Calculating the entire Lyapunov Spectra of the Lorenz Attractor

1 Introduction

The lorenz dynamical system is given by

$$\frac{dx_1}{dt} = f_1(x_1, x_2, x_3) = \sigma(x_2(t) - x_1(t))$$

$$\frac{dx_2}{dt} = f_2(x_1, x_2, x_3) = x_1(t)(\rho - x_3(t)) - x_2(t)$$

$$\frac{dx_3}{dt} = f_3(x_1, x_2, x_3) = x_1(t)x_2(t) - \beta x_3(t)$$

For purposes of this discussion, we take $\sigma = 10, \rho = 28, \beta = 8/3$.

2 Calculating Lyapunov exponents of a Discrete Flow

For a discrete mapping $x(t+1) = F(x(t))$ we calculate the local expansion of the flow by considering the difference of two trajectories

$$x(t+1) - y(t+1) = f(x(t)) - f(y(t)) \approx$$

$$\frac{\partial f}{\partial x}(x(t)).(x(t) - y(t)) = Df(x).(x(t) - y(t))$$

If this grows like

$$|x(t+1) - y(t+1)| \approx e^{\lambda}|x(t) - y(t)|$$

then the exponent λ is called the Lyapunov exponent. If it is positive, bounded flows will generally be chaotic. We can solve for this exponent, asymptotically, by

$$\lambda \approx \ln(|x_{n+1} - y_{n+1}|/|x_n - y_n|)$$

for two points x_n, y_n where are close to each other on the trajectory.

This allows you to estimate the Lyapunov exponent of a scalar map by only knowing the orbit. By searching through the list of all orbital points, you pick the two nearest points. (They can't be the same, or else the orbit is periodic!) There are ways of averaging, and doing regression, that enable better computations.

If the discrete flow comes from a vector map, then one needs information about the Jacobian

$$J_{m,n} = Df(v_0) = \frac{\partial f_m}{\partial x_n}$$

3 Calculating Lyapunov Exponents of a Continuous Flow

If the underlying dynamical system comes from an autonomous system of differential equations of the form

$$\frac{dx}{dt} = F(x)$$

then considering two neighboring trajectories

$$\frac{d(x-y)}{dt} = F(x) - F(y) \approx DF(x).(x(t) - y(t))$$

at a infitesimally later time, we have (asymptotically)

$$x(t+h) - y(t+h) = (x(t) - y(t)) + h * Df(x).(x(t) - y(t))$$

$$= [I + h * Df(x)].(x(t) - y(t))$$

This means the local expansion is given by $I + h * Df(v_0)$ instead of just $Df(v_0)$.

If $x - y$ is in the direction of an eigenvector of the matrix $Df(v_0)$ then you can recover the associated eigenvalue by

$$\frac{|x(t+h) - y(t+h)|}{|x(t) - y(t)|} \approx 1 + h\lambda_k \approx e^{h\lambda_k}$$

This gives rise to two different approximations:

$$\lambda_k \approx (\frac{|x(t+h) - y(t+h)|}{|x(t) - y(t)|} - 1)/h$$

or

$$\lambda_k \approx \ln(\frac{|x(t+h) - y(t+h)|}{|x(t) - y(t)|})/h$$

(the second is more accurate in general).

4 Jacobian of Lorenz flow

The jacobian of the Lorenz system is given by

$$J(x,y,z) = Df(x,y,z) = \begin{bmatrix} -\sigma & \sigma & 0 \\ -x_3 + \rho & -1 & -x1 \\ x_2 & x_1 & -\beta \end{bmatrix}$$

So the local expansion (and therefore the Lyapunov exponents) are given by behavior of $I + \Delta t J$.

5 Matlab Code

```
function lorenz_spectra(T,dt)
% Usage: lorenz_spectra(T,dt)
% T is the total time and dt is the time step

% parameters defining canonical Lorenz attractor
sig=10.0;
rho=28;
bet=8/3;

% dt=0.01; %time step
N=T/dt; %number of time intervals

% calculate orbit at regular time steps on [0,T]
% using matlab's built-in ode45 runke kutta integration routine

% begin with initial conditions (1,2,3)
x1=1; x2=2; x3=3;
% integrate forwards 10 units
[t,x] = ode45('g',[0:1:10],[x1;x2;x3]);
n=length(t);
% begin at this point, hopefully near attractor!
x1=x(n,1); x2=x(n,2); x3=x(n,3);
[t,x] = ode45('g',[0:dt:T],[x1;x2;x3]);

e1=0;
e2=0;
e3=0;

% show trajectory being analyzed
plot3(x(:,1),x(:,2),x(:,3),'.','MarkerSize',2);
JN = eye(3);
w = eye(3);
J = eye(3);

for k=1:N
    % calculate next point on trajectory
    x1 = x(k,1);
    x2 = x(k,2);
    x3 = x(k,3);
    % calculate value of flow matrix at orbital point
    % remember it is I+Df(v0)*dt not Df(v0)
    J =  (eye(3)+[-sig,sig,0;-x3+rho,-1,-x1;x2,x1,-bet]*dt);
    % calculate image of unit ball under J
    % remember, w is orthonormal ...
```

```
    w = ortho(J*w);
    % calculate stretching
    % should be e1=e1+log(norm(w(:,1)))/dt; but scale after summing
    e1=e1+log(norm(w(:,1)));
    e2=e2+log(norm(w(:,2)));
    e3=e3+log(norm(w(:,3)));
    % e1=e1+norm(w(:,1))-1;
    % e2=e2+norm(w(:,2))-1;
    % e3=e3+norm(w(:,3))-1;
    % renormalize into orthogonal vectors
    w(:,1) = w(:,1)/norm(w(:,1));
    w(:,2) = w(:,2)/norm(w(:,2));
    w(:,3) = w(:,3)/norm(w(:,3));
end

% exponent is given as average e1/(N*dt)=e1/T
e1=e1/T;    % Lyapunov exponents
e2=e2/T;
e3=e3/T;
l1=exp(e1); % Lyapunov numbers
l2=exp(e2);
l3=exp(e3);
[e1,e2,e3]
trace=e1+e2+e3
[l1,l2,l3]
```

The output is given by

```
>> lorenz_spectra(10,0.001)

ans =
0.8438   -0.0409  -14.5011

trace =
-13.6982

ans =
2.3253    0.9599    0.0000
```

One can show that the sum of the Lyapunov exponents must add up to the sum of the diagonal elements of the jacobian, hence we must have

$$\lambda_1 + \lambda_2 + \lambda_3 = -\sigma - 1 - \beta = -13.6666...$$

6 References

1. http://sprott.physics.wisc.edu/chaos/lespec.htm – contains a link to a basic program for calculating lyapunov spectra...

2. http://sprott.physics.wisc.edu/chaos/lyapexp.htm

3. http://sprott.physics.wisc.edu/chaos/lorenzle.htm

4. http://cse.ucdavis.edu/˜chaos/courses/nlp/Software/part7.html

5. http://cse.ucdavis.edu/˜chaos/courses/nlp/Software/Part7_Code/LorenzODELCE.py – (python code for calculating lyapunov spectrum)