

Week 4

程雨歌 12307110079

1 用 Taylor 级数法 (q=2,3) 计算 $\frac{du}{dt} = u - u^2$, 并与 Euler 显格式比较 (P74/1)

代码如下:

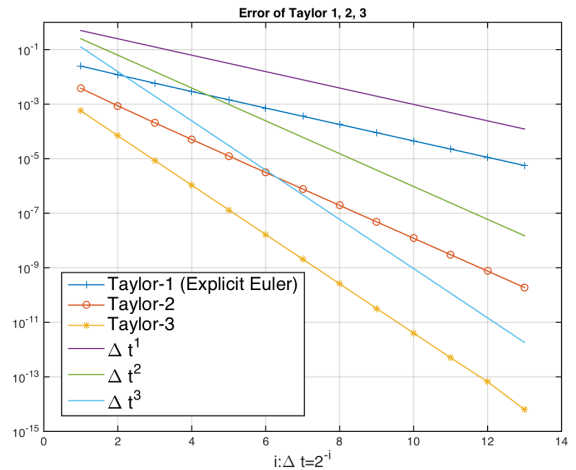
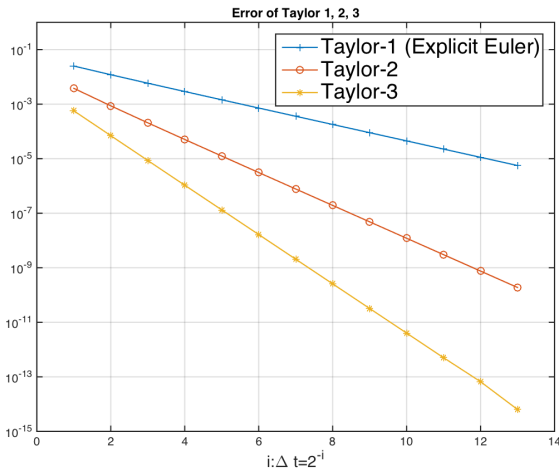
```
1 %% definitions
2 f = @(u) u - u.^2 ;
3 u = @(t, u0) 1 ./ ( ( 1 / u0 - 1 ) * exp( -t ) + 1 );
4 F = @(u) ( 1 - 2 .* u ) .* f(u) ;
5 G = @(u) f(u) .^ 2 .* (-2) ;
6 phi2 = @(u, dt) f(u) + dt ./ 2 .* F(u) ;
7 phi3 = @(u, dt) f(u) + dt ./ 2 .* F(u) ...
8       + dt .^ 2 ./ 6 .* ( G(u) + (1 - 2 .* u) .* F(u) );
9 t0 = 0 ; T = 8 ;
10 u0 = 0.5 ; % u0 = 1.5
11
12 %% compute error of Explicit Euler, phi2, phi3
13 r = 13;
14 dts = 2 .^ -(1: r);
15 e1 = zeros(1, r);
16 e2 = zeros(1, r);
17 e3 = zeros(1, r);
18 for i = 1 : r
19     dt = dts(i);
20     tdt = t0 : dt : T ;
21     n = length(tdt);
22     uu = u( tdt , u0 );
23     u1 = zeros ( 1 , n );
24     u2 = zeros ( 1 , n );
25     u3 = zeros ( 1 , n );
26     u1(1) = u0 ; u2(1) = u0 ; u3(1) = u0 ;
27     for j = 1 : n - 1
28         u1(j + 1) = u1(j) + dt * f(u1(j));
29         u2(j + 1) = u2(j) + dt * phi2(u2(j), dt);
30         u3(j + 1) = u3(j) + dt * phi3(u3(j), dt);
31     end
32     e1(i) = max(abs(u1 - uu));
33     e2(i) = max(abs(u2 - uu));
34     e3(i) = max(abs(u3 - uu));
35 end
36
37 %% plot
38 semilogy((1: r), e1, '-+'); hold on; grid on;
39 semilogy((1: r), e2, '-o');
40 semilogy((1: r), e3, '-*');
41 semilogy((1: r), dts);
```

```

42 semilogy((1: r), dts .^ 2);
43 semilogy((1: r), dts .^ 3);
44 set(gca,'ytick', 10 .^ (-15 : 2 : -1));
45 title('Error of Taylor 1, 2, 3');
46 h = legend('Taylor-1 (Explicit Euler)', 'Taylor-2', 'Taylor-3', ...
47 '\Delta t^1', '\Delta t^2', '\Delta t^3');
48 xlabel('i:\Delta t=2^{-i}', 'FontSize', 14);
49 set(h, 'FontSize', 16);
50
51 %% verify degree of convergence
52 figure(2);
53 for i = 1 : 4
54     d1 = e1 ./ (dts .^ i);
55     d2 = e2 ./ (dts .^ i);
56     d3 = e3 ./ (dts .^ i);
57     subplot(2,2,i);
58     semilogy((1: r), d1); hold on;
59     semilogy((1: r), d2);
60     semilogy((1: r), d3);
61     legend('Explicit Euler', 'Taylor-2', 'Taylor-3');
62     title(sprintf('p=%d', i));
63 end

```

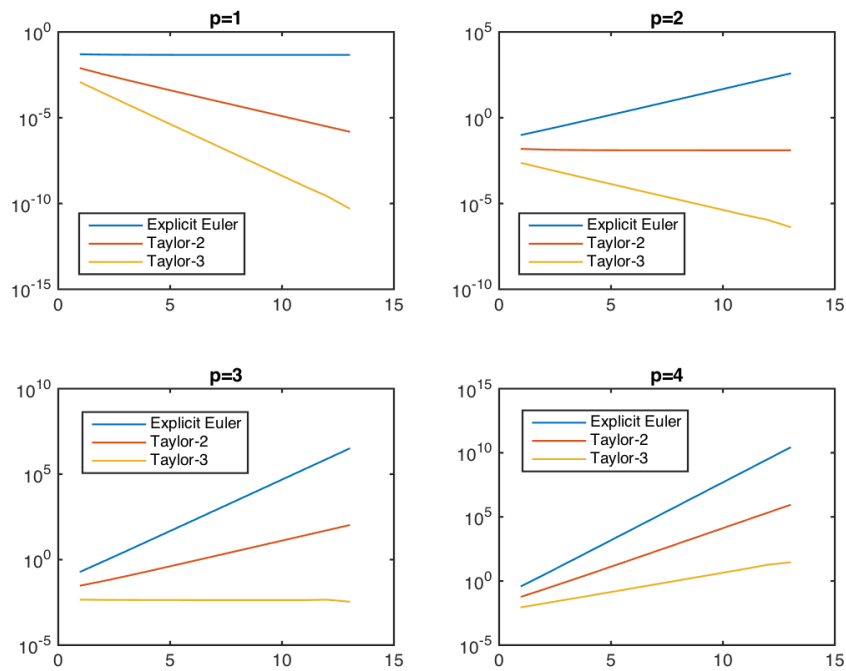
Euler 显式格式即当 $q=1$ 时的 Taylor 级数法。得到的误差比较如下：



将 $\Delta t^1, \Delta t^2, \Delta t^3$ 画出可以由平行关系看出各方法的收敛阶数。或者通过分别枚举 $p=1,2,3,4$, 并将每种方法得到的结果代入

$$\frac{|u(t) - u_n|}{\Delta t^p},$$

可以得到最高的具有常数上界结果的 p 即其收敛阶数：



可以看出，显式欧拉格式、2 阶、3 阶 Taylor 级数法的收敛阶数分别为 1、2、3。

2 用一到四阶格式的 Runge-Kutta 方法计算 $\frac{du}{dt} = u - u^2$ ，观察收敛阶 (P79/3)

代码如下：

```
1 %% definitions
2 f = @(u) u - u.^2 ;
3 u = @(t, u0) 1 ./ ( ( 1 / u0 - 1 ) * exp( -t ) + 1 );
4 phi2 = @(u, dt) 1/2 * ( f(u) + f(u + dt * f(u)) );
5 phi3 = @(u, dt) 1/6 * ( f(u) + 4 * f(u + dt/2 * f(u)) ...
6     + f(u + dt * (-f(u) + 2 * f(u + dt/2 * f(u)))) );
7 phi4 = @(u, dt) 1/8 * ( f(u) + 3 * f(u + dt/3 * f(u)) ...
8     + 3 * f(u + dt * (-1/3 * f(u) + f(u + dt/3 * f(u)))) ...
9     + f(u + dt * (f(u) - f(u + dt/3 * f(u)) ...
10     + f(u + dt * (-1/3 * f(u) + f(u + dt/3 * f(u)))))) );
11 t0 = 0 ; T = 8 ;
12 u0 = 0.5 ; % u0 = 1.5
13
14 %% compute error of Explicit Euler, phi2, phi3, phi4
15 r = 13;
16 dts = 2.^-(1:r);
17 e1 = zeros(1, r);
18 e2 = zeros(1, r);
19 e3 = zeros(1, r);
20 e4 = zeros(1, r);
21 for i = 1 : r
```

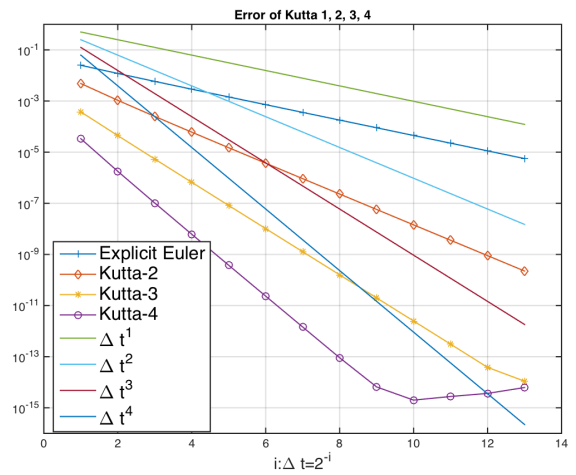
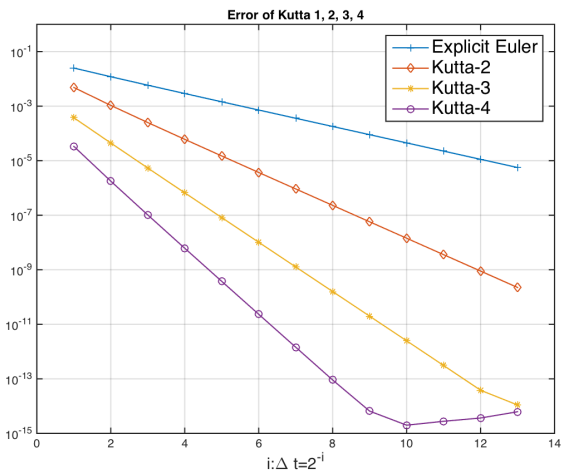
```

22     dt = dts(i);
23     tdt = t0 : dt : T ;
24     n = length(tdt);
25     uu = u( tdt , u0 );
26     u1 = zeros ( 1 , n );
27     u2 = zeros ( 1 , n );
28     u3 = zeros ( 1 , n );
29     u4 = zeros ( 1 , n );
30     u1(1) = u0 ; u2(1) = u0 ; u3(1) = u0 ; u4(1) = u0 ;
31     for j = 1 : n - 1
32         u1(j + 1) = u1(j) + dt * f(u1(j));
33         u2(j + 1) = u2(j) + dt * phi2(u2(j), dt);
34         u3(j + 1) = u3(j) + dt * phi3(u3(j), dt);
35         u4(j + 1) = u4(j) + dt * phi4(u4(j), dt);
36     end
37     e1(i) = max(abs(u1 - uu));
38     e2(i) = max(abs(u2 - uu));
39     e3(i) = max(abs(u3 - uu));
40     e4(i) = max(abs(u4 - uu));
41 end
42
43 %% plot
44 semilogy((1: r), e1, '-+'); hold on; grid on;
45 semilogy((1: r), e2, '-d');
46 semilogy((1: r), e3, '-*');
47 semilogy((1: r), e4, '-o');
48 semilogy((1: r), dts);
49 semilogy((1: r), dts .^ 2);
50 semilogy((1: r), dts .^ 3);
51 semilogy((1: r), dts .^ 4);
52 set(gca,'ytick', 10 .^ (-15 : 2 : -1));
53 title('Error of Kutta 1, 2, 3, 4');
54 h = legend('Explicit Euler', 'Kutta-2', 'Kutta-3', 'Kutta-4', ...
55 '\Delta t^1', '\Delta t^2', '\Delta t^3', '\Delta t^4');
56 xlabel('i:\Delta t=2^{-i}', 'FontSize', 14);
57 set(h, 'FontSize', 16);
58
59 %% verify degree of convergence
60 figure(2);
61 for i = 1 : 4
62     d1 = e1 ./ (dts .^ i);
63     d2 = e2 ./ (dts .^ i);
64     d3 = e3 ./ (dts .^ i);
65     d4 = e4 ./ (dts .^ i);
66     subplot(2,2,i);
67     semilogy((1: r), d1); hold on;
68     semilogy((1: r), d2);
69     semilogy((1: r), d3);
70     semilogy((1: r), d4);
71     legend('Explicit Euler', 'Kutta-2', 'Kutta-3', 'Kutta-4');
72     title(sprintf('p=%d', i));
73 end

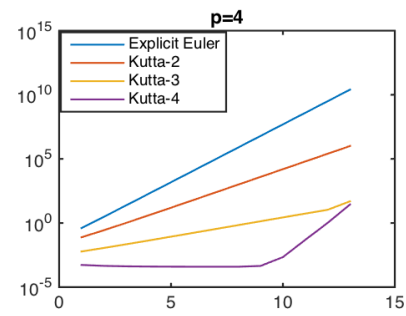
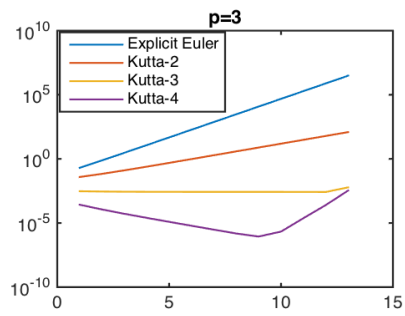
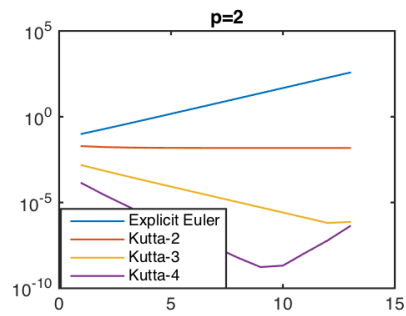
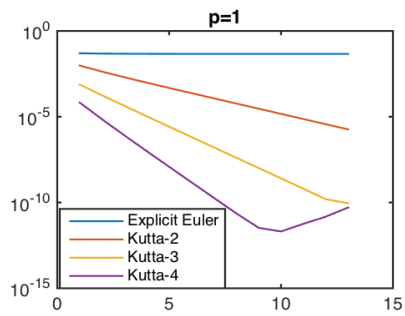
```

Euler 显式格式即当 $q=1$ 时的 Runge-Kutta 方法。4 阶采用的是 $3/8$ 的 Kutta 方法。

得到的误差比较如下图，可以看出 4 阶 Kutta 方法在步长为 2^{-10} 时误差就达到了 10^{-15} 的数量级，已经接近机器精度了，再减小步长已经不能提高精度了。



同样可通过两种方法考察其收敛阶数各为多少：



可以看出，显式欧拉格式、2 阶、3 阶、4 阶 Kutta 方法的收敛阶数分别为 1、2、3、4。