パイこね変換の数値計算誤差

(浮動小数点数計算の誤差)

次の関数を定義する

ψ(x) = 2x　　(0 < x < 0.5)
ψ(x) = 2 (1- x)　　(0.5 <= x <1)

つまり、パイの生地を2倍に伸ばして折り返して均等に練り上げる。

適当な初期値、例えば0.3の点に注目すると、伸ばして折り返すと0.6になる。
0.6の点は、0.8になる。

繰り返すとは、ψ(ψ(x)) = ψ^2(x)
やはり、0 <= ψ^2(x) <=1

初期値点X0とし、Xnの漸化式をつくる。
その一回前の点は、Xn-1記す

Xn = ψ(Xn-1) = ψ^n (X0)

これを、Cプログラミングとエクセルで計算してみよう。

```
/* パイこね変換の実験　S. Kusafusa */
#include <stdio.h>
int main(void){
 double x = 0.3; /* 初期値 */
 double psi;
 printf("i,  x,       ψ(psi)\n");
 for( int i = 1; i < 56; i++ ){
   if (x < 0.5)
   psi = 2 * x;
   else {
   psi = 2 * (1-x);
    }
   printf("%d %f %.40f\n", i, x, psi);
   x = psi;
 }
   return 0;
}
```

written by C. Кусафуса

計算結果

i, x, ψ(psi)

1 0.300000 0.599999999999999977795539507496869191527 4
2 0.600000 0.800000000000000044408920985006261616945 3
3 0.800000 0.399999999999999911182158029987476766109 5
4 0.400000 0.799999999999999822364316059974953532218 9
5 0.800000 0.400000000000000355271367880050092935562 1
6 0.400000 0.800000000000000710542735760100185871124 3
7 0.800000 0.399999999999998578914528479799628257751 5
8 0.400000 0.799999999999997157829056959599256515502 9
9 0.800000 0.400000000000005684341886080801486968994 1
10 0.400000 0.800000000000011368683772161602973937988 3
11 0.800000 0.399999999999977262632455676794052124023 4
12 0.400000 0.799999999999954525264911353588104248046 9
13 0.800000 0.400000000000090949470177292823791503906 2
14 0.400000 0.800000000000181898940354585647583007812 5
15 0.800000 0.399999999999636202119290828704833984375 0
16 0.400000 0.799999999999272404238581657409667968750 0
17 0.800000 0.400000000001455191522836685180664062500 0
18 0.400000 0.800000000029103830456733703613281250000
19 0.800000 0.399999999994179233908653259277343750000 0
20 0.400000 0.799999999883584678173065185546875000000
21 0.800000 0.400000000232830643653869628906250000000
22 0.400000 0.800000000465661287307739257812500000000
23 0.800000 0.399999999906867742538452148437500000000 0
24 0.400000 0.799999999813735485076904296875000000000
25 0.800000 0.400000000372529029846191406250000000000
26 0.400000 0.800000000745058059692382812500000000000
27 0.800000 0.399999998509883880615234375000000000000
28 0.400000 0.799999997019767761230468750000000000000
29 0.800000 0.400000005960464477539062500000000000000
30 0.400000 0.800000011920928955078125000000000000000
31 0.800000 0.399999976158142089843750000000000000000
32 0.400000 0.799999952316284179687500000000000000000
33 0.800000 0.400000095367431640625000000000000000000
34 0.400000 0.800000190734863281250000000000000000000
35 0.800000 0.399999618530273437500000000000000000000
36 0.400000 0.799999237060546875000000000000000000000
37 0.799999 0.400001525878906250000000000000000000000
38 0.400002 0.800003051757812500000000000000000000000
39 0.800003 0.399993896484375000000000000000000000000
40 0.399994 0.799987792968750000000000000000000000000
41 0.799988 0.400024414062500000000000000000000000000
42 0.400024 0.800048828125000000000000000000000000000
43 0.800049 0.399902343750000000000000000000000000000
44 0.399902 0.799804687500000000000000000000000000000
45 0.799805 0.400390625000000000000000000000000000000
46 0.400391 0.800781250000000000000000000000000000000
47 0.800781 0.398437500000000000000000000000000000000

48 0.398438 0.796875000000000000000000000000000000000
49 0.796875 0.406250000000000000000000000000000000000
50 0.406250 0.812500000000000000000000000000000000000
51 0.812500 0.375000000000000000000000000000000000000
52 0.375000 0.750000000000000000000000000000000000000
53 0.750000 0.500000000000000000000000000000000000000
54 0.500000 1.000000000000000000000000000000000000000
55 1.000000 0.000000000000000000000000000000000000000

驚くことに、55回以上こねると、初期値0.3の点は、パイ生地から消えてしまう！
これは何故だろう。

実は、doubleで定義した浮動小数点数の数値計算誤差によるものだ。

初期値をいろいろ変えて計算してみる。
0.2のとき55回で、0.6のとき54回で、0.8 のとき53回で、0になる。
ただし、初期値0.1のとき55回目で初期値に戻る。

さて、エクセルの有効桁数は16桁だ。
cプログラミングと同様に、計算してみると、次ページのように同じ結果が得られた。

0	0.3	0.300000000000000
1	0.6	0.600000000000000
2	0.8	0.800000000000000
3	0.4	0.400000000000000
4	0.8	0.800000000000000
5	0.4	0.400000000000000
6	0.8	0.800000000000001
7	0.4	0.399999999999999
8	0.8	0.799999999999997
9	0.4	0.400000000000006
10	0.8	0.800000000000011
11	0.4	0.399999999999977
12	0.8	0.799999999999955
13	0.4	0.400000000000091
14	0.8	0.800000000000182
15	0.4	0.399999999999636
16	0.8	0.799999999999272
17	0.4	0.400000000001455
18	0.8	0.800000000002910
19	0.4	0.399999999994179
20	0.8	0.799999999988358
21	0.4	0.400000000023283
22	0.8	0.800000000046566
23	0.4	0.399999999906868
24	0.8	0.799999999813735
25	0.4	0.400000000372529
26	0.800000001	0.800000000745058
27	0.399999999	0.399999998509884
28	0.799999997	0.799999997019768
29	0.400000006	0.400000005960464
30	0.800000012	0.800000011920929
31	0.399999976	0.399999976158142
32	0.799999952	0.799999952316284
33	0.400000095	0.400000095367432
34	0.800000191	0.800000190734863
35	0.399999619	0.399999618530273
36	0.799999237	0.799999237060547
37	0.400001526	0.400001525878906
38	0.800003052	0.800003051757812
39	0.399993896	0.399993896484375
40	0.799987793	0.799987792968750
41	0.400024414	0.400024414062500
42	0.800048828	0.800048828125000
43	0.399902344	0.399902343750000
44	0.799804688	0.799804687500000
45	0.400390625	0.400390625000000
46	0.80078125	0.800781250000000
47	0.3984375	0.398437500000000
48	0.796875	0.796875000000000
49	0.40625	0.406250000000000
50	0.8125	0.812500000000000
51	0.375	0.375000000000000
52	0.75	0.750000000000000
53	0.5	0.500000000000000
54	1	1.000000000000000
55	0	0.000000000000000
56	0	0.000000000000000
57	0	0.000000000000000
58	0	0.000000000000000
59	0	0.000000000000000
60	0	0.000000000000000