HOKKAIDO UNIVERSITY

Title	2005年度　混沌系工学特論講義ノート
Author(s)	井上, 純一
Issue Date	2005-11-30T05:17:27Z
Doc URL	http://hdl.handle.net/2115/828
Rights(URL)	http://creativecommons.org/licenses/by-nc-sa/2.1/jp/
Type	learningobject
Note	とりあげた各トピックスに関する最新情報は著者のホームページを参照ください。
Note(URL)	http://www005.upp.so-net.ne.jp/j_inoue/index.html; http://chaosweb.complex.eng.hokudai.ac.jp/~j_inoue/
Additional Information	There are other files related to this item in HUSCAP. Check the above URL.
File Information	konton2005_1.pdf (第1回講義ノート)

Instructions for use

Hokkaido University Collection of Scholarly and Academic Papers : HUSCAP

混沌系工学特論 配布資料 #1

担当：井上 純一 (情報科学研究科棟 8-13)

URL：http://chaosweb.complex.eng.hokudai.ac.jp/~j_inoue/

平成 17 年 10 月 17 日

〜 全般的な注意事項 〜

- この講義ではレポート問題を出します．講義ノート中に出てくる **問 #** がその問題です．これらの問題を各自考えてレポートにて提出して下さい．**何題解いて提出するかは自由**です．提出期限は最終講義まで．最終講義終了時に回収します．
- 講義ノート中に現れる **付録 (興味のある者は自習せよ)** は講義時間の関係で説明を省略する可能性の高い部分です．
- 各週に配布する講義ノートは**必ずしもその週の講義分とは限りません**．講義ノートは忘れずに持参してください．

今年度のスケジュール

(I) **入門・ウオーミングアップ編**

- 10/17：講義全体の概要説明と簡単な情報統計力学の入門
  ※ スケジュールの都合上, 1 回と少なめですが, 現代 GP の e-learning 教材等もご活用頂けます．

(II) **画像/符号/スペクトル拡散通信**

- 10/24：統計的手法を用いた画像復元/誤り訂正符号の原理とその情報統計力学 #1
- 10/31：統計的手法を用いた画像復元/誤り訂正符号の原理とその情報統計力学 #2
- 11/7：CDMA 通信方式の原理とその情報統計力学

(III) **ゲーム理論**

- 11/14：ゲーム理論 – 入門編 —
- 11/21：マイノリティ・ゲームとその情報統計力学 #1
- 11/28：マイノリティ・ゲームとその情報統計力学 #2

※ 今回配布の資料は昨年度の**混沌系工学特論講義ノート** #5 から一部を抜粋したものです．昨年度 HP：

`http://chaosweb.complex.eng.hokudai.ac.jp/~j_inoue/KONTON2004/konton2004.html`

も参照のこと．

※ 今年度の HP：

`http://chaosweb.complex.eng.hokudai.ac.jp/~j_inoue/KONTON2005/konton2005.html`

目 次

1　最適化問題と統計力学 **2**
1.1　ノイズの効果を利用したアルゴリズム 2
1.1.1　エネルギー 4 準位を持つ 2 体系の最適化 2
1.1.2　マスター方程式とその定常解 9
1.1.3　最適化問題：$E(s_1, s_2) = -Js_1s_2 - h_1s_1 - h_2s_2$ 再考 10
1.1.4　ノイズレベルのスケジューリング 13
1.2　平衡状態と物理量の期待値：内部エネルギー, 自由エネルギー, エントロピー *etc.* 13

目 次

1　最適化問題と統計力学

N 個の正の実定数 $\{a_1, a_2, \cdots, a_N\}$ が与えられたとき, それらを $\mathcal{A}_1$ と $\mathcal{A}_2$ の 2 つの組に分け,

$$\left| \sum_{a_j \in \mathcal{A}_1} a_j - \sum_{a_j \in \mathcal{A}_2} a_j \right| \tag{1}$$

を最小にする問題を考えよう. これは個々のアイテム a_j にラベル $s_j \in \{+1, -1\}$ を割り当て, アイテム a_j がグループ $\mathcal{A}_1$ に属するのであれば, $s_j = +1$, グループ $\mathcal{A}_2$ に属するのであれば, $s_j = -1$ と約束することにより次の量を最小化する問題に書き換えることができる.

$$E(\boldsymbol{s}) = \left| \sum_{j=1}^{N} a_j s_j \right| \tag{2}$$

これは **2 分割問題**と呼ばれ, 典型的な**組み合わせ最適化問題**の一つである. この量 $E(\boldsymbol{s})$ を最小にするアイテムの割り振り方 $\boldsymbol{s} = (s_1, s_2, \cdots, s_N)$ とそのときの $E(\boldsymbol{s})$ を求めるのがここでの課題となる.

さて, この種の問題を解かなければならない状況は実社会に数多く見受けられる. 例えば, プロ野球では近鉄バッファローズとオリックスブルーウエーブが合併してオリックスバッファローズができ, さらに楽天ゴールデンイーグルスが誕生し, 支配下選手 70 名ほどをオリックス ($\mathcal{A}_1$) とゴールデンイーグルス ($\mathcal{A}_2$) で分配しなければならなくなった. 選手によって年棒額 (a_j : $j = 1, \cdots, 70$) もまちまちであるから, 経営者側としては両球団で支払う総年棒額ができるだけ同じになるように $s_1, s_2, \cdots, s_{70}$ に ± 1 を割り振り, 選手を両球団 $\mathcal{A}_1, \mathcal{A}_2$ に分配したい. どうしたら良いだろう？ という具合に問題は設定されるわけである. アイテム数が 10 個程度の問題であれば, この解を探すことはさほど難しくはないが, アイテム数の増加とともに問題は困難になって, アイテム数が 70 であれば, 全件探索したとして 2^{70} 通り. これは非常に困難を極める. 従って, ここで我々が十分に時間をかけて取り組むにたる問題となるのである.

1.1　ノイズの効果を利用したアルゴリズム

1.1.1　エネルギー 4 準位を持つ 2 体系の最適化

いきなり, (2) 式のような多体系のエネルギー関数を扱うのではなく, 問題を恐ろしく簡単にしてしまおう. そこで次のような 2 体系のエネルギー関数の最小値を探す問題を考える.

$$E(\boldsymbol{s}) \quad = \quad -Js_1s_2 - h_1s_1 - h_2s_2 \tag{3}$$

ここで, $\boldsymbol{s} = (s_1, s_2)$ であり, それぞれの成分は ± 1 をとるものとする. この問題は簡単である. なぜならば, 全ての状態 $\boldsymbol{s}$ の組み合わせは $(s_1, s_2) = (1,1), (1,-1), (-1,1), (-1,-1)$ の 4 つしかない. 従って, それぞれに対応するエネルギーの値が簡単に求まって

$$\varepsilon_1 \equiv E(1,1) = -J - h_1 - h_2 \tag{4}$$

$$\varepsilon_2 \equiv E(1,-1) = J - h_1 + h_2 \tag{5}$$

$$\varepsilon_3 \equiv E(-1,1) = J + h_1 - h_2 \tag{6}$$

$$\varepsilon_4 \equiv E(-1,-1) = -J + h_1 + h_2 \tag{7}$$

がそれらの値である. 従って, $J > h_1 > h_2$, 例えば, $J = 1, h_1 = 0.5, h_2 = 0.1$ のように J_1, h_1, h_2 を決めれば, $\varepsilon_1, \cdots, \varepsilon_4$ はその大きさで順序付けられて

$$\varepsilon_1 < \varepsilon_4 < \varepsilon_2 < \varepsilon_3 \tag{8}$$

がそのエネルギー準位となる. これを模式的に描くと図 1 のようになる.

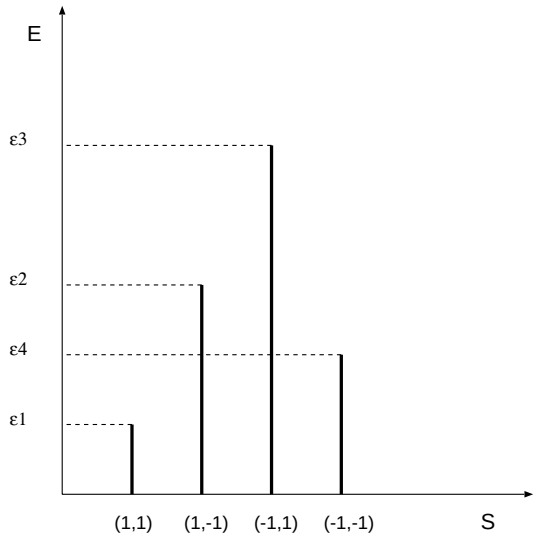

図 1: エネルギー 4 準位.

ここで, 我々が当初調べたかった問題 (2) と比べてかなりトーンダウンしてしまったではないか, と思われるかもしれない. たしかにこの 4 準位系は簡単ではあるが, 多数の要素からなるシステムに現れる現象である「相転移現象」を除けば我々がこれから直面する問題, そして, それをどのように回避するか等は全てこの簡単な問題 (3) を考えることから明らかになるので, まずはこれに集中しよう. 難しい問題 (2) はこれが十分理解できた後であらためて取り組むことにする. 何事も順序が大切なのである.

さて, ここでノイズを利用したアルゴリズムを次のように構成する.

ノイズを利用したアルゴリズム：

(i) 各時刻で任意に s_1, s_2 のどちらか 1 つを選び, その符号を変える. 符号を変える前後の状態を $\boldsymbol{s}, \boldsymbol{s}'$ と書こう.

(ii) (i) の前後でのエネルギー差：$\Delta E = E(\boldsymbol{s}') - E(\boldsymbol{s})$ を計算し, $\Delta E < 0$, つまり, (i) の操作でエネルギーが下がるのであれば, 新しい状態として $\boldsymbol{s}'$ を採用し, $\Delta E > 0$, **つまり, エネルギーが増加してしまった場合でも** $\mathrm{e}^{-\Delta E/T}$ **の確率で新しい状態** $\boldsymbol{s}'$ **を採用する**.

(iii) 上記 (i)(ii) を繰り返す.

このアルゴリズムは今後 [**ノイズを利用したアルゴリズム**] として本稿に現れ, 今回の内容の中心となるのでよく覚えておいてほしい. 上記の**太字**で書いた部分がノイズによる効果であり, そのノイズの大きさは T でコントロールされる. この効果により, エネルギーは単調には減少せず, ときどき増加することになる. このアルゴリズムを実際に計算機上で実行しょう. その際, 各時刻で状態が $(s_1, s_2) = (1,1), (1,-1), (-1,1), (-1,-1)$ の 4 つのそれぞれをとる確率：P_1, P_2, P_3, P_4 を各ステップで評価するために, 同じエネルギー関数 (3) 式を持つ複数の系 (**アンサンブル**) を用意し, (**システムの中でその状態をとる数**)/(**アンサンブルのシステム総数**) をもって, 各確率を評価することにしょう. この評価では例えば P_1 は

$$P_1 \quad = \quad \frac{\textbf{状態}\ (1,1)\ \textbf{をとるシステムの数}}{\textbf{アンサンブルを構成するシステムの総数}} \tag{9}$$

として見積もられることになる. 大して長いプログラムではないので, 下記に上記アルゴリズムのプログラム例を載せておく. これは**混沌系工学特論**のホームページ：

http://chaosweb.complex.eng.hokudai.ac.jp/~ j_inoue/KONTON2004/konton2004.html

からダウンロードできるので, 各自が必要とあらばこれを適時書き換えて実行してみると良いと思う.

```
/****************************************************/
/*          Algorithm for 2-spin problem            */
/*                        J.Inoue                   */
/****************************************************/
/****************************************************/
#include<math.h>
#include<stdio.h>
#include<stdlib.h>
/***************************************************/
/*--------Definitions of Parameters--------------*/
/***************************************************/
#define N 10000    /* アンサンブルのシステム総数 */
#define p 0.7    /* s1 の初期状態を選ぶための確率 */
#define pp 0.7   /* s2 の初期状態を選ぶための確率 */
#define tmax 500  /* アルゴリズムをまわす回数の上限 */
#define T 1.01   /* ノイズレベル. アニーリングする場合にはこれをコメントアウト */
#define Z (exp((J+h1+h2)/T)+exp((-J+h1-h2)/T)+exp((-J-h1+h2)/T)+exp((J-h1-h2)/T))
#define uu0 (exp((J+h1+h2)/T)/Z) /* e1 をとる定常状態確率 */
#define ud0 (exp((-J+h1-h2)/T)/Z) /* e2 をとる定常状態確率 */
```

```
#define du0 (exp((-J+h2-h1)/T)/Z) /* e3 をとる定常状態確率 */
#define dd0 (exp((J-h1-h2)/T)/Z)  /* e4 をとる定常状態確率 */
#define J 1.0
#define h1 0.5
#define h2 0.1
/*****************************************************************/
/*------ "Seeds" of rundomnumbers appearing in this program -----*/
/*****************************************************************/
#define SEEDNOISE 121
#define SEEDNOISE2 142
#define SEEDNOISE3 132
#define SEEDNOISE4 45
#define SEEDNOISE5 92
/*****************************************************************/
int s1[N];
int s2[N];
double uu;
double ud;
double du;
double dd;
// double T; /* アニールする場合にはこのコメントをはずす */
/*******************************************************/
/*     Function of generating randomnumber             */
/*       by Numerical recepies in C  ran3              */
/*******************************************************/
#define MBIG 1000000000
#define MSEED 161803398
#define MZ 0
#define FAC (1.0/MBIG)

float ran3(long *idum)
{
static int inext,inextp;
static long ma[56];
static int iff=0;
long mj,mk;
int i,ii,k;

if (*idum < 0 || iff == 0) {
iff=1;
mj=MSEED-(*idum < 0 ? -*idum : *idum);
mj %= MBIG;
ma[55]=mj;
mk=1;
```

```
for (i=1;i<=54;i++) {
ii=(21*i) % 55;
ma[ii]=mk;
mk=mj-mk;
if (mk < MZ) mk += MBIG;
mj=ma[ii];
}
for (k=1;k<=4;k++)
for (i=1;i<=55;i++) {
ma[i] -= ma[1+(i+30) % 55];
if (ma[i] < MZ) ma[i] += MBIG;
}
inext=0;
inextp=31;
*idum= 1;
}
if (++inext == 56) inext=1;
if (++inextp == 56) inextp=1;
mj=ma[inext]-ma[inextp];
if (mj < MZ) mj += MBIG;
ma[inext]=mj;
return mj*FAC;
}
#undef MBIG
#undef MSEED
#undef MZ
#undef FAC
/* (C) Copr. 1986-92 Numerical Recipes Software 1+5-5i. */

/* s1 の初期値を確率的に選ぶ */
void set_initial()
{
        long idum=(-SEEDNOISE2);
        int x;
for(x = 0; x < N-1;  x++){
  if(p > ran3(&idum)){
    s1[x] = 1;
  }else{
    s1[x] = -1;
  }
}
}

/* s2 の初期値を確率的に選ぶ */
```

```
void set_initial2()
{
        long idum=(-SEEDNOISE3);
        int x;
for(x = 0; x < N-1;  x++){
  if(p > ran3(&idum)){
    s2[x] = 1;
  }else{
    s2[x] = -1;
  }
}
}

/* 状態更新部分 */
void metro(int k)
{
        long idum=(-SEEDNOISE+k);
        long idum2=(-SEEDNOISE5+k);
        int x;
        double r;
           for(x = 0; x < N-1; x++){
              r = ran3(&idum2);
                if((r <= 0.5) && (ran3(&idum)
                  < exp(-(2.0*J*s1[x]*s2[x]+2.0*h1*s1[x])/T))){
       s1[x] = -s1[x];}else{
 s1[x]= s1[x];}
      if((r > 0.5) && (ran3(&idum)
                  < exp(-(2.0*J*s1[x]*s2[x]+2.0*h2*s2[x])/T))){
       s2[x] = -s2[x];}else{
 s2[x]= s2[x];}
   }
}
/*******************************************************/
/*                Main Program                         */
/*******************************************************/
main()
{
FILE *pt;
int i,t;
  set_initial();
  set_initial2();
          if((pt = fopen("result.dat", "wt")) != NULL){
    for(i=1,uu=0.0,ud=0.0,du=0.0,dd=0.0;i < N-1; i++){
      if((s1[i]==1) && (s2[i]==1)){uu = uu + 1.0/N;}
```

```
      if((s1[i]==1) && (s2[i]==-1)){ud = ud + 1.0/N;}
      if((s1[i]==-1) && (s2[i]==1)){du = du + 1.0/N;}
      if((s1[i]==-1) && (s2[i]==-1)){dd = dd + 1.0/N;}
    }
    fprintf(pt,"%d %lf %lf %lf %lf %lf %lf %lf %lf\n",0,uu,ud,du,dd,uu0,ud0,du0,dd0);
    for(t=0; t < tmax; t=t+1){
      /* アニールする場合には下記コメントをはずす */
      // T = 2.0/(log(1.0+2.0*t));
      // T = 2.0/(1.0+2.0*t);
      metro(t);
      for(i=1,uu=0.0,ud=0.0,du=0.0,dd=0.0; i < N-1; i++){
      if((s1[i]==1) && (s2[i]==1)){uu = uu + 1.0/N;}
      if((s1[i]==1) && (s2[i]==-1)){ud = ud + 1.0/N;}
      if((s1[i]==-1) && (s2[i]==1)){du = du + 1.0/N;}
      if((s1[i]==-1) && (s2[i]==-1)){dd = dd + 1.0/N;}
      }
       fprintf(pt,"%d %lf %lf %lf %lf %lf %lf %lf %lf\n",t,uu,ud,du,dd,uu0,ud0,du0,dd0);
    }
  }
            fclose(pt);
}
```

結果の一例を図 2 に載せる．この図より，十分に時間が経過すると，それぞれの状態をとる確率が定常値に

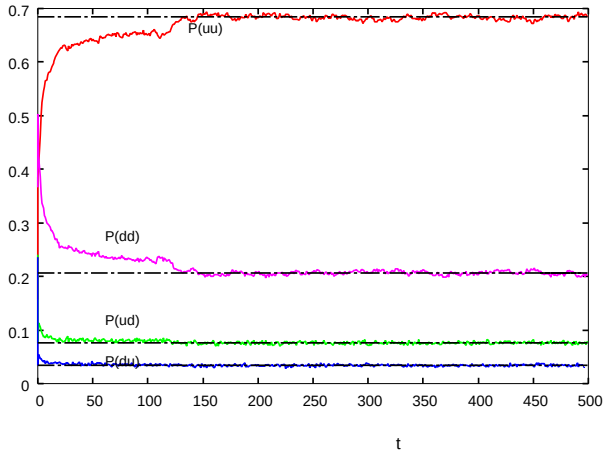

図 2: $(s_1, s_2) = (1,1), (1,-1), (-1,1), (-1,-1)$ をとる確率 $P(\mathsf{uu}) = P_1\,(E = \varepsilon_1), P(\mathsf{ud}) = P_2\,(E = \varepsilon_2), P(\mathsf{du}) = P_3\,(E = \varepsilon_3), P(\mathsf{dd}) = P_4\,(E = \varepsilon_4)$ の時間変化．アンサンブルのシステム総数 $N = 10000$，ノイズレベル $T = 1$．横に引かれた破線はマスター方程式：(12)(13)(14)(15) の定常解である平衡分布 (確率)．これらの定常確率はエネルギー 4 準位を反映している．つまり，$\varepsilon_1 < \varepsilon_4 < \varepsilon_2 < \varepsilon_3$ の順に $P_1 > P_4 > P_2 > P_3$ となっている．

収束していく様子がわかる．なお，この定常値への収束が初期値に依存するかどうかを見るために，s_1, s_2 の初期値を各アンサンブルで変えて $(s_1, s_2) = (1,1)$ となる確率 P_1(グラフ中では $P(\mathsf{uu})$ と表記) の時間変化をプロットしたものが図 3 である．この図から明らかに定常確率 (分布) は初期状態の選び方に依らないことがわかる．この性質，つまり，定常分布が初期状態に依らない性質を**エルゴード性**と呼び，この性質が

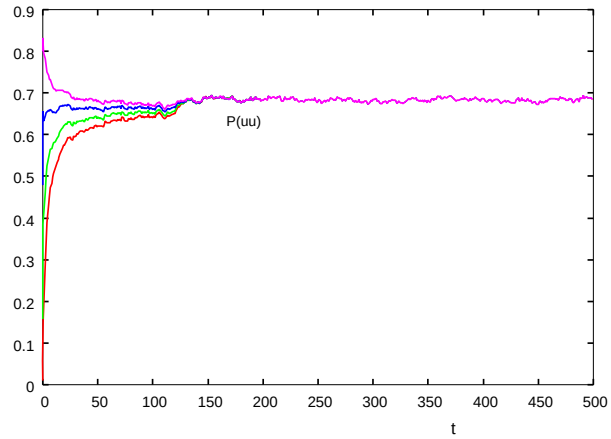

図 3: $(s_1, s_2) = (1, 1)$ をとる確率 $P(\mathsf{uu})$ の時間変化を初期状態を変えてプロット．アンサンブルのシステム数総 $N = 10000$, ノイズレベル $T = 1$.

満たされていれば, そのシステムの物理量 A_t の長時間平均は定常分布を作るアンサンブルでの平均 $\langle A \rangle$ に等しくなる．つまり, 式で書けば

$$\langle A \rangle \quad = \quad \lim_{\mathcal{T} \to \infty} \frac{1}{\mathcal{T}} \sum_{t=0}^{\mathcal{T}} A_t \tag{10}$$

が成り立つ．従って, 定常分布さえ書き下すことができれば, 後は学部で習った確率・統計の知識を用いて, その分布での期待値を計算することにより, 注目する一つのアンサンブルの物理量に関する長時間平均が計算できるということになる[1]．具体例は後に見ることにして, 次に, ここでの計算機シミュレーションで得られた定常分布が解析的に得られないかどうかを検討することにしょう．

1.1.2　マスター方程式とその定常解

図 2 で得られた計算機シミュレーションでの定常分布が解析的に得られれば何かと都合が良いであろう．そこで, これを調べるために「確率の保存式」である**マスター方程式**を用いる．これは一般的に

$$\frac{dP(\boldsymbol{s})}{dt} \quad = \quad \sum_{\boldsymbol{s}' \neq \boldsymbol{s}} w(\boldsymbol{s}' \to \boldsymbol{s}) P(\boldsymbol{s}') - \sum_{\boldsymbol{s} \neq \boldsymbol{s}'} w(\boldsymbol{s} \to \boldsymbol{s}') P(\boldsymbol{s}) \tag{11}$$

と書ける．ここで, $w(\boldsymbol{s}' \to \boldsymbol{s})$ は状態 $\boldsymbol{s}'$ から状態 $\boldsymbol{s}$ への**遷移確率**を表す．従って, 上式 (11) の右辺第 1 項は $\boldsymbol{s}$ 以外の全ての状態 $\boldsymbol{s}'$ から状態 $\boldsymbol{s}$ に流入する確率の流れを, また, 第 2 項は状態 $\boldsymbol{s}$ から, それ以外の状態 $\boldsymbol{s}'$ へ出て行く確率の流れを表しており, この両者の差がシステムが状態 $\boldsymbol{s}$ をとる確率の変化率を決める, という意味を (11) 式は持っている．そこで, このマスター方程式を我々が議論してきたエネルギー関数が (3) 式で与えられ, 状態が [**ノイズを利用したアルゴリズム**] で更新されるシステムに適用すれば, 4 状態のそれぞれをとる確率は次のマスター方程式に従う．

$$\frac{dP_1}{dt} \quad = \quad P_2 + P_3 - \left\{ \mathrm{e}^{-2(J+h_2)/T} + \mathrm{e}^{-2(J+h_1)/T} \right\} P_1 \tag{12}$$

[1] 式 (10) で長時間平均を計算する際にはサンプリング ((10) 式中の和) の開始時刻をアルゴリズムのスタート時刻 $t = 0$ ではなく, システムが平衡状態に落ち込んだ時刻 $t = t_0$ とする場合がある．$\mathcal{T}$ の大きさが十分に大きければ平衡状態へ落ち込むまでのサンプリング点の効果はほとんど結果に効かないであろうと予想できるが, 現実の計算機シミュレーションでは $\mathcal{T}$ を有限値で置き換えるので, この t_0 の設定は重要になってくる場合がある．

ここは 9 ページ目

$$\frac{dP_2}{dt} = P_3 - \left\{2 + \mathrm{e}^{-2(h_1-h_2)/T}\right\} P_2 + \mathrm{e}^{-2(J+h_2)/T} P_1 + \mathrm{e}^{-2(J-h_1)/T} P_4 \tag{13}$$

$$\frac{dP_3}{dt} = \mathrm{e}^{-2(h_1-h_2)/T} P_2 - 2P_3 + \mathrm{e}^{-2(J-h_2)/T} P_4 \tag{14}$$

$$\frac{dP_4}{dt} = P_2 + P_3 - \left\{\mathrm{e}^{-2(J-h_1)/T} + \mathrm{e}^{-2(J-h_2)/T}\right\} P_4 \tag{15}$$

ここで, 当然 $P_1 + P_2 + P_3 + P_4 = 1$ が成り立っていることに注意しなければならない.

さて, システムが定常状態になっているのであれば, $(dP_1/dt) = (dP_2/dt) = (dP_3/dt) = (dP_4/dt) = 0$ が成り立つはずであるから, これから得られる方程式 (12),(13),(14),(15) を P_1, P_2, P_3, P_4 に関して解くことにより, 多少面倒な計算の結果

$$P_1 = P(1,1) = \frac{\mathrm{e}^{(J+h_1+h_2)/T}}{\sum_{s_1=\pm1}\sum_{s_2=\pm1} \mathrm{e}^{(Js_1s_2+h_1s_1+h_2s_2)/T}} = \frac{\mathrm{e}^{(J+h_1+h_2)/T}}{Z} \tag{16}$$

$$P_2 = P(1,-1) = \frac{\mathrm{e}^{(-J+h_1-h_2)/T}}{\sum_{s_1=\pm1}\sum_{s_2=\pm1} \mathrm{e}^{(Js_1s_2+h_1s_1+h_2s_2)/T}} = \frac{\mathrm{e}^{(-J+h_1-h_2)/T}}{Z} \tag{17}$$

$$P_3 = P(-1,1) = \frac{\mathrm{e}^{(-J-h_1+h_2)/T}}{\sum_{s_1=\pm1}\sum_{s_2=\pm1} \mathrm{e}^{(Js_1s_2+h_1s_1+h_2s_2)/T}} = \frac{\mathrm{e}^{(-J-h_1+h_2)/T}}{Z} \tag{18}$$

$$P_4 = P(-1,-1) = \frac{\mathrm{e}^{(J-h_1-h_2)/T}}{\sum_{s_1=\pm1}\sum_{s_2=\pm1} \mathrm{e}^{(Js_1s_2+h_1s_1+h_2s_2)/T}} = \frac{\mathrm{e}^{(J-h_1-h_2)/T}}{Z} \tag{19}$$

が得られるので, 一般に状態 $\boldsymbol{s} = (s_1, s_2)$ をとる確率は

$$P(\boldsymbol{s}) = P(s_1, s_2) = \frac{\mathrm{e}^{(Js_1s_2+h_1s_1+h_2s_2)/T}}{\sum_{s_1=\pm1}\sum_{s_2=\pm1} \mathrm{e}^{(Js_1s_2+h_1s_1+h_2s_2)/T}} = \frac{\mathrm{e}^{-E(\boldsymbol{s})/T}}{Z} \tag{20}$$

と書ける. ここで, Z は確率の規格化因子に相当し,

$$Z = \sum_{s_1=\pm1}\sum_{s_2=\pm1} \mathrm{e}^{-E(\boldsymbol{s})/T} \tag{21}$$

と書き直すことができ, 統計力学ではこの Z のことを**分配関数**と呼んでいる. また, 分布 (20) は**ボルツマン分布**, あるいは, **ギブス分布**と呼ばれる. 従って, [**ノイズを利用したアルゴリズム**] の定常分布はボルツマン分布になることがわかった. なお, 図 2 にはこの定常分布から見積もられる各確率を横線で書き込んであるが, この図から [**ノイズを利用したアルゴリズム**] から求められる定常確率はいずれもこのボルツマン分布に収束している様子が見て取れる.

1.1.3　最適化問題：$E(s_1, s_2) = -Js_1s_2 - h_1s_1 - h_2s_2$ 再考

さて, 我々の当初の問題に戻ろう. 我々の問題はエネルギー関数 (3) 式の最小値とその最小値を与える状態 (s_1, s_2) を求めることであった. そこで, [**ノイズを利用したアルゴリズム**] から得られる定常分布 (20) 式の形を見てみると, この式でノイズレベル T を十分に小さく設定すると, 低いエネルギーを与える状態が比較的高頻度で出現し, $T = 0$ では最小エネルギーをとる状態 $(s_1, s_2) = (1, 1)$ のみが確率 1 で現れることがわかるであろう. これは我々の期待していた結果である. つまり, $P_1 = 1, P_2 = P_3 = P_4 = 0$ という結果が [**ノイズを利用したアルゴリズム**] から得られるのであれば, アンサンブルの中の全てのシステムの状態は $(s_1, s_2) = (1, 1)$ という「正解」となっているはずである. そこで, ノイズレベルを $T = 0.8, 0.6, 0.4, 0.2$ で固定した場合の計算機シミュレーション結果を見てみよう. その結果を図 4 に載せた. この図からまず, $T = 0.8$ ではエネルギーの 1 番高い $P(\mathsf{du})$, 及び, 2 番目に高い $P(\mathsf{ud})$ が比較的小さな値をとり, 一番エネルギーの低い $P(\mathsf{uu})$ が最も大きく, 次にエネルギーの低い $P(\mathsf{dd})$ が 2 番目に高い値に収束している. これは図 1 に示したエネルギー準位 $\varepsilon_1 < \varepsilon_4 < \varepsilon_2 < \varepsilon_3$ がきちんと反映されている. そして $T = 0.4$ ではエネル

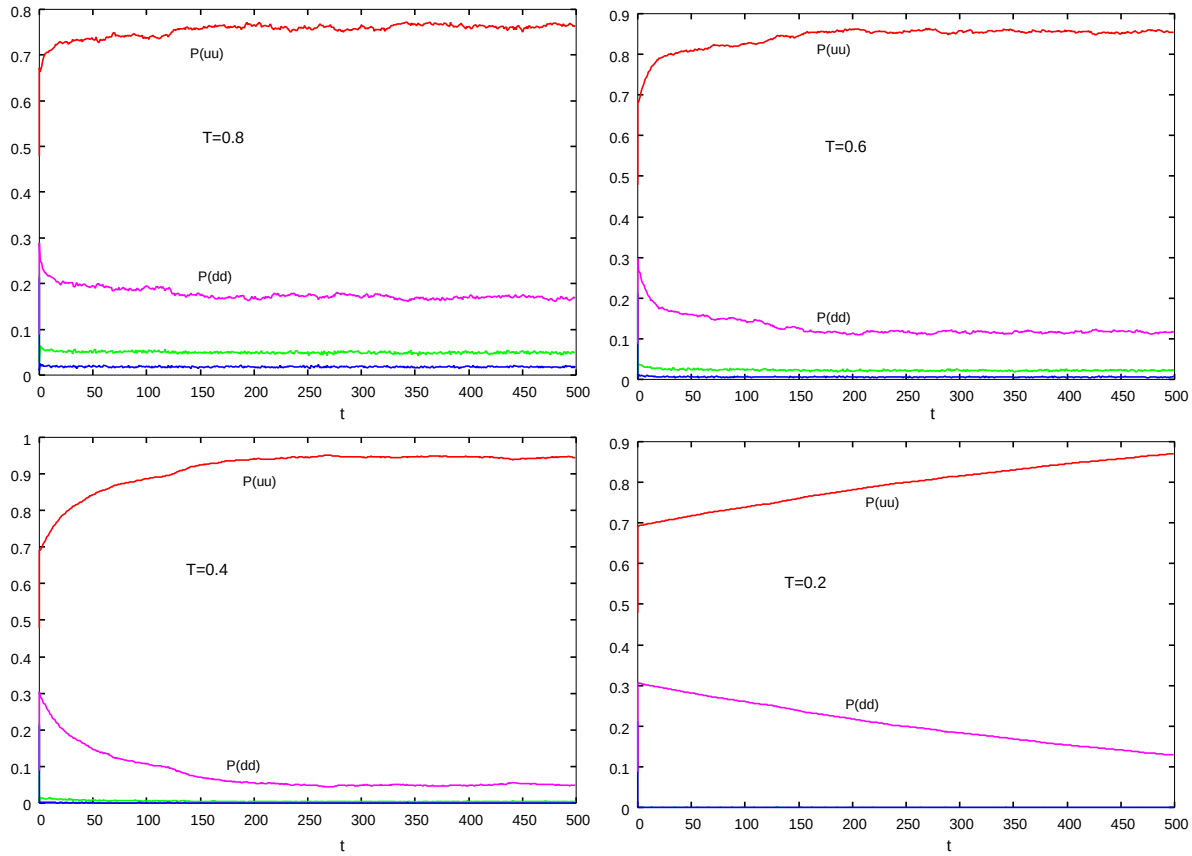

図 4: 左上から右下へ, $T = 0.8, 0.6, 0.4, 0.2$ とノイズレベルを固定した場合の各確率の時間変化の様子.

ギーの高い $P(\mathsf{du}), P(\mathsf{ud})$ が完全にゼロになり, 2 番目にエネルギーの低い $P(\mathsf{dd})$ が減少に転じ, $P(\mathsf{uu})$ が増加し始める. そして, $T = 0.2$ に固定して [**ノイズを利用したアルゴリズム**] を走らせると $t = 500$ 程度ではまだ定常分布に収束していないとは言え, この「好ましい傾向」は依然として見られる. そこで, 調子に乗って一挙に $T = 0.01$ までノイズレベルを下げて, このノイズレベルに固定してアルゴリズムを走らせて見よう. 結果を図 5 に載せる.

残念ながらこの図は我々の期待 (希望的観測) を裏切っている. $P(\mathsf{uu}), P(\mathsf{dd})$ とも完全に一定値に収束しているが, 我々が望む結果：$P(\mathsf{uu}) = 1, P(\mathsf{dd}) = 0$ ではない. 何が問題だったのであろうか？

そこで, 図 1 を見ながらもう一度考えてみると, [**ノイズを利用したアルゴリズム**] では 1 ステップで s_1 か s_2 のいずれか一方のみの状態変化しか許されていないので, **状態** $(-1, -1)$ **から状態** $(1, 1)$ **へ確率が流入するためには, 必ず, 状態** $(1, -1)$ **か** $(-1, 1)$ **を経由しなければならない**ことがわかる. 従って, 例えば, $(-1, -1)$ $(E = \varepsilon_4)$ から, $(-1, 1)$ $(E = \varepsilon_3)$ に移るためには $\Delta E = \varepsilon_3 - \varepsilon_4 = 2(J - h_2)$ の「エネルギー障壁」を飛び越えなければならない. この計算機シミュレーションでは $J = 1, h_2 = 0.1$ と選んでいるので, [**ノイズを利用したアルゴリズム**] より, このエネルギー障壁を飛び越える確率は $T = 0.01$ のとき

$$P((-1, -1) \to (-1, 1)) \quad = \quad \mathrm{e}^{-\Delta E/T} = \mathrm{e}^{-1.8/0.01} = \mathrm{e}^{-180} \tag{22}$$

であり, この逆数でもって状態 $(-1, -1)$ に留まる**滞在時間**を定めれば, アルゴリズムをスタートさせてから, おおよそ $t = \mathrm{e}^{180}$ までの間, この状態 $(-1, -1)$ に居座ることになる. これは事実上, $(-1, -1)$ に状態が落ち込めば, そこからは永遠に脱出できないことを意味する. 従って, 初期状態の選び方が悪く, 初期状

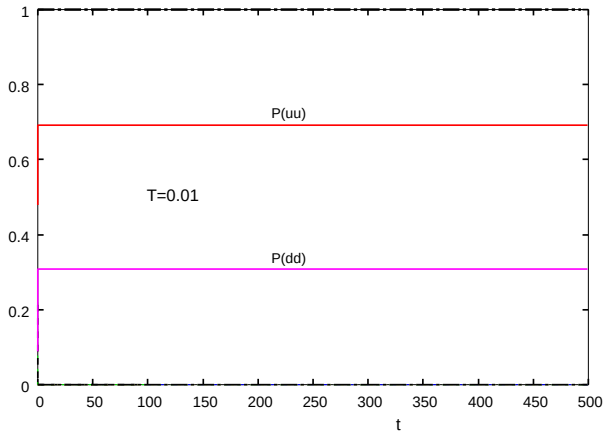

図 5: $T = 0.01$ にした場合の各確率時間発展. 横に走る破線はマスター方程式の定常解である平衡分布.

態 $(-1, -1)$ を比較的高頻度でアンサンブルの中の各システムに与えてしまえば $P(\mathsf{uu})$ は低くなってしまい, 逆にアンサンブルの中の全てのシステムの初期状態を $(1, 1)$ に選べば, $P(\mathsf{uu}) = 1$ となることが予想される. 定常分布が初期状態に依存するという意味において, 実質的にはエルゴード性が破れてしまっているわけである[2]. 実際にこれを見るために, $T = 0.01$ のときに複数の初期状態を選んだ場合の計算機シミュレーションによる $P(\mathsf{uu})$ の時間発展を図 6 に載せよう. この図から予想通り, $P(\mathsf{uu})$ の値は初期値の選び

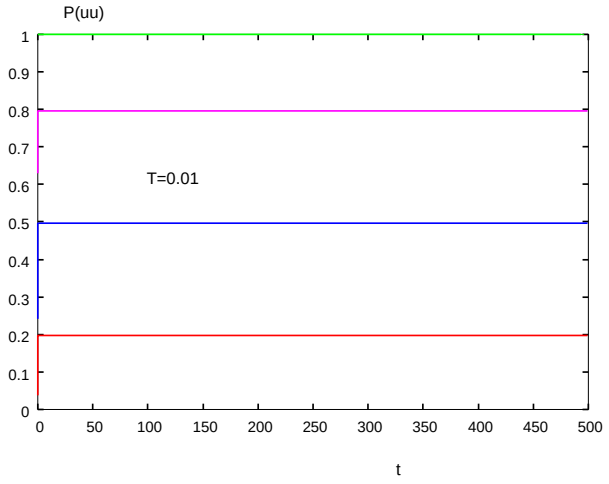

図 6: $T = 0.01$ の場合に初期状態を様々変えてみたときの $P(\mathsf{uu})$ の時間変化. 見かけ上, エルゴード性が破れている.

方に依存している.

[2] もちろん, ノイズレベル T が完全にゼロでは無いわけであるから, 本当はエルゴード性が破れているわけではなく, 無限の時間が経過しさえすれば, いずれは (熱) 平衡状態に達する. しかし, 我々の我慢できる時間の範囲内ではあたかもエルゴード性が破れているように見え, 平衡状態になることはとても期待できない.

1.1.4　ノイズレベルのスケジューリング

以上の結果から, いきなり $T = 0$ と置いてしまうと [**ノイズを利用したアルゴリズム**] は「ランダムシューティング」と何ら変わりなく, また, ゼロではないにしても T を低く設定しすぎれば $(-1,-1)$ から脱出するのは我慢できないほどの時間がかかってしまうことがわかった. では T を時間的に変化させ, アルゴリズムのスタート段階では T を大きめにとって全ての状態を等確率で生成させるようにし, ある程度のアンサンブルが状態 $(1,1)$ に落ち込んだのを見はからって T を徐々に下げていくという戦略はどうであろうか？ これはうまく行きそうである. なぜならば, $T = \infty$ では完全に等確率 $1/4$ で 4 つの状態 $(1,1),(1,-1),(-1,1),(-1,-1)$ が実現するが, 図 2 で見たようにノイズレベル T を $T = 1$ 程度まで下げていくとエネルギーの低い順：$\varepsilon_1 < \varepsilon_4 < \varepsilon_2 < \varepsilon_3$ に出現確率の大小関係：$P(\mathsf{uu}) > P(\mathsf{dd}) > P(\mathsf{ud}) > P(\mathsf{du})$ が決まった. 従って, **システムがあるノイズレベル T での (熱) 平衡状態に到達する限り, 得られる定常確率はエネルギー準位をそのまま反映する**のである. さらに, 各ノイズレベルではアルゴリズムの初期値を再設定するわけではなく, 直前のノイズレベルでのアンサンブルの状態をそのまま使うわけであるから, いきなりノイズレベルを $T = 0$ に設定したときのようなランダムシューティングにはならず, **必ずアルゴリズムの全プロセスを通じて最小エネルギー状態が最も高頻度に現れるという「好ましい性質」「好ましい状態」を引きずりながら新しい状態が生成されていく**ことになる. よって, ノイズレベルを徐々に下げつつ, つまり, 各ノイズレベルではシステムが平衡状態になるのを待ってから次の (より小さな) ノイズレベルに設定する, というように [**ノイズを利用したアルゴリズム**] を修正すれば, 十分に長い時間をかけさえすれば確率 1 で最小エネルギー状態が求められる (つまり, $P(\mathsf{uu}) = 1, P(\mathsf{dd}) = P(\mathsf{ud}) = P(\mathsf{du}) = 0$) と推察できる[3]. 前節：[**連想記憶の数理**] ではノイズと言えば神経素子の誤動作の原因としてネガティブな要素であると扱われていたわけであるが, これをうまく用いてアルゴリズムを構成すれば好ましい情報処理が実現できる, というこの講義で見る最初の例がこれである.

さて, 早速これを確かめるため, T に時間依存性を持たせ

$$T(t) = \frac{2}{1+2t} \tag{23}$$

$$T(t) = \frac{2}{\log(1+2t)} \tag{24}$$

のような 2 通りで下げた場合の結果を図 7 に載せる. この図より, $T \sim 1/t$：(23) でノイズレベルを下げてしまうと, 下げ方が早すぎてうまくいっていない. 一方, $T \sim 1/\log t$：(24) で下げると, $P(\mathsf{uu}) = 1$ へ順調に向かっていることがわかる. このようにノイズレベル T に時間的な変化をもたせて徐々に下げていく [**ノイズを利用したアルゴリズム**] のことを**シミュレーテッドアニーリング**と呼んでおり, 様々な工学的場面で用いられている. 一般的に, エネルギー関数が極めて特殊な状態依存性を持たなければノイズレベルを $1/\log t$ より十分に遅く下げていけば, 十分な時間の経過後には確率 1 で最小エネルギーを与える状態が得られることが知られている.

1.2　平衡状態と物理量の期待値：内部エネルギー, 自由エネルギー, エントロピー *etc.*

ここまでの議論で [**ノイズを利用したアルゴリズム**] を動作させることにより, アンサンブルの中の各システムが状態 $\boldsymbol{s} = (s_1, s_2)$ を割合：

$$P(s_1, s_2) = \frac{\mathrm{e}^{(Js_1s_2+h_1s_1+h_2s_2)/T}}{\sum_{s_1=\pm1}\sum_{s_2=\pm1}\mathrm{e}^{(Js_1s_2+h_1s_1+h_2s_2)/T}} = \frac{\mathrm{e}^{-E(\boldsymbol{s})/T}}{\sum_{\boldsymbol{s}\in\pm1}\mathrm{e}^{-E(\boldsymbol{s})/T}} \tag{25}$$

[3] 確率 1 で最小状態を得るためにはアニーリングプロセスの最後のつめの段階 ($T \simeq 0$) で熱平衡状態を作る必要があり, 前述のように, これには極めて長い時間がかかる. 従って, 実用的にはどこか満足の行くところでアニーリングプロセスを打ち切ってしまう場合が多い. 近似解, それも精度の良い近似解なら OK とするわけである.

ここは 13 ページ目

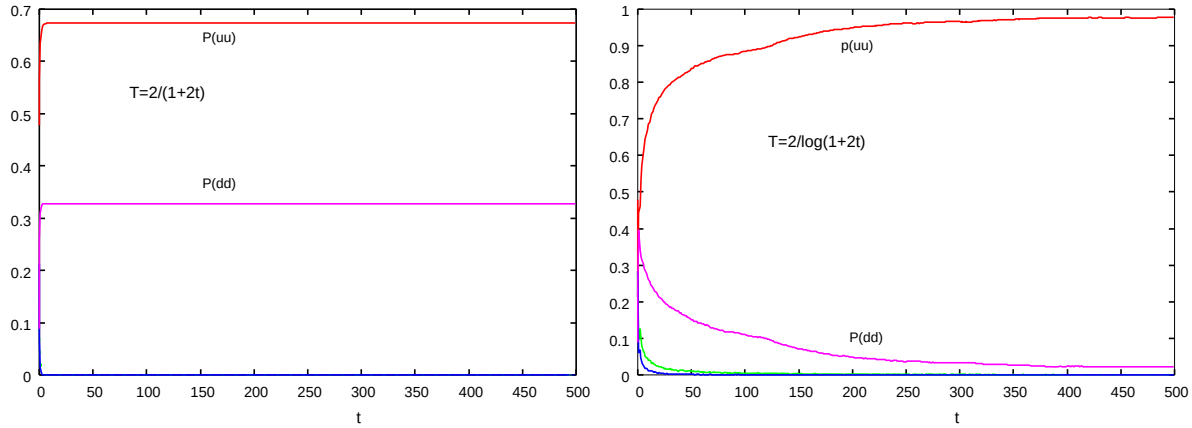

図 7: ノイズレベルを $T(t) = 2/(1+2t)$ (左) , $T(t) = 2/\log(1+2t)$ (右) で下げた場合の各確率の時間変化の様子.

でとることがわかった. 時間が十分に経過し, システムが平衡状態に達すれば, 注目する 1 つのシステムの物理量の長時間平均は分布 $P(s_1, s_2)$ に関する平均で置き換えることができる.

ところで, 我々はエネルギー：

$$E(\boldsymbol{s}) \quad = \quad -Js_1s_2 - h_1s_1 - h_2s_2 \tag{26}$$

がいったい何に関する物理的対象のモデルを表すのか, についてはここまで明確にしてこなかったのではあるが, これは例えば, 2 つの磁石が互いに大きさ J の相互作用を及ぼしあっており, 個々の磁石にはそれぞれ, 上向きに (N 極方向に) h_1, h_2 の強さの磁場がかかっている系として見ることができる. あるいは, 2 つの神経素子からなる回路が 1 つのパターンを Hebb 則により記憶しており, 個々の神経素子にはそのパターン方向へ素子を向けさせるようなバイアスが働いている神経系, つまり, 神経系のエネルギー関数が

$$E(\boldsymbol{s}) \quad = \quad -J\xi_1^1\xi_2^1 s_1s_2 - h_1\xi_1^1 s_1 - h_2\xi_2^1 s_2 \tag{27}$$

で与えられ, 記憶したパターンを $\boldsymbol{\xi}^1 = (\xi_1^1, \xi_2^1) = (1, 1)$ とした場合であると考えることができる.

さて, このシステムのエネルギー E のアンサンブルでの期待値は $\beta \equiv T^{-1}$ と置けば

$$\begin{aligned}
\langle E \rangle &= \sum_{s_1=\pm1}\sum_{s_2=\pm1} E(s_1, s_2)P(s_1, s_2) \\
&= -\left[\frac{\sum_{s_1=\pm1}\sum_{s_2=\pm1}(Js_1s_2 + h_1s_1 + h_2s_2)\mathrm{e}^{\beta(Js_1s_2+h_1s_1+h_2s_2)}}{\sum_{s_1=\pm1}\sum_{s_2=\pm1}\mathrm{e}^{\beta(Js_1s_2+h_1s_1+h_2s_2)}}\right] \\
&= -\frac{\partial}{\partial\beta}\log Z \qquad (28) \\
Z &\equiv \sum_{s_1=\pm1}\sum_{s_2=\pm1}\mathrm{e}^{\beta(Js_1s_2+h_1s_1+h_2s_2)} \\
&= \left[\mathrm{e}^{\beta(J+h_1+h_2)} + \mathrm{e}^{\beta(-J+h_1-h_2)} + \mathrm{e}^{\beta(-J-h_1+h_2)} + \mathrm{e}^{\beta(J-h_1-h_2)}\right] \qquad (29)
\end{aligned}$$

と表すことができる. 従って, エネルギーの期待値を求めるためには, 分配関数 Z, あるいは**自由エネルギー**と呼ばれる次の関数：

$$F \quad = \quad -T\log Z \tag{30}$$

から

$$\langle E \rangle = \frac{\partial}{\partial \beta}(\beta F) \tag{31}$$

という操作を通じて求めることができる．

この自由エネルギーはとても便利な関数である．なぜならば，任意の物理量 $A(s_1, s_2)$ の期待値 $\langle A(s_1, s_2) \rangle$ が算出したければ，$A(s_1, s_2)$ に共役な外場 (パラメータ) a を用いて $aA(s_1, s_2)$ を分配関数の指数部分に加えた自由エネルギー：

$$F = -T \log \left(\sum_{s_1=\pm 1} \sum_{s_2=\pm 1} \mathrm{e}^{\beta(Js_1s_2+h_1s_1+h_2s_2)+aA(s_1,s_2)} \right) \tag{32}$$

を作り，これから

$$\langle A(s_1, s_2) \rangle = -\frac{1}{T} \lim_{a \to 0} \left(\frac{\partial F}{\partial a} \right) = \frac{\sum_{s_1=\pm 1} \sum_{s_2=\pm 1} A(s_1, s_2)\, \mathrm{e}^{\beta(Js_1s_2+h_1s_1+h_2s_2)}}{\sum_{s_1=\pm 1} \sum_{s_2=\pm 1} \mathrm{e}^{\beta(Js_1s_2+h_1s_1+h_2s_2)}} \tag{33}$$

という操作を実行すればよい．この自由エネルギーはここで示したような計算技法上だけではなく，前節 [**連想記憶の数理**] で触れたような相転移現象に関する重要な情報を与えてくれる．それに関しては追々見ていくことになるなるであろう．

また，このシステムの**エントロピー** S は熱力学的関係式：

$$F = \langle E \rangle - TS \tag{34}$$

から，自由エネルギー，及び，**内部エネルギー** $\langle E \rangle$ を用いて

$$S = \frac{1}{T}(\langle E \rangle - F) \tag{35}$$

からみ積もることができる．図 8 に内部エネルギー，自由エネルギー，及び，エントロピーのノイズレベル依存性をプロットする．この図より，例えば，$T = 0$ のノイズゼロの場合には内部エネルギー，つまり，エネ

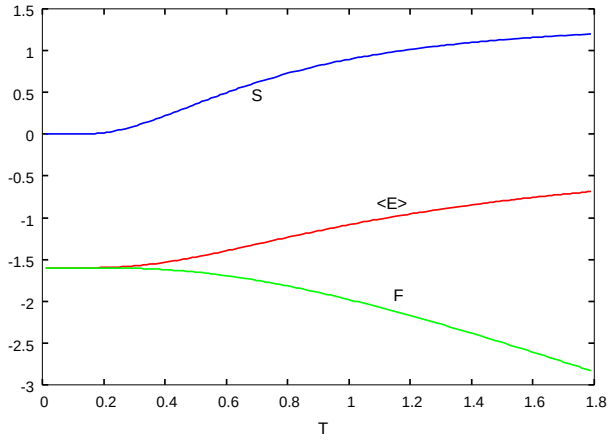

図 8: 我々が考えてきたシステム：$E = -Js_1s_2 - h_1s_1 - h_2s_2$ の内部エネルギー $\langle E \rangle$，自由エネルギー F，及び，エントロピー S のノイズレベル依存性．

ルギーの期待値は $\varepsilon_1 = -J - h_1 - h_2 = -1.0 - 0.5 - 0.1 = -1.6$ であるエネルギーの最小状態に一致する．ノイズレベルが有限であるのであれば，$\varepsilon_2, \varepsilon_3, \varepsilon_4$ のエネルギーをとるシステムも有限の割合で存在するの

で, その期待値は ε_1 からずれた値をとるが, 完全にノイズレベルがゼロであるのであれば, アンサンブルを構成する全てのシステムが $(s_1, s_2) = (1, 1)$ の状態をとり, 従って, アンサンブルの「期待値」といっても, それは完全に ε_1 に一致するわけである.

一方, エントロピーは全ての可能な状態が $(1, 1)$ の一つだけであることから, ノイズ $T = 0$ では状態数の対数として $\log 1 = 0$ となっていることがわかる. また, 十分にノイズが大きければ, 4 つの状態の全てが等確率で現れることになるので, $T \to \infty$ で $\log 4$ に漸近していくことが見て取れる.

ここまでの準備ができた段階であらためて我々が調べたかった問題：

$$E(\boldsymbol{s}) = \left| \sum_{j=1}^{N} a_j s_j \right| \tag{36}$$

を見ていくことにしよう **(次回)**.

問題 1：**[ノイズを利用したアルゴリズム] を用いて以下の問いに答えよ.**

(1) [ノイズを利用したアルゴリズム] をノイズレベル T で動作させることにより, 平衡状態を作り, 得られるアンサンブルを用いてエネルギーの期待値：

$$\langle E \rangle = \frac{\sum_i^{\text{アンサンブルのシステム総数 }(N)} E_i}{\textbf{アンサンブルのシステム総数 }(N)}$$

を求めよ. また, これを図 8 の $\langle E \rangle$ (つまり, 手計算からの結果) と比較せよ.

(2) s_1, s_2 の平衡分布での期待値 $\langle s_1 \rangle, \langle s_2 \rangle$ を自由エネルギー経由で手計算により求めよ.

(3) [ノイズを利用したアルゴリズム] でノイズレベル T の平衡状態を作り, 得られるアンサンブルより

$$\langle s_1 \rangle = \frac{\sum_i^{\text{アンサンブルのシステム総数 }(N)} [s_1]_i}{\textbf{アンサンブルのシステム総数 }(N)}$$

$$\langle s_2 \rangle = \frac{\sum_i^{\text{アンサンブルのシステム総数 }(N)} [s_2]_i}{\textbf{アンサンブルのシステム総数 }(N)}$$

を計算し, (2) で求めた $\langle s_1 \rangle, \langle s_2 \rangle$ と比較せよ.

(4) [ノイズを利用したアルゴリズム] を用いて, アンサンブル中の 1 つのシステムに注目した場合のエネルギー E の長時間平均：

$$\overline{E} = \lim_{\mathcal{T} \to \mathcal{T}_\infty} \frac{1}{\mathcal{T}} \sum_{t=0}^{\mathcal{T}_\infty} [E_t(s_1, s_2)]$$

($\mathcal{T}_\infty$ は十分に大きな数) を求め, (1) で得られた $\langle E \rangle$ の値と比較せよ. $\overline{E} = \langle E \rangle$ が成り立っているであろうか？