH2
RunT2
RunH3
RunT3
Axis
Clear
Exit

Jプログラム・リスト

```
NB. nfractsq.ijs
NB.  by Toshio Nishikawa 2021/3/12

wr =: 1!:2&2
require 'numeric'
require 'trig'
require 'gl2'

FRACT=: 0 : 0
pc fract;
menupop "File";
menu new "&New" "" "" "";
menu open "&Open" "" "" "";
menusep ;
menu exit "&Exit" "" "" "";
menupopz;
xywh 4 5 228 194;cc gfract isigraph;
xywh 240 177 34 12;cc ok button;cn "Axis";
xywh 277 190 34 12;cc cancel button;cn "Exit";
xywh 277 178 34 11;cc Clear button;
xywh 240 6 34 11;cc Run0 button;
xywh 240 19 34 11;cc Run1 button;
xywh 239 35 34 11;cc Run2 button;
xywh 239 52 34 11;cc Run3 button;
xywh 240 80 34 11;cc RunC button;
xywh 240 99 34 11;cc RunH0 button;
xywh 240 115 34 11;cc RunH1 button;
xywh 239 132 34 11;cc RunH2 button;
xywh 239 150 34 11;cc RunH3 button;
xywh 278 99 34 11;cc RunT0 button;
xywh 278 114 34 11;cc RunT1 button;
xywh 278 132 34 11;cc RunT2 button;
xywh 278 149 34 11;cc RunT3 button;
pas 6 6;pcenter;
rem form end;
```

```
)

run =: fract_run
fract_run=: 3 : 0
wd FRACT
NB. initialize form here
wd 'pshow;'
)

fract_close=: 3 : 0
wd'pclose'
)

fract_cancel_button=: 3 : 0
fract_close''
)

fract_Clear_button=: 3 : 0
glclear ''
glshow ''
)

adj =: 500&+@(250&*)

square =: 3 : 0
'x0 y0 a' =. y.
a2 =. a % 2
p1 =. (x0 + a2), (y0 + a2)
p2 =. (x0 - a2), (y0 + a2)
p3 =. (x0 - a2), (y0 - a2)
p4 =. (x0 + a2), (y0 - a2)
SQ =: p1;p2;p3;p4
)
```

```
NB. H-fractal ========================================
NB. draw H character
NB. hfract centerX, centerY, size(length of a square side)
NB.   => points of H character
hfract =: 3 : 0
HH0 =. y.
HH1 =. < -: (; 0{HH0) + (; 3{HH0)
HH2 =. < -: (; 1{HH0) + (; 2{HH0)
HH =. HH0, HH1, HH2
(;(0{HH), (3{HH));(;(1{HH), (2{HH));((;4{HH), (;5{HH))
)


fract_RunH0_button=: 3 : 0
glrgb 255 0 0
glpen 1, 0

NB. 0 order H ========================================
XY0DA =: square 0, 0, 2
HHH =. hfract XY0DA
HH0DA =: ''
 for_J. (i.3) do.
  HH0DA =: HH0DA, J{HHH
  gllines adj , ; J{HHH
 end.
glshow ''
)

NB. 1 order H-fractal ================================
fract_RunH1_button=: 3 : 0
XY1DA =: ''
SQ1 =. square(0, 0, 2)
for_J. (i.4) do.
  XYXY =. square (;J{XY0DA), 1
  XY1DA =: XY1DA, XYXY
  for_G. (i.3) do.
```

```
    gllines adj , ; G{hfract (square (; J{XY0DA), 1)
  end. NB. G
end. NB. J
glshow ''
)

NB. 2 order H-fractal ====================================
fract_RunH2_button=: 3 : 0
XY2DA =: ''
for_K. (i.16) do.
  XYXY =. square (; K{XY1DA), 0.5
  XY2DA =: XY2DA, XYXY
     for_G. (i.3) do.
       gllines adj , ; G{hfract (square (; K{XY1DA), 0.5)
     end. NB. G
end. NB. K
glshow ''

NB. 3 order H-fractal ====================================
fract_RunH3_button=: 3 : 0
XY3DA =: ''
for_L. (i.64) do.
  XYXY =. square (; L{XY2DA), 0.25
  XY3DA =: XY2DA, XYXY
     for_G. (i.3) do.
       gllines adj , ; G{hfract (square (; L{XY2DA), 0.25)
     end. NB. G
end. NB. K
glshow ''
)
```

```
NB. Tree Fractal ========================================
triangle =: 3 : 0
'X Y A' =. y.
T0 =. (A * (0, 1)) + (X, Y)
T1 =. (A * ((-%:3)%2), _0.5) + (X, Y)
T2 =. (A * ((%:3)%2), _0.5) + (X, Y)
T0;T1;T2
)

NB. Tree 0 order ========================================
fract_RunT0_button=: 3 : 0
glrgb 255 0 0
glpen 1, 0

TR0DA =: triangle (0, 0, 1)
for_G. (i.3) do.
  gllines adj , (; G{TR0DA), 0, 0
end. NB. G
glshow ''
)

NB. Tree 1 order ========================================
fract_RunT1_button=: 3 : 0
glrgb 255 0 0
glpen 1, 0

TR1DA =: ''
for_J. (i.3) do.
  TRR =. triangle (;J{TR0DA) , 0.5
  TR1DA =: TR1DA, TRR
  for_G. (i.3) do.
    gllines adj , (;G{triangle (;J{TR0DA), 0.5), (;J{TR0DA)
  end. NB. G
end. NB. J
glshow ''
)
```

```
NB. Tree 2 order ================================================
fract_RunT2_button=: 3 : 0
TR2DA =: ''
for_J. (i.9) do.
  TRR =. triangle (;J{TR1DA) , 0.25
  TR2DA =: TR2DA, TRR
  for_G. (i.3) do.
    gllines adj , (;G{triangle (;J{TR1DA), 0.25), (;J{TR1DA)
  end. NB. G
end. NB. J
glshow ''
)

NB. Tree 3 order ================================================
fract_RunT3_button=: 3 : 0
TR3DA =: ''
for_J. (i.27) do.
  TRR =. triangle (;J{TR2DA), 0.125
  TR3DA =: TR3DA, TRR
  for_G. (i.3) do.
    gllines adj , (;G{triangle (;J{TR2DA), 0.125), (;J{TR2DA)
  end. NB. G
end. NB. J
glshow ''
)
```