平成 19 年 6 月 11 日

複雑系科学実験１ 渡辺担当分

渡辺 宙志 *

名古屋大学大学院情報科学研究科複雑系科学専攻多自由度システム情報論講座

概要

目 次

*E-mail: hwatanabe@is.nagoya-u.ac.jp

4 モンテカルロ法

これまでの授業ではすべて**決定論的 (deterministic)** な問題、方程式を扱ってきた。決定論的な問題では、初期条件さえ決めればその後の時間発展がすべて決まる。今回は、**確率的 (stochastic)** な問題を扱う。

4.1 擬似乱数

4.1.1 擬似乱数の性質

確率的な問題では次の状態が確率的に決まる。このような確率的な系をシミュレートするためには、ランダムな数列、乱数列が必要になる。乱数列とは、これまでの数列 $x_1, x_2, \cdots, x_n$ から次の値 x_{n+1} が予測できず、さらに数の間に相関が無いような数列である。実際にはコンピュータで生成する乱数は決定論的なアルゴリズムによって生成されるため、次の値が予測できる。そのため計算機が作成する乱数を**擬似乱数 (pseudo random number)** という。

擬似乱数は、最初に乱数の初期値を与える。この初期値を**シード (seed)** と呼ぶ。同じシードからは必ず同じ乱数列が得られる。この性質はプログラムのデバッグなどに非常に有効である。また、実際の計算ではシードを変えて何回も計算することで統計平均を取ることが多い。

乱数は名前の通りランダムな数であるので、乱数列には周期はあってはならないが、擬似乱数は周期を持つ。そこで、まずなるべく長い周期を持つことが要請される。さらに、乱数列に相関があってはならない。たとえば「1 が出た後には 3 が出やすい」というサイコロがあっては困る。擬似乱数の発生アルゴリズムには線形合同法や M 系列などがある。最近ではメルセンヌ・ツイスタ法 [2] など、得られる乱数列の統計的性質が優れているアルゴリズムが提案されている。

4.1.2 線形合同法

最も簡単な擬似乱数生成法として線形合同法を扱う。線形合同法とは、現在の乱数 x_n から次の乱数 x_{n+1} を次の漸化式から生成する。

$$x_{n+1} = ax_n + c \mod m \tag{1}$$

ここで、a や c は定数、mod m とは、最大値 m で割ったあまりを取ることに対応する。a や c を適切に選べば、0 から $m-1$ までの乱数が得られる。数値計算においては、乱数は整数よりも 0 から 1 までの実数であった方が便利であることが多い。そこで、得られた乱数を m で割ることで実数の乱数を得る。線形合同法は簡単な方法であるが、周期がたかだか m となるなど乱数としての性質は良くないため、実際の計算にはそのままでは使われない。

4.1.3 課題 1

4.5.1 にあるソースを打ち込み、`rand.cc` という名前で保存した後、コンパイル、実行せよ。何度か実行し、同じ乱数列が得られることを確認せよ。また、出力された結果を gnuplot で表示し、$0 < x < 1$ の範囲で一様に分布していることを確認せよ。

4.1.4 課題２

4.5.1 にあるソースを修正し、$0 < x < 1$、$0 < y < 1$ となるような二次元の点 (x, y) を次々と表示するようにせよ。修正する際、ファイル名を `rand2d.cc` などと名前を変えて保存すること。

4.1.5 課題３

4.5.1 にあるソースを修正し、半径 1 の円の中を一様に分布する二次元の点 (x, y) を次々と表示するようにせよ。
ヒント：半径 r と角度 θ にわけて、$0 < r < 1$、$0 < \theta < 2\pi$ の乱数としてから、$x = r\cos\theta$、$y = r\sin\theta$ として表示せよ。ただし、そのままでは一様分布しないはずである。原因を考察し、可能ならば一様分布するように改良すること。

4.2 確率変数と確率分布

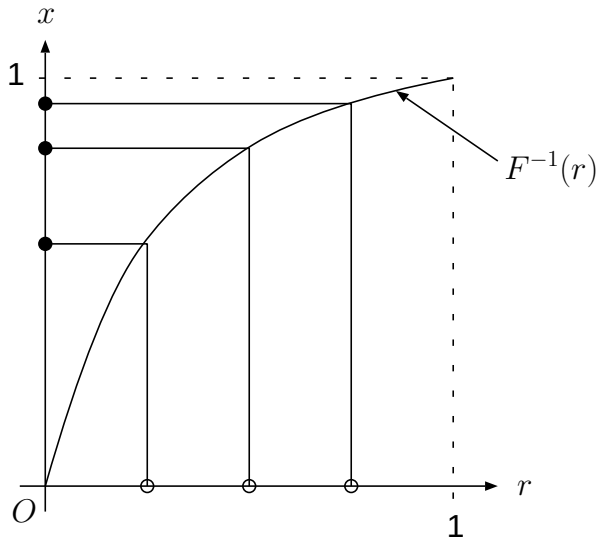


図 1: 一様乱数 r から確率密度 $f(x)$ であるような確率変数 X を生成する方法は図のように写像として理解できる。一様に並んだ点 (図の白丸) を、ある関数を使って x 軸上に移動することを考える。このとき、うつされた点 (図の黒丸) の**密度**が $f(x)$ となるように変換しなければならない。したがって $0 \le x \le x'$ までの点の数は、$f(x')$ の原始関数 $F(x')$ である。したがって、その逆関数 $F^{-1}(r)$ を使って r を変換すればよいことがわかる。図では $f(x) = x$ であるような場合を例示してある。このとき $F^{-1}(r) = r^{1/2}$ となる。

乱数について、もう少し詳しく解説する。ある変数 X を考えよう。この X は、観測するたびに異なる値 x_i を持つとする。このような変数を**確率変数 (random variable)** と呼ぶ。サイコロの目は確率変数である。確率変数は次にどんな値をとるかは確率的にしか決まらない。このとき、どんな値がどのような確率で出るかを考える。サイコロの目の場合は、X は 1 から 6 までの値をとりうる。また、公平なサイコロであればそれぞれの値をとる確率は同じだろう。したがって、

$$P(X = x) = \frac{1}{6} \quad (x = 1, 2, \cdots, 6) \tag{2}$$

である。サイコロの場合は、X がとりうる値は離散的であった。X が連続変数である場合には、その値が x から $x + \mathrm{d}x$ までの値をとる確率を

$$P(x \le X < x + \mathrm{d}x) = f(x)\mathrm{d}x \tag{3}$$

とあらわす。このとき、関数 $f(x)$ を X の**確率密度関数 (probability density function, PDF)** と呼ぶ。X が一様乱数であれば、$f(x)$ は定数関数となる。この式を

$$P(X = x) = f(x) \tag{4}$$

などと表現、理解しないようにしよう。式 (4) は「確率変数 X が、値 x を持つ確率は $f(x)$ であらわされる」という意味で、式 (3) より理解しやすいかもしれない。サイコロなど、離散変数の場合はこれでも良いが、「連続変数がピンポイントである値を持つ」という現象はどうやって定義されるのだろうか？そのあたりは**測度 (measure)** をしっかり定義しないと意味のある議論はできない。とりあえずここでは、式 (3) が定義だと覚えておけばよい。

さて、式 (3) は微分の形をしており、このままでは扱いにくい。そこで、X が x 以下の値をとる確率 $P(0 \leq X < x)$ を考える。これは式 (3) を積分することで得られ、

$$\begin{aligned} P(0 \leq X < x) &= \int_0^x \mathrm{d}x f(x) & (5) \\ &= F(x) & (6) \end{aligned}$$

とあらわされる。この関数 $F(x)$ を、**累積密度関数 (cumulative distribution function, CDF)** と呼ぶ。

いま、確率密度関数が $f(x)$ であるような確率変数 X を計算機で作りたいとする。このとき、使える乱数は一様乱数 r だけであるから、なんらかの変換 $x = g(r)$ を施して、r から X を作らねばならない。では、変換 g と確率密度関数 f はどのような関係となっているだろうか。結論を先に言ってしまうと

$$g(r) = F^{-1}(r) \tag{7}$$

である。$F(x)$ は累積密度関数であり、$f(x)$ の原始関数であるから、結局 $g(r)$ として「確率密度関数の原始関数の逆関数」を用いればよい (図 1 を参照のこと)。

4.2.1 課題４

4.5.2 のスクリプトを `hist.pl` という名前で作成せよ。これは出力されたデータからヒストグラムを作るスクリプトである。作成したら、4.5.1 の出力結果から次のようにしてヒストグラムを作成せよ。

```
% g++ rand.cc          #←コンパイル
% ./a.out > data.dat   #実行して結果をファイルへ出力
% perl hist.pl data.dat > hist.dat # ヒストグラムを作成
```

作成されたヒストグラムデータ `hist.dat` を gnuplot で確認せよ。このとき、

```
gnuplot> plot "hist.dat"
```

とすると密度関数 $f(x)$ が、

```
gnuplot> plot "hist.dat" using 1:3
```

とすると分布関数 $F(x)$ が表示される。$f(x)$、$F(x)$ はどのような関数か？

4.2.2 課題５

4.5.1 のソースを修正し、確率密度関数 $f(x)$ が、$f(x) = x$ であるような、擬似乱数 X を出力するプログラムを書け。課題４と同様にヒストグラムを作成し、$f(x)$、$F(x)$ が正しく再現されているか確認せよ。

4.2.3 課題６

4.5.3 のソースを入力し、randave.cc という名前で保存せよ。これは一様乱数を 10 個ずつ平均をとって出力するコードである。課題４と同様にヒストグラムを作成し、$f(x)$、$F(x)$ がどのような関数となるか確認せよ。この結果は**中心極限定理**として知られている。

4.3 円周率と統計処理

4.3.1 モンテカルロ法による円周率の計算

擬似乱数を用いて系をシミュレーションし、物理量を推定する手法を一般に**モンテカルロ法 (Monte Carlo method)** と呼ぶ。モンテカルロ法の歴史は古いが、実際に応用されるようになったのは高速な計算機が実用化されてからである。ここでは、モンテカルロ法のもっとも簡単な応用の一つ、円周率を求める問題を取り上げる。

まず、$0 \leq x \leq 1$,$0 \leq y \leq 1$ の正方形の領域を考えよう。この中にダーツの要領でランダムに点を打つ。そのときの点の座標が (x, y) だったとすると、$x^2 + y^2 \leq 1$ ならその点は４分の１円の中に含まれる。点が一様に分布するのであれば、円の中に点が入る確率は正方形に対する４分の１円の面積比に等しいはずである。したがって、N 回点を打ち、そのうち k 点円の中に入ったとすると、

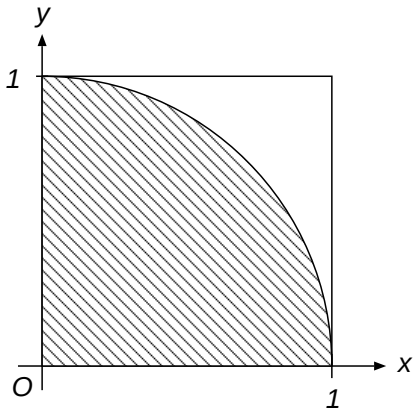
$$\frac{k}{N} = \frac{1/4\text{ 円の面積}}{\text{正方形の面積}} \tag{8}$$

$$= \frac{1}{4}\pi \tag{9}$$

したがって、円周率 π は

$$\pi \sim 4\frac{k}{N} \tag{10}$$

と推定できる。円周率を使わずに円周の長さや円の面積を計算するのは難しいが[1]、ある点が円に含まれているかどうかの判定はやさしい。この手法はそれを利用して円周率の計算を行っている。



モンテカルロ法により、円周率を求める。左図で、$0 \leq x \leq 1$,$0 \leq y \leq 1$ の正方形の範囲にランダムに点を打つ。そのとき、$x^2 + y^2 \leq 1$ である点が４分の１円に含まれる点である。点が正方形の中に一様に分布するのであれば、４分の１円の中 (斜線部分) に含まれる点の数の割合は、正方形の面積に対する斜線部分の面積の比と同じはずである。

4.3.2 課題１

4.5.4 にあるソースを打ち込み、pi.cc という名前で保存した後、コンパイル、実行せよ。N_TRIAL の値を増やし、結果がどうなるか考察せよ。

[1] アルキメデスは、円に内接及び外接する９６角形を作図することで円周率を計算し、$223/71 < \pi < 22/7$ を得ている。少数で表すと、$3.1408 < \pi < 3.1428$ である。

4.3.3 統計誤差

モンテカルロ法で計算された円周率は推定量であるから、統計誤差を含む。統計誤差はサンプル数が多いほど小さくなり、信頼できる値となるだろう。それでは、どれだけのサンプル数があればどれだけ信頼できるだろうか？ここでは分散、標準偏差、正規分布などについて学ぶ。

値が揺らぐ変数 X があったとする。その変数を N 回測定した時の値を $X_1, X_2, \cdots, X_N$ としよう。これらの平均値 $\bar{X}$ は

$$\bar{X} = \frac{\sum_i^N X_i}{N} \tag{11}$$

と、単純に算術平均で求めることができる。それでは、これらのデータのばらつきを評価しよう。先ほど求めた平均を用いて、平均からの差 (偏差) の二乗の平均、V を求める。

$$V = \frac{1}{N-1}\sum_i^N (X_i - \bar{X})^2 \tag{12}$$

と求められる。この値 V は**分散 (variance)** と呼ばれ、データのばらつき具合を表す指標である。ここで N ではなく $N-1$ で割るのは、不偏分散を求めるためである (詳しくは参考書を参照のこと)。分散は二乗和であるから、元のデータの単位と異なる。たとえばもとのデータが身長 [cm] であれば、分散の単位は [cm^2] となる。そこで次元をあわせるために、分散の平方根を取り、σ で表す。

$$\sigma = \sqrt{V} \tag{13}$$

$$= \sqrt{\frac{1}{N-1}\sum_i^N (X_i - \bar{X})^2} \tag{14}$$

この σ は、**標準偏差 (standard deviation)** と呼ばれる。標準偏差もデータのばらつき具合を示す量であるが、単位がもとの測定値と同じであることに注意。

さて、ここでもとめた分散は、確率変数 X の分散である。X の平均値である $\bar{X}$ もまた確率変数であるから、なんらかの分布を持つであろう。その分布は、サンプル数 N を増やせば増やすほど鋭い (幅の狭い) 分布になることが予想される。すなわち、$\bar{X}$ の分散は N を増やせば小さくなる。ここで、$\bar{X}$ の分散 $V(\bar{X})$ は

$$V(\bar{X}) = V\left(\frac{X_1 + X_2 + \cdots + X_N}{N}\right) \tag{15}$$

$$= V(X_1/N) + V(X_2/N) + \cdots + V(X_N/N) \tag{16}$$

$$= V/N^2 + V/N^2 + \cdots + V/N^2 \tag{17}$$

$$= V/N \tag{18}$$

余談コラム 5

円周率 π という数は、古くから人々を魅了してきた数である。円周率の近似の中でもっとも古いものの一つは紀元前 1700 年頃のエジプトの $256/81 \sim 3.16$ などがある。ギリシア時代からの三大作図問題の一つ、「与えられた円と同じ面積の正方形をコンパスと定規のみで作画できるか」という問題は、1882 年になってドイツの数学者 C. Lindemann が円周率は超越数であることを示すことで否定的に解かれた。超越数とは実係数代数方程式の解では表せない数のことで、π の他には自然対数 e や、$2^{\sqrt{2}}$ の形をしたものなど、限られた数しか見つかっていない。

円周率を求める公式はたくさんあるが、なかでも $\pi/4 = 1 - 1/3 + 1/5 - 1/7 + \cdots$ は、円周率とすべての奇数の関係を示しており興味深い (収束が遅いので実際の計算には向かない)。19 世紀、W. Shanks という人はほぼ一生を円周率の計算にささげ、Machin の公式を使って 707 桁まで計算したが、後に 528 桁目に計算ミスがあることが判明してしまった。最近では、東京大学の金田教授らが、日立製作所が納入した計算機を使って円周率の約 1 兆 2,400 億桁の計算に成功している。ちなみに円周率の暗唱記録も日本人で、2006 年に原口さんという人が 10 万桁の暗唱に成功している (暗唱するだけで 16 時間 30 分かかっている)。

と表される。ただし、X_i が互いに独立で、かつ標本分散と母分散が一致していることを仮定していることに注意。したがって、$\bar{X}$ の標準偏差 $D(\bar{X})$ は、

$$D(\bar{X}) = \sqrt{V(\bar{X})} \tag{19}$$

$$= \sigma/\sqrt{N} \tag{20}$$

となる。すなわち、N **サンプルの平均の標準偏差は、$\sqrt{N}$ に反比例する**。これは重要な結果である。すなわち、精度を倍にしようと思ったら、サンプル数を４倍に、精度１０倍にしようとしたらサンプル数は１００倍とらなくてはいけない。

4.3.4 正規分布

データの精度とは何かをもう少し考えよう。ある変数 X が、$x < X < x + \mathrm{d}x$ の範囲内にある確率が $f(x)\mathrm{d}x$ であるとき、$f(x)$ を X の確率密度関数と呼ぶことはすでに述べた。特に、$f(x)$ が次のような関数、

$$f(x) = \frac{1}{\sqrt{2\pi}\sigma} \exp\left\{-\frac{(x-\mu)^2}{2\sigma^2}\right\} \tag{21}$$

で表されるとき、この分布を**正規分布 (normal distribution)** と呼ぶ[2]。この分布は、平均が μ、標準偏差が σ で表されるような分布である。標準偏差 σ が小さければ小さいほど、この分布は平均値の周りに集まり、より値が予想しやすい分布となる。

平均値 μ、標準偏差 σ の正規分布に従う確率変数 X の値が、平均値 μ のまわり Δ に収まる確率、すなわち $P(\mu - \Delta < X < \mu + \Delta)$ を考えよう。この確率は、確率分布 $f(x)$ を $\mu - \Delta < x < \mu + \Delta$ の範囲で積分すれば得られる。特に、$\Delta = \sigma$ とした時、この区間を **1 シグマ**と呼び、およそ 68% である。世の中、サンプル数が多く、それぞれが独立とみなせる場合はほとんどこの正規分布に従うことが知られている。確率変数 X が測定値であった場合、σ が小さければ小さいほど、その測定は精度が高いと言う事ができる。

4.3.5 課題２

4.5.5 にある Perl スクリプトを打ち込み、`calcave.pl` という名前で保存せよ。円周率を計算するプログラム、`pi.cc` の試行回数 `N_TRIAL` を 100、サンプル数 `N_SAMPLE` も 100 として

```
% g++ pi.cc
% ./a.out | perl calcave.pl
```

を実行せよ。すると、平均値と標準偏差が出力されるはずである。以後、試行回数`N_TRIAL`を 1000,10000,100000 と増やし、出てきた結果を

```
100 3.17 0.10
1000 3.12 0.04
```

のように、試行回数、平均値、標準偏差の順番でファイルに保存せよ。ファイル名は `pi.dat` とすること。このファイルを gnuplot で以下のように表示せよ。

```
% gnuplot                                  ← gnuplot 起動
gnuplot> set log x                         ← x 座標を対数表示に
gnuplot> plot "pi.dat" with error,pi       ←エラーバー付きで結果を表示
```

このとき、サンプル数が増えるにしたがってデータがどのようになっているか考察せよ。

[2]ガウス分布 (Gaussian distribution) とも呼ばれる。もちろん数学者 C. F. ガウスに由来する。ガウスは天文学の観測データを調べるなかで誤差理論を確立した。

4.3.6 課題３

先ほどのデータファイル pi.dat のうち、サンプル数と標準偏差の関係がどのようになっているか考察せよ。特に、両対数表示にして、$1/\sqrt{N}$ と比較せよ。

```
% gnuplot                              ← gnuplot 起動
gnuplot> set log xy                    ←両対数表示に
gnuplot> plot "pi.dat" using 1:3             ←エラーバー付きで結果を表示
gnuplot> plot "pi.dat" using 1:3,x**-0.5    1/√x と比較
```

4.4 2 次元イジング模型

4.4.1 イジング模型について

モンテカルロ法は物理に限らず様々な分野で応用されているが、中でも広く応用されている、古典スピン系へのモンテカルロ法を見てみよう。ここでは、非自明な相転移を持つ系のなかで、もっとも簡単なモデルの一つである**イジング模型 (Ising Model)** を扱う。

イジング模型とは、上向き (+1) か下向き (−1) にのみ方向を持つスピン σ_i が、相互作用をするようなモデルで、そのエネルギーは

$$H = -\frac{J}{2}\sum_{i,j}\sigma_i\sigma_j \tag{22}$$

とあらわすことができる。ただし、和はすべての隣接ペアについて取る。このスピンは、温度が低い時にはエネルギーを最低にしようとして、同じ方向にそろうであろう。また、温度が高い時には、エントロピーを増やそうと方向はばらばらになるだろう。その中間で、スピンがそろうかそろわないか微妙な温度があると予想できる。そのような点を**臨界点 (critical point)** と呼び、その周りでは様々な興味深いことがおきる。臨界点よりも高温側では、秩序が無い (スピンがそろっていない) ので、**無秩序相 (disordered phase)**、低温側ではスピンに秩序がある (そろっている) ので**秩序相 (ordered phase)** と呼ぶ。

4.4.2 状態数とエントロピー

熱力学で学ぶように、系は自由エネルギー

$$F = U - TS \tag{23}$$

を最小にするような状態に落ち着く。ただし U は内部エネルギー、T は温度、S はエントロピーである。エントロピーというのは、大雑把に言えば、あるエネルギーを取る状態の、取りうる状態の数である[3]。し

余談コラム６

本文で述べた通り、サンプル数 N に対して、誤差は約 $1/\sqrt{N}$ となる。したがって、100 個のサンプルを取ったとき、データが 23% とでたとしても、信頼できるのは一桁目 (20%) まで、ということになる。どこまでがどのくらい信頼できるかを示すのに σ(シグマ) という単位が使われる。通常 1 シグマ (信頼度 68%) もしくは 2 シグマ (信頼度 95%) を取る。1 シグマの例としては台風の予報円がある。たとえば５時間後の予報円が与えられたとき、その意味は (正確な表現ではないが)「５時間後に台風の中心がその円の中に入る確率は 68%」という意味である。また、視聴率を計算するのに、600 世帯で調査し、その結果 60 世帯が視聴していたとすると視聴率は 10% と計算できる。このとき、信頼度 95%の範囲は、10% ± 2.4% となる。これは、「同条件で視聴率を 100 回測定した場合、5 回は 7.6% 以下もしくは 12.4% 以上の結果が出る」という意味である。したがって、視聴率 10% と 11% の違いは統計的にほとんど意味が無い。さらに小数点以下を表示するのはまったくのナンセンスである。視聴率の調査方法については、ビデオリサーチ社のウェブサイト (http://www.videor.co.jp/index.htm) に詳しいので一度見てみると面白いだろう。

[3]実際にはその対数にボルツマン定数をかけたもの、$S = k_B \log W$ をエントロピーと呼ぶ。

たがって乱雑であればあるほど大きく、系に秩序があると小さくなる。温度が低い時 ($T \sim 0$)、(23) 式の右辺第二項は無視できるので系はエネルギーが低くなるように、イジング模型ならスピンはそろうような状態が支配的 (秩序相) になる。逆に、温度が高いときにはエントロピーを大きくするような、すなわちスピンが乱雑な方向を向いたような状態が支配的 (無秩序相) となる。これが (だいぶ大雑把であるが) 秩序ー無秩序相転移の仕組みである。

あるスピンの状態 (configuration) が与えられたとしよう。この時、その状態のエネルギー U は式 (22) で求めることができる。しかし、エントロピーはどうであろうか。小さな系、たとえば 10×10 の大きさで、エネルギーがちょうどゼロとなるような状態が何個あるか考えてみよ。

いま、あるエネルギー E を取る状態の数が $W(E)$ と求まったとすると、そのエネルギーを取る確率はボルツマン重み $\exp -E/(k_B T)$ に比例するので温度 T におけるエネルギーの期待値は

$$\langle E \rangle = \frac{EW(E)}{\sum_E W(E)} \tag{24}$$

と求めることができる。ここで分母に すべての取りうるエネルギーについての和 が入っていることに注意しよう。これはすなわち すべての取りうる状態についての和 が必要になることを意味する。

たとえば、先ほどの小さな系 10×10 の系では、どれだけの状態があるだろうか。スピンが 100 個あり、それぞれが上向き、下向きの 2 通りの状態を持つので、合計 $2^{100} \sim 10^{30}$ ほどの状態を持つ[4]。これらをすべて手で計算するのは非現実的だ。そこで、必要そうなところのみサンプリングして、期待値を求めようというのがモンテカルロ法の趣旨である。

4.4.3 秩序変数

系がどれだけそろっているか調べるための変数を導入しよう。システムサイズ L の正方格子イジング模型において、スピンの平均の値、すなわち、

$$m = \langle \sigma_i \rangle \tag{25}$$

$$= \frac{1}{L^2} \sum_i \sigma_i \tag{26}$$

を平均磁化、もしくは単に**磁化 (magnetization)** と呼ぶ。温度が高い時 (無秩序相) には、スピンは勝手な方向を向くであろうから磁化は 0 となるが、温度が低い時 (秩序相) スピンはそろおうとするから 0 で無い値をとるだろう。このように、相を特徴づけるような変数を**秩序変数 (order parameter)** と呼ぶ。

4.4.4 メトロポリス法

スピン系を効率よくシミュレートする方法のうち、ここでは広く使われている**メトロポリス法 (Metropolis method)** を用いる。メトロポリス法は以下のようなステップで行われる。

1. 全系のスピンの中から確率 $1/N$ で一つスピンを選ぶ
2. 現在のスピンを $S_{\rm old}$、ひっくり返した状態を $S_{\rm new}$ とし、エネルギーをそれぞれ $E(S_{\rm old})$、$E(S_{\rm new})$ とする。
3. エネルギーの比較を行い、$E(S_{\rm old}) > E(S_{\rm new})$ ならスピンをひっくり返す。$E(S_0) < E(S_i)$ なら確率 $\exp[\beta(E(S_{\rm old}) - E(S_{\rm new}))]$ でスピンをひっくり返す
4. 1 へ戻る

[4] 2^{10} が 1024、すなわち 10^3 に近いことは覚えていて損は無い。計算機は 2 進法が基本なので、たとえば 64GB(ギガバイト) などの数字を良く見かけるだろう。これは $64 \cdot 10^9 = 2^6 \cdot (10^3)^3 = 2^6 \cdot (2^{10})^3 = 2^{36}$ である。

ただし、N はスピンの個数で $N = L^2$、β は逆温度と呼ばれ、$\beta = 1/k_B T$ である。このアルゴリズムは、確率が大きいところのみをサンプリングしつつ、**詳細釣合 (detailed balance)** の条件を満たすという特徴を持つ。ここでは詳しく触れないが、この条件により、熱平衡状態への緩和が保証される。

4.4.5 課題 1

4.5.6 のソースコードを打ち込み、実行せよ。結果をファイルに保存し、gnuplot で表示せよ。臨界点はどのくらいか。

4.4.6 課題 2

Onsager による厳密解によれば、

$$2 \tanh^2 \left(\frac{2J}{k_B T} \right) = 1 \tag{27}$$

となるような温度が臨界点 T_c である。このシミュレーションでは、$k_B = 1$、$J = 1$ としている。gnuplot で上記の関数を表示し、臨界点を小数点以下 3 桁まで求めよ。その臨界点と、先ほど求めた結果を比較せよ。

4.5 ソースコード

4.5.1 擬似乱数

```
#include <stdio.h>
#include <math.h>

const unsigned int MAX_INT = (unsigned int)-1;
//------------------------------------------------------------------------
double
myrand(void){
  static unsigned int x = 1;
  x = 1664525*x + 1013904223;
  return (double)x/(double)MAX_INT;
}
//------------------------------------------------------------------------
int
main(void){
  const int N = 10000;
  for(int i=0;i<N;i++){
   printf("%f␣\n",myrand());
  }
}
```

4.5.2 ヒストグラムを求めるスクリプト

```
$i=0;
while(<>){
  chomp;
  $r = $_;
  $data[$i] = $r;
  if($i==0){
    $min = $r;
    $max = $r;
  }
  $i++;
  if($r < $min){
    $min = $r;
  }elsif($r > $max){
    $max = $r;
  }
}

$datanum = $i;
$num = $datanum/100;
$d = ($max-$min)/$num;

for($i=0;$i<$num;$i++){
  $hist[$i] = 0;
}

for($i=0;$i<$datanum;$i++){
  $index = int (($data[$i]-$min)/$d);
  $hist[$index]++;
}

$sum = 0;
for($i=0;$i<$num;$i++){
  $x = $d*($i+0.5)+$min;
  $y = $hist[$i]/$datanum;
  $sum += $y;
  print "$x␣$y␣$sum\n";
}
```

4.5.3 乱数の平均

```
#include <stdio.h>
const unsigned int MAX_INT = (unsigned int)-1;
//------------------------------------------------------------------------
double
myrand(void) {
  static unsigned int x = 1;
  x = 1664525*x + 1013904223;
  return (double)x/(double)MAX_INT;
}
//------------------------------------------------------------------------
int
main(void) {
  const int N = 10000;
  const int N_AVE = 10;
  for(int i=0;i<N;i++) {
    double x = 0;
    for(int j=0;j<N_AVE;j++) {
      x+= myrand();
    }
    x /= (double) N_AVE;
    printf("%f␣\n",x);
  }
}
```

4.5.4 円周率

```
#include <stdio.h>
#include <stdlib.h>
const int N_TRIAL = 100; //試行数
const int N_SAMPLE = 1;//サンプル数
int
main(void){
  for(int j=0;j<N_SAMPLE;j++){
    double n=0;
    for(int i=0;i<N_TRIAL;i++){
      double x = (double)(rand()/(double)RAND_MAX);
      double y = (double)(rand()/(double)RAND_MAX);
      if(x*x + y*y<1.0){
        n = n + 1;
      }
    }
    double pi = (double)n/(double)N_TRIAL*4.0;
    printf("%f␣\n",pi);
  }
}
```

4.5.5 平均、標準偏差を求める Perl スクリプト

```
$num = 0;
while(<>){
  chomp;
  $x[$num] = $_;
  $num++;
}

$sum = 0;
for($i=0;$i<$num;$i++){
  $sum+= $x[$i];
}
$ave = $sum/$num;
$v = 0;
for($i=0;$i<$num;$i++){
  $v = $v + ($ave-$x[$i])*($ave-$x[$i]);
}
$v = $v / ($num-1);
$s = sqrt $v;
print "$ave␣$s\n";
```

4.5.6 2 次元 Ising Model

```
#include <stdio.h>
#include <stdlib.h>
#include <math.h>

const int L = 32; // システムサイズ
const int ND = 100; // 測定点の数
const int T_LOOP = 100;//緩和時間
const int S_LOOP = 100;//測定時間

int spin[L][L]; //スピン

//------------------------------------------------------------------------
inline double
myrand(void){
  return (double)rand()/(double)RAND_MAX;
}
//------------------------------------------------------------------------
double
calc_magnetization(void){
  double m = 0;
  for(int ix=0;ix<L;ix++){
    for(int iy=0;iy<L;iy++){
      m += spin[ix][iy];
    }
  }
  m = m /(double)(L*L);
  return m;
}
//------------------------------------------------------------------------
void
mcsub(double T){
  double beta = 1.0/T;
  for(int ix=0;ix<L;ix++){
    for(int iy=0;iy<L;iy++){
      //エネルギーの計算
      double de = 0;
      if(ix!=0)de += spin[ix-1][iy];
      else de += spin[L-1][iy];
```

```
      if(iy!=0)de += spin[ix][iy-1];
      else de += spin[ix][L-1];
      if(ix!=L-1)de += spin[ix+1][iy];
      else de += spin[0][iy];
      if(iy!=L-1)de += spin[ix][iy+1];
      else de += spin[ix][0];
      de = de*2.0*spin[ix][iy];
      if(de<0 || exp(-beta*de) > myrand()){
        spin[ix][iy] = - spin[ix][iy];
      }
    }
  }
}
//--------------------------------------------------------------------------
double
mc(double T){
  for(int ix=0;ix<L;ix++){
    for(int iy=0;iy<L;iy++){
      spin[ix][iy] = 1;
    }
  }
  for(int i=0;i<T_LOOP;i++){
    mcsub(T);
  }
  double m = 0;
  for(int i=0;i<S_LOOP;i++){
    mcsub(T);
    m += calc_magnetization();
  }
  m = m / (double)S_LOOP;
  return m;
}
//--------------------------------------------------------------------------
int
main(void){
  double ts = 1;
  double te = 3;
  for(int i=0;i<ND;i++){
    double t = ts + (te-ts)*(double)i/(double)ND;
    double m = mc(t);
    printf("%f␣%f␣\n",t,m);
  }
}
//--------------------------------------------------------------------------
```

4.6 出席及び課題の提出について

授業終了時に、学生番号、名前、課題の回答、感想などをまとめ、hwatanabe@is.nagoya-u.ac.jp までメールにて提出すること。その際、メールの題名 (subject) は、「実験レポート (日付) 学籍番号」とせよ。たとえば、2007 年 6 月 11 日、学籍番号が 050500000 なら、「実験レポート (070611) 050500000」とせよ。授業全体に関する感想および、「ここはこうしたら方が良い」「ここがよくわからなかった」などの修正すべき点についても歓迎する。

参考文献

[1] 統計学の参考書としては、東京大学出版会の「統計学入門」などがある。

[2] M. Matsumoto and T. Nishimura, Mersenne twister: A 623-dimensionally equidistributed uniform pseudorandom number generator, ACM Trans. on Modeling and Computer Simulations, 1998. なお、メルセンヌ・ツイスタ法のウェブサイトは http://www.math.sci.hiroshima-u.ac.jp/˜m-mat/MT/mt.html にある。