The Lorentz attractor and the construction of a higher-order symplictic method

Nikiforov Vladimir

august 2023

1 math method

We start by constructing the first-order Euler method for the system of Lawrence equations:

$$\begin{cases} \dot{x} = \sigma(y(t) - x(t)) \\ \dot{y} = x(t)(r - z(t)) - y(t) \\ \dot{z} = x(t)y(t) - bz(t) \end{cases}$$

where real r, b, σ, variables are system parameters

Or in terms of stepwise iterations:

$$\dot{x} = \frac{x(t + \Delta t) - x(t)}{\Delta t} \Rightarrow \dot{x}\Delta t = x(t + \Delta t) - x(t)$$

$$\dot{x}h_t = x_{i+1} - x_i$$

$$\begin{cases} x_{i+1} - x_i = h_t(\sigma(y_i - x_i)) \\ y_{i+1} - z_i = h_t(x_i(r - z_i) - y_i) \\ z_{i+1} - z_i = h_t(x_i y_i - bz_i) \end{cases}$$

omitting algebraic calculations we have:

$$\begin{cases} x_{i+1} = h_t\sigma y_i + (1 - h_t\sigma)x_i \\ y_{i+1} = h_t(r - z_i)x_i + (1 - h_t)y_i \\ z_{i+1} = h_t y_i x_i + (1 - h_t b)z_i \end{cases}$$

or in the method operator view:

$$\begin{pmatrix} x_{i+1} \\ y_{i+1} \\ z_{i+1} \end{pmatrix} = \begin{pmatrix} 1 - h_t\sigma & h_t\sigma & 0 \\ h_t(r - z_i) & 1 - h_t & 0 \\ h_t y_i & 0 & 1 - h_t b \end{pmatrix} \begin{pmatrix} x_i \\ y_i \\ z_i \end{pmatrix}$$

our method will look like this:

$$\phi = \begin{pmatrix} 1 - h_t\sigma & h_t\sigma & 0 \\ h_t(r - z_i) & 1 - h_t & 0 \\ h_t y_i & 0 & 1 - h_t b \end{pmatrix}$$

using the method composition theorem, you can construct a second-order method that performs the following calculation:

$$\phi^2 = \phi^* \phi_h = \phi_{-h}^{-1} \phi_h$$

$$\phi^2 = \begin{pmatrix} 1 - h_t\sigma & h_t\sigma & 0 \\ h_t(r - z_i) & 1 - h_t & 0 \\ h_t y_i & 0 & 1 - h_t b \end{pmatrix}^{-1} \begin{pmatrix} 1 + h_t\sigma & -h_t\sigma & 0 \\ -h_t(r - z_i) & 1 + h_t & 0 \\ -h_t y_i & 0 & 1 + h_t b \end{pmatrix}$$

Accordingly, for the older method, the algorithm is the same. since the calculations are carried out by a computer, the dependence of the method on the step does not play a significant role, we simply recalculate this matrix at each step.

2 code

in matlab:

```
clc
clearvars
cla
close all

%% system parameter

ht = 0.0001;

r = 28;

sig = 10;

b = 8/3;

NIT = 100000;

%% boundary conditions

x = zeros(1,NIT);
y = zeros(1,NIT);
z = zeros(1,NIT);

x(1) = 10;
y(1) = 10;
z(1) = 10;

x2 = zeros(1,NIT);
y2 = zeros(1,NIT);
z2 = zeros(1,NIT);

x2(1) = 10;
y2(1) = 10;
z2(1) = 10;

x4 = zeros(1,NIT);
y4 = zeros(1,NIT);
z4 = zeros(1,NIT);

x4(1) = 10;
y4(1) = 10;
z4(1) = 10;

%% solve system equation
for i = 1:NIT-1
% matrix first order method
fi = [(1 - ht * sig), ht * sig, 0;
      ht * (r - z(i)), (1 - ht), 0;
      ht * y(i), 0, (1 - ht * b)];

% matrix second order method
% fi_-h ^ -1
fii = [(1 - ht * sig), ht * sig, 0;
```

```
      ht * (r - z2(i)), (1 - ht), 0;
      ht * y2(i), 0, (1 - ht * b)];
fimh = [(1 - (-ht) * sig), (-ht) * sig, 0;
      (-ht) * (r - z2(i)), (1 - (-ht)), 0;
      (-ht) * y2(i), 0, (1 - (-ht) * b)];
fimhm = fimh ^-1;
fi2 = fimhm * fii;

% matrix fourth order method
% % fi2_-h ^ -1

fiIV1 = [(1 - ht * sig), ht * sig, 0;
         ht * (r - z4(i)), (1 - ht), 0;
         ht * y4(i), 0, (1 - ht * b)];

fiIV1nh = [(1 - (-ht) * sig), (-ht) * sig, 0;
           (-ht) * (r - z4(i)), (1 - (-ht)), 0;
           (-ht) * y4(i), 0, (1 - (-ht) * b)] ^-1;
% -ht:
fiIV2 = [(1 - (-ht) * sig), (-ht) * sig, 0;
           (-ht) * (r - z4(i)), (1 - (-ht)), 0;
           (-ht) * y4(i), 0, (1 - (-ht) * b)];

fiIV2nh = [(1 - ht * sig), ht * sig, 0;
         ht * (r - z4(i)), (1 - ht), 0;
         ht * y4(i), 0, (1 - ht * b)] ^-1;

fi4 = fiIV1nh * fiIV1 * ((fiIV2 * fiIV2nh) ^ -1);

% calculation first order
x(i+1) = fi(1,1) * x(i) + fi(1,2) * y(i) + fi(1,3) * z(i);
y(i+1) = fi(2,1) * x(i) + fi(2,2) * y(i) + fi(2,3) * z(i);
z(i+1) = fi(3,1) * x(i) + fi(3,2) * y(i) + fi(3,3) * z(i);

% calculation second order
x2(i+1) = fi2(1,1) * x2(i) + fi2(1,2) * y2(i) + fi2(1,3) * z2(i);
y2(i+1) = fi2(2,1) * x2(i) + fi2(2,2) * y2(i) + fi2(2,3) * z2(i);
z2(i+1) = fi2(3,1) * x2(i) + fi2(3,2) * y2(i) + fi2(3,3) * z2(i);

% calculation fourth order
x4(i+1) = fi4(1,1) * x4(i) + fi4(1,2) * y4(i) + fi4(1,3) * z4(i);
y4(i+1) = fi4(2,1) * x4(i) + fi4(2,2) * y4(i) + fi4(2,3) * z4(i);
z4(i+1) = fi4(3,1) * x4(i) + fi4(3,2) * y4(i) + fi4(3,3) * z4(i);


end

%% output graphic

% dx = diff(x);
% dy = diff(y);
% dz = diff(z);
% x(end) = [];
% y(end) = [];
% z(end) = [];
figure(1);
```

```
plot3(x,y,z);
hold off;

figure(2);
plot3(x2,y2,z2);
hold off;

figure(3);
plot3(x4,y4,z4);
hold off;
```

3 results