計算機実習 問題 6.6 - ファイゲンバウム定数の評価

早稲田大学先進理工学部物理学科 B4 藤本將太郎

2014/05/01

1　シミュレーションの目的

すでに問題 6.2 で見たように、隣り合う分岐の間の r の領域は周期の増大にしたがって小さくなる (表 1)。例えば、$b_2 - b_1 = 0.2398$、$b_3 - b_2 = 0.023624$、$b_4 - b_3 = 0.00508$ となっているので、$b_k - b_{k-1}$ が等比的、つまり一定の比 $(b_k - b_{k-1})/(b_{k+1} - b_k)$ で減少すると推定してよいであろう。この比は正確に一定にはならないが、k が大きくなるとともに一定値に近づくことを確かめることができる。これは b_k の数列には極限があり、

$$b_k \approx r_\infty - c\delta^{-k} \quad (c \text{ は定数}) \tag{1}$$

のように等比的に漸近すると考えられる。ここで δ はファイゲンバウム (Feigenbaum) 定数と呼ばれている。式 (1) から δ は比

$$\delta = \lim_{k\to\infty} \frac{b_k - b_{k-1}}{b_{k+1} - b_k} \tag{2}$$

で与えられる。

表 1　k 番目の分岐が生じる点における制御パラメータ b_k の値

k	b_k
1	0.750 000
2	0.862 372
3	0.886 023
4	0.891 102
5	0.892 100
6	0.892 423
7	0.892 473
8	0.892 484

2　作成したプログラム

Python を用いて作成したプログラムを以下に示す。

```python
#! usr/bin/env python
# coding:utf-8

""" 計算機実習
問題 6.6 ファイゲンバウム定数の評価
-a 相加平均でファイゲンバウム定数を求める
作成者：藤本將太郎
"""

import matplotlib.pylab as plt
from Tkinter import *
import numpy

b=[0.750000, 0.862372, 0.886023, 0.891102, 0.892190, 0.892423, 0.892473, 0.892484]
#        b1          b2          b3          b4          b5          b6          b7          b8

def delta(k):
    return (b[k-1] - b[k-2])/(b[k]-b[k-1])

plt.gca().set_xlim(1.5, len(b)-0.5)
plt.xlabel(r'$k$')
plt.ylabel(r'$\delta_{k}$')
plt.title('Feigenbaum constant')
sum_delta_k=0

for k in range(2, len(b)):
    plt.scatter(k, delta(k), color='r', s=8, marker='o')
#    sum_delta_k=sum_delta_k+delta(k)
#ave_delta_k=sum_delta_k/6
#
# -- edited --

for k in range(3, len(b)-1):
    sum_delta_k = sum_delta_k+delta(k)
ave_delta_k = sum_delta_k/4

#plt.plot([2, len(b)-1], [ave_delta_k]*2,
#                         label=r'$\mathrm{average}\ \mathrm{of}\ \delta_{k}\ :\ \delta \ =\ $'
#                         + str(ave_delta_k)
#                         )
```

```
# -- edited --

plt.plot([3, len(b)-2], [ave_delta_k]*2,
        label=r'$\mathrm{average}\ \mathrm{of}\ \delta_{k}\ :\ \delta \ =\ $'
             + str(ave_delta_k)
        )

plt.legend(loc="best")
plt.show()

```

上のプログラムでは表 1 の数値 b_k を使って $\delta_k = (b_k - b_{k-1})/(b_{k+1} - b_k)$ を k に対してプロットし、単純な相加平均によって δ を求めることができる。

```
#! usr/bin/env python
# coding:utf-8
""" 計算機実習
問題 6.6 ファイゲンバウム定数の評価
-a 最小 2 乗法で r_infinity を求める。
作成者：藤本將太郎
"""
import scipy.optimize as optimize
from numpy import *

delta=4.669201609102991
#b=[0.750000, 0.862372, 0.886023, 0.891102, 0.892190, 0.892423, 0.892473, 0.892484]
#k=range(1, len(b)+1)
b=[0.886023, 0.891102, 0.892190, 0.892423]
k=range(3, 7)
parameter0=[1.0, 1.0] # c, r_m  初期値

def fit_func(parameter0,k,b):
    c = parameter0[0]
    r_infinity = parameter0[1]
    residual=b-(r_infinity-c*delta**(-k))
    return residual

result=optimize.leastsq(fit_func, parameter0, args=(array(k),array(b)))

print 'c=', result[0][0]
print 'r_infinity=', result[0][1]
```

上のプログラムを実行すると、プログラム内で指定されたパラメータ初期値を用いて、最小 2 乗法により式 (1) におけるパラメータ c、r_∞ を求めることができる。モジュールとして scipy モジュールの optimize を用いている。

3 実習課題

a. 作成したプログラムを使って、$\delta_k = (b_k - b_{k-1})/(b_{k+1} - b_k)$ を k に対してプロットし、δ を求めよ。b_k の表 1 に与えられた値の桁数はどの k についても十分か。最も精度よく求められている δ の値は

$$\delta = 4.669\ 201\ 609\ 102\ 991\cdots \tag{3}$$

である。式 (3) の小数点以下の桁は、δ が高い精度で求められていることを示している。式 (1)、式 (3) および b_k の値を使って r_∞ の値を求めよ。

まず、$\delta_k = (b_k - b_{k-1})/(b_{k+1} - b_k)$ を k に対してプロットしたグラフを、図 1 に示した。ここで単純に全体の相加平均を求め、その結果を直線にしてグラフに描いた。また、表 1 に与えられた b_k の桁数について、b_k の間隔が k が大きくなるにつれて減少していくことを考えると、b_8 の桁数などは十分であるとは言えないだろう。実際、平均値として得られた δ が、精度よく求められている δ の値に近いのに対して、本来 k が大きいところでは収束するはずの δ_k が、そこからずれた値となっていることも、b_8 などの桁数が不足していることを表していると言える。

次に、最小 2 乗法により式 (1) におけるパラメータ c、r_∞ を求め、その結果を表 2 に示す。ここで、得られた値 $r_\infty = 0.892546164091$ は非常に正確に調べられていて、その値は $r_\infty = 0.892486417967\cdots$ である。すなわち、得られた r_∞ はよく知られている値に対してわずか約 0.007 %の誤差で精度よく求めることができていることがわかる。

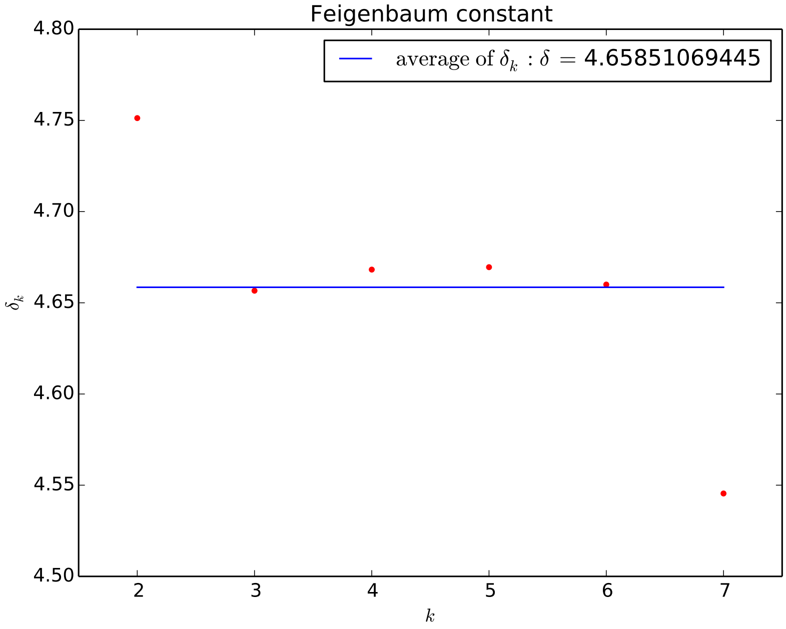

図 1　δ_k を k に対してプロットしたグラフ

表 2　最小 2 乗法によって式 (1) にフィッティングした時のパラメータの値

パラメータ	値
c	0.665237682254
r_∞	0.892546164091

4　まとめ

ファイゲンバウム定数の算出を行い、その普遍的性質への理解を深めることができた。

5　参考文献

- **ハーベイ・ゴールド, ジャン・トボチニク, 石川正勝・宮島佐介訳『計算物理学入門』, ピアソン・エデュケーション**, 2000.