地震と自己組織化臨界現象

山崎淳子

羽田野研究室

平成 12 年 2 月 29 日

概 要

地震現象にはグーテンベルグ・リヒター則という経験則が知られている。この経験則から地震の頻度とエネルギーの関係がべき乗法則に従っていることが導ける。本研究ではこれを再現することを目的として、地殻にたまった歪みの応力の放出に着目した 2 つのモデルをシミュレーションした。ここで用いたモデルでは応力を放出する際、わずかなことが原因となり連鎖反応を起こして全体に影響が及ぶことがある。これは自己組織化臨界現象の特徴である。この自己組織化臨界現象の性質を取り入れたモデルによって、地震の頻度とエネルギーのべき乗法則を再現することができた。

目 次

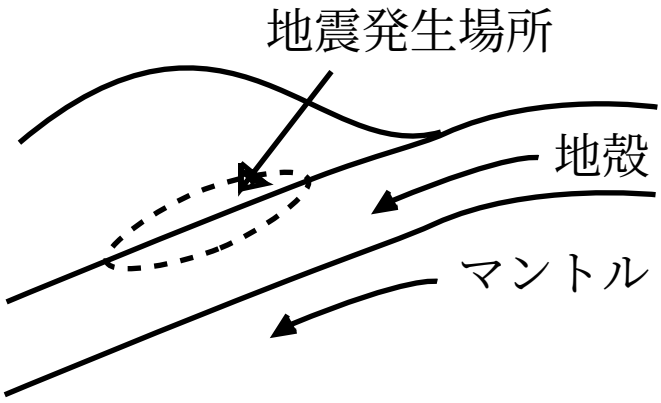

図 1: 地震の起こる仕組み。マントルの対流にひきずられて地殻が動く。それによって地殻に歪みの力がかかっていく。歪みの力がしきい値を超えたら断層がずれると地震が生じる。

1 はじめに

日頃、小さな地震は頻繁に起きて、大きな地震はわずかにしか起きないという規則性を感覚的に知っている。実際に、地震にはグーテンベルグ・リヒター則という経験式が知られている。これによると、マグニチュード M と, そのマグニチュードの地震がおこる頻度 $N(M)$ の関係が

$$\log N = a - bM \tag{1}$$

のように表わされる。ここで a と b は定数で、特に b はほぼ 1 である。また、マグニチュード M とエネルギー E の関係が

$$\log E = \frac{3}{2}M + 11.8 \tag{2}$$

と表わされる。従って地震の頻度とエネルギーの関係が、

$$N \propto E^{-\frac{2}{3}b} \tag{3}$$

という、べき乗法則になることを導ける [1]。

地震は、地球の表面にある地殻が他の地殻に潜り込むことなどによって、ひずみがたまっていき、しきい値を超えて歪みのエネルギーが一気に開放される時に起こる（図 1）。今回の研究では、これから説明する自己組織化臨界現象の性質を取り入れたモデルでべき乗法則を再現することが目的である。

２節では自己組織化臨界現象について説明をして、３節ではモデルの説明をする。４節で結果の報告を行う。

2　自己組織化臨界現象とは

自己組織化臨界現象の理論はP. Bak らが提唱した [2]。この理論の特徴を以下に述べる。

(i) 相互作用のある大規模で複雑な系は自然に臨界状態へと自己を組織している。

(ii) 臨界状態にあるとき、ちょっとしたことが連鎖反応を起こし、系の無限個の要素にも影響を及ぼすことがある。

この自己組織化臨界現象は、株式市場や生態系などにもあてはまると期待されている。

この特徴を上に書いた地震の起こり方にあてはめると、(i) においては、地殻に歪みの応力がかかっていき、しきい値まで近づいていくことに相当する。(ii) においては、地殻のどこかで歪みの応力がしきい値を超えると、その影響が周りに広がって連鎖反応が起こることに相当する。自己組織化臨界現象において重要なことはこの連鎖反応である。一部分から影響を受けたところが、その影響を受けたことによって、しきい値をこえることがある。そして、さらに影響がひろがっていって、次々としきい値をこえる場所があらわれ、系全体に影響が及ぶ可能性があるのだ。この連鎖反応によって、少しだけずれたはずの断層の影響が広い範囲までおよび、大きな地震になることがある。

本研究においては地震の広がりかたに連鎖反応が起こるようなモデルを２つ用いた。１つめは連鎖反応が起こる時に、広がり方が決定論的に起こるもの、２つめはそれが確率的に起こるというものである。

3　モデルの説明

地震は、地殻にたまった歪みのエネルギーが開放されることでおこる。地殻は、岩石から構成されていて弾性的な性質をもっている。そこで、その弾性的な性質を表わすために、ブロックをバネで繋いだモデルが考案された。そのモデルをもとにつくられたのが以下のモデルである。

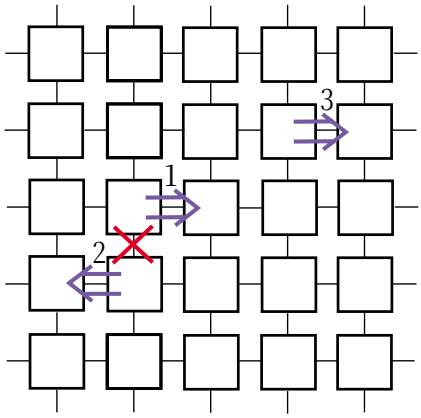

図 2: 決定論的モデル。× 印の場所でしきい値が超えたとすると 1 や 2 の方向に力が放出される。しきい値を超えていない 3 のブロックは 1 の方向に力が働くが 1 や 2 の力の影響を受けている。

3.1 決定論的モデル

どこかで歪みの力がしきい値をこえとき、そこから広がるひずみの応力を決定論的に決定するモデルである [2]。(このモデルはブロックの動きそのものを表わそうとはしていない。) 地震の説明でででてきた地殻を、2 次元の $N \times N$ 個のブロックで見立てる (図 2)。以下のような記号を用いる：

- $\vec{r}$: ブロックの位置を表わす。
- $\vec{r} + \vec{e}_x$: $+x$ 方向の隣のブロックの位置
- $\vec{r} + \vec{e}_y$: $+y$ 方向の隣のブロックの位置
- $\sigma_x(\vec{r})$:$\vec{r} + \vec{e}_x$ にあるブロックから $+x$ 方向へ引っ張られる垂直応力
- $\sigma_y(\vec{r})$:$\vec{r} + \vec{e}_y$ にあるブロックから $+x$ 方向へ引っ張られるせん断応力
- $\sigma_x^{thr}(\vec{r})$ と $\sigma_y^{thr}(\vec{r})$:断層が耐えられる応力の限界を表わすしきい値

次にモデルの構造を説明する。

(i) 時間とともに、すべての $\vec{r}$ について歪みの力が増えていく：

$$\sigma_y(\vec{r}) \to \sigma_y(\vec{r}) + p \tag{4}$$

ブロック間はばねでつながっていて、時間毎にひずみの力 σ がたまっていくことを、$\sigma_y(\vec{r})$ に p を足すことで表現している。

(ii) ある場所 $\vec{r_0}$ でひずみの力がしきい値をこえる時 ($|\sigma_i(\vec{r_0}) \equiv \sigma_0| \geq \sigma_i^{thr}$ ただし i は x または y) 断層がずれて、地震が起こる。周りへの影響は距離に関係してひろがっていく。その広がり方は、

(a) しきい値をこえたところでは

$$\sigma_i(\vec{r_0}) \to \sigma_i(\vec{r_0}) - \sigma_0 \tag{5}$$

(b) それ以外のブロックでは $j = x, y$ に対して、

$$\sigma_j(\vec{r}) \to \sigma_j(\vec{r}) + \frac{d}{d-1}\sigma_0(G_j(\vec{r} - \vec{r_0}) - G_j(\vec{r} - \vec{r_0} + e_i)) \tag{6}$$

であると定める。しきい値を超えたところの広がりが止まったら 1 回の地震とする。どこかで、しきい値を超えるところがあるとき、G_x, G_y を周りの σ に加えていくので、連鎖反応が起こる可能性がある。

ここで $G_x(\vec{r})$ と $G_y(\vec{r})$ は力の広がりかたを表わすグリーン関数である。以下でこれを具体的に説明する。まず $\sigma_x(\vec{r})$ と $\sigma_y(\vec{r})$ が満たす 2 つの方程式を書く。地震が起きていないときは各ブロックは静止しているから、働いている力の合計が 0 にならなくてはいけない。そこで、釣り合いの式がかける：

$$\sigma_x(\vec{r}) + \sigma_y(\vec{r}) + \sigma_x(\vec{r} - \vec{e_x}) + \sigma_y(\vec{r} - \vec{e_y}) = 0 \tag{7}$$

次に、ある 1 単位の閉曲面をとると応力の合計は 0 になる。そこで、次の式がかける。

$$\sigma_x(\vec{r}) + \sigma_y(\vec{r} + \vec{e_x}) - \sigma_x(\vec{r} + \vec{e_y}) - \sigma_y(\vec{r}) = 0 \tag{8}$$

地震が起こると、しきい値を超えたところから応力 σ_0 が開放される。開放される前後で共に式 (7) と式 (8) が成り立たなければならないことから、周囲への影響は自動的に決まる。力が開放される時、図 2 のように、$\vec{r} + \vec{e_y}$ のブロックには $+x$ 方向に F という力がかかり、$\vec{r} - \vec{e_y}$ のブロックには $-F$ の力がかかるとする。その影響を σ' とすると、

$$\sigma'_{x,y}(\vec{r}) = -FG_{x,y}(\vec{r} - \vec{r_0}) + FG_{x,y}(\vec{r} - (\vec{r_0} + \vec{e_y})) \tag{9}$$

と書くことができる。 このとき、

$$G_x(\vec{r}) = \frac{1}{(2\pi)^2}\int_0^{2\pi}\int_0^{2\pi} dk_x dk_y \frac{(1-e^{ik_x}e^{ikr})}{4-2(\cos k_x + \cos k_y)} \tag{10}$$

$$G_y(\vec{r}) = \frac{1}{(2\pi)^2}\int_0^{2\pi}\int_0^{2\pi} dk_x dk_y \frac{(1-e^{ik_y}e^{ikr})}{4-2(\cos k_x + \cos k_y)} \tag{11}$$

となる。式 (10),(11) は $0 \sim 2\pi$ までの連続的波数で表わされている。モデルで実際に扱う時は $k = 1 \sim N$ までの離散座標に変換して、$k = l = 0$ の場合、

$$G_{x,y} = \frac{1}{N^2}\sum_{k=1}^{N}\sum_{l=1}^{N}\frac{2(x+y)+1}{4} \tag{12}$$

それ以外については、

$$G_x = \frac{1}{N^2}\sum_{k=1}^{N}\sum_{l=1}^{N}\frac{\cos(\frac{2\pi}{N}kx + \frac{2\pi}{N}ly) - \cos(\frac{2\pi}{N}k(x+1) + \frac{2\pi}{N}ly)}{4-2(\cos(\frac{2\pi}{N}k) + \cos(\frac{2\pi}{N}l))} \tag{13}$$

$$G_y = \frac{1}{N^2}\sum_{k=1}^{N}\sum_{l=1}^{N}\frac{\cos(\frac{2\pi}{N}kx + \frac{2\pi}{N}ly) - \cos(\frac{2\pi}{N}kx\frac{2\pi}{N}l(y+1))}{4-2(\cos(\frac{2\pi}{N}k) + \cos(\frac{2\pi}{N}l))} \tag{14}$$

とする。

つぎに、シミュレーションプログラムの具体的な説明をする (図 3):

(i) $N, p, t_{\max}$ の入力

N は 1 辺当たりのブロックの数、p は時間毎に加えられる歪みの応力、$t_{\max}$ はシミュレーションする時間である。

(ii) $\sigma_{x,y}^{thr}$ を決定する

$\sigma_{x,y}^{thr}$ は $[0,1]$ の一様乱数を与える。

(iii) G_x, G_y を読み込む。

あらかじめ G_x, G_y を式 (12),(13),(14) に従って求めておき読み込む。

(iv) 時間のループ

0〜$t_{\max}-1$ まで時間発展させる。

(v) $\sigma_y + p \to \sigma_y$

時間毎に p をたして、歪みの応力がかかることを表現する。

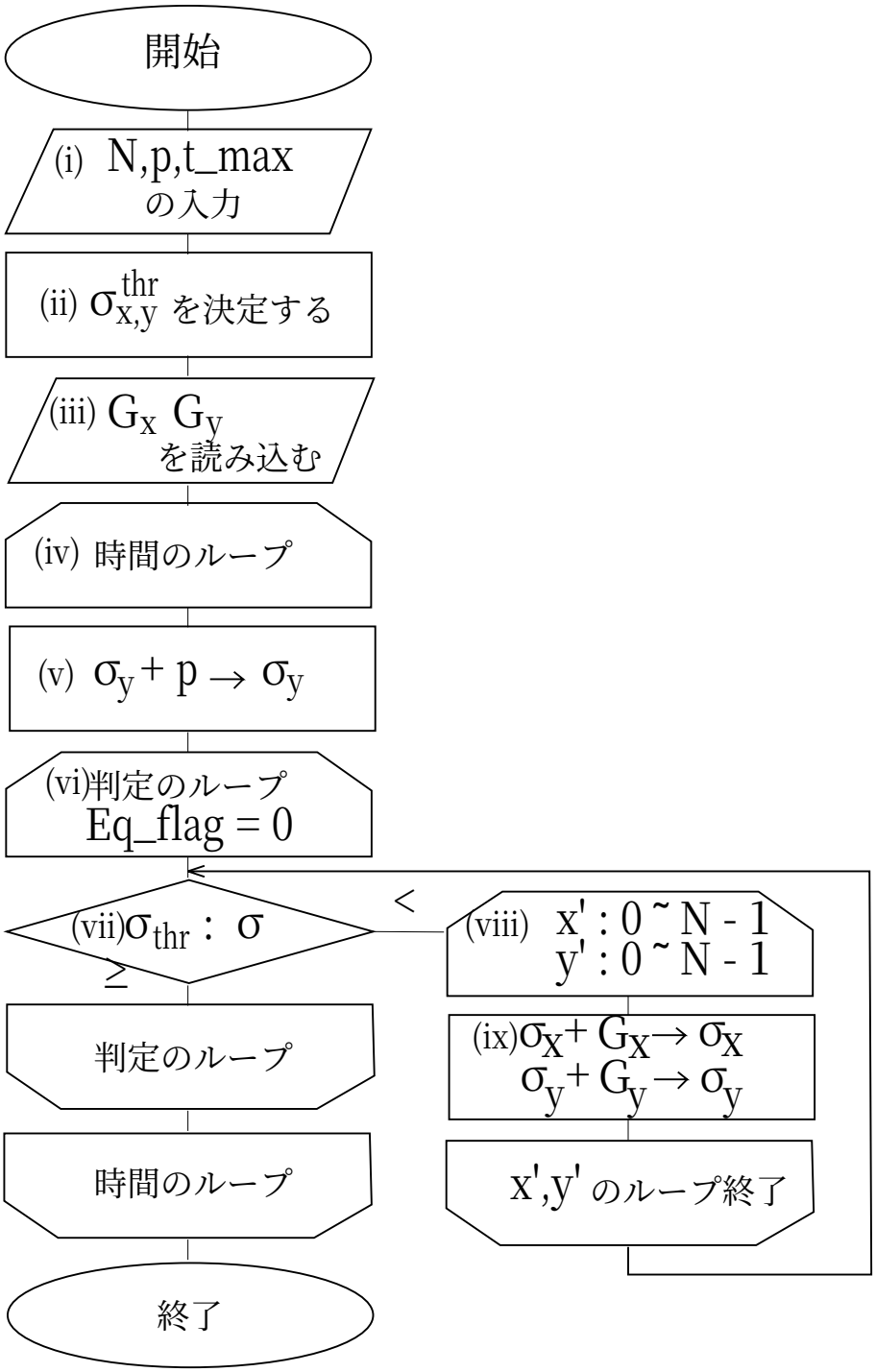

図 3: 決定論的モデルの流れ図

(vi) 判定のループ

どこかしきい値を超えるところがあったら $Eq_{\text{flag}} = 1$ として、地震が起きたことを表わす。もし、全ての場所で $Eq_{\text{flag}} = 0$ だった場合、判定のループを出ることになる。

(vii) $\sigma^{thr} : \sigma_{x,y}$

各ブロックにおいて、時間発展毎に大きくなる $\sigma_{x,y}$ が σ^{thr} より大きいかどうか判定する。σ^{thr} より大きくなれば (viii) に進み、そうでなければループを抜ける。ここでしきい値の起こった回数を数える。

(viii) しきい値を超えたところ以外 (影響を受けるところ) のループ

x', y' は、影響を受けるところで、周期的境界条件を用いている。

(ix) $G_{x,y}$ を加える。

式 (12),(13),(14) に従って求めた $G_{x,y}$ をしきい値を超えたところ以外に加えていく。

その後、しきい値の超えるところがなくなるまで、(vi) のループを続ける。

3.2 確率論的モデル

2つめのモデルでは、歪みの力の広がりを確率的に起こす [3]。碁盤を、地殻にみたて、しきい値を超えた場所に白石を置くものとする。はじめに、一ケ所でしきい値を超えたとして碁盤の中央に白い碁石をおく。その白い石の最隣接4ケ所に確率 p で白い碁石を、確率 $1-p$ で黒い石をおく。白い石がおかれたら、さらにその周り 3 ケ所に同様に石を置いていく。なお、すでに石の置かれているところには置くことはできない。最終的に白い石がすべて黒い碁石に囲まれると 1 回の地震は終了する。白い石の広がり方は、白い石の最隣接の場所におくことから菱形状であることがわかる。ひし形の辺上にある、石の置いていないところのまわり4ケ所に白い石があるかどうか調べ、もしあれば白い石か黒い石を確率的に置く。以下、モデルのプログラムを具体的に説明する (図 4)

(i) $N, p, t_{\max}$ の入力ひし形の対角線上の数が N、白い石の置かれる確率 p、地震の起きる回数を $t_{\max}$ を入力する。

(ii) はじめの白い石を中央に置く

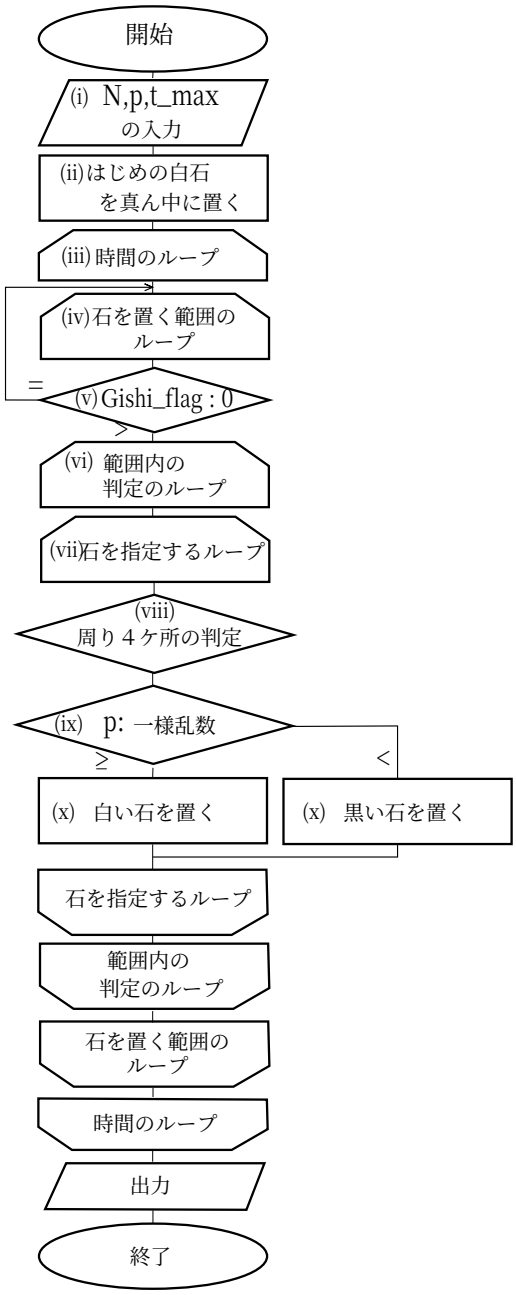

図 4: 確率論的モデルの流れ図

(iii) 時間のループ

(iv) 石を置く範囲のループ

石を置く範囲を広げていくとき、範囲全体をスキャンするループである。このモデルは決定論的モデルとは異なり、周期的境界条件を用いていない。

(v) $Goishi_{\mathrm{flag}}$ の判定

ある大きさの範囲の中で白い石が置かれたら $Goishi_{\mathrm{flag}}$ が増えていく。ここで、$Goishi_{\mathrm{flag}} \geq 1$ のときは更に石の置かれる可能性のある範囲を広げる。$Goishi_{\mathrm{flag}} = 0$ のときは新たに白い石が置かれていないので、黒い石に囲まれているか、石を置くところがなくなったことのどちらかが考えられる。

(vi) 範囲内の判定のループ

(iv) で石を置く範囲を決めたら、中央の白い石から順にスキャンして判定をしなおす。これによって、まわり込みながら広がる白い石をカバーすることができる。

(vii) 石を決定するループ

具体的に一つずつ石が置かれていない所を指定するループである。

(viii) 周り４ケ所の判定

石の置かれていない所の周り４ケ所に白石が置かれているかどうか判定する。

(ix) p による判定

(viii) において、周りに白石が置かれていたら新たに石を置くために、$[0, 1]$ の一様乱数を与える。その乱数が確率 p より大きいか、小さいか判定する。

(x) 白石か黒石を置く。

(ix) において、与えられた乱数が確率 p より小さいと白石を置く。p より大きいと黒石を置く。

以上のように、範囲を広げていきながら石を置いていく作業を行う。

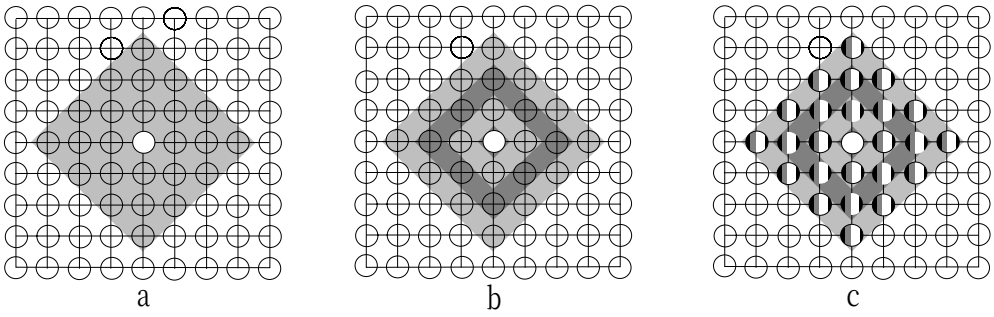

図 5: 石の判定の方法。(a) は流れ図の (iv) において石を置く範囲を決めるのに相当する。(b) は (vi) の範囲内の判定のループを示すもので、中心から広げて判定を行う。(c) は (vii) において１つ１つ石を指定して実際に判定を行う様子を示す。

この碁石のモデルでは確率 p が小さい時には白石は広がっていかず、逆に p が大きすぎると毎回範囲を石が覆い尽くしてしまう。このモデルはパーコレーションモデルの一種で、ある臨界確率 p_c が存在する。$p < p_c$ では白石の領域は有限の大きさで止まるが $p = p_c$ で無限の大きさのクラスターができる。臨界確率 p_c の時に白石の大きさの分布はべき乗法則になると期待される。これは、自己組織化臨界現象の特徴である「臨界状態においてちょっとしたことが連鎖反応を起こし、系の無限個の要素にも影響を及ぼすことがある」ということにあてはまる。つまり $p = p_c$ のときに地震の性質が現れることがわかる。予備的計算で臨界確率を調べたところ、およそ $p_c = 0.582$ であった。そこで以降は $p = p_c$ として数値計算を行った。

4　モデルの結果

4.1　決定論的モデル

シミュレーションから時間としきい値を超えた場所の数の関係が得られ、図 6 のようになった。小さな地震が頻繁に起こり、大きな地震は少ないことを見ることができる。また、図 6 では常に地震が起きているように見えるが、例えば $t = 6000$〜7000 に起きた地震の様子を図 7 で見てみる。すると、次の地震が起こるまでに間隔があり、実際の地震の特徴を

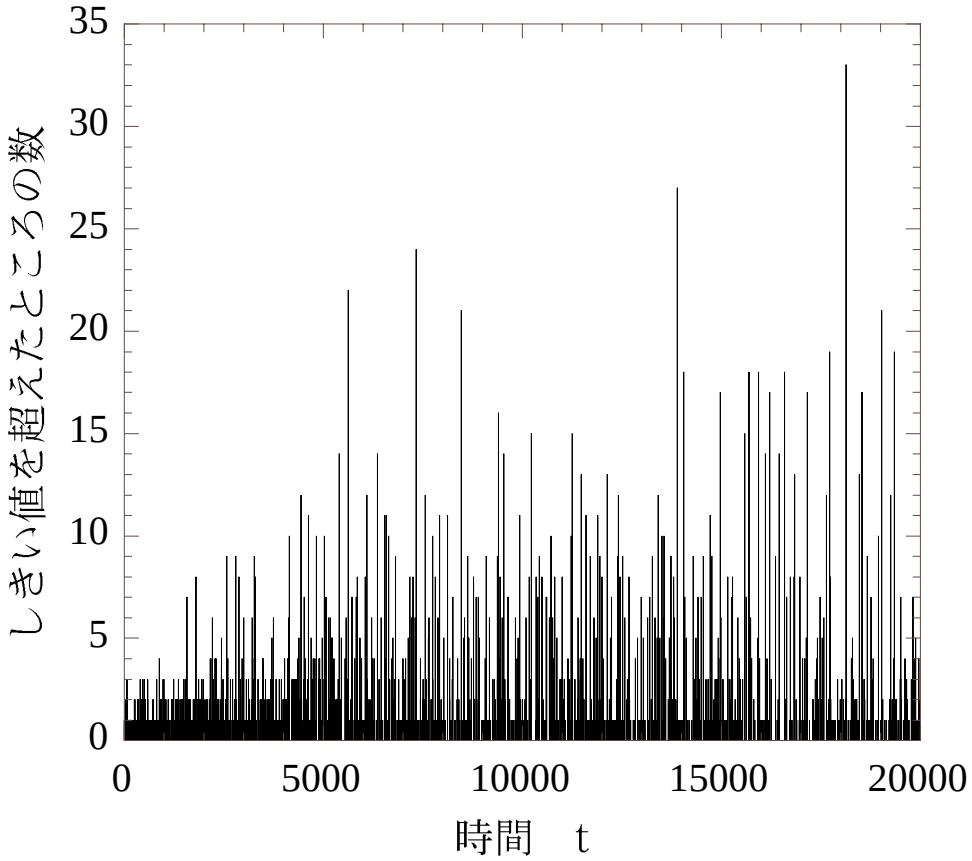

図 6: シミュレーション時間としきい値を超えた場所の数。$N = 40, t_{\max} = 20000, p = 10^{-3}$ の場合。

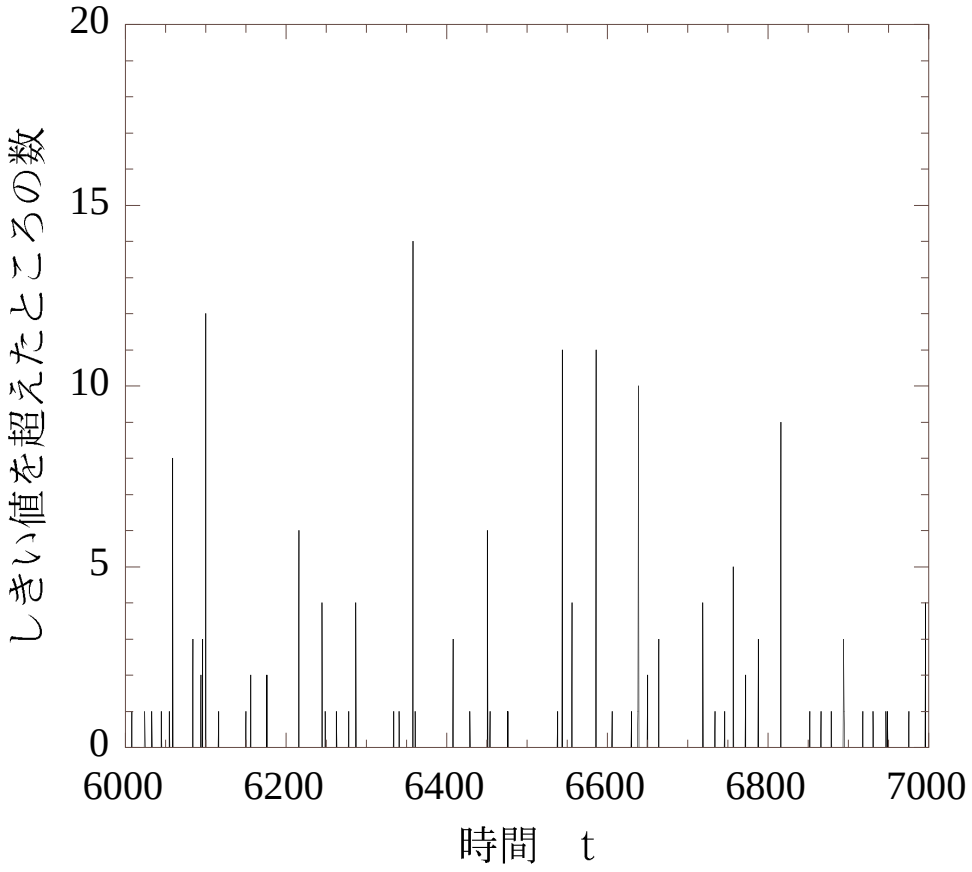

図 7: シミュレーション時間としきい値を超えた場所の数。$N = 40, t_{\max} = 20000, p = 10^{-3}$ の場合 。図 6 のうち $t = 6000$〜7000 の部分を拡大した。

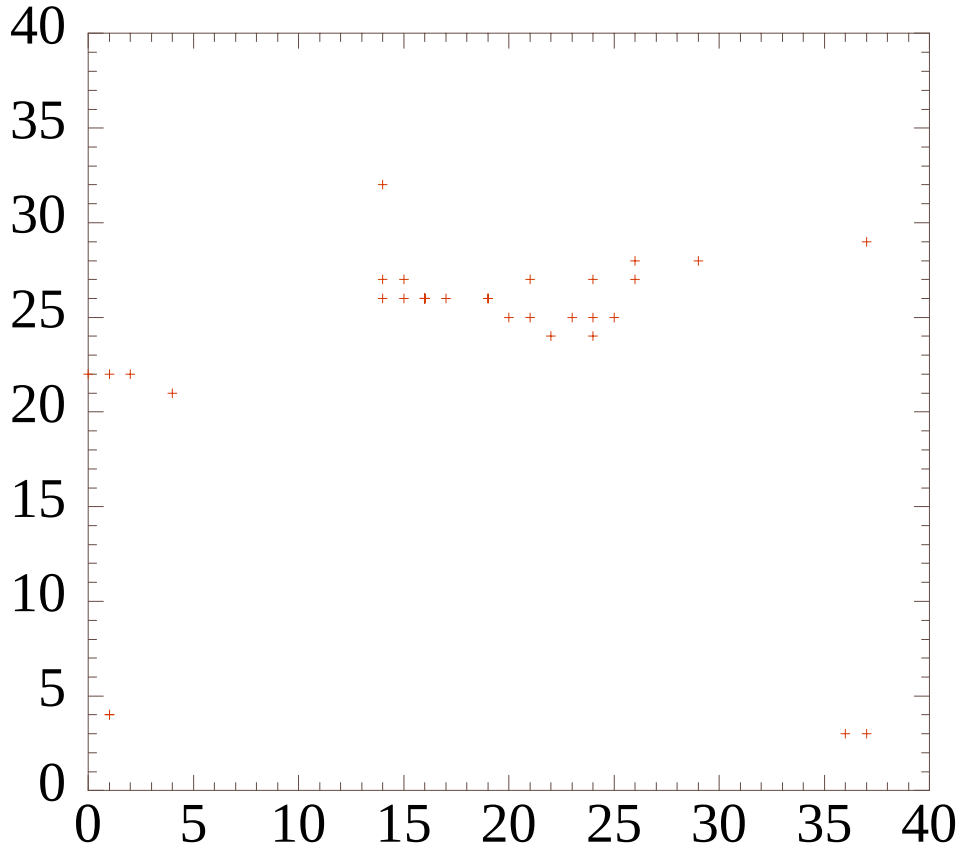

図 8: しきい値を超えた場所。図 6 における $t = 9740$ のとき、しきい値を超えた場所をプロットした図。連鎖反応が起こっていることがわかる。

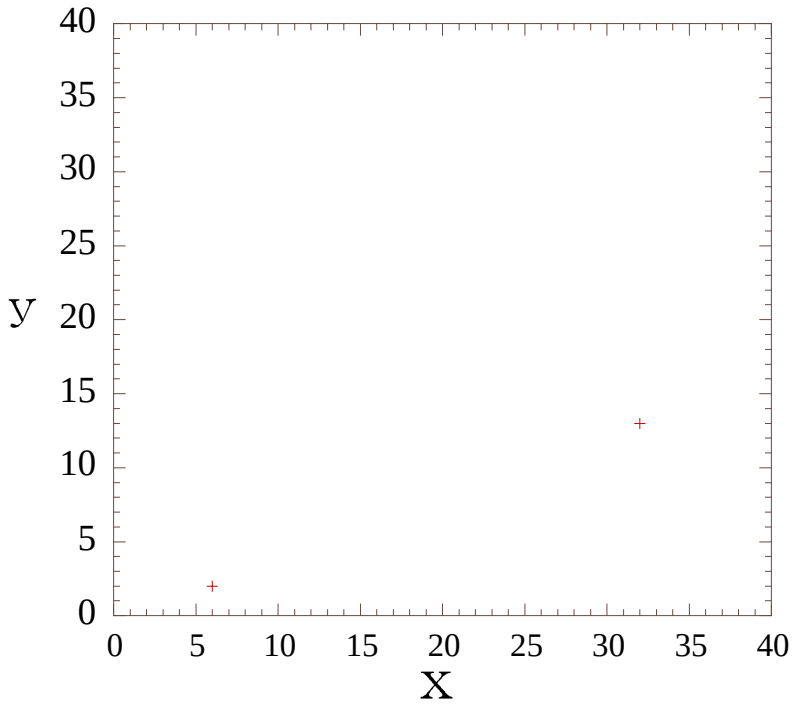

図 9: しきい値を超えた場所。図 6 における $t = 2$ のとき。この場合は連鎖反応が起こっていない。

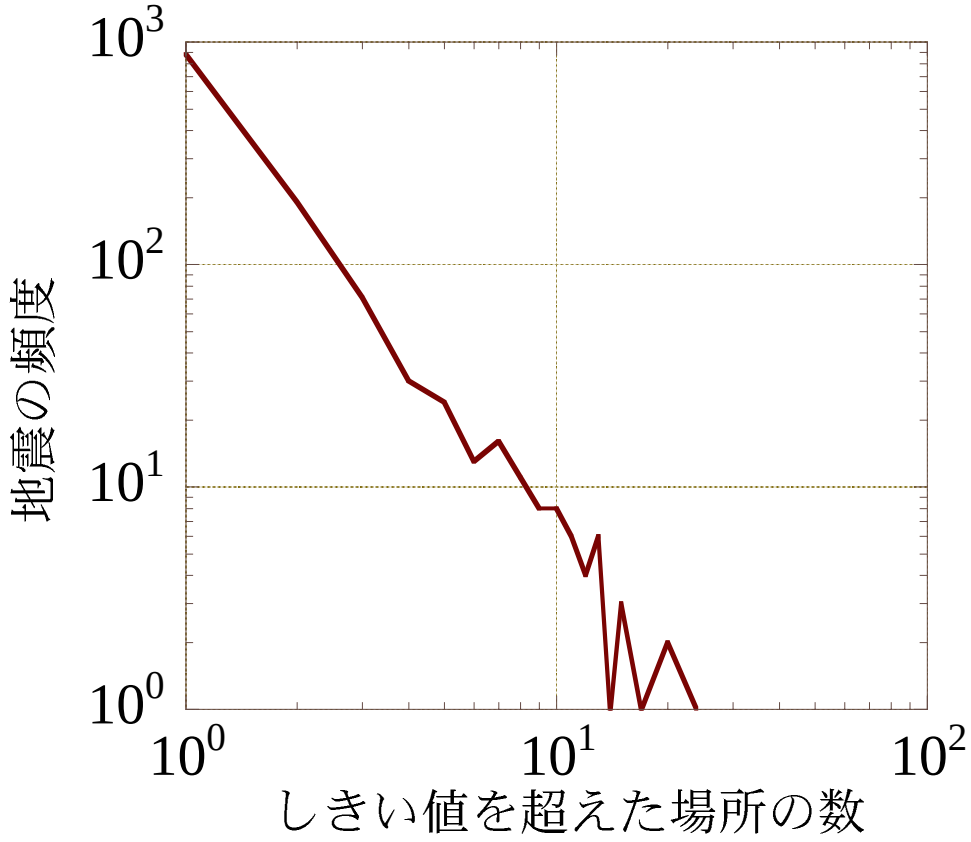

図 10: しきい値を超えた場所の数と、そのような地震の頻度の関係。$p = 0.0001, N = 40, t_{\max} = 10000$ のときの結果である。べき乗法則が現れている。

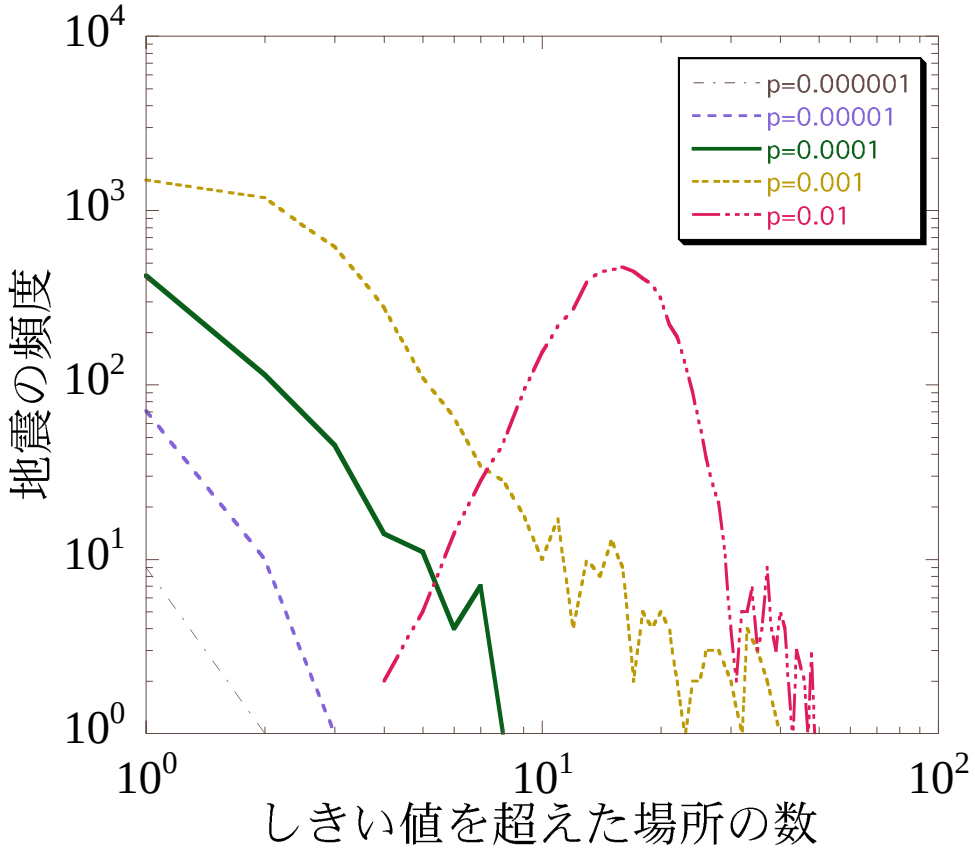

図 11: p を変化させたときの地震の規模と頻度の関係の変化の様子。$N = 40, t_{\max} = 10000$ のときに p を変化させたところ、$p = 0.0001$ のときべき乗法則がはっきりと現れることがわかる。p がそれ以下の場合地震が少ししか起こらない。また p がそれ以上の場合、べき乗法則から外れていくことがわかる。

とらえていることが分かる。次に、図 6 の $t = 9740$ と $t = 2$ において, しきい値を超えた場所をプロットしたのが図 8 と図 9 である。図 8 では壊れたところ（しきい値を超えたところ）が横に連なって広がっていて、連鎖反応が起こっていることが分かる。一方、図 9 では、壊れたところが 2 点だけで連鎖反応は起こっておらず、ブロック全体が臨界状態に達していないことが分かる。

しきい値をこえた場所の数と地震の頻度の関係は図 10 のようになった。両対数プロットで、直線に乗っているのでべき乗法則が現れていることがわかる。

なお、時間毎に加える p を変化させたのが図 11 である。$p = 10^{-2}$〜10^{-6} の 5 つで変化させたところ、$p = 10^{-4}, 10^{-5}$ のときはべき乗法則が保たれた。しかし、$p = 10^{-3}$ のときにはべき乗法則からは遠ざかっている。このモデルにおいては、p が大きくなれば地殻が動きが大きくなり、小さくなれば地殻の動きがゆっくりになることに相当する。p が大きすぎるとべき乗法則からはずれて小さい地震は減ってしまう。つまり、p は重要な要素であることがわかる。モデルでは、p を調節することができるが、本当の地震では自然にこの p が決定されているようなメカニズムがなくてはならない。

式 (3) における地震のエネルギー E は、このモデルでは 1 回の地震で越えられたしきい値の合計に相当すると考えられる。エネルギー（σ^{thr} の合計) と地震の頻度の関係をプロットしたのが図 12 である。両対数プロットで直線になっていて、式 (3) のべき乗法則を表わすことができた。

4.2 確率論的モデル

白石の置かれる確率 p が臨界確率 $p_c \cong 0.582$ より低いときと高いときに、白石が広がる様子をそれぞれ図 13 と図 14 に示す。$p = 0.53$ で p_c より白石が置かれる確率が低いとき、図 13 は黒石が白石を囲んでいて、白石の広がりは止まっている。しかし確率 $p = 0.8$ のときの図 14 では、碁盤が黒石と白石の両方で被われている。シミュレーションでは碁盤のサイズが有限であるが、実際には白石は更に広がって無限サイズの碁盤をも埋め尽くすであろう。

確率が $p = p_c$ のときには、シミュレーションをくり返すたびに図 13 や図 14 のようなあらゆる白石の広がり方を見ることができる。つまり小さな地震や大きな地震が取り混ぜて起こる。

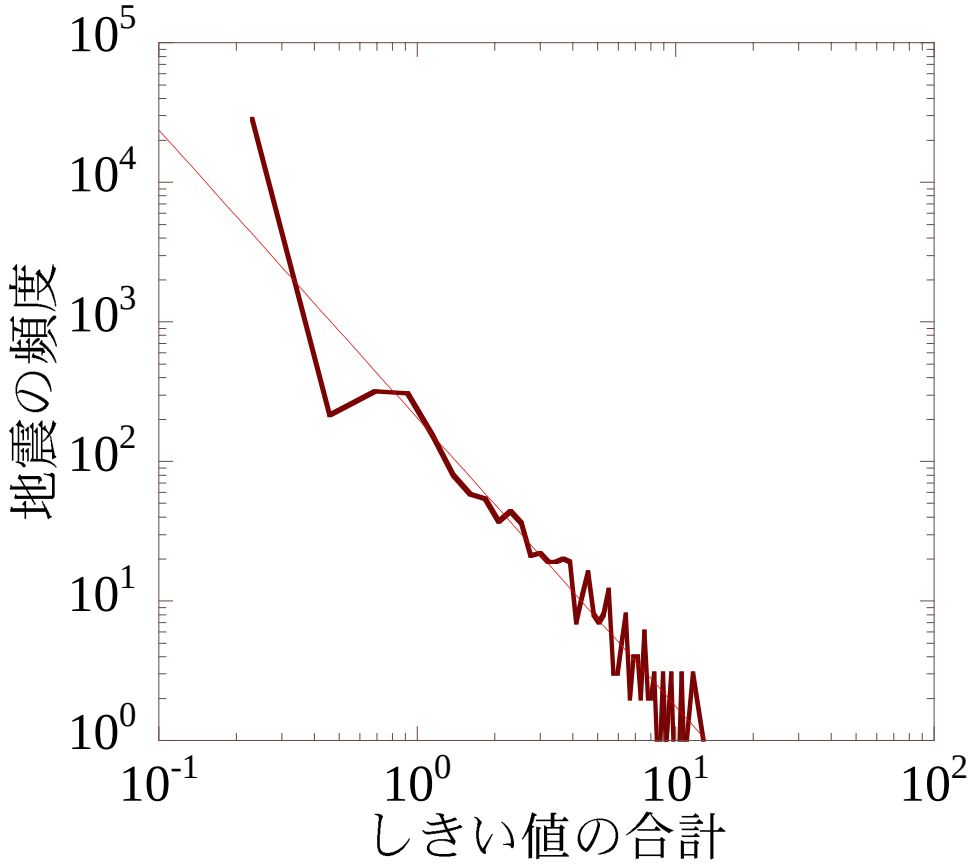

図 12: しきい値の合計と頻度の関係。$N = 40, p = 0.0001, t_{\max} = 10000$ のとき,σ^{thr} の合計を 0.23 の幅で区切ってヒストグラムをとった。σ^{thr} の合計と、地震の頻度の関係がべき乗法則になっていることがわかる。

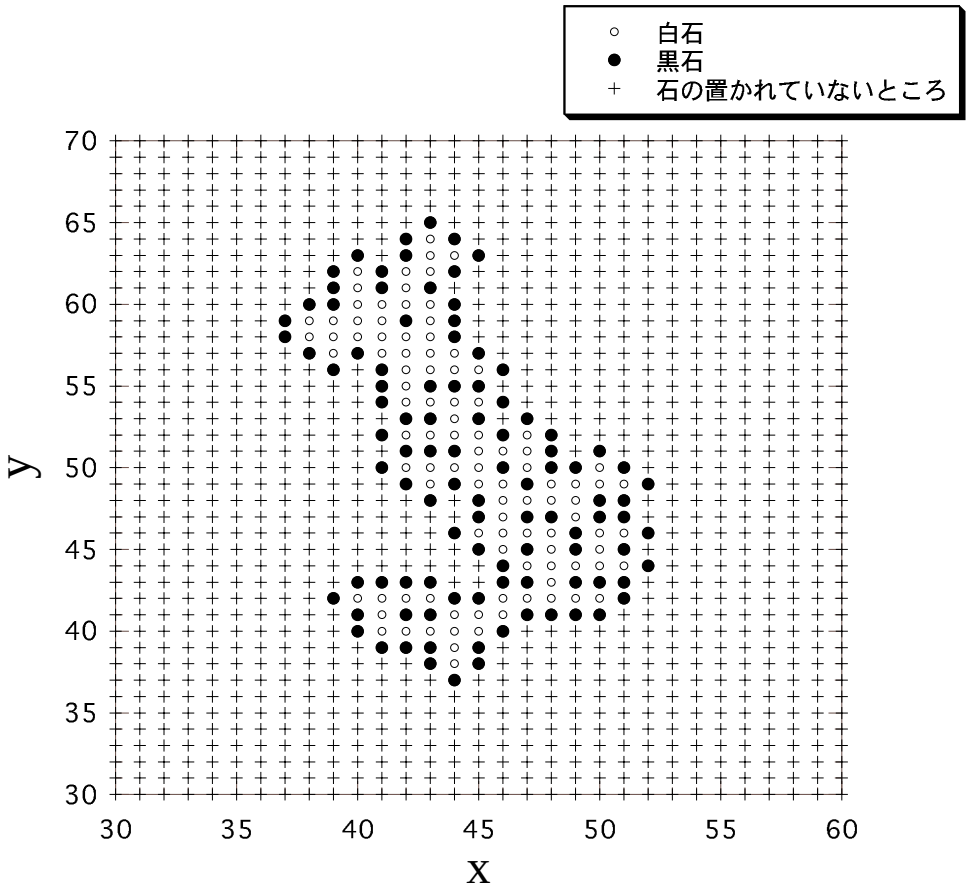

図 13: 白石の広がった様子。$p = 0.53, N = 100$ のとき置かれた白石の数は 100 個。まだ空間を埋め尽くすことはできない。

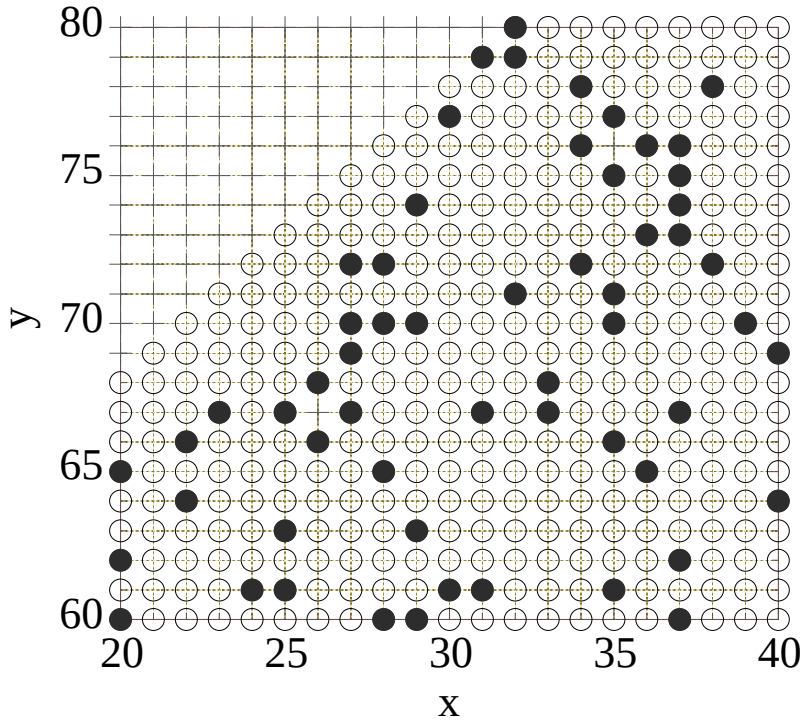

図 14: 白石の広がった様子 (一部分を拡大した)。$p = 0.80, N = 100$ のとき置かれた白石の数は 3758 個。石を置くところが有限でそれ以上石を置くことはできないが、まだ更に範囲を広げれば白石を置くことができ、無限大に広がることが予想される。

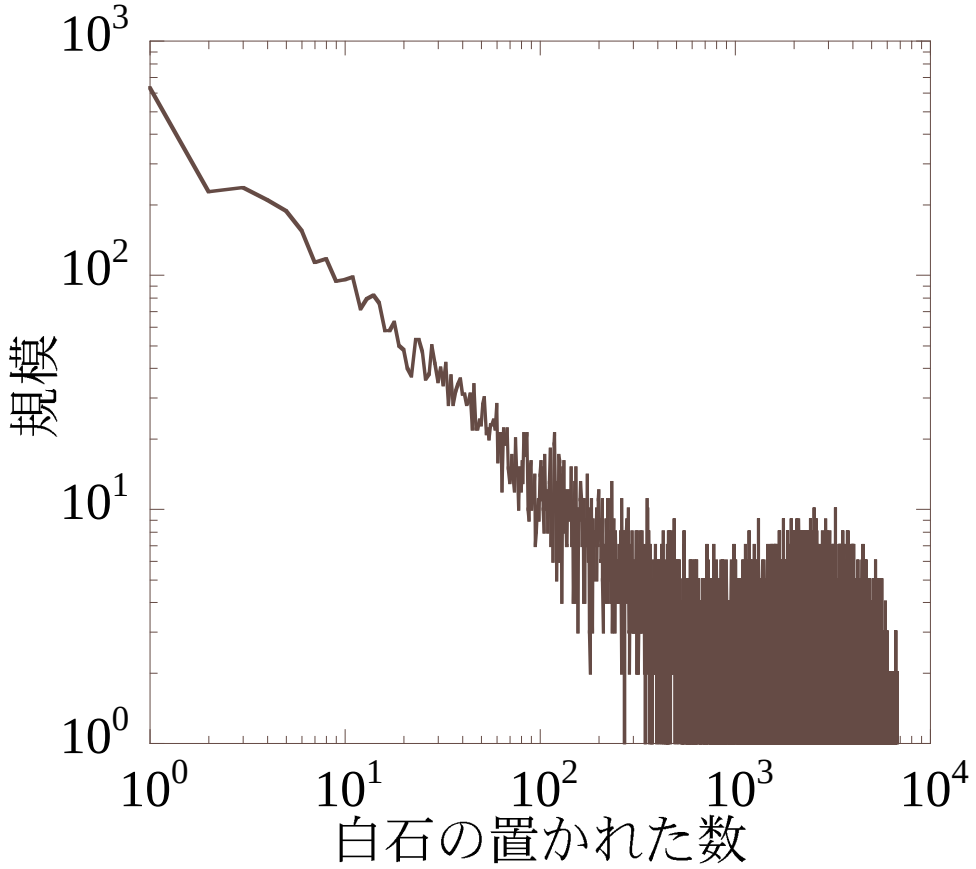

図 15: 白石の置かれた数と頻度の関係。$p = 0.582$、$N = 200$ のときの白石の置かれた数と頻度の関係でべき乗法則があらわれた。

つぎに、モデルの結果を示す。１つの白石から開放される断層のエネルギーが一定であるとすると、白石の数が地震の全エネルギー E に相当しているといえる。図 15 は置かれた白石の数とその大きさの地震の頻度の関係である。両対数プロットで直線になっていることから、べき乗法則が現れたことがわかる。つまり、このモデルで、地震のエネルギーと頻度の関係がべき乗法則であるといえた。ただし、直線になっているのは横軸の白石の数が 1000 位のところまでで、それより白石の数が大きくなると山のようになっている。これは、碁盤が有限であるために、本来なら石を置くことができるはずの碁盤の目に石を置けなくなってしまったためと考えられる。

5　まとめ

本研究で用いた２つのモデルから地震の頻度とエネルギーの関係をベキ乗法則で表すことができた。自己組織化臨界現象において重要な連鎖反応が地震の性質を捉えていると思われる。したがって、簡単な２つのモデルで地震を再現できたといえる。

決定論的モデルではしきい値の合計と頻度の関係が

$$N \propto (\text{しきい値の合計})^{2.0} \tag{15}$$

となった。一方、確率的モデルでは地震の頻度と白石の数の関係は、

$$N \propto (\text{白石の数})^{0.8} \tag{16}$$

となった。式 (3) から E の指数は $-\frac{2}{3}b$ となっている。ここで分母の 3 は実際には次元なので、これらのモデルでは 2 とすべきである。また b は経験的に $b \simeq 1$ となっている。従って、E の指数はおよそ -1 になるはずである。よって指数が 0.8 である確率論的モデルの方が地震の性質をよく表せていることがわかった。

確率論的モデルでは $p = p_c$ のときに地震が起こっている。確率モデルの結果が正しいとすると、現実の地震は自然と臨界状態に向かっているのではなく、常に臨界状態が続いていると思われる。ところが、決定論的モデルでは系全体が臨界状態でないときでも、しきい値を超えて地震が起こる可能性がある。そのため現実よりも地震の頻度が多くなってしまうことが考えられる。また、決定論的モデルの結果において横軸そのものの検討をするべきである。

今後の課題は、一点目は 2 つのモデルを 3 次元化することである。2 点目は確率論的モデルを、決定論的モデルのように、地震の起こる時間間隔を取り入れたモデルにしていきたい。

参考文献

[1] 高安秀樹『フラクタル科学』(朝倉書店、1987)

[2] Kan Chen and Par Bak,Phys.Rev.A43,625(1991)

[3] 大塚道男、地震第 2 号、24(1971),13 および 215

A　決定論的モデルのプログラムリスト

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

void Sxy_0andSxy_thr(double Sx[][100],double Sy[][100],
               double Sx_thr[][100],double Sy_thr[][100],
               int N );

void Sy_plus_p (double Sy[][100],double p,int N);

void ReadGxGy  (double Gx[][100], double Gy[][100], int N);

void histogram (int kibo[] , int t_max);
double  kousin    (int Sx_flag[][100],int Sy_flag[][100],int N,
       double Sx[][100],double Sy[][100],
       double Sx_thr[][100],double Sy_thr[][100]);

int  hantei( double Sx[][100],double Sy[][100],
                  double Sx_thr[][100],double Sy_thr[][100],
                  double dSx[][100],double dSy[][100],
                  double Gx[][100],double Gy[][100] ,
                  int N,int d,
                  int *Count_x_p,int *Count_y_p,int *Eq_flag_p,
                  int Sx_flag[][100],int Sy_flag[][100],int t);


/*    void hindo(int kibo[], int t_max);   */

     /*"d" ha delta , "S" ha siguma  */
     /* N : site no okisa            */
     /* p : time goto ni fueru forse */
     /* d : demension                */

int main()
```

```
      {
           FILE    *f_count,*f_kibo ,*f_hindo;
           int     x,y,N,i,j,l,m,d,t,t_max ;
           int     x_dash ,y_dash;
           int     Count_x,Count_y,Count[100000];
           int     kibo[10000],h;
           int     xx,yy;
           double  p;
           double  Sx[100][100],Sy[100][100];
           double  dSx[100][100], dSy[100][100];
           double  Gx[100][100],Gy[100][100];
           int     Sx_flag[100][100] ,  Sy_flag[100][100];
           int     Eq_flag;
           double  Sx_thr[100][100]  ,  Sy_thr[100][100];

        f_count =   fopen("count_t.dat","w");
f_kibo  =   fopen("kibo.dat","wr");

    /* shokiti no settei */

        printf(" N ( <=100 ) = "); scanf("%d",&N);
        printf(" p =  "); scanf("%lf",&p);
        printf(" t_max =  "); scanf("%d",&t_max);
        printf(" d =  "); scanf("%d",&d);

     /* siguma wo zero ni suru */
     /* Siguma_threshold wo motomeru */

        Sxy_0andSxy_thr( Sx,Sy,Sx_thr,Sy_thr,N );

    /* Gx,Gy wo yomikomu */

        ReadGxGy(Gx,Gy,N);
```

```
/********keisan hajime!!*****************************/

  for( t = 0 ; t < t_max ; t++ ){             /* loop_1 */

        Count_x =0;
        Count_y =0;

    /* p wo tasu*/
      Sy_plus_p( Sy,p,N );

    for( x = 0 ; x <= N-1 ; x++ ){
    for( y = 0 ; y <= N-1 ; y++ ){

     Sx_flag[x][y] = 0;
     Sy_flag[x][y] = 0;

    }}

  do                    /*  do loop (jikan no loop) */
   {
      Eq_flag=0;
    for( x_dash =0 ; x_dash <= N-1 ; x_dash++ ){
    for( y_dash =0 ; y_dash <= N-1 ; y_dash++ ){
       dSx[x_dash][y_dash]=0.0;
       dSy[x_dash][y_dash]=0.0;
    }}

   /* Sx[x][y] to Sy[x][y] no hantei */

     hantei( Sx,Sy,Sx_thr,Sy_thr, dSx,dSy,Gx,Gy,N,d,
             &Count_x,&Count_y,&Eq_flag,Sx_flag,Sy_flag,t);



   /* Sx[x][y] to Sy[x][y] wo motomeru */
```

```
        for( x = 0 ; x <= N-1 ; x++ ){        /* loop_6*/
        for( y = 0 ; y <= N-1 ; y++ ){
               Sx[x][y] += dSx[x][y];
               Sy[x][y] += dSy[x][y];
 }}                          /*end of roop_6*/

  }while(  Eq_flag  == 1  );   /*end of do loop*/


   /* count S_flag */

        Count[t] =  Count_x + Count_y ;

          fprintf( f_count,"%d %d\n",t,Count[t] );
          fprintf( f_kibo ,"%d \n" , Count[t] );
  /*  printf( "%d\n",Count[t]);*/
   /* Sx_thr to Sy_thr no kousin */
        kousin(  Sx_flag,Sy_flag,N,Sx,Sy,Sx_thr,Sy_thr);

  }                           /*end of loop_1*/
   /* write a histogram */
               histogram(Count, t_max);
               fclose( f_count );
        fclose( f_kibo );

 }                                    /*end*/

void Sxy_0andSxy_thr(double Sx[][100],double Sy[][100],
double Sx_thr[][100],double Sy_thr[][100],int N ){

     int x,y,xx,yy;

     /*format Sx Sy */
       for ( x = 0 ; x <= N-1 ; x++ ){
```

```
        for ( y = 0 ; y <= N-1 ; y++ ){

          Sx[x][y] = 0.0;
          Sy[x][y] = 0.0;
}}
      /*Sxy_thr*/
        for ( x = 0 ; x <= N-1 ; x++ ){
        for ( y = 0 ; y <= N-1 ; y++ ){
           Sx_thr[x][y] = (double)rand() / RAND_MAX   ;
           Sy_thr[x][y] = (double)rand() / RAND_MAX  ;


       /*   printf( "%lf %lf \n",Sx_thr[x][y],Sy_thr[x][y] );*/

       }}
}

void Sy_plus_p(double Sy[][100],double p,int N){

     int x,y;

     for( x = 0 ; x <= N-1 ; x++ ){
     for( y = 0 ; y <= N-1 ; y++ ){

          Sy[x][y] += p  ;
       }}
 }

void ReadGxGy(double Gx[][100], double Gy[][100], int N) {

     FILE *f_gx,*f_gy;
     int x,y;

     f_gx =   fopen("gx.dat","r");
     f_gy =   fopen("gy.dat","r");
```

```
        for ( x = 0 ; x <= N-1 ; x++ ){
        for ( y = 0 ; y <= N-1 ; y++ ){

         fscanf(f_gx,"%lf\n",&Gx[x][y]);
         fscanf(f_gy,"%lf\n",&Gy[x][y]);
        }}

     fclose(f_gx);
     fclose(f_gy);


  }

double kousin(int Sx_flag[][100],int  Sy_flag[][100],int N,
             double Sx[][100],double Sy[][100],
             double Sx_thr[][100],double Sy_thr[][100]){

        int x,y;

        for ( x = 0 ; x <= N-1 ; x++ ){         /*loop_7*/
        for ( y = 0 ; y <= N-1 ; y++ ){
       if(    Sx_flag[x][y] == 1  ){

           Sx_thr[x][y] = (double)rand() / RAND_MAX  ;
           Sx[x][y] = 0;

       } }}                                /*end of loop_7*/

     /*  Sy_thr no kousin */

        for ( x = 0 ; x <= N-1 ; x++ ){   /*loop_7dash*/
        for ( y = 0 ; y <= N-1 ; y++ ){
       if(    Sy_flag[x][y] == 1   ){
```

```
            Sy_thr[x][y] = (double)rand() / RAND_MAX   ;
            Sy[x][y]= 0;

        } }}                        /*end of loop_7dash*/

}

#include<stdio.h>
#include<math.h>
#define PI 3.14159265358979
#include<stdlib.h>
#include<string.h>
  int main(int argc, char *argv[])

{
   int      N,x,y,k,l;
   double   Gx[100][100],ky,kx;
   FILE *fp;

  if ( argc != 2) {
     printf("Write a filename .\n");
     exit(1);
    }

   if ((fp = fopen(argv[1], "w")) == NULL) {
     printf("The file doesn't open.\n");
     exit(1);
   }


   printf(" Gx wo motomeru ");
   printf(" N ( <= 100 ") ;  scanf("%d",&N);

   for(x=0;x<N;x++){
   for(y=0;y<N;y++){
```

```
    Gx[x][y] = 0.0 ;

    for(k=0;k<N;k++){
      for(l=0;l<N;l++){

 if(k==0 && l==0){
           Gx[x][y]+=(2.0*(x+y)+1)/4.0;
 }
         else{
           kx=2.0*PI/N*k;
           ky=2.0*PI/N*l;

          Gx[x][y]+= (cos(kx*x+ky*y)-cos(kx*(x+1)+ky*y))/
                         (4.0-2.0*(cos(kx)+cos(ky))) ;
         }
       }}
       Gx[x][y]= Gx[x][y]/N/N;
       fprintf(fp, "%lf\n",Gx[x][y]);
   }}

#include <stdio.h>


double hantei( double Sx[][100],double Sy[][100],
             double Sx_thr[][100],double Sy_thr[][100],
             double dSx[][100],double dSy[][100],
             double Gx[][100],double Gy[][100] ,
             int N,int d,
             int *Count_x_p,int *Count_y_p,int *Eq_flag_p,
             int Sx_flag[][100],int Sy_flag[][100],
             int t ){


        int x,y,x_dash,y_dash;
```

```
        int a,a_dash,b,b_dash;
        int alpha,beta;
        double S0;

        for( x = 0 ; x <= N-1 ; x++ ){            /*loop3*/
        for( y = 0 ; y <= N-1 ; y++ ){

           if( Sx[x][y] > Sx_thr[x][y] ){        /* if */

 printf("An earthquake occured at Sx[%d][%d] at t=%d.\n", x,y,t);

                        S0        = Sx[x][y];
                        Sx_flag[x][y] = 1;
                        *Count_x_p += 1;
                        *Eq_flag_p = 1;
                        alpha   = x;
                        beta    = y;
                        Sx[x][y]= 0 ;

for(x_dash= 0 ; x_dash<= N - 1; x_dash++ ){     /* loop_4   */
  for(y_dash= 0 ; y_dash<= N - 1; y_dash++ ){     /*(mawarino */
                                                /* eikyou) */
 if( x_dash != alpha || y_dash != beta ){

                        a      = (x_dash-x+N)%N;
                        a_dash = (x_dash-(x+1)+N)%N;
                        b      = (y_dash-y+N)%N;

                      dSx[x_dash][y_dash] += (double)d/(d-1)*S0
                                      *(Gx[a][b]-Gx[a_dash][b]);

                      dSy[x_dash][y_dash] += (double)d/(d-1)*S0
                                      *(Gy[a][b]-Gy[a_dash][b]);
 }
```

```
            }}                    /*end of loop_4*/
              Sx[x][y]= 0 ;

           }                        /*end of if */

          /* Sy[x][y] no hantei*/

if( Sy[x][y] > Sy_thr[x][y] ){                    /* if */

    /*  printf("An earthquake occured at
                    Sy[%d][%d] at t=%d.\n",  x,y,t);*/

                  S0        = Sy[x][y];

                  Sy_flag[x][y] = 1 ;
                  *Count_y_p  += 1 ;
                  *Eq_flag_p = 1;
                  alpha   = x;
                  beta    = y;
                  Sy[x][y]=0;

 for(x_dash= 0 ; x_dash<= N - 1; x_dash++ ){    /*loop_5*/

             for(y_dash= 0 ; y_dash<= N - 1; y_dash++ ){

if( x_dash != alpha || y_dash != beta ){


                    a      = (x_dash-x+N)%N;
                    b      = (y_dash-y+N)%N;
                    b_dash = (y_dash-(y+1)+N)%N;

                  dSx[x_dash][y_dash] += (double)d/(d-1)*S0
                     *(Gx[a][b]-Gx[a][b_dash]);
```

```
                dSy[x_dash][y_dash] += (double)d/(d-1)*S0
                   *(Gy[a][b]-Gy[a][b_dash]);
  }

              }}                        /*end of loop_5*/
                  Sy[x][y]=0;

              }                        /*end of else */


    }}                           /*end of loop_3*/

#include <stdio.h>


void histogram(int kibo[] , int t_max) {

             FILE    *f_hindo;
             int     hindo[10000],h;
             int     time;

     f_hindo  =  fopen("hindo.dat","w");

     for(h=0; h<10000;h++){
           hindo[h] = 0;}

     for(h=0; h<10000 ; h++ ){
     for(t=0; time<t_max ; t++ ){
              if( kibo[time]==h ){
hindo[ kibo[time] ] ++ ;}}}

     for(h=0; h<10000; h++){
       if(hindo[h] != 0){
 fprintf( f_hindo ,"%d  %d\n",h,hindo[h] );}}
```

```
     fclose( f_hindo );

}

double  energy_n(  double Energy[], int t ){

             FILE    *f_Energy;
             int     hindo[10000];float h;
             int     t;

     f_Energy  =  fopen("energa_hindo.dat","w");

     for(h=0; h<10000;h++){
           hindo[h] = 0;}

     for(h=0; h<200 ; h+=0.5 ){
     for(t=0; t<t_max ; t++ ){
              if( Energy[t]==h ){
hindo[ (float)Energy[t]/0.5 ] ++ ;}}}

     for(h=0; h<10000; h++){
       if(hindo[h] != 0){
 fprintf( f_hindo ,"%f  %d\n",h,hindo[h] );}}


     fclose( f_hindo );
}
```

B　確率論的モデルのプログラムリスト

```
#include<stdio.h>
#include<stdlib.h>
#define _N 200
#define Time 10000
```

```
int  hantei_goishi ( int x,int y, int diamond ,
      int goishi[][_N+2],int *Goishi_flag,int size[],
      double P,int number,int t );

int  jouken(  int goishi[][_N+2],int a, int b,int *p_h_count );

int histgram ( int _Time , int size[] );
void write(  int goishi[][_N+2] );
                                          /* non_stone  0  */
int main()                                /* black      2  */
                                          /* white_new  -1 */
{                                         /* white_old  1  */
      FILE *f_size,*f_p;
      static int goishi[_N+2][_N+2];
      int x0,y0,x,y,x_0,y_0,k;
      int i,diamond,count;
      int number;
      int Goishi_flag, size[100000];
      int okisa;
      int _T,j,F,H[7] ;
      float  r_size=0;
      double P=0.585000000000;


         _T=(_N/2)-1;
      /**** P wo kimeru   ****/
     for( i=0;i<Time;i++ ){
        size[i]=0;
       }

         /*  printf(" P = "); scanf("%lf",&P);*/
 f_p = fopen("P_size.dat" ,"w");
 f_size = fopen( " size.dat ","w" );
```

```
 r_size=0;
      /****   format ****/
for( j=0;j<Time;j++ )
{

   for(x=0;x<=_N+1;x++){
   for(y=0;y<=_N+1;y++){
     goishi[x][y]=0;
   }}
      /****  1ko goishi wo oku ****/

   x_0 = (_N/2);
   y_0 = (_N/2);


   goishi[x_0][y_0] = 1 ;
   Goishi_flag = 1;
   size[j]=1;
       /****   goishi wo sagasu  ****/

for( okisa = 1; okisa<_T ; okisa++ ){
  count = 0;
  if( Goishi_flag == 0){ break ;}
  else{
    Goishi_flag = 0;
    do{

       for( diamond=1 ; diamond<= okisa ; diamond++ ){

        for( i=0 ; i<diamond ; i++ ){

  x = x_0+diamond-i;  y= y_0-i;
         hantei_goishi
         ( x, y ,diamond,goishi,&Goishi_flag,size,P,number,j );
         x = x_0-i;  y= y_0-diamond+i;
```

```
          hantei_goishi
          ( x, y ,diamond,goishi,&Goishi_flag,size,P,number,j );
          x = x_0-diamond+i;  y= y_0+i;
          hantei_goishi
          ( x, y ,diamond,goishi,&Goishi_flag,size,P,number,j );
          x = x_0+i;  y= y_0+diamond-i;
          hantei_goishi
          ( x, y ,diamond,goishi,&Goishi_flag,size,P,number,j );

          x = 0; y = 0;
        }

      }

      count++;

    }while( count < 2 );
  }

 }
      fprintf(f_size,"%d %d\n",j,size[j]);
}/*for j*/
      histgram( Time,size );
      fclose(  f_size );
      fclose(f_p);
}


int  hantei_goishi ( int x,int y, int diamond ,
    int goishi[][_N+2],int *Goishi_flag,int size[]
   ,double P, int number,int t ){


       int A_plus; int B_plus; int A_minus; int B_minus;
       double p_goishi;int h_count;int A1;int A2;int A3;
```

```
        int A4;int A;
        int B1;int B2;int B3;int B4;int B;
        double _l;
        if( goishi[x][y] == 0 ){

     A_plus = x+1; A_minus = x-1; A = x;
     B_plus = y+1; B_minus = y-1; B = y;

          h_count = 0;

         jouken( goishi,A_plus ,B ,&h_count );
         jouken( goishi,A,B_plus ,&h_count );
         jouken( goishi,A_minus ,B,&h_count );
         jouken( goishi,A,B_minus,&h_count );


if( h_count >0 ){
           p_goishi = ((double)rand() / RAND_MAX)  ;

        if( p_goishi > P ){

 goishi[x][y]= 2;}

 /*  printf("goishi[%d][%d]=2\n",x,y );} */

        else{
          goishi[x][y]=1;

 /*   printf( " goishi[%d][%d]=1\n ",x,y ); */
         ( *Goishi_flag ) ++;
          ( size[t] )++;}
         }
        }
}
```

```
int jouken(  int goishi[][_N+2],int a, int b,int *p_h_count ){

       if( goishi[a][b]==1 ){

         ( *p_h_count)++;

       }
}

/*void write(  int goishi[][_N+2] ){

     FILE  *f_goishi_white;
     FILE  *f_goishi_black;
      FILE  *f_goishi_zero;
     int x;int y;

         f_goishi_white =   fopen("goishi_white.dat","w");
         f_goishi_black =   fopen("goishi_black.dat","w");
         f_goishi_zero  =   fopen("goishi_zero.dat","w");

        for( x=1; x<=F ; x++ ){
         for( y=1; y<=F ; y++ ){

           if( goishi[x][y] == 1 ){

          fprintf( f_goishi_white ,"%d %d \n",x,y );}

           else if( goishi[x][y] == 2 ){

          fprintf( f_goishi_black ,"%d %d \n",x,y );}

   else{

          fprintf( f_goishi_zero ,"%d %d \n",x,y );}
```

```
  }}
            fclose(  f_goishi_white );
            fclose(  f_goishi_black );
            fclose(  f_goishi_zero  );

     }*/

int histgram ( int _Time , int size[] )
{

            FILE    *f_hindo;
            int     hindo[10000],h;
            int     t;

     f_hindo  =  fopen("goishi_hindo.dat","w");

     for(h=0; h<10000;h++){          /*  hindo no shokika */
           hindo[h] = 0;}

     for(h=0; h<10000 ; h++ ){      /*  hindo wo siraberu */
     for(t=0; t<_Time ; t++ ){
              if( size[t]==h ){
hindo[ size[t] ] ++ ;}}}

     for(h=0; h<10000; h++){      /*  hindo wo kakikomu  */
       if(hindo[h] != 0){
 fprintf( f_hindo ,"%d  %d\n",h,hindo[h] );}}


     fclose( f_hindo );
}
```