Міністерство освітиі науки України  
Національний технічний університет України

«Київський політехнічний інститут»  
Факультет прикладної математики  
Кафедра спеціалізованих комп’ютерних систем

**Лабораторна робота №5**

з дисципліни

«Моделювання»

Перевірив:

НаливайчукМ.В.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Київ 2012**

**Завдання:**

1. Розробити алгоритм та написати програму в системі Matlabдля розв'язання рівняння Лоренца з хаотичним аттрактором.
2. Використовуючи змінну x(t)рівняння Лоренца, обчислити кореляційний інтеграл.
3. Графічним способом обчислити кореляційну розмірність.

**Лістинги та результати:**

1.

function lorenz(action)

%LORENZ Plot the orbit around the Lorenz chaotic attractor.

% This demo animates the integration of the

% three coupled nonlinear differential equations

% that define the "Lorenz Attractor", a chaotic

% system first described by Edward Lorenz of

% the Massachusetts Institute of Technology.

%

% As the integration proceeds you will see a

% point moving around in a curious orbit in

% 3-D space known as a strange attractor. The

% orbit is bounded, but not periodic and not

% convergent (hence the word "strange").

%

% Use the "Start" and "Stop" buttons to control

% the animation.

% Adapted for Demo by Ned Gulley, 6-21-93; jae Roh, 10-15-96

% Copyright 1984-2005 The MathWorks, Inc.

% $Revision: 5.13.4.3 $ $Date: 2005/12/15 20:52:53 $

% The values of the global parameters are

global SIGMA RHO BETA

SIGMA = 10.;

RHO = 28.;

BETA = 8./3.;

% Possible actions:

% initialize

% close

% Information regarding the play status will be held in

% the axis user data according to the following table:

play= 1;

if nargin<1,

action='initialize';

end

switch action

case 'initialize'

oldFigNumber=watchon;

figNumber=figure( ...

'Name','Lorenz Attractor', ...

'NumberTitle','off', ...

'Visible','off');

colordef(figNumber,'black')

axes( ...

'Units','normalized', ...

'Position',[0.05 0.10 0.75 0.95], ...

'Visible','off');

text(0,0,'Press the "Start" button to see the Lorenz demo', ...

'HorizontalAlignment','center');

axis([-1 1 -1 1]);

%===================================

% Information for all buttons

xPos=0.85;

btnLen=0.10;

btnWid=0.10;

% Spacing between the button and the next command's label

spacing=0.05;

%====================================

% The CONSOLE frame

frmBorder=0.02;

yPos=0.05-frmBorder;

frmPos=[xPos-frmBorder yPos btnLen+2\*frmBorder 0.9+2\*frmBorder];

uicontrol( ...

'Style','frame', ...

'Units','normalized', ...

'Position',frmPos, ...

'BackgroundColor',[0.50 0.50 0.50]);

%====================================

% The START button

btnNumber=1;

yPos=0.90-(btnNumber-1)\*(btnWid+spacing);

labelStr='Start';

callbackStr='lorenz(''start'');';

% Generic button information

btnPos=[xPos yPos-spacing btnLen btnWid];

startHndl=uicontrol( ...

'Style','pushbutton', ...

'Units','normalized', ...

'Position',btnPos, ...

'String',labelStr, ...

'Interruptible','on', ...

'Callback',callbackStr);

%====================================

% The STOP button

btnNumber=2;

yPos=0.90-(btnNumber-1)\*(btnWid+spacing);

labelStr='Stop';

% Setting userdata to -1 (=stop) will stop the demo.

callbackStr='set(gca,''Userdata'',-1)';

% Generic button information

btnPos=[xPos yPos-spacing btnLen btnWid];

stopHndl=uicontrol( ...

'Style','pushbutton', ...

'Units','normalized', ...

'Position',btnPos, ...

'Enable','off', ...

'String',labelStr, ...

'Callback',callbackStr);

%====================================

% The INFO button

labelStr='Info';

callbackStr='lorenz(''info'')';

infoHndl=uicontrol( ...

'Style','push', ...

'Units','normalized', ...

'position',[xPos 0.20 btnLen 0.10], ...

'string',labelStr, ...

'call',callbackStr);

%====================================

% The CLOSE button

labelStr='Close';

callbackStr= 'close(gcf)';

closeHndl=uicontrol( ...

'Style','push', ...

'Units','normalized', ...

'position',[xPos 0.05 btnLen 0.10], ...

'string',labelStr, ...

'call',callbackStr);

% Uncover the figure

hndlList=[startHndl stopHndl infoHndl closeHndl];

set(figNumber,'Visible','on', ...

'UserData',hndlList);

set(figNumber, 'CloseRequestFcn', 'clear global SIGMA RHO BETA;closereq');

watchoff(oldFigNumber);

figure(figNumber);

case 'start'

axHndl=gca;

figNumber=gcf;

hndlList=get(figNumber,'UserData');

startHndl=hndlList(1);

stopHndl=hndlList(2);

infoHndl=hndlList(3);

closeHndl=hndlList(4);

set([startHndl closeHndl infoHndl],'Enable','off');

set(stopHndl,'Enable','on');

% ====== Start of Demo

set(figNumber,'Backingstore','off');

% The graphics axis limits are set to values known

% to contain the solution.

set(axHndl, ...

'XLim',[0 40],'YLim',[-35 10],'ZLim',[-10 40], ...

'Userdata',play, ...

'XTick',[],'YTick',[],'ZTick',[], ...

'Drawmode','fast', ...

'Visible','on', ...

'NextPlot','add', ...

'Userdata',play, ...

'View',[-37.5,30]);

xlabel('X');

ylabel('Y');

zlabel('Z');

% The orbit ranges chaotically back and forth around two different points,

% or attractors. It is bounded, but not periodic and not convergent.

% The numerical integration, and the display of the evolving solution,

% are handled by the function ODE23P.

FunFcn='lorenzeq';

% The initial conditions below will produce good results

%y0 = [20 5 -5];

% Random initial conditions

y0(1)=rand\*30+5;

y0(2)=rand\*35-30;

y0(3)=rand\*40-5;

t0=0;

tfinal=100;

pow = 1/3;

tol = 0.001;

t = t0;

hmax = (tfinal - t)/5;

hmin = (tfinal - t)/200000;

h = (tfinal - t)/100;

y = y0(:);

% Save L steps and plot like a comet tail.

L = 50;

Y = y\*ones(1,L);

cla;

head = line( ...

'color','r', ...

'Marker','.', ...

'markersize',25, ...

'erase','xor', ...

'xdata',y(1),'ydata',y(2),'zdata',y(3));

body = line( ...

'color','y', ...

'LineStyle','-', ...

'erase','none', ...

'xdata',[],'ydata',[],'zdata',[]);

tail=line( ...

'color','b', ...

'LineStyle','-', ...

'erase','none', ...

'xdata',[],'ydata',[],'zdata',[]);

% The main loop

while (get(axHndl,'Userdata')==play) && (h >= hmin)

if t + h > tfinal, h = tfinal - t; end

% Compute the slopes

s1 = feval(FunFcn, t, y);

s2 = feval(FunFcn, t+h, y+h\*s1);

s3 = feval(FunFcn, t+h/2, y+h\*(s1+s2)/4);

% Estimate the error and the acceptable error

delta = norm(h\*(s1 - 2\*s3 + s2)/3,'inf');

tau = tol\*max(norm(y,'inf'),1.0);

% Update the solution only if the error is acceptable

if delta <= tau

t = t + h;

y = y + h\*(s1 + 4\*s3 + s2)/6;

% Update the plot

Y = [y Y(:,1:L-1)];

set(head,'xdata',Y(1,1),'ydata',Y(2,1),'zdata',Y(3,1))

set(body,'xdata',Y(1,1:2),'ydata',Y(2,1:2),'zdata',Y(3,1:2))

set(tail,'xdata',Y(1,L-1:L),'ydata',Y(2,L-1:L),'zdata',Y(3,L-1:L))

drawnow;

end

% Update the step size

if delta ~= 0.0

h = min(hmax, 0.9\*h\*(tau/delta)^pow);

end

% Bail out if the figure was closed.

if ~ishandle(axHndl)

return

end

end % Main loop ...

% ====== End of Demo

set([startHndl closeHndl infoHndl],'Enable','on');

set(stopHndl,'Enable','off');

case 'info'

helpwin(mfilename);

end % if strcmp(action, ...

%===============================================================================

function ydot = lorenzeq(t,y)

%LORENZEQ Equation of the Lorenz chaotic attractor.

% ydot = lorenzeq(t,y).

% The differential equation is written in almost linear form.

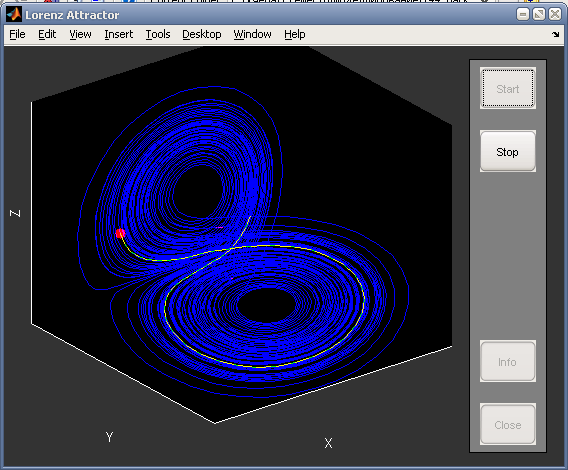
global SIGMA RHO BETA

A = [ -BETA 0 y(2)

0 -SIGMA SIGMA

-y(2) RHO -1 ];

ydot = A\*y;



2,3.

% Code to compute correlation dimension for an attractor

% generated by an orbit of a 2D map

% Ricardo Carretero 11/30/04

clear

format compact

figure(1)

clf

figure(2)

clf

trans = 10; % transients

Npts = 20; % number of pts (no more than a couple of K's)

x0=0.1; y0=0.1; % ICs

a=1.2; b=0.4; % parameters of map

xold=x0;

yold=y0;

for jj=1:trans % transients

xn=a-xold.^2+b\*yold;

yn=xold;

xold=xn;yold=yn;

end

x=zeros(Npts,1);

y=zeros(Npts,1);

x(1)=xn;

y(1)=yn;

for jj=1:Npts-1 % generating orbit

x(jj+1)=a-x(jj)^2+b\*y(jj);

y(jj+1)=x(jj);

end

D = sparse(Npts,Npts); % generating distance matrix

for jj=1:Npts

for ii=jj+1:Npts

dd = (x(ii)-x(jj))^2 + (y(ii)-y(jj))^2;

D(ii,jj)=dd;

end

end

D=sqrt(D);

rm = double(min(min(D+(1000\*D==0))));

rM = double(max(max(D)));

rM = 2^ceil(log(rM)/log(2));

ndiv = floor(double(log(rM/rm)/log(2)));

nr = ndiv+1;

rvec=rM\*2.^(-((1:nr)'-1));

Npairs=Npts\*(Npts-1)/2;

Cr=[]; % Correlation function

for jj=1:nr

r = rvec(jj)

N = (D<r & D>0);

S = double(sum(sum(N)));

Cr = [Cr; S/Npairs];

end

figure(2)

plot(rvec,Cr,'o-'); % plot results

hold on

xlabel('r');

ylabel('C(r)');

grid

discard=3; % discard a few points on either ends

n1=discard+1;

n2=nr-discard;

inside=n1:n2;

xx=log(rvec)/log(2);

yy=log(Cr)/log(2);

xxx=xx(inside);

yyy=yy(inside);

[coeff,temp]=polyfit(xxx,yyy,1); % fit a line to compute Dc

Dc=coeff(1)

yfit=Dc\*xx+coeff(2);

figure(1)

plot(xx,yy,'o-'); % plot results

hold on

plot(xx,yfit,'r-');

axis tight

plot([xx(n1),xx(n1)],[-30,30],'k--');

plot([xx(n2),xx(n2)],[-30,30],'k--');

xlabel('log\_2(r)');

ylabel('log\_2(C(r))');

title(['D\_c=',num2str(Dc)]);

grid

