Project 1: Taylor Series

Numerical Methods for ODEs

Jiale.Wang Tao.Luo Wenchao.Zhang[1]

South University of Science and Technology of China

September 25, 2012

[1] In alphabetical order

Outline

南方科技大学
SOUTH UNIVERSITY OF SCIENCE AND TECHNOLOGY OF CHINA

Outline

1. Elementary
   - The Basic Problem That We Studied
   - Works with Matlab
2. Our Results
   - Primary Results
   - Improved Method
3. Others

Initial Value Problem

$$\begin{cases} y'(x) = f(x,y) & a \leq x \leq b \\ y(x_0) = y_0 \end{cases} \tag{1}$$

where y_0 is a known constant.

- If $f(x,y)$ satisfies Lipschitz condition, y=y(x) has unique solution
- Can be solved by numerical methods

Initial Value Problem

$$\begin{cases} y'(x) = f(x,y) & a \leq x \leq b \\ y(x_0) = y_0 \end{cases} \tag{1}$$

where y_0 is a known constant.

- If $f(x,y)$ satisfies Lipschitz condition, y=y(x) has unique solution
- Can be solved by numerical methods

Basic Ideas and Methods

- Euler Method(EM)
- Right-hand Euler Method(REM)
- Trapezoidal Method(TM)
- Simpson Method(SM)
- Taylor Series Method(TSM)

Basic Ideas and Methods

- Euler Method(EM)
- Right-hand Euler Method(REM)
- Trapezoidal Method(TM)
- Simpson Method(SM)
- Taylor Series Method(TSM)

Basic Ideas and Methods

- Euler Method(EM)
- Right-hand Euler Method(REM)
- Trapezoidal Method(TM)
- Simpson Method(SM)
- Taylor Series Method(TSM)

Basic Ideas and Methods

- Euler Method(EM)
- Right-hand Euler Method(REM)
- Trapezoidal Method(TM)
- Simpson Method(SM)
- Taylor Series Method(TSM)

Basic Ideas and Methods

- Euler Method(EM)
- Right-hand Euler Method(REM)
- Trapezoidal Method(TM)
- Simpson Method(SM)
- Taylor Series Method(TSM)

Basic Ideas and Methods

- From Picard Method
-
  -
$$
\begin{aligned}
\int_{x_k}^{x_{k+1}} y'(x)dx &= \int_{x_k}^{x_{k+1}} f(x,y(x))dx \\
i.e.\ y(x_{k+1}) - y(x_k) &= \int_{x_k}^{x_{k+1}} f(x,y(x))dx
\end{aligned}
$$
- EM: Left rectangle replace right-hand integral
- REM:Right rectangle replace the integral
- TM:Trapezoidal replace the intergral
- SM:Parabola given by endpoints and midpoint

Basic Ideas and Methods

- From Picard Method
-
  -
$$\begin{aligned}\int_{x_k}^{x_{k+1}} y'(x)dx &= \int_{x_k}^{x_{k+1}} f(x,y(x))dx \\ i.e.\ y(x_{k+1}) - y(x_k) &= \int_{x_k}^{x_{k+1}} f(x,y(x))dx\end{aligned}$$
- EM: Left rectangle replace right-hand integral
- REM:Right rectangle replace the integral
- TM:Trapezoidal replace the intergral
- SM:Parabola given by endpoints and midpoint

Basic Ideas and Methods

- From Picard Method
-
  - $$\begin{aligned} \int_{x_k}^{x_{k+1}} y'(x)dx &= \int_{x_k}^{x_{k+1}} f(x,y(x))dx \\ i.e.\ y(x_{k+1}) - y(x_k) &= \int_{x_k}^{x_{k+1}} f(x,y(x))dx \end{aligned}$$
- EM: Left rectangle replace right-hand integral
- REM:Right rectangle replace the integral
- TM:Trapezoidal replace the intergral
- SM:Parabola given by endpoints and midpoint

Basic Ideas and Methods

- From Picard Method
-
  - $$\begin{aligned} \int_{x_k}^{x_{k+1}} y'(x)dx &= \int_{x_k}^{x_{k+1}} f(x,y(x))dx \\ i.e.\ y(x_{k+1}) - y(x_k) &= \int_{x_k}^{x_{k+1}} f(x,y(x))dx \end{aligned}$$
- EM: Left rectangle replace right-hand integral
- REM:Right rectangle replace the integral
- TM:Trapezoidal replace the intergral
- SM:Parabola given by endpoints and midpoint

Basic Ideas and Methods

- From Picard Method
-
  - $$\begin{aligned}\int_{x_k}^{x_{k+1}} y'(x)dx &= \int_{x_k}^{x_{k+1}} f(x,y(x))dx \\ i.e.\ y(x_{k+1})-y(x_k) &= \int_{x_k}^{x_{k+1}} f(x,y(x))dx\end{aligned}$$
- EM: Left rectangle replace right-hand integral
- REM:Right rectangle replace the integral
- TM:Trapezoidal replace the intergral
- SM:Parabola given by endpoints and midpoint

Basic Methods

- EM: $y(x_{k+1}) - y(x_k) \approx hf(x_k, y(x_k))$
  then $y_{k+1} = y_k + hf(x_k, y_k)$;
- REM: $y_{k+1} = y_k + hf(x_{k+1}, y_{k+1})$;
- TM: $y_{k+1} = y_k + \frac{h}{2}[f(x_k, y_k) + f(x_{k+1}, y_k + hf(x_k, y_k))]$;
- SM: $y_{k+1} = y_{k-1} + \frac{h}{3}(f(x_{k+1}, y_{k+1}) + 4f(x_k, y_k) + f(x_{k-1}, y_{k-1}))$

南方科技大学
SOUTH UNIVERSITY OF SCIENCE AND TECHNOLOGY OF CHINA

Basic Methods

- EM:$y(x_{k+1}) - y(x_k) \approx hf(x_k, y(x_k))$
  then $y_{k+1} = y_k + hf(x_k, y_k)$;
- REM:$y_{k+1} = y_k + hf(x_{k+1}, y_{k+1})$;
- TM:$y_{k+1} = y_k + \frac{h}{2}[f(x_k, y_k) + f(x_{k+1}, y_k + hf(x_k, y_k))]$;
- SM:$y_{k+1} = y_{k-1} + \frac{h}{3}(f(x_{k+1}, y_{k+1}) + 4f(x_k, y_k) + f(x_{k-1}, y_{k-1}))$

Basic Methods

- EM:$y(x_{k+1}) - y(x_k) \approx hf(x_k, y(x_k))$
  then $y_{k+1} = y_k + hf(x_k, y_k)$;
- REM:$y_{k+1} = y_k + hf(x_{k+1}, y_{k+1})$;
- TM:$y_{k+1} = y_k + \frac{h}{2}[f(x_k, y_k) + f(x_{k+1}, y_k + hf(x_k, y_k))]$;
- SM:$y_{k+1} = y_{k-1} + \frac{h}{3}(f(x_{k+1}, y_{k+1}) + 4f(x_k, y_k) + f(x_{k-1}, y_{k-1}))$

Basic Methods

- EM:$y(x_{k+1}) - y(x_k) \approx hf(x_k, y(x_k))$
  then $y_{k+1} = y_k + hf(x_k, y_k)$;
- REM:$y_{k+1} = y_k + hf(x_{k+1}, y_{k+1})$;
- TM:$y_{k+1} = y_k + \frac{h}{2}[f(x_k, y_k) + f(x_{k+1}, y_k + hf(x_k, y_k))]$;
- SM:$y_{k+1} = y_{k-1} + \frac{h}{3}(f(x_{k+1}, y_{k+1}) + 4f(x_k, y_k) + f(x_{k-1}, y_{k-1}))$

Taylor Series Method

- TSM needs more information of the points
- Iteration: $y_{k+1} = y_k + hy'_k + \frac{h^2}{2}y''_k + \cdots + \frac{h^p}{p!}y_k^{(p)}$
- Way to get differentials

$$y' = f(x,y) \equiv f^{(0)}$$
$$y'' = \frac{\partial f}{\partial x} + \frac{\partial f}{\partial y}\frac{dy}{dx} = \frac{\partial f^{(0)}}{\partial x} + \frac{\partial f^{(0)}}{\partial y} f \equiv f^{(1)}$$
$$y''' = \frac{\partial f^{(1)}}{\partial x} + \frac{\partial f^{(1)}}{\partial y} f \equiv f^{(2)}$$
$$\cdots$$
$$y^{(j)} = \frac{\partial f^{(j-2)}}{\partial x} + \frac{\partial f^{(j-2)}}{\partial y} f \equiv f^{(j-1)}$$
$$\cdots$$

Taylor Series Method

- TSM needs more information of the points
- Iteration: $y_{k+1} = y_k + hy'_k + \frac{h^2}{2}y''_k + \cdots + \frac{h^p}{p!}y_k^{(p)}$
- Way to get differentials

$$y' = f(x,y) \equiv f^{(0)}$$
$$y'' = \frac{\partial f}{\partial x} + \frac{\partial f}{\partial y}\frac{dy}{dx} = \frac{\partial f^{(0)}}{\partial x} + \frac{\partial f^{(0)}}{\partial y}f \equiv f^{(1)}$$
$$y''' = \frac{\partial f^{(1)}}{\partial x} + \frac{\partial f^{(1)}}{\partial y}f \equiv f^{(2)}$$
$$\cdots$$
$$y^{(j)} = \frac{\partial f^{(j-2)}}{\partial x} + \frac{\partial f^{(j-2)}}{\partial y}f \equiv f^{(j-1)}$$
$$\cdots$$

Taylor Series Method

- TSM needs more information of the points
- Iteration:$y_{k+1} = y_k + hy'_k + \frac{h^2}{2}y''_k + \cdots + \frac{h^p}{p!}y_k^{(p)}$
- Way to get differentials

$$y' = f(x,y) \equiv f^{(0)}$$
$$y'' = \frac{\partial f}{\partial x} + \frac{\partial f}{\partial y}\frac{dy}{dx} = \frac{\partial f^{(0)}}{\partial x} + \frac{\partial f^{(0)}}{\partial y} f \equiv f^{(1)}$$
$$y''' = \frac{\partial f^{(1)}}{\partial x} + \frac{\partial f^{(1)}}{\partial y} f \equiv f^{(2)}$$
$$\ldots$$
$$y^{(j)} = \frac{\partial f^{(j-2)}}{\partial x} + \frac{\partial f^{(j-2)}}{\partial y} f \equiv f^{(j-1)}$$
$$\ldots$$

Outline

1. Elementary
   - The Basic Problem That We Studied
   - Works with Matlab
2. Our Results
   - Primary Results
   - Improved Method
3. Others

SM with Matlab

```
k = c(1,2) + subs(f,[x,y],[c(1,1),c(1,2)])*h;
q = c(1,2) - subs(f,[x,y],[c(1,1),c(1,2)])*h;
for i = 2:n
if i==2
c(i,1) = c(i-1,1) + h;
w = c(i-1,1) - h;
k1 = subs(f,[x,y],[w,q]);
k2 = subs(f,[x,y],[c(i-1,1),c(i-1,2)]);
k3 = subs(f,[x,y],[c(i,1),k]);
c(i,2) = q + h*(k1+4*k2+k3)/3;
else
kk = c(i-1,2) + subs(f,[x,y],[c(i-1,1),c(i-1,2)])*h;
c(i,1) = c(i-1,1) + h;
kk1 = subs(f,[x,y],[c(i-2,1),c(i-2,2)]);
kk2 = subs(f,[x,y],[c(i-1,1),c(i-1,2)]);
kk3 = subs(f,[x,y],[c(i,1),kk]);
c(i,2) = c(i-2,2) + h*(kk1+4*kk2+kk3)/3;
```

Figure:

TSM with Matlab

```
c(i) = diff(c(i-1),x) + diff(c(i-1),y)*c(1);
end
subsc = subs(c,[x,y],[x0,y0]);
taylor = y0 ;
for i= 1:n
taylor = taylor + (subsc(i)*(x-x0)∧i)/factorial(i);
end
```

Figure:

Comparison of Methods

$$\begin{cases} y'(x) = 2y \\ y(0) = 1 \end{cases}$$

Black:REM; Blue:Solution; Red:SM;Yellow:EM

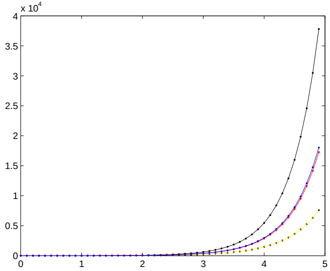


Figure:

Outline

1. Elementary
   - The Basic Problem That We Studied
   - Works with Matlab
2. Our Results
   - Primary Results
   - Improved Method
3. Others

Primary Ideas

Fact

Appropriate Changes of Step Length Helps Accuracy;
Areas Change Fast Deserved Small Steps,and vice versa

Corollary

Steps Are Anti-related with $|\frac{dy}{dx}|=|f(x,y)|$;

Fact

Steps could not be 0 or much huge;

Primary Ideas

Theorem

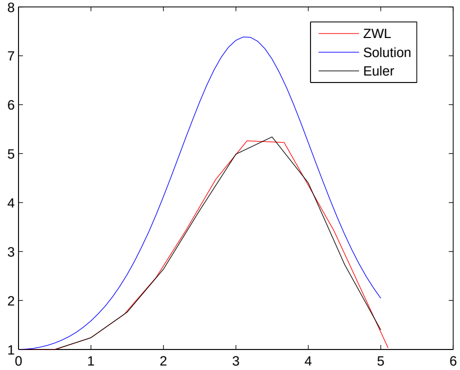
(Primary ZWL Method) $h_k = \frac{A}{|f(x_k, y_k)| + B} + C$, *A B C are Parameters*

Fact

Primary ZWL Method Compare to Euler Method

Primary Ideas

$$\begin{cases} y'(x) = y\sin x \\ y(0) = 1 \end{cases}$$



Why?

- $\left(\frac{A}{|f(x_k,y_k)|+B}+C\right)f(x_k,y_k)\approx\left(\frac{A}{|f(x_k,y_k)|}\right)f(x_k,y_k)=A$
- Fixed Steps of y but not x;
- Has the Same Order Error of Euler;

Why?

- $\left(\frac{A}{|f(x_k,y_k)|+B}+C\right)f(x_k,y_k)\approx\left(\frac{A}{|f(x_k,y_k)|}\right)f(x_k,y_k)=A$
- Fixed Steps of y but not x;
- Has the Same Order Error of Euler;

Why?

- $\left(\frac{A}{|f(x_k,y_k)|+B}+C\right)f(x_k,y_k)\approx\left(\frac{A}{|f(x_k,y_k)|}\right)f(x_k,y_k)=A$
- Fixed Steps of y but not x;
- Has the Same Order Error of Euler;

Outline

Improved ZWL Method

Fact

Steps Length Rely on Curvature

Curvature Formula

$$\rho = \frac{|y''|}{\sqrt{1+y'^2}^3}$$

Corollary

Steps Are Anti-related with Curvature

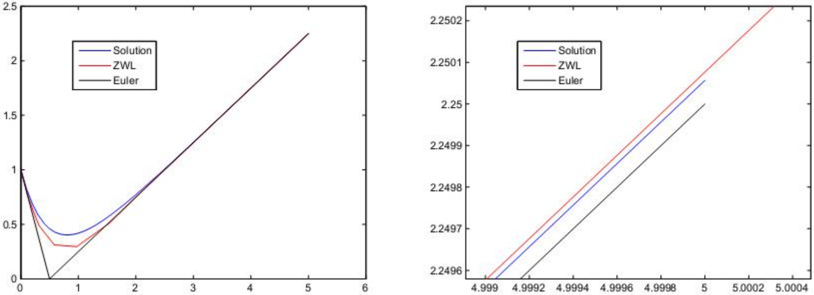
Theorem

$h_k = \frac{A}{\frac{|y''(x_k)|}{f(x_k,y_k)^3}+B} + C$ *where A B C are parameters.*

Comparison of ZWL Method and EM

$$\begin{cases} y'(x) = x - 2y \\ y(0) = 1 \end{cases}$$

Figure:



Convergence of ZWL Method

Proof.

Since we have

$$y_{k+1} = y_k + h_k f(x_k, y_k) = y_k + \left(\frac{A}{\frac{|y''(x_k)|}{f(x_k,y_k)^3} + B} + C \right) f(x_k, y_k)$$

For A B C are positive number, If we have $\frac{A}{B} + C < 1$,then

$$|y_{k+1} - y_k| \leq L^k |y_1 - y_0|$$

where,$0 < L \leq 1$.Hence,

$$\lim_{k\to\infty} |y_{k+1} - y_k| = 0$$

i.e. ZWL Method is Convergent. □

Division of Work

- Jiale.Wang: Give Matlab codes for several common methods; Participate to discover the primary ZWL methods;Give Matlab codes of question b);
- Tao Luo: GIve Matlab codes for Primary ZWL method and Taylor Series Method;Participate to discover the primary ZWL methods;Give the Matlab codes of question a)&c);
- Wenchao.Zhang: Write project article and this beamer; Do some researches on ZWL methods especially the Improved one;Give Matlab codes for ZWL method(Improved one).

Division of Work

- Jiale.Wang: Give Matlab codes for several common methods; Participate to discover the primary ZWL methods;Give Matlab codes of question b);
- Tao Luo: GIve Matlab codes for Primary ZWL method and Taylor Series Method;Participate to discover the primary ZWL methods;Give the Matlab codes of question a)&c);
- Wenchao.Zhang: Write project article and this beamer; Do some researchs on ZWL methods especially the Improved one;Give Matlab codes for ZWL method(Improved one).

Division of Work

- Jiale.Wang: Give Matlab codes for several common methods; Participate to discover the primary ZWL methods;Give Matlab codes of question b);
- Tao Luo: GIve Matlab codes for Primary ZWL method and Taylor Series Method;Participate to discover the primary ZWL methods;Give the Matlab codes of question a)&c);
- Wenchao.Zhang: Write project article and this beamer; Do some researches on ZWL methods especially the Improved one;Give Matlab codes for ZWL method(Improved one).

Codes List

- a.m
- airy.m
- b.m
- eular_method.m
- eular_PriZWL.m
- Simposonsrule.m
- ZWL.m

Summary

- Collect Several Common Methods for Numerical Solution of ODE.
- Make Matlab codes for these Methods.
- Discover a New Method For the Problem and it's Better than EM.

Summary

- Collect Several Common Methods for Numerical Solution of ODE.
- Make Matlab codes for these Methods.
- Discover a New Method For the Problem and it's Better than EM.

Summary

- Collect Several Common Methods for Numerical Solution of ODE.
- Make Matlab codes for these Methods.
- Discover a New Method For the Problem and it's Better than EM.

For Further Reading

Matlab codes for Ariy Equation

```
syms x y x0 y0 y1;
f = y*x ; n = 5 ;
x0 = 4;%initial condition
y0 = 1;%initial condition
y1 =2 ;%initial condition
c = f;
for i = 2:5
c(i) = diff(c(i-1),x) + diff(c(i-1),y)*c(1);
end
subs-c = subs(c,[x,y],[x0,y0]);
taylor = y0 +y1*(x-x0);
for i= 1:n
taylor = taylor + (subs-c(i)*(x-x0)∧(i+1))/factorial(i);
end
%
```

Figure: taylor