数値計算のスクエア
ジュリア集合 (ニュートン・ラフソン法の周辺)

Shimura Masato
JCD02773@nifty.ne.jp

2009 年 3 月 2 日

目次

概要

ニュートン・ラフソン法は非線形方程式の強力な反復解法である。反復法の前史としては古代メソポタミアのルートの求根法がある。ニュートン・ラフソン法を複素数に拡大するとジュリア集合が得られる。

1 メソポタミアの $\sqrt{.}$ 計算

1.1 反復法

3000 年前に反復法を用いた平方根の計算がメソポタミアで行われていた。
$\sqrt{5}$ を求める。　初期値は 1 とする。　(2,3,7/3 などでもよい）
この方法で、少ない反復数でいろいろな数の平方根が求められる。

$$\frac{1}{2}(1+5)=3$$
$$\frac{1}{2}(3+\frac{5}{3})=\frac{7}{3}$$
$$\frac{1}{2}(\frac{7}{3}+\frac{5}{\frac{7}{3}})=\frac{47}{21}$$
$$\vdots$$

$$x_1=\frac{1}{2}(x_0+\frac{a}{x_0})$$
$$x_2=\frac{1}{2}(x_1+\frac{a}{x_1})$$
$$x_3=\frac{1}{2}(x_3+\frac{a}{x_2})$$
$$\vdots$$

1.2 Script と計算

f(関数) ^: (i.n) 初期値

mf0=: 4 : '1r2 * y + x % y'	$x_1=\frac{1}{2}(x_0+\frac{a}{x_0})$
5 mf0 ^:(i.7) 1	x ルートを求める数 y 初期値

```
1 3 2.33333 2.2381 2.23607 2.23607 2.23607
```

mf は $mf0$ を組み込んで使いやすく見やすくしたものである。

```
    5 mf0 ^:(_) 1
2.23607

    1 x:5  mf0 ^:(_) 1   NB. ^:(_) is convergence in fraction
454110520844r203084398781

2 20 mf 5  NB. find sqrt(5) with initial is 2 repeat is 20

mf 5     NB. find sqrt (5) (default x is 1 10)
```

0	1	1
1	3	3
2	2.33333	7r3
3	2.2381	47r21
4	2.23607	2207r987
5	2.23607	31469859094513r14073748835533
6	2.23607	454110520844r203084398781
7	2.23607	454110520844r203084398781
8	2.23607	454110520844r203084398781
9	2.23607	454110520844r203084398781

$\sqrt{11}$

$\frac{1}{2}(1 + 11) = 6$
$\frac{1}{2}(11 + \frac{11}{6}) = \frac{47}{12}$
$\frac{1}{2}(\frac{47}{12} + \frac{11}{\frac{47}{12}}) = \frac{3793}{1128}$

```
   mf 11
+-+-------+-----------------------------+
|0|1      |1                            |
+-+-------+-----------------------------+
|1|6      |6                            |
+-+-------+-----------------------------+
|2|3.91667|47r12                        |
+-+-------+-----------------------------+
|3|3.36259|3793r1128                    |
+-+-------+-----------------------------+
|4|3.31694|53350589108795r16084284383466|
+-+-------+-----------------------------+
|5|3.31662|4837030517153r1458419568449  |
+-+-------+-----------------------------+
|6|3.31662|1819779504139r548684165128   |
+-+-------+-----------------------------+
|7|3.31662|1819779504139r548684165128   |
+-+-------+-----------------------------+
```

接続詞　接続詞は (2 : 0) で定義する。　`^:(_)` とすると収束するまで反復計算を行うが、最初は `^:(i.10)` などと回数を指定した方が無限ループを避けられ, 安全である。

反復の接続詞　J はパワー接続詞（*power conjunction*）という強力な機能を備えている。関数を外部で定義して左引数とし、右引数に初期値をとり、自身は接続（反復回転）を行って解を求める。

[*1]

1.3　Script

```
NB. Methopotamian fraction
mf0=: 4 : '1r2 * y + x  % y'

mf=: 3 : 0
NB. Usage:  1 10 mf  5
NB. Usage: mf 5
NB. y is sqre(5)
```

[*1] 左右は逆にし、初期値を左に、関数を右に定義することもできる。

```
NB. x is primary-value/ repeat-times
TMP=.y mf0 ^:(i.10) 1
|:({@>i. 10) ,({@>TMP),: {@>1 x: TMP
:
if.1=  # x do. x=. x,.10 end.
'S N'=: x
TMP=. y mf0 ^:(i.N) S
|:({@>i. N) ,({@>TMP),: {@>1 x: TMP
)
```

2　ニュートン/ラフソン法

2000 年以上の時の隔たりを経て反復法が蘇った。

ニュートン · ラフソン法

$$x_1 = x_0 - \frac{P(x_0)}{P'x_0)}$$
$$x_2 = x_1 - \frac{P(x_1)}{P'x_1)}$$
$$x_3 = x_2 - \frac{P(x_2)}{P'x_2)}$$
$$\vdots$$

$f = x^2 - 10$ をニュートン · ラフソン法の公式に入れる

$$x - \frac{x^2 - 10}{2x} = \frac{x^2 + 10}{2x} = \frac{1}{2}(x + \frac{10}{x})$$

ニュートン · ラフソン法

$$z_{n+1} = z_n - \frac{P(x_n)}{P'x_n)}$$

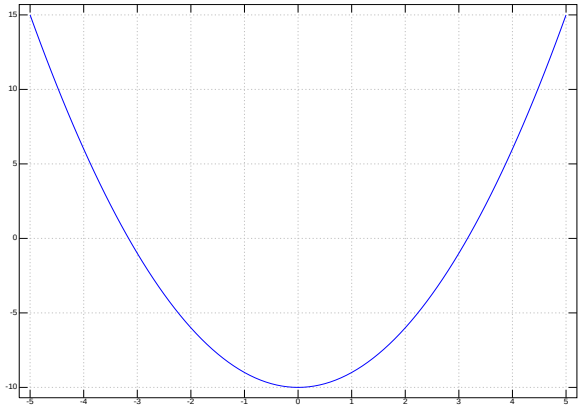
は非線形方程式を効率よく解くが、初期値によっては解が不安定になるので、近傍で初期値を取り直すと安定することが知られている。

ニュートン · ラフソン法で解く。0 のポイントはエラーになる。

`>:i.5` の各ポイントを初期値とした解である。

```
 _10 0 1&p. new_1  >: i.5
3.16228 3.16228 3.16228 3.16228 3.16228
```

```
plot a;  _10 0 1&p. a=. steps _5 5 100
```



この $f = x^2 - 10$ はニュートン・ラフソン法を持ち出さなくとも J では簡単に解とグラフを得ることができる。

```
  p. _10 0 1
+-+----------------+
|1|3.16228 _3.16228|
+-+----------------+
```

図 1　$f = x^2 - 10$

2.1　Script

```
new_1=: 1 : ' ] - x % x D.1' (^:_)("0)
```

3　ジュリア集合

イングランドのケーリー卿は多項式に関するニュートン・ラフソン法の適用範囲を複素数に広げ、根の近くから始点を選ぶのではなく固定吸引点を求める問題に拡張した。

$z^2 - 1, z^3 - 1$ などの複数の解の吸引圏の境界問題を懸賞問題とした。(1879 年)

この問題はファトウとジュリアにより解決された。

複数の解の吸引圏の境界を始点に選んだ場合は数列

$$z_{n+1} = z_n - \frac{P(x_n)}{P' x_n)}$$

は P の根に近づくことなく、境界上を彷徨う。その集合がジュリア集合である。

■Julia　*Gaston Maurice Julia*(1893 – 1978)*France*

フランス人の父は貧しい機械工でアルジェリアで働いているときに生まれた。アルジェリアのクリスチャンスクールで早くから数学と音楽で注目され、シスターは母に奨学金を得て学業を続けるように薦めた。

1911 年に奨学金でパリに出て、数学で優れていたエコール・ノルマルで学ぶ。第 1 次大戦に歩兵(少尉)として西部戦線に従軍し顔面に銃撃を受け鼻を失ったが、病院のベッドで(生涯顔をマスクで覆って)数学の研究を再開した。1917 年に博士論文。1918 年に看護婦で(バイク事故で既に亡くなっていた作曲家 *Ernst Chausson* の娘)*Marianne* と結婚した。1925 年にジュリア集合を含む著書を出版し科学アカデミーのグランプリを得た。

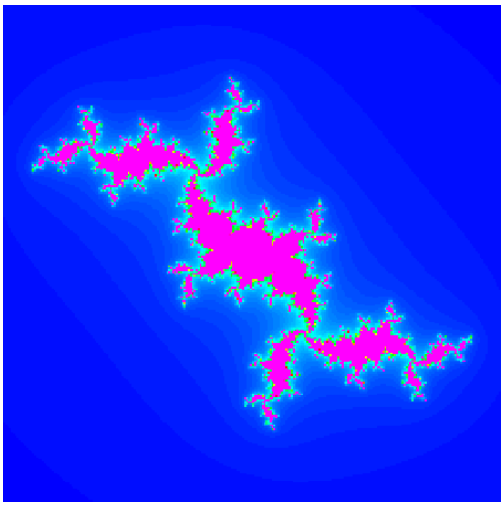


図 2　Julia −0.2 *j*0.8

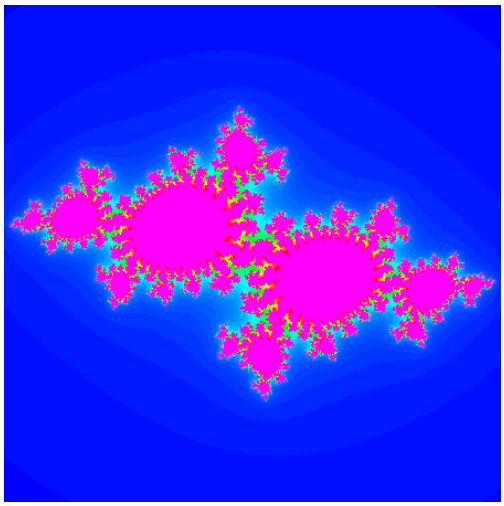


図 3　Julia −0.687 *j*0.312

1919 年から *Colle'ge de France* やエコールノルマルで教え、1937 年エコール・ポリテクニークの数学教授に就任した。1925 年にはジュリア集合の国際セミナーがベルリンで開催。業績は 50 年後マンデルブロート集合で再び注目されるに至った。

岡潔がドイツでなくフランスへ留学したのはジュリアの講義を聴くためであった。

3.1　ジュリア集合を描く

ジュリア集合のアルゴリズムとスクリプトの多くは *C.Reiter* に依った。

描画は J の *viewmat* を用いる。

マンデルブロート集合やジュリア集合の複素図形は次の 3 ステップで描く

step 1　複素数を敷き詰めたキャンバスを用意する。　z をマトリクスとする。

step 2　キャンバス上の各複素数の点に $f = z^2 + c$ を計算して書き込む (z を 2 乗し c を加える。)

step 3　図形を形と色の整数データに変換する。*viewmat* で表示する。

3.1.1　viewmat

```
require 'viewmat'
```

ビットマップのビューア *viewmat* のチュートリアルは LAB に入っている。

viewmat は 1600 万色と複素数をサポートし、複素数は矢印で示す。図形と色調を整数データで表す。

次は 1600 万色の BMP 画像（熱帯の鳥）の一部である。

```
   5{.5{."1 ]120}.a
0 486656 486656 486656 271106
0 486656 486656 486656 271106
0 486656 486656 486656 271106
0 486656 486656 486656 271106
0 486656 486656 486656 271106
```

3.1.2 複素数のキャンバス

z の集合。サイズは 256×256 や 512×512 を用いる。

5×5 の種	`  _1.5+ 3*(i.%<:)n=.5   NB.i.n (%) <:n fork`
	
	`_1.5 _0.75 0 0.75 1.5`

Jullia 用の複素キャンバス
`|. |:`
rotate ← transpose
`_1.5 n=512(usual)` は位置とサイズである。
	`|.|: j./~ _1.5+ 3*(i.%<:)5`
	
	`  _1.5j1.5   _0.75j1.5   0j1.5   0.75j1.5   1.5j1.5`
	`_1.5j0.75  _0.75j0.75  0j0.75  0.75j0.75  1.5j0.75`
	`     _1.5       _0.75       0       0.75       1.5`
	`_1.5j_0.75 _0.75j_0.75 0j_0.75 0.75j_0.75 1.5j_0.75`
	`_1.5j_1.5  _0.75j_1.5  0j_1.5  0.75j_1.5  1.5j_1.5`

3.1.3 $f = z^2 + c$ を定義

$f5 = z^2 - 0.2j0.8$

```
f5=:  +&_0.2j0.8@*: NB. square -> add _0.2j0.8
```

$f = z^2 + C$

```
f5=:  +&_0.2j0.8@*:  NB. square -> add _0.2j0.8
```

```
  fjx _1.5 5
 _1.5j1.5   _0.75j1.5   0j1.5   0.75j1.5   1.5j1.5
_1.5j0.75  _0.75j0.75  0j0.75  0.75j0.75  1.5j0.75
     _1.5       _0.75       0       0.75       1.5
_1.5j_0.75 _0.75j_0.75 0j_0.75 0.75j_0.75 1.5j_0.75
 _1.5j_1.5  _0.75j_1.5  0j_1.5  0.75j_1.5  1.5j_1.5
```

3.2 step 3

escapet C.Reiter の作成した関数のバージョンの差異による異同を修正したもの。

```
escapet=: 2 : 0
NB. (f5 esc  10 3) 0.3 1 // OK
NB. Usage: (f5 escapet 10 3 ) 0.3 1
NB. exchange complex number to color code
'a0 b0'=: n
TMP=.(,u@{: )^:(<&b0@# *. <&a0@:(+/)@:|@:{: )^:_ ,: y
# TMP
)
```

julia のエンジンで接続詞で定義されている。少し丁寧に分解する。

パラメータは 5

1	f		ダイレクトに指定できるようにもした
2	n_1	上限	*color*(256)
3	n_2	反復数	
4	y_0	初期値	中心位置（ほぼ固定）
5	y_1	初期値	画面サイズ 16, 256, 512*etc*

3.2.1 経過と解説

関数 f f = .+ : NB. double 2 倍する

f を適用 初期値 1 で 20 を超える最初の数まで 2 倍する

```
(,f4@{:) ^:(<&20@{:)^:_ ] 1
```

$y_0 = 1, n_2 = 20$

1 2 4 8 16 32

初期値を 2 個 20 を超えない範囲の解を得る

$y_0, y_1 = 0.31$

```
(,f5@{:) ^:(<&20@(+/)@:|@{:)^:_ ,: 0.3 1
0.3  1
0.6  2
1.2  4
2.4  8
4.8 16
```

反復を 3 回に指定 $n_1 = 3$

```
   (,f5@{:) ^:(<&3@# *. <&20@(+/)@:|@{:)^:_ ,: 0.3 1
0.3 1
0.6 2
1.2 4
```

個数を数える　　`# (,f5@{:) ^:(<&20@(+/)@:|@{:)^:_ ,: 0.3 1`

```
5
```

3.2.2　データ変換

複素数の $f = z^2 + c$ を *viewmat* データに変換する

```
((+&0j1@*:) julia0 4 256) fjx _1.5 12
3 3 3  3  3  3  3  3  3 2 2 2
3 3 4  4  4  5  4  3  3 3 2 2
3 5 7  6  6 10  5  4  3 3 3 2
4 9 7 12 10  8  5  4  4 3 3 3
3 4 5  6  7 13  9  5  4 3 3 3
3 4 4  5  6  8 15  5  4 4 3 3
3 3 4  4  5 15  8  6  5 4 4 3
3 3 3  4  5  9 13  7  6 5 4 3
3 3 3  4  4  5  8 10 12 7 9 4
2 3 3  3  4  5 10  6  6 7 5 3
2 2 3  3  3  4  5  4  4 4 3 3
2 2 2  3  3  3  3  3  3 3 3 3
```

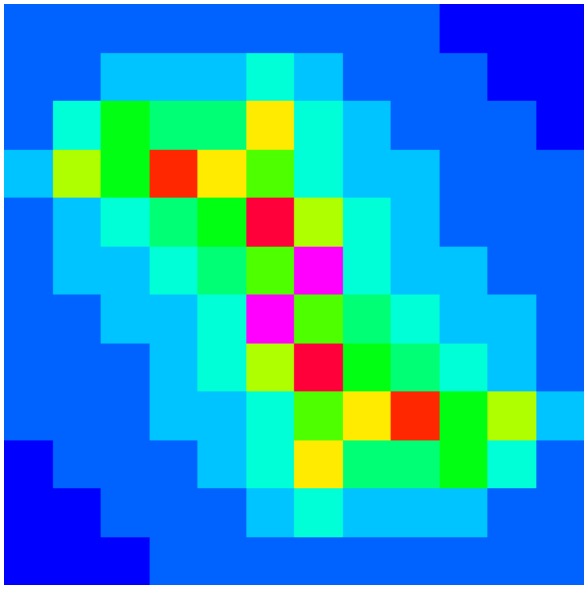


図 4　Jullia 12×12

3.3　ジュリア集合を描く

CPU のクロックとメモリに依存するので一呼吸する。

関数を予め定義する　`f5=:  +&_0.2j0.8@*: NB. square -> add _0.2j0.8`

```
f6=:  +&_0.2j0.6@*:
f7=:  +&_0.2j0.8@(^&3) NB. (fine!) triple -> add _0.2j0.8
f8=:  +&_0.4j0.6@*: NB. fine square -> add _0.4j0.6
```

```
viewmat  f5 julia ''
```

関数をダイレクトに入力する　いろいろなサンプルからパラメータ c の値を探し、入力してみよう

```
viewmat (+&0j1@*:)  julia ''

viewmat ((f5) julia0 4 100) fjx _1.5 512
```

3.4　ギャラリー

```
viewmat (+&_0.123j0.745@*:)  julia ''

viewmat (+&_0.391j_0.587@*:)  julia ''
```

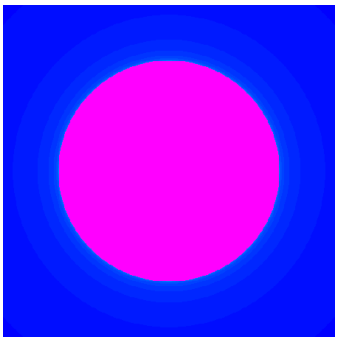


図 5　0

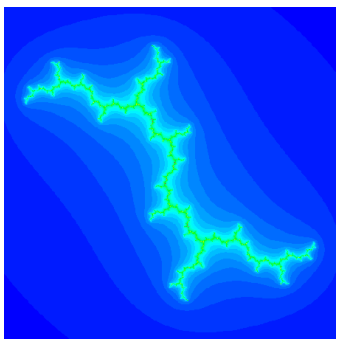


図 6　0*j*1

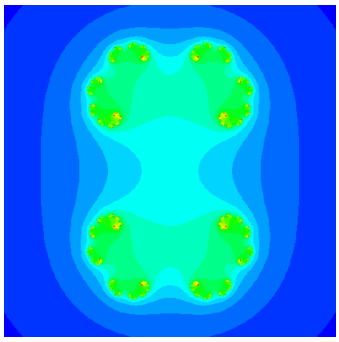


図 7　1*r*2

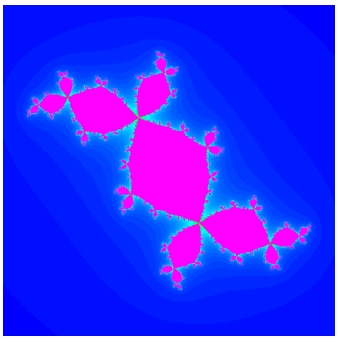


図　　8　Douady'srabbit −0.123*j*0.745

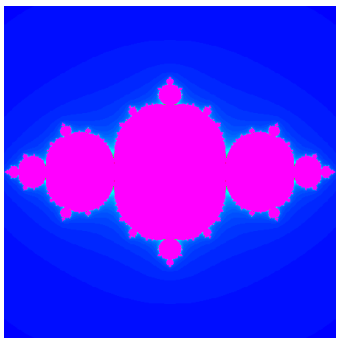


図 9　San Marco −0.75

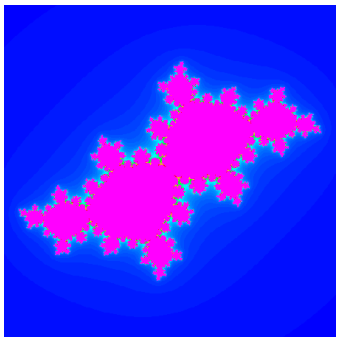


図 10　Siegel disk −0.391*j*0.

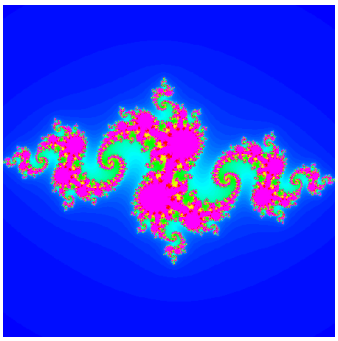


図 11　$-0.8\,j0.156$

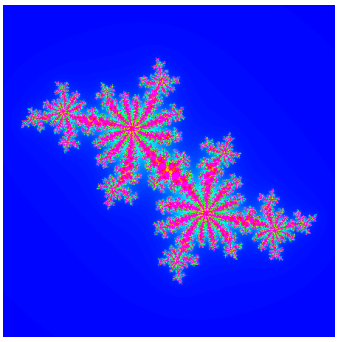


図 12　$-0.297491\,j0.641051$

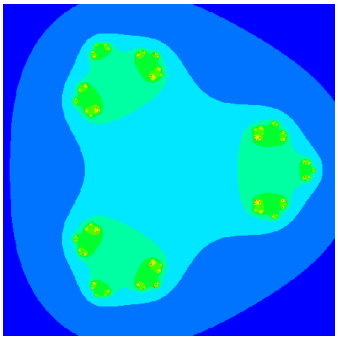


図 13　$z^3 - 1$ $+0.1$

4　マンデルブロート (Mandelbrot)

マンデルブロートは戦乱を避けながらエコール・ポリテクニークでジュリアとレビに学んでいる。

```
   viewmat mandelt fmx _2j_1.5  256
```

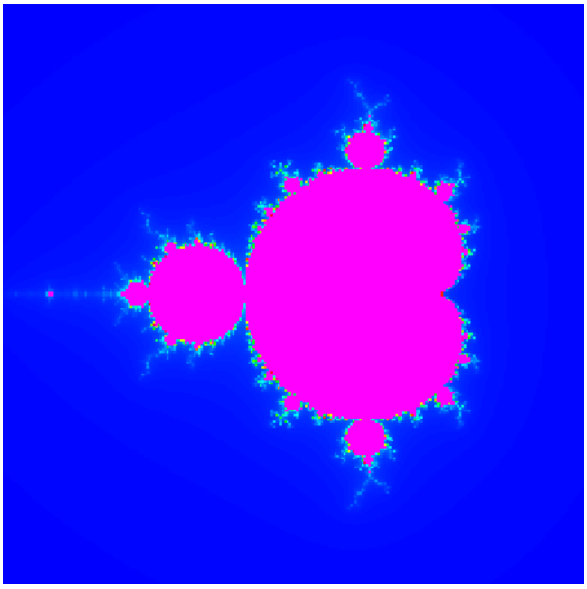


図 14　Mandelbrot

複素キャンバスの作成（1）
シーズ
	`   (i.%<:) 5   NB. (i.5) % 4`
	
	`0 0.25 0.5 0.75 1`

参考:マトリクスに展開	`  +/~ (i.5) % 4`
	`   0 0.25  0.5 0.75    1`
	`0.25  0.5 0.75    1 1.25`
	`0.5 0.75    1 1.25  1.5`
	`0.75    1 1.25  1.5 1.75`
	`   1 1.25  1.5 1.75    2`

複素マトリクスのキャンバス	

```
  j./~ (i.%<:) 5

   0    0j0.25    0j0.5    0j0.75    0j1
0.25 0.25j0.25 0.25j0.5 0.25j0.75 0.25j1
 0.5  0.5j0.25  0.5j0.5  0.5j0.75  0.5j1
0.75 0.75j0.25 0.75j0.5 0.75j0.75 0.75j1
   1    1j0.25    1j0.5    1j0.75    1j1
```

_2j_1.5 を描く
j の前後の数はポジションで-2j-i.5 が中央に来る。

```
 _2j_1.5 + 3*|.|: j./~ (i.%<:) 5

    4.5j_6      3j_7.125    1.5j_8.25      0j_9.375 _1.5j_10.5
3.375j_4.5  1.875j_5.625  0.375j_6.75 _1.125j_7.875  _2.625j_9
   2.25j_3   0.75j_4.125  _0.75j_5.25  _2.25j_6.375 _3.75j_7.5
1.125j_1.5 _0.375j_2.625 _1.875j_3.75 _3.375j_4.875  _4.875j_6
         0   _1.5j_1.125     _3j_2.25   _4.5j_3.375    _6j_4.5
```

カラー表示データに変換	

```
 mandelt   f6 5
  3 3 3 3 2
  3 3 3 3 3
  3 3 3 3 3
  4 4 3 3 3
255 5 3 3 3
```

4.1　Script

```
NB. ---mandelbrot----------------------
mandelt0=: 3 : 'y &+@*: escapet(10 255) 0'
mandelt=: 3 : '> mandelt0 L:0 {@> y'
NB. mandelbrot fx
fmx=: 3 : ' ({. y) + 3*|.|: j./~ (i.%<:) {: y'
```

```
NB. viewmat mandelt fmx _2j_1.5  256
NB. fx (x=)_2j_1.5  (n=)5/15/256
NB. (mjn) position parameter
NB.  m is left<-->right n is up/down
NB. _2j_1.5 is position <-- center
NB. 3* is scale  3 is just!!
```

5 References

C.Reiter [Fractal visualization and J] 2000 Jsoftware