ローレンツモデルにおける
カオスの定量的解析

藤渕 智康
羽田野研究室

平成12年2月29日

概 要

自然界には予測が不可能なカオス現象が多く存在している。それに対応して理論的なカオスモデルも幾つか提案されている。しかし、特に多次元のカオスモデルは、その解析が必ずしも容易ではない。本研究では3次元ローレンツモデルを取り上げる。そこから導かれるローレンツマップという1次元写像の特性を活かしてカオスの定量的解析の簡略化を試みた。その結果、3次元のローレンツモデル自体の性質と1次元ローレンツマップの性質が対応していることを確認した。

目 次

1　はじめに

身近な自然の中には予測不可能な系の振る舞いをするカオスの現象が数多い。天気予報で問題となる大気の対流もその 1 つである。カオスへの関心は 1963 年の E.N.Lorentz の対流モデルから提案されたローレンツンモデルの定量的解析が発祥となり、これを境に多くのカオスの研究がなされてきている。

今回研究を進めるにあたって、カオスの研究が発展する要因となった 3 次元ローレンツモデルを取り上げる。連続的な時間発展によるカオスは 3 次元以上でないと起こらない。その点で 3 次元ローレンツモデルは最も簡単なカオスのモデルである。しかし、その定量的解析は後述するように必ずしも容易ではない。それに対して離散的な時間発展によるカオスは 1 次元でも起こる。ロジスティック写像がその代表的な例である。

実は、ローレンツマップという 1 次元の離散的な時間のモデルを 3 次元ローレンツモデルから導くことができる。さらに、1 次元ローレンツマップは 3 次元ローレンツモデルよりも定量的解析が容易であると推測される。そこで 1 次元ローレンツマップと 3 次元ローレンツモデルのカオスの性質を比較した。両者に対応があれば、ローレンツモデルを 1 次元のローレンツマップを使って容易に解析できることになる。

実際に両者の対応があることが本研究で確認された。これによってローレンツモデルのカオスの解析が簡略化できると期待される。

2　カオスの振る舞い

カオスの大きな特徴として以下の 2 点が挙げられる。

（1） 方程式から得られる解が非周期的。

（2） わずかに異なる初期値からの解の違いが指数関数的に増大。

解が非線形微分方程式で与えられる場合、初期値を与えれば解は一意的に決定される。しかし、初期値のわずかな違いが鋭敏に作用して方程式の解の違いが時間と共に指数関数的に拡大される場合がある。このような力学系の非周期解が示す複雑な運動をカオスと呼んでいる。

線形の系における 2 つの初期値のわずかな違いによる差異は時間変化と共に線形に広がっていくため予測が可能である。それに対してローレンツモデルなどの非線形の系では、パラメータによっては 2 つのわずかに異なる初期値を与えると差異は時間変化と共に短時間のうちに指数関

数的に増大し、予測が不可能になる。

カオスの例として、今回研究の解析で用いたロジスティック写像とローレンツモデルについて見ていく。

2.1 ロジスティック写像

ロジスティック写像は $0 \leq x \leq 1$ における 1 次元の差分方程式、

$$x_{n+1} = rx_n(1 - x_n) \tag{1}$$

で与えられる写像である [1]。

図 1 は横軸に x_n、縦軸に x_{n+1} をプロットし、放物線 $x_{n+1} = rx_n(1-x_n)$ と対角線 $x_{n+1} = x_n$ の反復として写像の動きを見ている。式 (1) のパラメータ r は範囲 $0 \leq r \leq 4$ で定義されている。$0 \leq r \leq 1 + \sqrt{6}$ では x はある値に収束するか 2 周期解となる。$1 + \sqrt{6} < r \leq 4$ においては x が領域内に広く分布してカオスの振る舞いをする（図 2)。

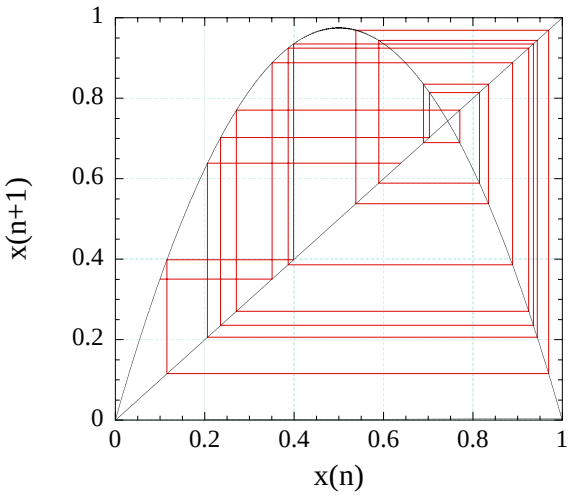

図 1: ロジスティック写像（式 (1) において $r = 3.9, x_0 = 0.1$）。

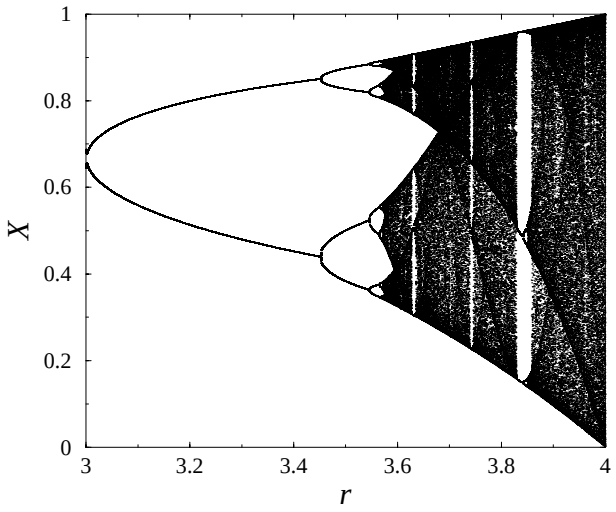

図 2: ロジスティック写像の分岐図。縦軸 X は各 r に対する x_n の平衡値。

2.2 ローレンツモデル

ローレンツモデルは気象学者 E.N.Lorentz が対流の研究をする際に、カオスの振る舞いを数値的に研究するために提案したモデルである。時間に依存する変数 $X(t), Y(t), Z(t)$ に対する 3 次元の非線形常微分方程式は次式のように表される [1]。

$$\begin{cases} \frac{d}{dt}X = -10(X - Y) \\ \frac{d}{dt}Y = -XZ + rX - Y \\ \frac{d}{dt}Z = XY - \frac{8}{3}Z \end{cases} \tag{2}$$

$X(t), Y(t), Z(t)$ は対流の研究で用いられていた変数に比例しており、

X：対流運動の強度に比例する変数

Y：上昇する流れと下降する流れの温度差に比例する変数

Z：垂直方向の温度の歪みに比例する変数

となっている [2]。ただし、本研究では $X(t), Y(t), Z(t)$ は位相空間の変数として扱う。

式 (2) のローレンツモデルでは、パラメータ r の値によって解の振る舞いが変化する。$r < 24.74$ では解は時間と共にある値に収束する（図 3）。$24.74 < r < 145$ では $X(t), Y(t), Z(t)$ の解が非周期的となり（図 4）、カオスの第 1 の性質を示す。

次に初期値がわずかに異なる 2 つのローレンツモデルの解の時間変化を考える。非カオス状態においては 2 つの解が大きく離れることなく共にある点に収束していく（図 5）。これに対して、カオス状態においては２つの解が大きく離れていき全く異なる振る舞いをする（図 6）。2 つの解の間の位置空間における距離 L の時間変化を計算すると、非カオス状態では指数関数的に値が小さくなり（図 7）、カオス状態では指数関数的に増大していることが確認できる（図 8）。このようにカオスの第 2 の性質が示される。

なお、本研究の数値計算においては、時間の刻み幅 dt を $0.001 \sim 0.0001$ として 4 次のルンゲ・クッタ法を用いている [3]。

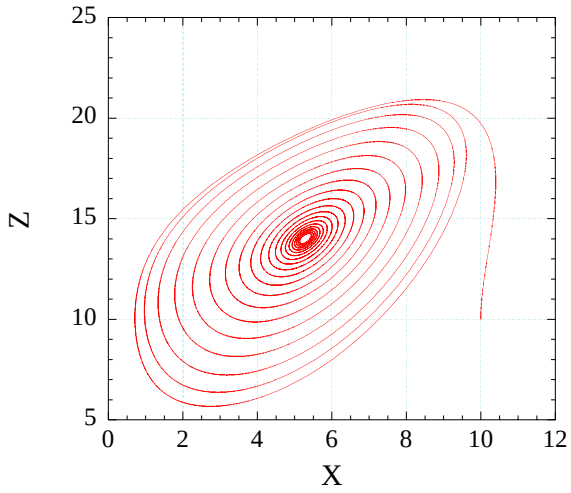

図 3: $r = 15$ の時のローレンツモデルの解。初期値 $X_0 = Y_0 = Z_0 = 10.0$ から始めると、X $= 5.30$,Y $= 5.30$,Z $= 14.00$ 付近に収束する

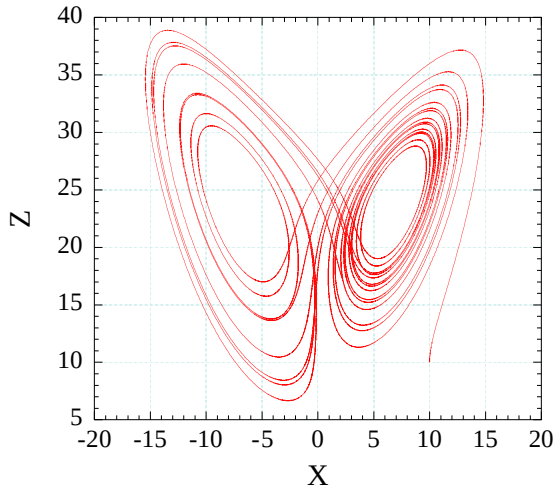

図 4: $r = 25$ の時のローレンツモデルの解。初期値 $X_0 = Y_0 = Z_0 = 10.0$ から始めると二度と同じ場所に戻らず、非周期解である。

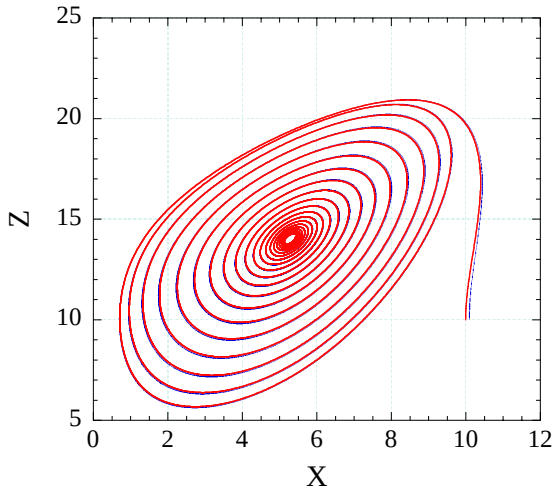

図 5: $r = 15$ の時のローレンツモデルの解。赤線が初期値 $X_0 = Y_0 = Z_0 = 10.0$, 青線が $X_0 = Y_0 = Z_0 = 10.1$ から始めたもの。いずれの解も同じ点に収束している。

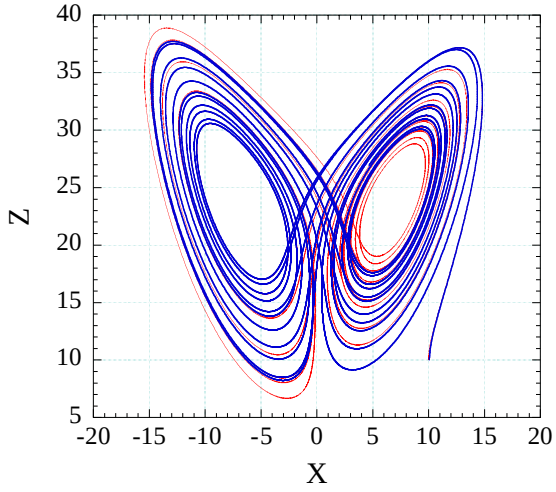

図 6: $r = 25$ の時のローレンツモデル。赤線が初期値 $X_0 = Y_0 = Z_0 = 10.0$, 青線が $X_0 = Y_0 = Z_0 = 10.1$ から始めたもの。2 つの解は全く異なる軌道を描く。

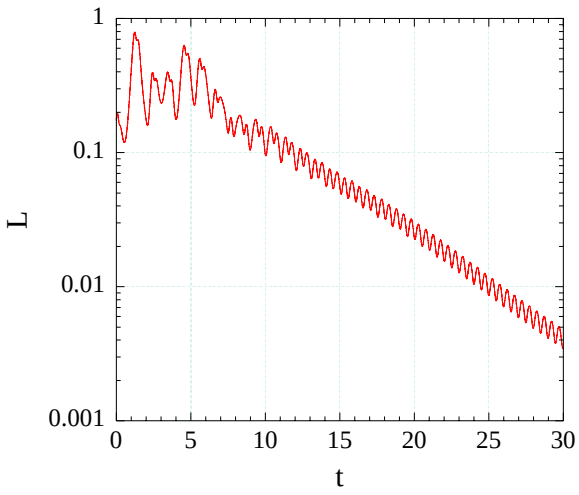

図 7: 図 5 の 2 つの解（$r = 15$ の場合）の間の距離の時間変化。L は指数関数的に減少する。

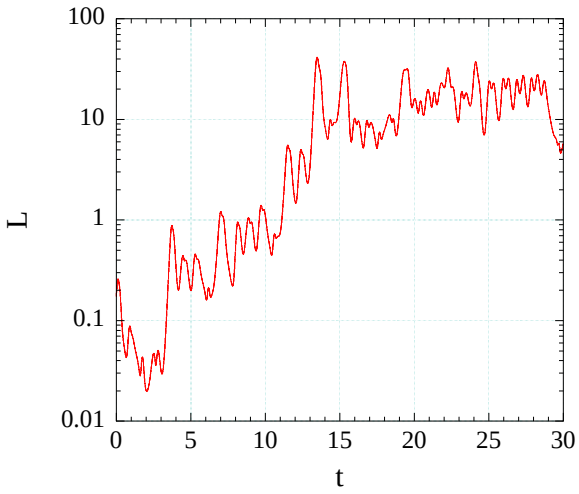

図 8: 図 6 の 2 つの解（$r = 25$ の場合）の間の距離の時間変化。L は指数関数的に増大する。

3　リアプノフ指数

カオスの振る舞いを定量的に表わす量として軌道の拡大率を表わすリアプノフ指数 (Lyapunov exponent) が用いられる。これは図 9 のように各時刻における 2 つの軌道の間の広がりの率として定義される。指数関数 $\exp(\lambda t)$ で近似した時に傾き λ で表わされる量である（3.2 節参照)。

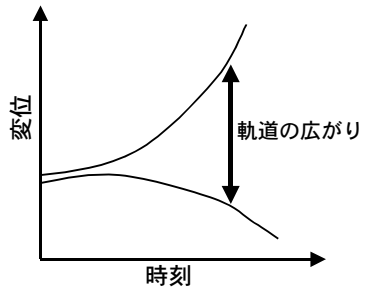

図 9: 2 つの軌道の広がり

重要となるのがこのリアプノフ指数 λ の正負の符号である。$\lambda > 0$ ならばカオス状態、$\lambda < 0$ ならば非カオス状態と判断出来る。つまりこれによってカオス状態と非カオス状態を特徴付けることができる。

3.1　ロジスティック写像におけるリアプノフ指数

離散的時間発展の写像におけるリアプノフ指数は比較的簡単に計算することが出来る。わずかに異なる 2 つの初期条件 x,$x + \varepsilon$ から写像が始まったとした時、n 回の写像後の軌道間距離 $\varepsilon(n)$ が、

$$\varepsilon(n) \approx \varepsilon \exp(\lambda n) \tag{3}$$

と近似できると仮定する。この時の λ がリアプノフ指数である。

$x_{n+1} = f(x_n)$ で与えられる 1 次元写像の λ は以下のようにして正確に計算することができる。2 つの状態間の n 回目の写像後の差 $f_n(x+\varepsilon) - f_n(x)$ が、

$$f_n(x + \varepsilon) - f_n(x) \approx \varepsilon \exp(n\lambda) \tag{4}$$

と表わせるとする。両辺の対数をとり整理すると、

$$\log\left\{\frac{f_n(x+\varepsilon)-f_n(x)}{\varepsilon}\right\}\approx n\lambda \tag{5}$$

となる。この式の左辺において、ε を小さく取ると微係数の定義となる。よって λ について解けば、

$$\lambda\approx\frac{1}{n}\log\left|\frac{df_n}{dx}\right| \tag{6}$$

となる。写像の回数 n を多く取り平均することでリアプノフ指数、

$$\lambda=\lim_{n\to\infty}\frac{1}{n}\sum_{i=0}^{n-1}\log\left|\frac{df_i}{dx}\right| \tag{7}$$

を得る [4]。

実際のロジスティック写像では f_n に (1) 式を用いて $n=50000$ といった有限の値をとって (7) 式を計算する。その結果、図 10 を得た。図 2 と図 10 は良く対応しており、$1+\sqrt{6}=3.449...$ からリアプノフ指数が正であることが確認できる。

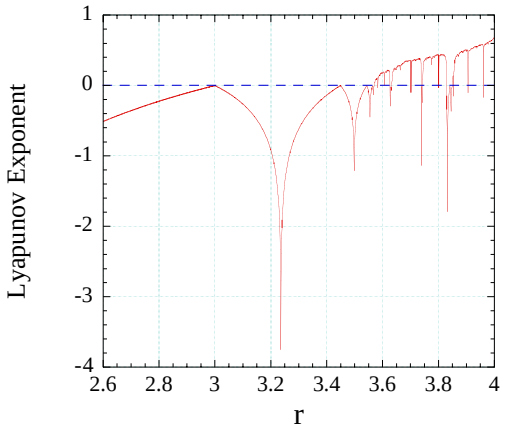

図 10: ロジスティック写像におけるリアプノフ指数。式 (7) に基づいて計算。

3.2 ローレンツモデルにおけるリアプノフ指数

2 章 2.2 節で説明した、わずかに初期値の異なる 2 つのローレンツモデルの解を考える。図 11 のように各時刻における 2 点間の距離が指数関数的に振る舞う部分を指数関数 $\exp(\lambda t)$ で近似することにより、リアプノフ指数 λ を計算する。

指数関数近似式を、

$$L(t) \propto \exp(\lambda t) \tag{8}$$

とすると、ある時間 t_1 と t_2 における $L(t)$ は、

$$L(t_1) \propto \exp(\lambda t_1) \tag{9}$$

$$L(t_2) \propto \exp(\lambda t_2) \tag{10}$$

である。この 2 式より、

$$\frac{L(t_2)}{L(t_1)} = \exp\{\lambda(t_2 - t_1)\} \tag{11}$$

となる。両辺の対数を取り λ について解くと、

$$\lambda = \frac{1}{t_2 - t_1} \log_e \left\{ \frac{L(t_2)}{L(t_1)} \right\} \tag{12}$$

としてリアプノフ指数 λ を得る [1]。

実際の計算では精度を上げるため多くの初期条件を取り相加平均してリアプノフ指数を計算した。図 12 がその結果である。今回の計算では $r = 20.7$ からカオスの振る舞いを観測出来た。

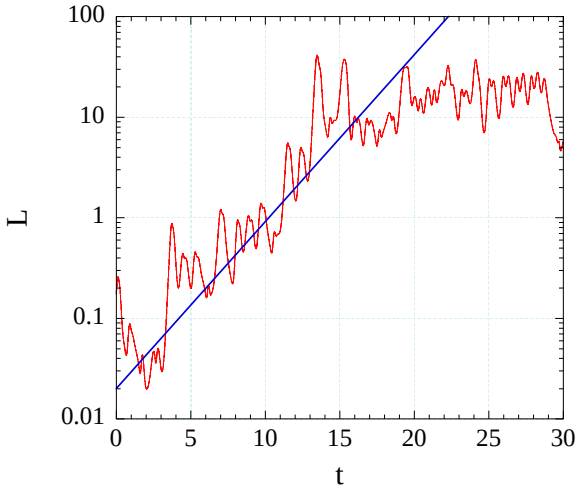

図 11: 2 つの解の間の距離の指数関数による近似。

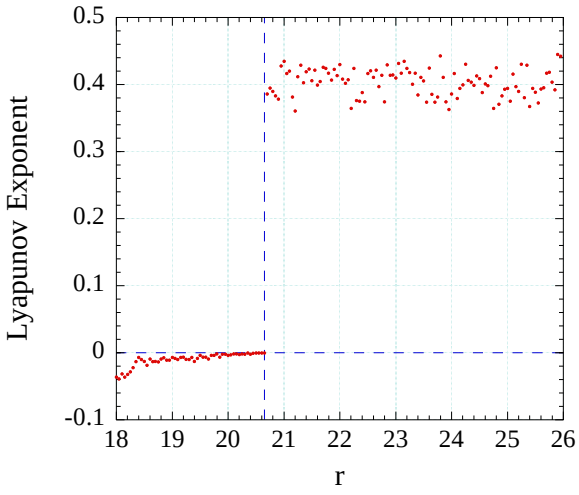

図 12: ローレンツモデルにおけるリアプノフ指数の r による変化。

4　ローレンツマップ

ローレンツマップとは気象学者 E.N.Lorentz がローレンツモデルの研究をしている際に見つけた離散時間の 1 次元写像である。ローレンツモデルの Z の極大値に着目する（図 13）。

横軸に i 番目の Z の極大値 M_i、縦軸に $i+1$ 番目の Z の極大値 M_{i+1} をプロットすると、すべての点が図 14 のような曲線上に存在する。これをローレンツマップと呼ぶ [2]。この曲線を図 1 のロジスティック写像に相当するものとみなせば、図 14 もカオスのモデルと考えることができる。

次節で 1 次元のローレンツマップの解析について述べるが、その前にまず 3 次元ローレンツモデルのリアプノフ指数の計算における問題点を挙げる。

- 各時刻の 2 点間距離を取る必要がある。
- 指数関数的な振る舞いをする時間範囲が初期値 $X(0), Y(0), Z(0)$ によって異なる。
- 指数関数で近似する時間幅の取り方に手間がかかる。

以上の問題点のため、3 次元ローレンツモデルのリアプノフ指数の算出は様々な段階を踏んでいかなければならない。

一方、ローレンツマップの場合はリアプノフ指数を求める際に次のような利点がある。

- 精度の良い計算方法の式 (7) が与えられる。
- 時間が離散的である。
- 扱う変数が 1 つしかない。
- r を固定すればローレンツマップの曲線の形はローレンツモデルの初期値に依らない。

以上の利点から、ローレンツマップの方がより簡単に解析できる。本研究では図 14 のローレンツマップを使ってカオスの定量的解析を行い、ローレンツモデルにおけるカオスの定量的な解析と比較した。

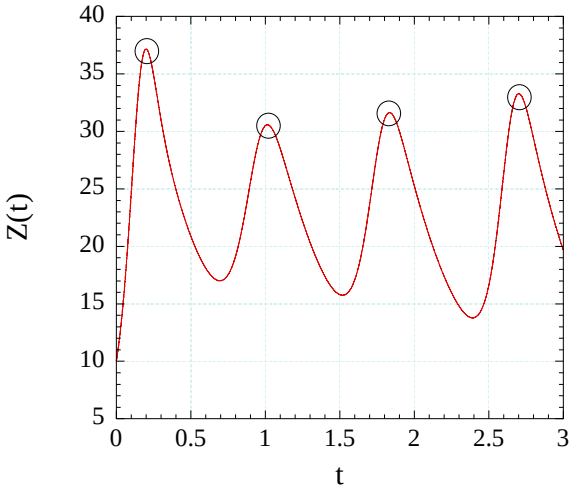

図 13: カオス状態 ($r = 25$) の時のローレンツモデルの Z の時間変化。丸印の極大値に注目して図 14 を描く。

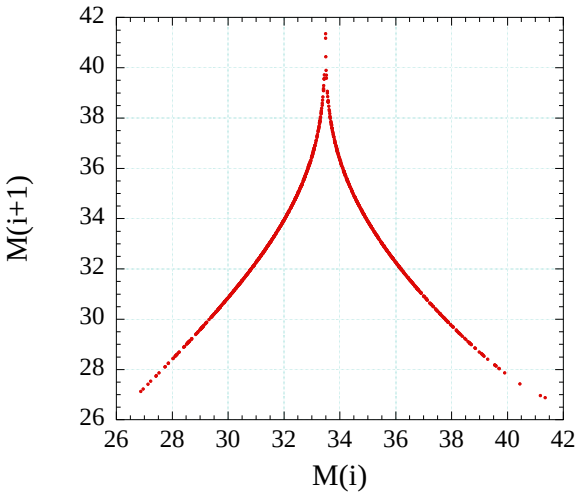

図 14: $r = 25$ のローレンツモデルから導かれたローレンツマップ

5　ローレンツマップにおける解析

本研究ではローレンツマップの写像がロジスティック写像に似ていることを利用し、3 章 3.1 節の解析方法を用いてリアプノフ指数を計算した。

第 3.1 節の解析方法の式 (7) を用いるためには、図 14 の曲線を表わす近似式、

$$M_{i+1} = f(M_i) \tag{13}$$

を見つける必要がある。この節では関数 f の定め方を述べる。

まず近似式としてカスプ形、

$$f(M_i) = \rho \exp\left(-a\,|M_i - b|^{\alpha}\right) \tag{14}$$

を仮定した（図 15)。式 (14) において、ρ はローレンツマップにおける縦軸 M_{i+1} の最大値と定め、b は最大値 ρ を取る時の横軸 M_i の値と定める。定数 a と α は次のように定める。

近似式として式 (14) が成り立つと仮定した時、左辺を M_{i+1} と置き換え、

$$M_{i+1} = \rho \exp\left(-a\,|M_i - b|^{\alpha}\right) \tag{15}$$

とおく。この式を変形すると、

$$\log\left(\log\rho - \log M_{i+1}\right) = \alpha \log|M_i - b| + \log a \tag{16}$$

となる。そこで、データ、(M_i, M_{i+1}) を、

$$P = \log|M_i - b| \tag{17}$$

$$Q = \log\left(\log\rho - \log M_{i+1}\right) \tag{18}$$

と変換すると、

$$Q = \alpha P + \log a \tag{19}$$

という線形の近似式が得られる。実際にローレンツマップのデータを基に横軸に P、縦軸に Q を取りプロットすると式 (19) で近似できるグラフが得られた（図 16)。そこで式 (14) の仮定が成り立つとして、図 16 における切片と傾きから定数 a、α を定める。

つまり、式 (14) で用いられている定数は、r を決めてローレンツモデルの解の Z の極大値を算出すれば一意的に決定できる値となっている。実際に決定した定数の値を図 17 に示す。

このようにして得た式 (14) と式 (7) を利用し、パラメータ r を変化させながらローレンツマップによるリアプノフ指数 λ を算出した。ここで強調すべき点は、近似式 (14) を定めたために式 (7) によって精度良くリアプノフ指数を計算できるということである。

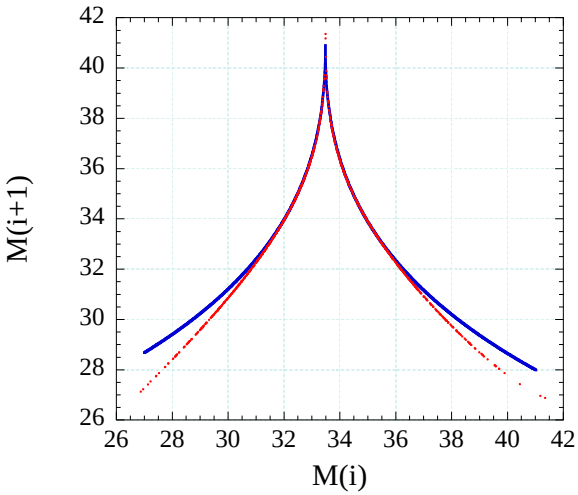

図 15: ローレンツマップのカスプ波形による近似。

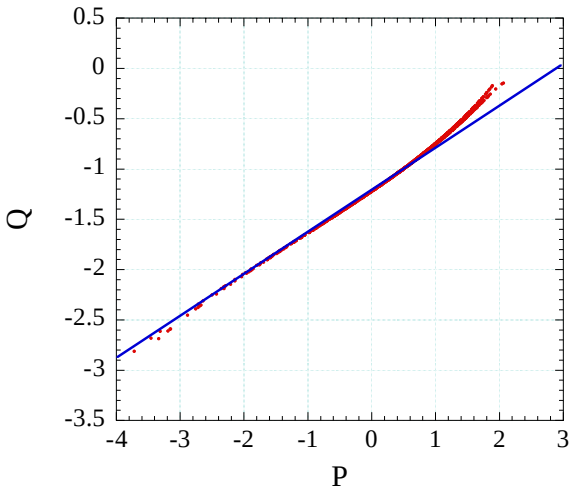

図 16: 赤の散布図が式 (17),(18) に従ってローレンツマップのデータをプロットしたもの。青線がそれを直線で式 (19) の形に近似した線。

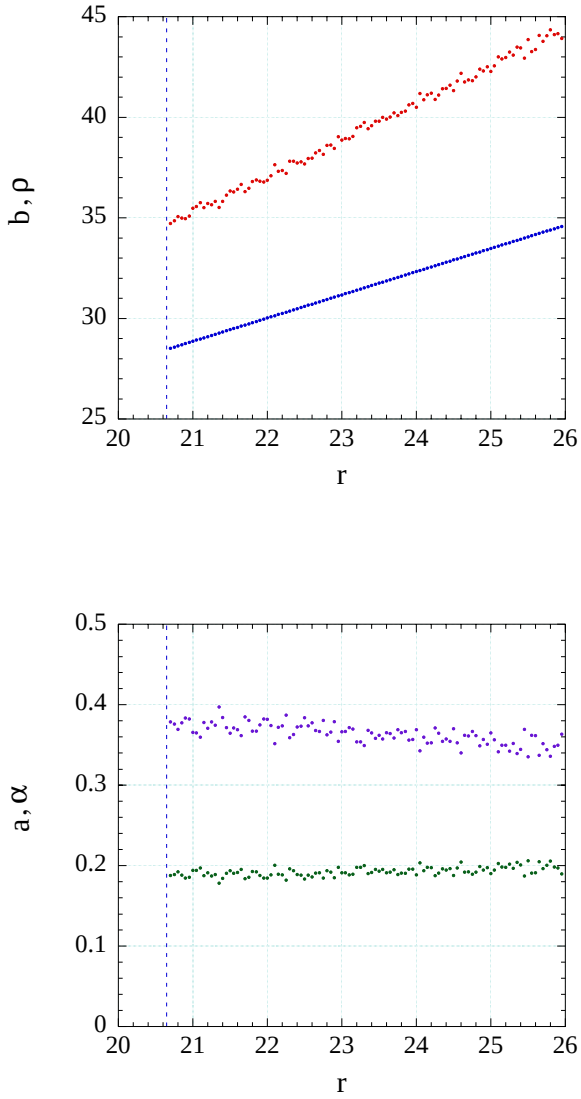

図 17: 各 r に対する近似式 (14) の定数。上図の赤の散布図が ρ、青の散布図が b。下図の緑の散布図が a、紫の散布図が α。

6　解析結果

1 次元のローレンツマップによるリアプノフ指数 λ を式 (7) を使って算出し、3 次元ローレンツモデルのリアプノフ指数と比較した（図 18）。1 次元のローレンツマップのリアプノフ指数も 3 次元ローレンツモデルの時と同様に $r = 20.7$ 付近から正の値になることが確認できる。また、3 次元のローレンツモデルよりも、1 次元のローレンツマップの方が精度良くリアプノフ指数を計算できていることが分かる。なお、$r < 20.7$ でリアプノフ指数の値がないのは、ローレンツモデルの極大値が収束してしまうためである。

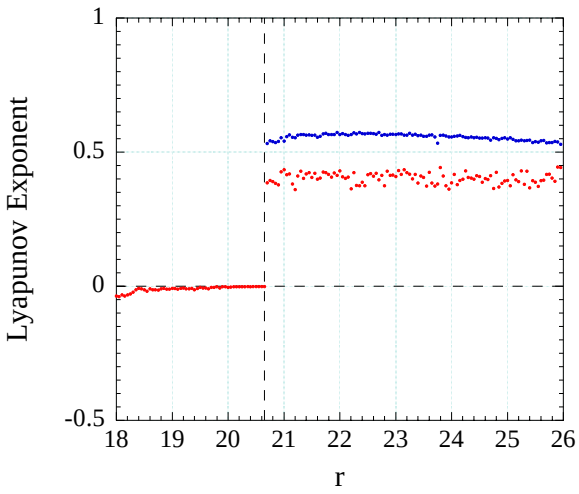

図 18: ローレンツマップから求めたリアプノフ指数（青の散布図）とローレンツモデルから得たリアプノフ指数（赤の散布図）。

7　考察およびまとめ

ローレンツマップの近似としてカスプ形の式 (14) を導入した。これによって、ローレンツモデルの定量的解析が簡略化できた。今後、もし定数 ρ,b,a,α と r の関係が正確に分かれば、ローレンツモデルの計算をすることなくローレンツマップの解析ができることになろう。

最後に、本研究における問題点を取り上げておく。ローレンツモデルにおいてパラメータ r が約 30 よりも大きくなるとローレンツマップの左右の対称性が崩れてくる。この時はローレンツマップをカスプ形で近似するのは難しい。そのため今回の解析はパラメータを $18 \leq r \leq 26$ に限定した。

また、本研究では 4 次のルンゲ・クッタ法を用いて時系列の計算をし、時間の上限を 5×10^4 秒と定めた。この時間の範囲内で解 X, Y, Z が収束せず非周期解となる場合をカオスと判断した。理論上収束するはずの $r = 21.0$ においては特に時間 $t \leq 3 \times 10^6$ まで時系列の計算を行ったが、解 X, Y, Z が収束しなかった。今回の解析で $r = 20.7$ 付近からカオスとなっているのは $t \leq 5 \times 10^4$ という範囲に依るものと考えられる。改善策としては、

- 4 次以上の差分法を使う。
- 時間の刻み幅 dt をさらに細かくする。
- 計算時間を更に長く取ることで計算精度を上げる。

ということが考えられる。

8　謝辞

今回研究を進めてくるにあたり、研究室の方々から多くのアドバイスを頂き心から感謝致します。特に指導をして下さった羽田野直道先生、昨年ローレンツモデルの研究をされていた及川さんには大変お世話になり、この場を借りて深くお礼申し上げます。

参考文献

[1] 高安 秀樹 , フラクタル , 朝倉書店 (1986)

[2] E.N.Lorentz , J.Atom.Sci.40,130. (1963)

[3] 薮下 信 , 計算物理 (I) , 地人書館（1982）

[4] G.L.Baker , J.P.Gollub 著 松下 貢 訳 , カオス力学入門 , 啓学出版 (1992)

A　ローレンツモデル

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

int main()
{
  FILE *datafile;
  double x,y,z,t=0,dt,t_min,t_max,r,count=0;
  double ka,kb,kc,kd,kax,kbx,kcx;
  double la,lb,lc,ld,lay,lby,lcy;
  double ma,mb,mc,md,maz,mbz,mcz;

  printf("r    =");scanf("%lf",&r);
  printf("x    =");scanf("%lf",&x);
  printf("y    =");scanf("%lf",&y);
  printf("z    =");scanf("%lf",&z);
  printf("t_min=");scanf("%lf",&t_min);
  printf("t_max=");scanf("%lf",&t_max);
  printf("dt   =");scanf("%lf",&dt);

  datafile=fopen("lorentz-rk3.dat","w");

  for(t=0;t<=t_max;t=t+dt){
    if(t>=t_min){  /*** start if 1 ***/
      fprintf(datafile,"%f %f %f %f\n",t,x,y,z);
       if(count<=t){  /*** start if 2 ***/
         printf("t=%10f x=%10f y=%10f z=%10f\n",t,x,y,z);
         count=count+10;
       }  /*** end if 2 ***/
    }  /*** end if 1 ***/

    ka=dt*(-10*(x-y));   kax=x+(ka/2);
    la=dt*(-x*z+r*x-y);  lay=y+(la/2);
    ma=dt*(x*y-(8/3)*z); maz=z+(ma/2);
```

```
    kb=dt*((-10)*(kax-lay));       kbx=x+(kb/2);
    lb=dt*((-kax)*(maz)+(r*kax)-lay); lby=y+(lb/2);
    mb=dt*((kax*lay)-((8/3)*maz));  mbz=z+(mb/2);

    kc=dt*((-10)*((kbx-lby)));       kcx=x+kc;
    lc=dt*((-kbx)*mbz+(r*kbx)-lby); lcy=y+lc;
    mc=dt*(kbx*lby-((8/3)*mbz));  mcz=z+mc;

    kd=dt*((-10)*(kcx-lcy));
    ld=dt*((-kcx)*mcz+(r*kcx)-lcy);
    md=dt*(kcx*lcy-((8/3)*mcz));

    x=x+((ka+(2*kb)+(2*kc)+kd)/6);
    y=y+((la+(2*lb)+(2*lc)+ld)/6);
    z=z+((ma+(2*mb)+(2*mc)+md)/6);
  }                                /*** end for loop ***/
  fclose(datafile);
}
```

B　ローレンツモデルによるリアプノフ指数

```
/*** Lyapunov Exponent with Lorentz Model ***/

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

int main()
{
  FILE *datafile;
  FILE *Loutputfile;
  int L_output;
  long int i,i_Lp,i_Lq,i_Lpp=0,i_Lqq=0;
  long double L;
  double L_start,L_sab,r,rr,rrr,dr;
  double t=0,dt,t_min=0.000000,t_max,tp=0,tq,ttq;
  double n,i_max,count_break=0;
  double Lp,Lq,Lpp,Lqq,rogu,rogu_sum=0,count=0,ramuda=0,tau;
  double x,y,z,xv,yv,zv;
  double xo,yo,zo,xvo,yvo,zvo;
  double ka,kb,kc,kd,kax,kbx,kcx;
  double la,lb,lc,ld,lay,lby,lcy;
  double ma,mb,mc,md,maz,mbz,mcz;
  double kav,kbv,kcv,kdv,kaxv,kbxv,kcxv;
  double lav,lbv,lcv,ldv,layv,lbyv,lcyv;
  double mav,mbv,mcv,mdv,mazv,mbzv,mczv;

  printf("r_start=");scanf("%lf",&rr);
  printf("r_end  =");scanf("%lf",&rrr);
  printf("dr     =");scanf("%lf",&dr);
  printf("dt     =");scanf("%lf",&dt);
  printf("x      =");scanf("%lf",&xo);
  printf("y      =");scanf("%lf",&yo);
  printf("z      =");scanf("%lf",&zo);
  printf("xv     =");scanf("%lf",&xvo);
  printf("yv     =");scanf("%lf",&yvo);
```

```
  printf("zv     =");scanf("%lf",&zvo);
  printf("t_max  =");scanf("%lf",&t_max);
  printf("tp     =");scanf("%lf",&tp);
  printf("tq     =");scanf("%lf",&tq);

  printf("If L_output=1, output L(t)\n");
  printf("L_output =");scanf("%d",&L_output);

  i_max=t_max/dt;
  printf("i_max=%f\n",i_max);

  tp=0.00000;
  tq=20.000;
  i_Lp=0;
  ttq=tq+5.00000;

  datafile=fopen("lyapounov3.dat","w");

  if(L_output==1)
    {
      Loutputfile=fopen("distance-rk1.dat","w");
    }

  for(r=rr;r<=rrr;r=r+dr) /** start for(r...) **/
    {
      printf("r=%f\n",r);
      rogu_sum=0;
      count=0;

      for(n=xo;n<=(xo+0.010);n=n+0.00100) /** start for(X(0)...) **/
{
 x=n;  xv=n+0.000100;
 y=yo;   yv=yvo;
 z=zo;   zv=zvo;
```

```
        L_start=sqrt((x-xv)*(x-xv)+(y-yv)*(y-yv)+(z-zv)*(z-zv));
Lp=L_start;
L_sab=L_start*0.1000;

for(i=0;i<=i_max;i=i+1)  /** start for(t_min...) **/
  {
    t=i*dt;
    L=sqrt((x-xv)*(x-xv)+(y-yv)*(y-yv)+(z-zv)*(z-zv));

    if(L_output==1 && i%1000==0)
      {
fprintf(Loutputfile,"%f %f\n",t,L);
      }

    if(t<ttq)
      {
if(t>tq-0.00001 && t<tq+0.00001)
  {
    Lq=L;
    i_Lq=i;
    count_break=0;
    if(r>20.8500)
      {
i_max=(tq+1.0000)/dt;
      }
  }
      }
    if(t>ttq)
      {
if(L<L_sab)
  {
    Lqq=L;
    i_Lqq=i;
    rogu_sum=0;
    count_break=1;
```

```
  break;
}
    }

  /** Lorentz 1 start **/
  ka=dt*(-10*(x-y));   kax=x+(ka/2);
  la=dt*(-x*z+r*x-y);  lay=y+(la/2);
  ma=dt*(x*y-(8/3)*z); maz=z+(ma/2);

  kb=dt*((-10)*(kax-lay));      kbx=x+(kb/2);
  lb=dt*((-kax)*(maz)+(r*kax)-lay); lby=y+(lb/2);
  mb=dt*((kax*lay)-((8/3)*maz));  mbz=z+(mb/2);

  kc=dt*((-10)*((kbx-lby)));      kcx=x+kc;
  lc=dt*((-kbx)*mbz+(r*kbx)-lby); lcy=y+lc;
  mc=dt*(kbx*lby-((8/3)*mbz));  mcz=z+mc;

  kd=dt*((-10)*(kcx-lcy));
  ld=dt*((-kcx)*mcz+(r*kcx)-lcy);
  md=dt*(kcx*lcy-((8/3)*mcz));

  x=x+((ka+(2*kb)+(2*kc)+kd)/6);
  y=y+((la+(2*lb)+(2*lc)+ld)/6);
  z=z+((ma+(2*mb)+(2*mc)+md)/6);
  /** Lorentz 1 end  **/

  /** Lorentz 2 start **/
  kav=dt*(-10*(xv-yv));   kaxv=xv+(kav/2);
  lav=dt*(-xv*zv+r*xv-yv);  layv=yv+(lav/2);
  mav=dt*(xv*yv-(8/3)*zv); mazv=zv+(mav/2);

  kbv=dt*((-10)*(kaxv-layv));      kbxv=xv+(kbv/2);
  lbv=dt*((-kaxv)*(mazv)+(r*kaxv)-layv); lbyv=yv+(lbv/2);
  mbv=dt*((kaxv*layv)-((8/3)*mazv));  mbzv=zv+(mbv/2);
```

```
    kcv=dt*((-10)*((kbxv-lbyv)));      kcxv=xv+kcv;
    lcv=dt*((-kbxv)*mbzv+(r*kbxv)-lbyv); lcyv=yv+lcv;
    mcv=dt*(kbxv*lbyv-((8/3)*mbzv));  mczv=zv+mcv;

    kdv=dt*((-10)*(kcxv-lcyv));
    ldv=dt*((-kcxv)*mczv+(r*kcxv)-lcyv);
    mdv=dt*(kcxv*lcyv-((8/3)*mczv));

    xv=xv+((kav+(2*kbv)+(2*kcv)+kdv)/6);
    yv=yv+((lav+(2*lbv)+(2*lcv)+ldv)/6);
    zv=zv+((mav+(2*mbv)+(2*mcv)+mdv)/6);
    /** Lorentz 2 end **/

  }    /** start for t_min **/

if(count_break>0.5)
  {
    Lq=Lqq;
    i_Lq=i_Lqq;
  }

printf("X(0)=%6.3f i=%d Lp=%e  i=%d Lq=%e\n",n,i_Lp,Lp,i_Lq,Lq);

if(count_break<0.5)
  {
    rogu=(log(Lq)-log(Lp))/(dt*(i_Lq-i_Lp));
    rogu_sum=rogu_sum+rogu;
    count=count+1;
  }

if(count_break>0.5)
  {
    rogu=(log(Lq)-log(Lp))/(dt*(i_Lq-i_Lp));
    rogu_sum=rogu;
    count=1.0000;
```

```
      break;
    }

 if(L_output==1)
    {
      fprintf(Loutputfile,"&\n");
    }

}    /** end for{X(0),Y(0),Z(0)} **/

       printf("count=%3.0f\n",count);
       ramuda=(rogu_sum/count);

       fprintf(datafile,"%6.3f %10.6f\n",r,ramuda);
       printf("r= %6.3f Lyapounov = %10.6f\n",r,ramuda);
       printf("\n");


    }/** end for(r....) **/


  fclose(datafile);

  if(L_output==1)
    {
      fclose(Loutputfile);
    }

}
```

C　ローレンツマップによるリアプノフ指数

```
/*** Lyapunov Exponent with Lorentz Map ***/

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

#define kosu 1000000

int main()
{
  FILE *writefile;
  FILE *readfile;
  FILE *datafile;
  long int i,kazu;
  double x,y,z,x_start,y_start,z_start;
  double r,r_start,r_end,dr, t,dt,t_min,t_max,s;
  double ka,kb,kc,kd,kax,kbx,kcx;
  double la,lb,lc,ld,lay,lby,lcy;
  double ma,mb,mc,md,maz,mbz,mcz;
  double z_now,z_old1,z_old2,z_old3,z_old4,z_old5;
  double z_old6,z_old7,z_old8,z_old9,z_old10,sa_z;
  double M_max,M_min,MMM,MN,MO,MP,MQ,count_M=0;
  static double P[kosu],Q[kosu],PP[kosu],QQ[kosu];
  double Q_max,Q_zero,QQQ,n,a,b,c,beki,w1,w2;
  double Pua,Pub,Qua,Qub,count_ua,count_ub,u;
  double dQdP,lyap,rogu,shiguma,count_lyap,u1,u2,u3,u4,u5;

  printf("r_start =");scanf("%lf",&r_start);
  printf("r_end   =");scanf("%lf",&r_end);
  printf("dr      =");scanf("%lf",&dr);
  printf("x_start =");scanf("%lf",&x_start);
  printf("y_start =");scanf("%lf",&y_start);
  printf("z_start =");scanf("%lf",&z_start);
  printf("t_min   =");scanf("%lf",&t_min);
  printf("t_max   =");scanf("%lf",&t_max);
```

```
 printf("dt      =");scanf("%lf",&dt);

 datafile=fopen("lyapounov.dat","w");

 for(r=r_start;r<=r_end;r=r+dr)
   {
    printf("/*** r=%5.2f loop ***/\n",r);
    w1=-1;  w2=1;
    x=x_start;  y=y_start;  z=z_start;
    z_now=z_old1=z_old2=z_old3=z_old4=z_old5=-0;
    z_old6=z_old7=z_old8=z_old9=z_old10=-1;
    t=M_max=b=count_M=0.000000;
    QQQ=Q_zero=Q_max=0;
    n=kazu=shiguma=count_ua=count_ub=0;
    Pua=Pub=Qua=Qub=a=0;
    count_lyap=1;    M_min=c=99999;

    writefile=fopen("lyap_prot.dat","w");

    for(t=0;t<=t_max;t=t+dt)
      {

if(t>=t_min && z>0.0001) /** start if t_min **/
  {
    z_old10=z_old9;    z_old9=z_old8;
    z_old8=z_old7;     z_old7=z_old6;
    z_old6=z_old5;     z_old5=z_old4;
    z_old4=z_old3;     z_old3=z_old2;
    z_old2=z_old1;     z_old1=z_now;
    z_now=z;


    if(z_old2 < z_old1 && z_now < z_old1) /** start if z_old **/
      {
MP=MQ;
```

```
MQ=z_old1;

if(M_max<MQ)
  {
    M_max=MQ;
    b=MP;
  }

if(M_min>MQ)
  {
    M_min=MQ;
  }
if(MP>0.00001 && MQ>0.00001)
  {
    count_M=count_M+1;
    fprintf(writefile,"%10.6f %10.6f\n",MP,MQ);
  }

      } /** end if z_old **/

  }   /** end if  t_min **/

ka=dt*(-10*(x-y));   kax=x+(ka/2);
la=dt*(-x*z+r*x-y);  lay=y+(la/2);
ma=dt*(x*y-(8/3)*z); maz=z+(ma/2);

kb=dt*((-10)*(kax-lay));      kbx=x+(kb/2);
lb=dt*((-kax)*(maz)+(r*kax)-lay); lby=y+(lb/2);
mb=dt*((kax*lay)-((8/3)*maz));  mbz=z+(mb/2);

kc=dt*((-10)*((kbx-lby)));      kcx=x+kc;
lc=dt*((-kbx)*mbz+(r*kbx)-lby); lcy=y+lc;
mc=dt*(kbx*lby-((8/3)*mbz));  mcz=z+mc;

kd=dt*((-10)*(kcx-lcy));
```

```
ld=dt*((-kcx)*mcz+(r*kcx)-lcy);
md=dt*(kcx*lcy-((8/3)*mcz));

x=x+((ka+(2*kb)+(2*kc)+kd)/6);
y=y+((la+(2*lb)+(2*lc)+ld)/6);
z=z+((ma+(2*mb)+(2*mc)+md)/6);

      }    /** end for t  **/

    fclose(writefile);

    sa_z=fabs(z_now-z_old10);
    if(z_old10>0.0 && sa_z>0.00000001)
      {

printf("count_M = %6.0f\n",count_M);


/**  Start  reading "lyapounov.dat"  **/

readfile=fopen("lyap_prot.dat","r");

i=0;

while(fscanf(readfile,"%lf %lf",&P[i],&Q[i]) != EOF)
  {
    i++;
  }
kazu=i;
printf("kazu  =%d\n",kazu);

fclose(readfile);

/** End reading "lyapounov.dat"  **/
```

```
for(i=0; i<kazu; i++)
  {
    if(Q[i]>Q_max)
      {
Q_max=Q[i];
/* b=P[i];*/
      }
  }

for(i=0; i<kazu; i++)
  {
    if(P[i]!=b && Q[i]!=c)
      {
PP[i]=log(fabs(P[i]-b));
QQ[i]=log(log(Q_max)-log(fabs(Q[i]-c)));

if(PP[i]>-0.001 && PP[i]<0.001)
  {
    QQQ=QQQ+QQ[i];
    n=n+1;
  }
if(PP[i]>w1-0.01 && PP[i]<w1+0.01)
  {
    Pua=Pua+PP[i];
    Qua=Qua+QQ[i];
    count_ua=count_ua+1;
  }
if(PP[i]>w2-0.01 && PP[i]<w2+0.01)
  {
    Pub=Pub+PP[i];
    Qub=Qub+QQ[i];
    count_ub=count_ub+1;
  }
      }
  }
```

```
 Q_zero=QQQ/n;
 a=exp(Q_zero);
 beki=((Qub/count_ub)-(Qua/count_ua))/((Pub/count_ub)-(Pua/count_ua));

 printf("n counts = %3.0f\n",n);
 printf("Pua=%10.6f Pub=%10.6f\n",Pua/count_ua,Pub/count_ub);
 printf("beki=%f a=%f b=%f c=%f\n",beki,a,b,c);

 /** Start lyapounov value **/

 for(i=0;i<=kazu;i++)
   {
     if(P[i]!=b && Q[i]!=c)
       {
 dQdP=fabs(Q_max*a*beki*pow(fabs(P[i]-b),(beki-1))
*exp(-a*pow(fabs(P[i]-b),beki)));

 rogu=log(dQdP);
 shiguma=shiguma+rogu;
 count_lyap=count_lyap+1;
       }
   }
 lyap=shiguma/count_lyap;
 printf("r=%4.2f lyapounov =%10.6f\n",r,lyap);
 fprintf(datafile,"%6.3f %10.6f\n",r,lyap);

       }
     else
       {
 lyap=-10;
 printf("r=%4.2f lyapounov =%10.6f\n",r,lyap);
 fprintf(datafile,"%6.3f %10.6f\n",r,lyap);
       }
```

```
   }  /** End for(r=r_start;...) ***/

 fclose(datafile);

}
```