Foundations of Computational Math 2

Program 4

Chenchen Zhou

1. Program description

Apply three explicit methods:

Forward Euler method
Explicit midpoint method
Runge Kutta 4-stage 4th order

Apply one implicit method:

Back Euler method

2. Code

- Forward Euler:

```
function [error] = Forward_euler(h)
%%% Apply the Forward Euler method%%%%%%%%
t=1; y0=1;
while (t<25)

    y1=y0+h*f(t,y0);
    y0=y1;
    t=t+h;

end

error=abs(y0-1/25);
```

- Explicit Midpoint:

```
function [error] = Midpoint(h)
%%% Apply Explicit Midpoint method%%%%%%%
t=1; y0=1;
while(t<25)

    f1=f(t,y0);
    y1=y0+h*f(t+h/2,y0+(h/2)*f1);
    y0=y1;
    t=t+h;
end
error=abs(y0-1/25);
```

- Classical Explicit Runge Kutta 4-stage 4th order:

```
function [error] = RK(h)
%%% Apply Runge kutta method%%%%%%%
t=1; y0=1;
while(t<25)

    f1=f(t,y0);
    f2=f(t+h/2,y0+(h/2)*f1);
    f3=f(t+h/2,y0+(h/2)*f2);
    f4=f(t+h,y0+h*f3);
    y1=y0+h*(f1+2*f2+2*f3+f4)/6;
    y0=y1;
    t=t+h;

end
error=abs(y0-1/25);
```

Backward Euler

```
function [error] = Backward_euler(h)
%%% Apply the Backward Euler method%%%%%%%
t=1+h;    y0=1;
while(t<25)

    y1=-1/(10*t*h)+sqrt(1/(100*t^2*h^2)+y0/(5*t*h)+1/(t^2)-1/(5*t^3));
```

```
    y0=y1;
    t=t+h;
end
error=abs(y0-1/25);
```

test code

```
function [err1,err2,err3,err4]=test(I)
for i=1:7
    err1(i)=Forward_euler(I(i));
    err2(i)=Midpoint(I(i));
    err3(i)=RK(I(i));
    err4(i)=Backward_euler(I(i));
end
err1
err2
err3
err4
```

3. Result

	Forward euler	midpoint	R-K	Backward
h=0.2	4.684494348E-03	1.012363252E-03	3.168176042E-04	1.306343604E-06
h=0.1	6.451561900E-07	3.285230800E-07	2.161955089E-08	1.613009596E-04
h=0.05	3.232390000E-07	5.436446000E-08	1.070657010E-09	8.048685449E-05
h=0.02	3.210356384E-05	3.196720438E-05	3.197439677E-05	1.296675881E-07
h=0.01	6.475395000E-08	1.712340000E-09	1.411370000E-12	1.607128787E-05
h=0.005	8.030764440E-06	7.997983560E-06	7.998400233E-06	3.239692351E-08
h=0.002	3.212695940E-06	3.199678330E-06	3.199744014E-06	1.295718070E-08

Analysis:

As we know, or the method the truncation error will be O(h^n), n is some number. So the globe error is related to local error or truncation error. If we decrease the stepsize h, we will get more accurate answer. But in my code, it has a strange outcome I think. Because when h decrease, the error is not absolutely decreasing. I guess the reason for it is we decrease the h, but for the same interval, [1,25], the smaller h is , more evaluation of f we will compute. So it accumulates the error.

In conclude, we need to choose our h in practice to get efficiency and accuracy. It's not true that the smaller h is, the more accurate result we will get.

(4.1.b) Consider the Jacobian of $f(t, y)$ and determine the inteval, $1 \le t \le t_{stab}$ for which each method/stepsize combination is absolutely stable. (For this problem λ is a function of t.) Use the figures from the notes and textbook (p. 491) to estimate the extent of the region of absolute stability.

For this question:

For explicit methods, we can write the difference equation like: $y_n = R(z)y_{n-1}$

In our methods, R(z) is (1+h*F(tn,yn)/yn). This function is related to t. put it another way, R(z) varies with time t. We just need to estimate h*F(tn,yn)/yn every step and to see when it will reach -2. Because F(tn,yn) is the combination of fi. F(tn,yn) is always less than 1, yn is the approximate of y, so when t>1, it will always less than 1 and decrease with t. R(z) will be about 0.2 when t is very close to 1. Then R(z) will decrease and for several steps it will reach -1. Then this is the max t for absolute stability. By considering all combination of methods with stepsize, we just to see the smallest t we get. For simplify this problem, first take F(tn,yn) equals to f(tn,yn). Then we will consider the Jacobi of f(tn,yn). We do this for estimate the value of R(z).

$$f(t, y) = \frac{5}{t} - \frac{1}{t^2} - 5ty^2,$$

We will see ft=2/t^3-5/t^2-5y^2; fy=-10ty;

If we investigate the value of ft and fy as t increase, we will see that, ft will be very small in several steps comparing to fy. And fy will have less change than ft. So the most contribution for change of f(t,y) comes from fy(yn+1-yn). Because we just need to get a bound. So let's take a very rough estimate. We just replace (h/yn) to 1, and replace f(tn,yn) to be -10tyn.(Because comparing to the initial value of f(1,1)=1, it is the most important role to influence the value of f(tn,yn).Then now R(z)=1-10t*h.
We need R(z)=-1. So we get the relationship between tmax=tstab and stepsize h.

t=0.2/h

By using this, we can calculate t as follows:

h =0.2	Tstab=1
h =0.1	Tstab=2
h =0.05	Tstab=4
h =0.02	Tstab=10
h =0.01	Tstab=20
h =0.005	Tstab=40
h =0.002	Tstab=100