計算機実習 問題 12.1 離散的な時間の 1 次元ランダムウォーク

早稲田大学先進理工学部物理学科 B4 藤本將太郎

2014/05/28

1　シミュレーションの目的

本シミュレーションでは、ランダムウォークの最も簡単な場合として、並進対称な 1 次元の格子上を一定の時間間隔で遷移するモデルを考える。ランダムウォークの分散について知られていることとして、十分大きな N に対して $< \Delta x^2(N) >$ はべき乗則

$$< \Delta x^2(N) > \sim N^{2\nu} \tag{1}$$

を満たす。ここで記号 $\sim$ は”漸近的に等しい”ことを意味し、式 (1) は漸近的なスケーリング則の 1 例となっている。今簡単な 1 次元ランダムウォークのモデル (右と左に進む確率が等しいとき) では、すべての N で式 (1) が成り立ち、$\nu = 1/2$ となる。

2　作成したプログラム

本シミュレーションで作成したプログラムを以下に示す。

2.1　1 次元ランダムウォークのシミュレーション (12-1_random_walk_d1.py)

このプログラムでは、右に遷移する確率を prob として指定し、numpy モジュールの乱数生成メソッドを用いて、ランダムな [0,1) の数を配列 p に格納している。各ステップごとに prob の値と乱数の値とを比較して、右か左に l だけ変化させた値を次の時間での変位として記録する (今は l=1)。関数 calc_ave では、$< x(N) >$、$< x^2(N) >$ の値を計算する。関数 show を用いると、上の計算結果をもちいて、N に対する $< x(N) >$、$< x^2(N) >$、$< \Delta x^2(N) >$ のグラフを表示することができる。関数 caluculate_error は問題 b で使用し、引数に与えた整数値までのランダムウォークの計算を行って、試行回数を増やしていったときに $< \Delta x^2(N) >$ の精度が 1 %未満になっているかどうかを判定する。

```
#! /usr/bin/env python
# -*- coding:utf-8 -*-
#
# written by Shotaro Fujimoto, May 2014.

import numpy as np
import matplotlib.pyplot as plt
```

```
class RandomWalk1():

    def __init__(self, prob=0.7, l=1, nwalkers=1000, x0=0):
        """ Initial function in RandomWalk1.

        prob     : probability that a particle moves right
        l        : step length
        nwalkers : number of trials
        x0       : initial position
        """
        self.prob = prob
        self.l = l
        self.nwalkers = nwalkers
        self.x0 = x0

    def random_walk_d1(self, N):
        """ Caluculate the displacements of each walkers.

        N : A list of walk steps
        """
        x = np.zeros([self.nwalkers, max(N)], 'i')
        p = np.random.random([self.nwalkers, max(N)-1]) # generate random number in [0,1)
        prob = self.prob
        l = self.l
        x0 = self.x0

        for n in range(self.nwalkers):
            x[n][0] = x0
            for i in range(1,max(N)):
                d = +l if p[n][i-1] < prob else -l
                x[n][i] = x[n][i-1] + d
        self.x = x
        self.N = N

    def calc_ave(self):
        """ Caluculate the average of displacements after max(N) steps.

        You can call the results by "self.N", "self.x_ave", and "self.x_2_ave"
        """
```

```
        x = self.x
        x_ave = np.zeros(len(self.N))
        for i,nvalue in enumerate(self.N):
            x_ave[i] = sum([x[n][nvalue-1]*1. for n in xrange(self.nwalkers)])/self.nwalkers

        x_2_ave = np.zeros(len(self.N))
        for i,nvalue in enumerate(self.N):
            x_2_ave[i] = sum([x[n][nvalue-1]**2. for n in xrange(self.nwalkers)])/self.nwalkers

        self.x_ave = x_ave
        self.x_2_ave = x_2_ave

    def show(self):
        """ Show the graph.
        """
        fig = plt.figure('random walk',figsize=(8,8))

        ax1 = fig.add_subplot(311)
        ax1.plot([n for n in self.N], self.x_ave)
        ax1.set_ylabel(r'$<x(N)>$', fontsize=16)

        ax2 = fig.add_subplot(312)
        ax2.plot([n for n in self.N], self.x_2_ave)
        ax2.set_ylabel(r'$<x^{2}(N)>$', fontsize=16)

        ax3 = fig.add_subplot(313)
        ax3.set_ylabel(r'$<\Delta x^{2}(N)>$', fontsize=16)
        ax3.plot([n for n in self.N], self.x_2_ave-self.x_ave**2)
        ax3.set_xlabel(r'$N$')

        plt.show()

    def caluculate_error(self, N):
        """ Caluculate the error of <\Delta x^{2}(N)> and preview.

        N : (int)
        """
        resN_0 = 4.*self.prob*(1.-self.prob)*(self.l**2)*N
        _N = range(1,N+1)

```

```
        M = 2
        count = 0
        while count < 15:
            resN = np.zeros(M, 'f')
            for m in range(M):
                self.random_walk_d1(_N)
                t = self.calc_ave()
                resN[m] = self.x_2_ave[N-1]-self.x_ave[N-1]**2
            std_resN = np.std(resN)

            if M > (std_resN*100./resN_0)**2:
                print str(M) + " & $>$ & " + str((std_resN/(0.01*resN_0))**2) \
                      + " & " + str(count+1) + " \\\\"
                count +=1
            else:
                print str(M) + " & $<$ & " + str((std_resN/(0.01*resN_0))**2) + " & \\\\"
            M += 1
        return None

if __name__ == '__main__':

    rw1 = RandomWalk1()
    # --- 問題 a ---
    N = [4, 8, 16, 32] # caluculate when N = *
    rw1.random_walk_d1(N)
    rw1.calc_ave()
    rw1.show()

    # --- 問題 b ---
#    rw1.caluculate_error(8) # 8 or 32

```

3 実習課題

a. 右に動く確率を $p = 0.7$ とする。$< x(N) >$ と $< x^2(N) >$ を $N = 4, 8, 16, 32$ について計算せよ。この場合の $< x(N) >$ はどのように説明できるか。$< \Delta x^2(N) >$ がどう N に依存するか定性的に答えよ。$< x^2(N) >$ は単純な N 依存性を示すか。

  $< x(N) >$ と $< x^2(N) >$、$< \Delta x^2(N) >$ について、$\mathrm{nwalkers} = 1000$ としてそれぞれの N について計算を行い、この結果を横軸を N としてグラフにしたものを図 1 に示す。このグラフから読み

取れることとして、まず、$< x(N) >$ は N に対して線形に増加しており、これは以下のような簡単な計算の結果と一致している。また、傾きの大きさも $2p-1=0.4$ となっていることが分かる。

$$< x(N) > = \sum_{i=1}^{N} \{p \times 1 + (1-p) \times (-1)\} \tag{2}$$

$$= \sum_{i=1}^{N} (2p-1) = (2p-1)N \tag{3}$$

次に、$< \Delta x^2(N) >$ については、N の 1 乗に比例していることが見て取れる (すなわち $\nu = 1/2$ である)。これも、一般の場合に $< \Delta x^2(N) >= 4pql^2 N$ と表せることと合致している。

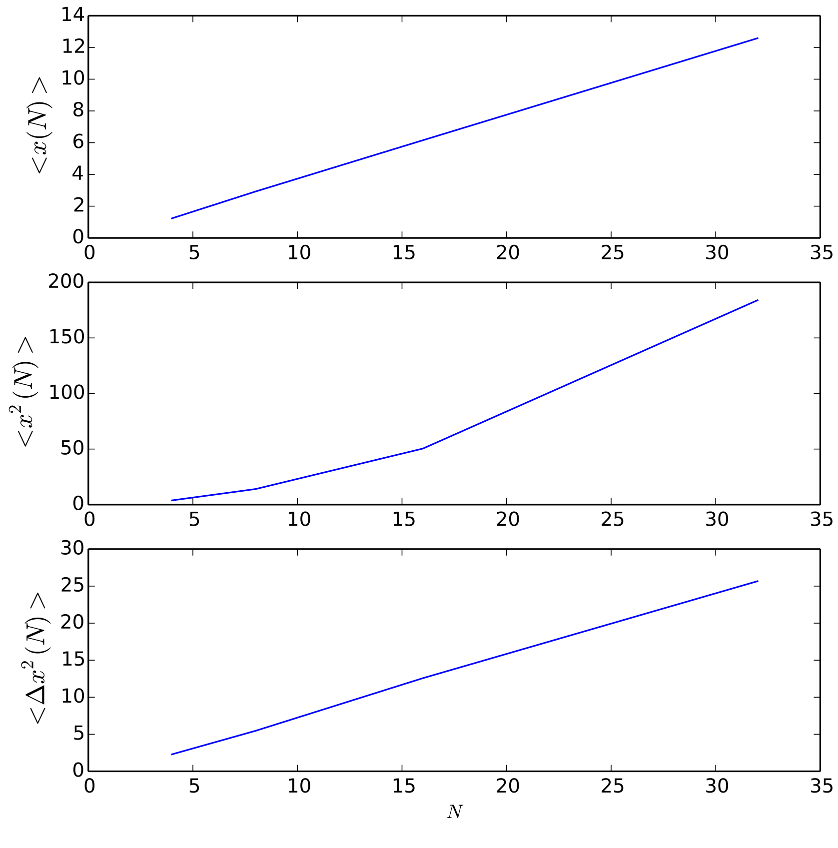

図 1　N に対する $< x(N) >$ と $< x^2(N) >$、$< \Delta x^2(N) >$ のグラフ

b. 第 11.4 節で述べた誤差解析の方法を用いて、$N=8$ と $N=32$ の場合の $< \Delta x^2(N) >$ を精度 1 %で得るために必要な試行の回数を求めよ。

(a) で述べたように、解析的に $< \Delta x^2(N) >$ の値は求められるので、その値を真の値 $< \Delta x^2(8) >_0 = 4 \times 0.7 \times 0.3 \times 8 = 6.72$、$< \Delta x^2(32) >_0 = 4 \times 0.7 \times 0.3 \times 32 = 26.88$ として、それとの相対誤差が 1 %となるような試行回数 M を求めればよい。すなわち標準誤差を σ_m

として

$$\frac{\sigma_m}{< \Delta x^2(N) >_0} \times 100 \leq 1 \tag{4}$$

$$\sigma_m \leq 0.01 < \Delta x^2(N) > \tag{5}$$

となる。ここで、付録に示すように、n 回の試行を行う 1 回の測定で得られた分散を σ とすると

$$\sigma_m = \frac{\sigma}{\sqrt{n}} \tag{6}$$

が成り立つので、これを代入すると (n を M と読み替えて)

$$M \geq \left(\frac{\sigma \times 100}{< \Delta x^2(N) >_0} \right)^2 \tag{7}$$

が得られる。σ が M の関数として決まっている場合は、M の値を解析的に求めることができるが、今の場合 σ を M の関数として求める方法はわからない。したがって、M の値を 2 から順に大きくしていき、M の値と式 (7) の右辺の計算値とを比較していくことにする。この結果をまとめたものを表 1,2 に示す。これらの試行から、$< \Delta x^2(N) >$ を精度 1 %で得るために必要な試行の回数 M は、nwalkers = 1000 であるときには、$N = 8$ のとき $M \geq 19$、$N = 32$ のとき $M \geq 26$ ほどであれば良いことが分かる。それより小さい M では、精度 1 %で求められることもあるが、下限値として適切ではない。

M		$\left(\frac{\sigma\times 100}{<\Delta x^2(N)>_0}\right)^2$	count
2	$<$	11.7080095291	
3	$<$	10.5683527683	
4	$<$	5.6449783206	
5	$<$	13.1767199401	
6	$<$	9.31017539915	
7	$<$	11.298006385	
8	$<$	11.6122000702	
9	$<$	11.8588271339	
10	$<$	13.2805462695	
11	$>$	5.40663459242	1
12	$>$	11.6587313198	2
13	$>$	5.20198341922	3
14	$<$	14.0913796119	
15	$>$	13.2349559502	4
16	$>$	9.20886425624	5
17	$>$	10.1345045153	6
18	$<$	18.5593052103	
19	$>$	11.0536600843	7
20	$>$	11.5386838619	8
21	$>$	13.1460283992	9
22	$>$	8.39227718357	10
23	$>$	15.8071381158	11
24	$>$	8.9317879812	12
25	$>$	17.2226656569	13
26	$>$	12.3538617431	14
27	$>$	11.7846823015	15

表 1　$N = 8$ のとき、M と式 (7) の右辺との比較 (nwalkers = 1000)

M		$\left(\frac{\sigma\times 100}{<\Delta x^2(N)>_0}\right)^2$	count
2	$>$	0.32149282727	1
3	$>$	1.02405794329	2
4	$<$	30.0521604962	
5	$<$	7.13793428824	
6	$<$	17.593509665	
7	$<$	17.952294418	
8	$>$	5.56611217356	3
9	$<$	13.868141118	
10	$<$	17.3761782997	
11	$>$	10.5719588472	4
12	$<$	17.9530873904	
13	$<$	17.977542833	
14	$<$	21.0129775506	
15	$<$	16.105658749	
16	$<$	17.2696667207	
17	$<$	29.1796778766	
18	$>$	17.1884240052	5
19	$>$	14.4165685045	6
20	$>$	16.5500505218	7
21	$>$	9.49101301793	8
22	$>$	10.7188319627	9
23	$<$	29.1758785194	
24	$>$	13.1087274658	10
25	$<$	26.9290290068	
26	$>$	20.6962324661	11
27	$>$	21.9648359983	12
28	$>$	20.7435187729	13
29	$>$	18.3682158692	14
30	$>$	24.8443920402	15

表 2　$N = 32$ のとき、M と式 (7) の右辺との比較 (nwalkers = 1000)

4　まとめ

このシミュレーションでは、離散時間の 1 次元ランダムウォークの簡単な例を実施することができた。また、測定の精度を上げるために試行回数を増やすことなど、定量的な誤差について学ぶ機会となった。

5 付録: 平均値の標準偏差

σ を測定の標準偏差とすると、n 回の試行からなる単独の測定の誤差が $\sigma/\sqrt{n}$ に等しくなることを、解析的に導く。注目する測定量を x で表し、それぞれが n 回の試行からなる m 組の、合計して mn 回の試行からなる測定の組を考える。特定の測定を表すために添字 α を使い、ある測定の i 回目の試行を表すために添字 i を用いる。測定 α の i 回目の試行の結果を $x_{\alpha,i}$ で表すと、測定の値は

$$M_\alpha = \frac{1}{n}\sum_{\alpha=1}^{n} x_{\alpha,i} \tag{8}$$

で与えられる。さらに mn 回のすべての試行についての平均 $\bar{M}$ は

$$\bar{M} = \frac{1}{m}\sum_{\alpha=1}^{m} M_\alpha = \frac{1}{mn}\sum_{\alpha}^{m}\sum_{i=1}^{n} x_{\alpha,i} \tag{9}$$

となる。α 番目の測定値とすべての測定の平均値との差は

$$e_\alpha = M_\alpha - \bar{M} \tag{10}$$

である。平均値の分散は

$$\sigma_m^2 = \frac{1}{m}\sum_{\alpha=1}^{m} e_\alpha^2 \tag{11}$$

と書くことができる。
σ_m と各測定の試行の分散との関係を調べることにしよう。個々の試行結果 $x_{\alpha,i}$ と平均値との差 $d_{\alpha,i}$ は

$$d_{\alpha,i} = x_{\alpha,i} - \bar{M} \tag{12}$$

で与えられる。したがって、nm 回の試行についての分散 σ^2 は

$$\sigma^2 = \frac{1}{mn}\sum_{\alpha=1}^{m}\sum_{i=1}^{n} d_{\alpha,i}^2 \tag{13}$$

である。また、

$$e_\alpha = M_\alpha - \bar{M} = \frac{1}{n}\sum_{i=1}^{n}(x_{\alpha,i} - \bar{M}) \tag{14}$$

$$= \frac{1}{n}\sum_{i=1}^{n} d_{\alpha,i} \tag{15}$$

である。したがって、式 (15) を (11) に代入すると、

$$\sigma_m^2 = \frac{1}{m}\sum_{\alpha-1}^{m}\left(\frac{1}{n}\sum_{i=1}^{n} d_{\alpha,i}\right)\left(\frac{1}{n}\sum_{j=1}^{n} d_{\alpha,j}\right) \tag{16}$$

が得られる。式 (16) の組 α についての試行 i,j に関する和には 2 種類の項、つまり、$i = j$ の項と $i \neq j$ の項が含まれている。$d_{\alpha,i}$ と $d_{\alpha,j}$ は互いに独立で、平均値としては正と負の値を同程度に取ることが予想されるので、測定回数の大きい極限では、式 (16) で $i = j$ の項だけが和に寄与すると考えてよいだろう。したがって、

$$\sigma_m^2 = \frac{1}{mn^2}\sum_{\alpha=1}^{m}\sum_{i=1}^{n} d_{\alpha,i} \tag{17}$$

と書く。式 (17) と (13) を組み合わせると、求めていた式

$$\sigma_m^2 = \frac{\sigma^2}{n} \tag{18}$$

が導かれる。

6 追記: $< x^2(N) >$ の解析的な値 (2014/06/09)

問題 a では N に対する $< x^2(N) >$ について図 1 を用いて定性的に述べたが、これを解析的に求めるとするとどうなるか。確率 p で右に移動し、確率 q で左に移動する場合を考えると、このとき $x^2(N)$ は 2 つの項の和で表すことができて、$x_0 = 0$ ならば

$$x^2(N) = \sum_{i=1}^{N} s_i^2 + \sum_{i \neq j=1}^{N} s_i s_j \tag{19}$$

である。ここで $s_i = \pm l$ とする。上の式を利用して $x^2(N)$ の期待値を計算すると、

$$< x^2(N) >= \sum_{i=1}^{N} \left[p(+l)^2 + q(-l)^2\right] + \sum_{i \neq j=1}^{N} \left[p(+l) + q(-l)\right]^2 \tag{20}$$

である。右辺第 2 項の和は、(i, j) の組み合わせ (区別できる) から $i = j$ の場合の N 通りを除いた数だけの場合があるので

$$< x^2(N) >= N(p+q)l^2 + N(N-1)(p-q)^2 l^2 \tag{21}$$

となる。したがって

$$\begin{aligned} < x^2(N) > &= Nl^2 + N(N-1)(p-q)^2 l^2 \\ &= Nl^2 \left[(p+q)^2 - (p-q)^2\right] + N^2(p-q)^2 l^2 \\ &= 4pql^2 N + N^2(p-q)^2 l^2 \end{aligned}$$

である。また、これより $< \Delta x^2(N) >$ は

$$\begin{aligned} < \Delta x^2(N) > &= < x^2(N) > -(< x(N) >)^2 = 4pql^2 N + N^2(p-q)^2 l^2 - N^2(p-q)^2 l^2 \\ &= 4pql^2 N \end{aligned}$$

と求められる。以上から $< x^2(N) >$ は N の 2 乗に比例しており、実際にシミュレーションで行った $\alpha = 0.7$、$N = 30$ のときの値を計算してみると、$< x^2(30) >= 4 \times 0.7 \times 0.3 \times 30 + 30^2(0.7 - 0.3)^2 = 169.2$ であり、図 1 で見た値と一致していることが確かめられる。

7 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.
- 鈴木武・山田作太郎著『数理統計学―基礎から学ぶデータ解析―』, 内田老鶴圃, 2008.