

## 複雑系科学演習 第2回 レポート

複雑系知能学科 複雑系コース 3年 Iクラス番号 1019086 岩上慎之介

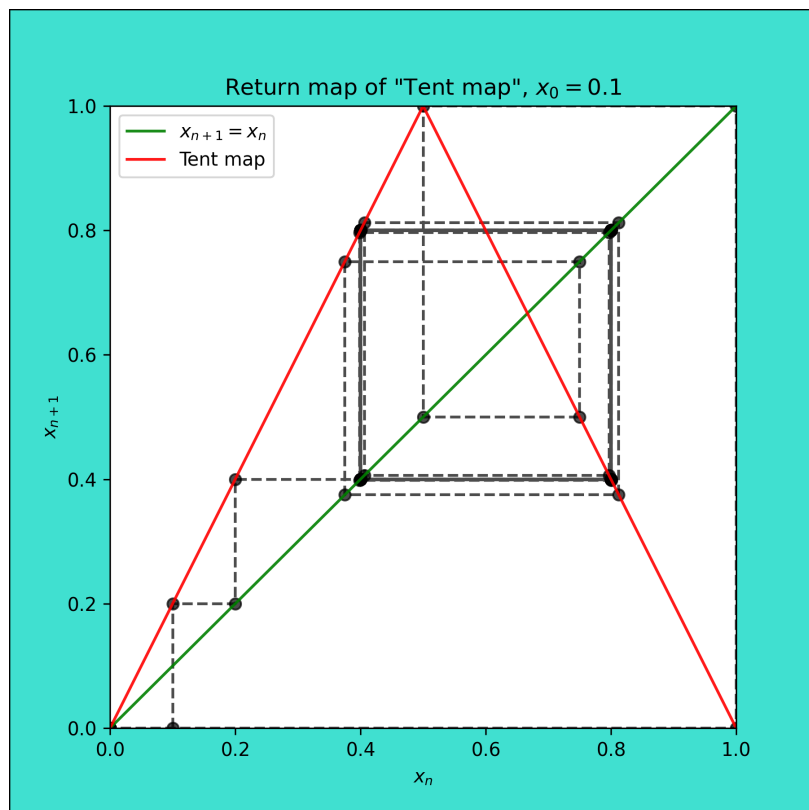
2021年11月15日

# 1 レポート課題 1st

## 1.1 課題 1

前頁の関数 `next` をテント写像に書き換え，リターンマップを描け

### 1.1.1 画像



### 1.1.2 ソースコード

Listing 1 task6

```
1 from matplotlib import pyplot as plt
2 import numpy as np
3
4
5 class Task6():
6     def __init__(self) -> None:
7         self.x = 0.1 # 初期値 x0 = 0.1
8         self.xn = np.linspace(0, 1, 1000) # 横軸の範囲と刻み幅
9         self.tent_y = [] # テント写像のグラフを描くための配列
```

```

10         for i in self.xn:
11             if 0 <= i <= 0.5:
12                 self.tent_y.append(2 * i)
13             else:
14                 self.tent_y.append(2 * (1 - i))
15         self.filepath = 複雑系科学演習'/Week6/images/task6'
16
17     def tent(self) -> list:
18         リターンマップでのテント写像の座標を持つ配列を返す""
19         calc_x = self.x
20         x_array = [calc_x]
21         for _ in range(100):
22             if 0 <= calc_x <= 0.5:
23                 calc_x = 2 * calc_x
24             else:
25                 calc_x = 2 * (1 - calc_x)
26             x_array.append(calc_x)
27         return x_array
28
29     def plot_return_map(self) -> None:
30         リターンマップの描画 (テント写像) ""
31         plt.figure(figsize=(6, 6), facecolor='turquoise',
32                 linewidth=1, edgecolor='black')
33         n = self.tent()
34         spiper_plot_x = [self.x]          # クモの巣図用の配列 (x)
35         spiper_plot_y = [0]              # クモの巣図用の配列 (y)
36         for i in range(1, len(n)):
37             spiper_plot_x.append(n[i - 1])
38             spiper_plot_x.append(n[i])
39             spiper_plot_y.append(n[i])
40             spiper_plot_y.append(n[i])
41
42         plt.plot(spiper_plot_x, spiper_plot_y, marker='o',
43                 linestyle='dashed', color='black', alpha=0.7)
44         plt.plot(self.xn, self.xn, color='green',
45                 alpha=0.9, label="$x_{n+1} = x_n$")
46         plt.plot(self.xn, self.tent_y, color='red',
47                 alpha=0.9, label="Tent map")
48         plt.title("Return map of \"Tent map\", $x_0 = $" + str(self.x))
49         plt.xlim(0, 1)
50         plt.ylim(0, 1)
51         plt.xlabel("$x_n$")
52         plt.ylabel("$x_{n+1}$")
53         plt.legend(loc='best')
54         plt.savefig(self.filepath, dpi=300)
55

```

```

56
57 task = Task6()
58 task.plot_return_map()

```

---

## 1.2 課題 2

$x_0 = \frac{1}{10}$  として、どのような周期解になるか厳密にて計算せよ。

手計算で  $x_i$  の値を計算すると、

$$\begin{aligned}
 x_0 &= \frac{1}{10} \\
 x_1 &= 2 \times \frac{1}{10} = \frac{1}{5} \\
 x_2 &= 2 \times \frac{1}{5} = \frac{2}{5} \\
 x_3 &= 2 \times \frac{2}{5} = \frac{2}{5} \\
 x_4 &= 2 \times \left(1 - \frac{4}{5}\right) = \frac{2}{5} \\
 x_5 &= 2 \times \frac{2}{5} = \frac{2}{5} \\
 x_6 &= 2 \times \left(1 - \frac{4}{5}\right) = \frac{2}{5} \\
 &\vdots \\
 &\vdots \\
 &\vdots
 \end{aligned} \tag{1}$$

となるため  $\frac{2}{5}$  と  $\frac{4}{5}$  の2つの周期解 (2周期解) を取り続けていく。

## 1.3 課題 3

$x_0 = \frac{1}{10}$  として、前のプログラムを実行せよ。ただし  $N = 100$  とし  $x_{100}$  まで数値計算せよ。どのような結果となったか。理由も述べよ。

### 1.3.1 出力

Listing 2 out

---

```

1  0.700000 0.600000
2  0.600000 0.600000
3  0.600000 0.800000
4  0.800000 0.800000
5  0.800000 0.400000
6  0.400000 0.400000
7  0.400000 0.800000
8  0.800000 0.800000

```

9	0.800000	0.400000
10	0.400000	0.400000
11	0.400000	0.800000
12	0.800000	0.800000
13	0.800000	0.400000
14	0.400000	0.400000
15	0.400000	0.800000
16	0.800000	0.800000
17	0.800000	0.400000
18	0.400000	0.400000
19	0.400000	0.800000
20	0.800000	0.800000
21	0.800000	0.400000
22	0.400000	0.400000
23	0.400000	0.800000
24	0.800000	0.800000
25	0.800000	0.400000
26	0.400000	0.400000
27	0.400000	0.800000
28	0.800000	0.800000
29	0.800000	0.400000
30	0.400000	0.400000
31	0.400000	0.800000
32	0.800000	0.800000
33	0.800000	0.400000
34	0.400000	0.400000
35	0.400000	0.800000
36	0.800000	0.800000
37	0.800000	0.400000
38	0.400000	0.400000
39	0.400000	0.800000
40	0.800000	0.800000
41	0.800000	0.400000
42	0.400000	0.400000
43	0.400000	0.800000
44	0.800000	0.800000
45	0.800000	0.400000
46	0.400000	0.400000
47	0.400000	0.800000
48	0.800000	0.800000
49	0.800000	0.400000
50	0.400000	0.400000
51	0.400000	0.800000
52	0.800000	0.800000
53	0.800000	0.400000
54	0.400000	0.400000

55	0.400000	0.800000
56	0.800000	0.800000
57	0.800000	0.400000
58	0.400000	0.400000
59	0.400000	0.800000
60	0.800000	0.800000
61	0.800000	0.400000
62	0.400000	0.400000
63	0.400000	0.800000
64	0.800000	0.800000
65	0.800000	0.400000
66	0.400000	0.400000
67	0.400000	0.799999
68	0.799999	0.799999
69	0.799999	0.400002
70	0.400002	0.400002
71	0.400002	0.800003
72	0.800003	0.800003
73	0.800003	0.399994
74	0.399994	0.399994
75	0.399994	0.799988
76	0.799988	0.799988
77	0.799988	0.400024
78	0.400024	0.400024
79	0.400024	0.800049
80	0.800049	0.800049
81	0.800049	0.399902
82	0.399902	0.399902
83	0.399902	0.799805
84	0.799805	0.799805
85	0.799805	0.400391
86	0.400391	0.400391
87	0.400391	0.800781
88	0.800781	0.800781
89	0.800781	0.398438
90	0.398438	0.398438
91	0.398438	0.796875
92	0.796875	0.796875
93	0.796875	0.406250
94	0.406250	0.406250
95	0.406250	0.812500
96	0.812500	0.812500
97	0.812500	0.375000
98	0.375000	0.375000
99	0.375000	0.750000
100	0.750000	0.750000

101	0.750000	0.500000
102	0.500000	0.500000
103	0.500000	1.000000
104	1.000000	1.000000
105	1.000000	0.000000
106	0.000000	0.000000
107	0.000000	0.000000
108	0.000000	0.000000
109	0.000000	0.000000
110	0.000000	0.000000
111	0.000000	0.000000
112	0.000000	0.000000
113	0.000000	0.000000
114	0.000000	0.000000
115	0.000000	0.000000
116	0.000000	0.000000
117	0.000000	0.000000
118	0.000000	0.000000
119	0.000000	0.000000
120	0.000000	0.000000
121	0.000000	0.000000
122	0.000000	0.000000
123	0.000000	0.000000
124	0.000000	0.000000
125	0.000000	0.000000
126	0.000000	0.000000
127	0.000000	0.000000
128	0.000000	0.000000
129	0.000000	0.000000
130	0.000000	0.000000
131	0.000000	0.000000
132	0.000000	0.000000
133	0.000000	0.000000
134	0.000000	0.000000
135	0.000000	0.000000
136	0.000000	0.000000
137	0.000000	0.000000
138	0.000000	0.000000
139	0.000000	0.000000
140	0.000000	0.000000
141	0.000000	0.000000
142	0.000000	0.000000
143	0.000000	0.000000
144	0.000000	0.000000
145	0.000000	0.000000
146	0.000000	0.000000

147	0.000000	0.000000
148	0.000000	0.000000
149	0.000000	0.000000
150	0.000000	0.000000
151	0.000000	0.000000
152	0.000000	0.000000
153	0.000000	0.000000
154	0.000000	0.000000
155	0.000000	0.000000
156	0.000000	0.000000
157	0.000000	0.000000
158	0.000000	0.000000
159	0.000000	0.000000
160	0.000000	0.000000
161	0.000000	0.000000
162	0.000000	0.000000
163	0.000000	0.000000
164	0.000000	0.000000
165	0.000000	0.000000
166	0.000000	0.000000
167	0.000000	0.000000
168	0.000000	0.000000
169	0.000000	0.000000
170	0.000000	0.000000
171	0.000000	0.000000
172	0.000000	0.000000
173	0.000000	0.000000
174	0.000000	0.000000
175	0.000000	0.000000
176	0.000000	0.000000
177	0.000000	0.000000
178	0.000000	0.000000
179	0.000000	0.000000
180	0.000000	0.000000
181	0.000000	0.000000
182	0.000000	0.000000
183	0.000000	0.000000
184	0.000000	0.000000
185	0.000000	0.000000
186	0.000000	0.000000
187	0.000000	0.000000
188	0.000000	0.000000
189	0.000000	0.000000
190	0.000000	0.000000
191	0.000000	0.000000
192	0.000000	0.000000



```
193 0.000000 0.000000
194 0.000000 0.000000
195 0.000000 0.000000
196 0.000000 0.000000
197 0.000000 0.000000
198 0.000000 0.000000
199 0.000000 0.000000
200 0.000000 0.000000
201 0.000000 0.000000
202 0.000000 0.000000
```

---

### 1.3.2 結果と考察

実行した結果、0.4 と 0.8 でずっとループすることはなく途中から値が変化した。最終的には、 $(x_n, x_{n+1}) = (0, 0)$  を取り続けるようになった。これは、コンピュータの丸め誤差が影響し値が変化したと考えられる。

### 1.3.3 ソースコード

Listing 3 tent.c

---

```
1  #include<stdio.h>
2  #define N 100
3  double next(double x, double r) {
4      if (0 <= x && x <= 0.5) {
5          return 2 * x;
6      } else {
7          return 2 * (1 - x);
8      }
9  }
10 int main(void){
11     int n;
12     double r;
13     double xn = 0.7;
14     double xn1;
15     scanf("%lf", &r);
16     for(n = 0; n <= N; n++) {
17         xn1 = next(xn, r);
18         printf("%lf %lf\n", xn, xn1);
19         printf("%lf %lf\n", xn1, xn1);
20         xn = xn1;
21     }
22     return 0;
23 }
```

---

#### 1.4 課題 4

テント写像のリアプノフ指数を計算せよ。プログラミングがなければ手計算でよい。  
リアプノフ指数  $\lambda$  は、

$$\lambda = \lim_{n \rightarrow \infty} \frac{1}{n} \sum_{i=0}^{n-1} \log |f'(x_i)| \quad (2)$$

と表すことができる。

今回はテント写像のリアプノフ指数について調べていくので、関数  $f(x_i)$  は、

$$f(x_i) = \begin{cases} 2x_i & \left(0 \leq x_i \leq \frac{1}{2}\right) \\ 2(1-x_i) & \left(\frac{1}{2} < x_i \leq 1\right) \end{cases} \quad (3)$$

となる。さらに (3) を用いて計算すると  $f'(x_i)$  は、

$$f'(x_i) = \begin{cases} 2 & \left(0 \leq x_i \leq \frac{1}{2}\right) \\ -2 & \left(\frac{1}{2} < x_i \leq 1\right) \end{cases} \quad (4)$$

となる。よって、 $|f'(x_i)| = 2$  である。したがって、テント写像のリアプノフ指数  $\lambda$  は、

$$\begin{aligned} \lambda &= \lim_{n \rightarrow \infty} \frac{1}{n} \sum_{i=0}^{n-1} \log 2 \\ &= \lim_{n \rightarrow \infty} \frac{1}{n} \times n \log 2 \\ &= \lim_{n \rightarrow \infty} \log 2 \\ &= \log 2 \end{aligned} \quad (5)$$

と計算することができ  $\lambda = \log 2$  となる。

#### 1.5 課題 5

テント写像は  $x_n$  から  $x_{n+1}$  を求める写像である。  $y_n = \sin^2\left(\frac{\pi}{2}x_n\right)$  と定義して、  $y_{n+1}$  を  $y_n$  の式として表わせ。