Math55: Differential Equations

Variable Step Size Differential Equation Solvers

Jason Brewer and George Little

Introduction

The purpose of developing numerical methods is to approximate the solution to the well posed initial value problem

$$\frac{dy}{dt} = f(t, y), a \leq t \leq b, y(a) = \alpha$$

Single Step Solvers

Taylor's Method

The first step to developing a numerical solver is to represent the solution to the Differential Equation as a Taylor power series

$$y(t) = y(a) + y'(a)(t-a) + \frac{y''(a)}{2!}(t-a)^2 + \cdots$$

where $y(t)$ represents the exact solution.

The Difference Equation

- After manipulating the power series, we replace the exact solutions with approximated solutions, giving us the equation

  $$w_{i+1} = w_i + h[f(t_i, w_i) + \frac{h}{2!}f'(t_i, w_i) + \cdots + \frac{h^{n-1}}{n!}f^{(n-1)}(t_i, w_i)]$$

  where w represents the approximated solution and h represents the step size.

The Difference Equation

- After manipulating the power series, we replace the exact solutions with approximated solutions, giving us the equation

$$w_{i+1} = w_i + h[f(t_i, w_i) + \frac{h}{2!} f'(t_i, w_i) + \cdots + \frac{h^{n-1}}{n!} f^{(n-1)}(t_i, w_i)]$$

  where w represents the approximated solution
  and h represents the step size

- This equation can now be put into the form

$$w_{i+1} = w_i + h \cdot T^{(n)}(y_i, w_i)$$

This equation is called the Taylor Method of order n

- Applying The Taylor method of order 1,2, and 4 to the initial value problem

$$\frac{dy}{dt} = -y + t + 1, 0 \leq t \leq 1, y(0) = 1$$

gives us the the following table (figure 1) in result

Analysis of Euler's Method

t	Exact value	Euler's Method	Error
0.0	1.000000	1.000000	0
0.1	1.004837	1.000000	4.837×10^{-3}
0.2	1.018730	1.010000	8.731×10^{-3}
0.3	1.040818	1.029000	1.182×10^{-2}
0.4	1.070320	1.056100	1.422×10^{-2}
0.5	1.106530	1.090490	1.604×10^{-2}
0.6	1.148811	1.131441	1.737×10^{-2}
0.7	1.196585	1.178297	1.829×10^{-2}
0.8	1.249328	1.230467	1.887×10^{-2}
0.9	1.306569	1.287420	1.915×10^{-2}
1.0	1.367879	1.348678	1.920×10^{-2}

Analysis of Taylor's Method Second Order

t	Exact value	Taylor's Method Order Two	Error
0.0	1.000000	1.000000	0
0.1	1.004837	1.005000	1.626×10^{-4}
0.2	1.018730	1.019025	2.942×10^{-4}
0.3	1.040818	1.041218	3.998×10^{-4}
0.4	1.070320	1.070802	4.820×10^{-4}
0.5	1.106530	1.107076	5.453×10^{-4}
0.6	1.148811	1.149404	5.924×10^{-4}
0.7	1.196585	1.197211	6.257×10^{-4}
0.8	1.249328	1.249976	6.470×10^{-4}
0.9	1.306569	1.307227	6.583×10^{-4}
1.0	1.367879	1.368541	6.616×10^{-4}

Analysis of Taylor's Method Fourth Order

t	Exact value	Taylor's Method Order Four	Error
0.0	1.000000	1.000000	0
0.1	1.004837	1.004837	8.200×10^{-8}
0.2	1.018730	1.018730	1.483×10^{-7}
0.3	1.040818	1.040818	2.013×10^{-7}
0.4	1.070320	1.070320	2.429×10^{-7}
0.5	1.106530	1.106530	2.747×10^{-7}
0.6	1.148811	1.148811	2.983×10^{-7}
0.7	1.196585	1.196585	3.149×10^{-7}
0.8	1.249328	1.249329	3.256×10^{-7}
0.9	1.306569	1.306569	3.125×10^{-7}
1.0	1.367879	1.367879	3.332×10^{-7}

Error Control

- As can be seen from the table in figure 1, as the order of the method increases, the error produced by Taylor's method decreases. The price for the decrease in error, however, is an increase of the number of calculations.
- For Taylor's method, this increase means the calculation of higher order derivatives

Rung-Kutta Methods

Because the higher order Taylor's Methods require the calculation of higher order derivatives, they are rarely ever used in practice. The problem of eliminating the upper order derivatives while keeping the higher order accuracy and error control is solved by converting the Taylor Methods into Rung-Kutta methods.

The Second order Taylor Method can be rewritten into a second order Rung-Kutta Method

$$w_{i+1} = w_i + h[f(t_i + \frac{h}{2}, w_i + \frac{h}{2} f(t_i, w_i))]$$

This particular RK2 Method is called the Midpoint Method and can be rewritten into the form commonly seen in text books.

$$\begin{aligned} w_0 &= \alpha \\ k_1 &= f(t_i, w_i) \\ k_2 &= f(t_i + \frac{h}{2}, w_i + \frac{h}{2} k_1) \\ w_{i+1} &= w_i + hk_2 \end{aligned}$$

The Fourth order Taylor method can be rewritten into a fourth order Rung-Kutta Method

$$\begin{aligned}
w_0 &= \alpha \\
k_1 &= hf(t_i, w_i) \\
k_2 &= hf(t_i + \frac{h}{2}, w_i + \frac{1}{2}k_1) \\
k_3 &= hf(t_i + \frac{h}{2}, w_i + \frac{1}{2}k_2) \\
k_4 &= hf(t_i, w_i + k_3) \\
w_{i+1} &= w_i + \frac{1}{6}(k_1 + 2k_2 + 2k_3 + k_4)
\end{aligned}$$

At this point it is important to note that there are many different Rung-Kutta Methods of n order. A method with a different set of coefficients is a new method.

Error Control and Variable Step Size

The main concern with numerical solvers is the error made when they approximate a solution. The second concern is the number of computations that must be performed. Both of these can be addressed by creating solvers that use a variable step size in order to keep the error within a specified tolerance. By using the largest step size allowable while keeping the error within a tolerance, the error made is reduced.

Determining the Error Made per Step

The way to keep the error under control is to determine the error made at each step. A common way to do this is to use two solvers of orders n and $n+1$. If you will remember Taylor's Theorem

$$y(t) = y(a) + y'(a)(t-a) + \frac{y''(a)}{2!}(t-a)^2 + \cdots + \frac{y^{(n)}(a)}{n!}(t-a)^n$$
$$+\frac{y^{(n+1)}(a)}{(n+1)!}(t-a)^{n+1} + \cdots$$

Determining the Error Made per Step

The way to keep the error under control is to determine the error made at each step. A common way to do this is to use two solvers of orders n and $n+1$. If you will remember Taylor's Theorem

$$y(t) = y(a) + y'(a)(t-a) + \frac{y''(a)}{2!}(t-a)^2 + \cdots + \frac{y^{(n)}(a)}{n!}(t-a)^n + \frac{y^{(n+1)}(a)}{(n+1)!}(t-a)^{n+1} + \cdots$$

Any approximations made of order n will have an error no larger than the value of the $n+1$ term. This leads us to take the difference of our two solvers to find the value the the error term.

Since the downside of using two distinct methods is a dramatic increase in computations, another method is typically used. An example of this method is the Rung-Kutta-Fehlberg Algorithm. The Rung-Kutta-Fehlberg Algorithm shown here combines Rung-Kutta methods of order four and order five into one algorithm. Doing this reduces the number of computations made while returning the same result.

Rung-Kutta-Fehlberg Algorithm

1. Set $t = a$, $w = \alpha$, $h = hmax$, FLAG$= 1$
   OUTPUT(t,w)
2. While $(FLAG = 1)$ do Steps 3-11
3. Set

$$K_1 = hf(t, w)$$

$$K_2 = hf(t + \frac{1}{4}h, w + \frac{1}{4}K_1$$

$$K_3 = hf(t + \frac{3}{8}h, w + \frac{3}{32}K_1 + \frac{9}{32}K_2$$

$$K_4 = hf(t + \frac{12}{13}h, w + \frac{1932}{2197}K_1 - \frac{7200}{2197}K_2 + \frac{7296}{2197}K_3$$

$$K_5 = hf(t + h, w + \frac{439}{216}K_1 - 8K_2 + \frac{3680}{513}K_3 - \frac{845}{4104}K_4$$

$$K_6 = hf(t + \frac{1}{2}h, w - \frac{8}{27}K_1 + 2K_2 - \frac{3544}{2565}K_3 + \frac{1859}{4104}K_4 - \frac{11}{40}K_5$$

```
4. Set $R = \frac{\left| \frac{1}{360}K_1 - \frac{128}{4275}K_3 - \frac{2197}{75240}K_4 + \frac{1}{50}K_5 + \frac{2}{55}K_6 \right|}{h}$

5. Set $\delta = 0.84(\frac{TOL}{R})^{\frac{1}{4}}$

6. If $(R \leq TOL)$ then do Steps 7 and 8

7. Set $t = t + h$, $w = w + \frac{25}{216}K_1 + \frac{1408}{2565}K_3 + \frac{2197}{4104}K_4 - \frac{1}{5}K_5$

8. OUTPUT(t,w,h)

9. If $(\delta \leq 0.1)$ then set $h = 0.1h$
   else If $(\delta \geq 4)$ then set $h = 4h$
   else set $h = \delta h$

10. If $(h > hmax)$ then set $h = hmax$

11. If $(t \geq b)$ then set $FLAG = 0$
    else If $(t + h > b)$ then set $h = b - t$
```

else If $(h < hmin)$ then set $FLAG = 0$ OUTPUT(”Minimum h exceeded’)

Summary

From the humble beginnings of Euler's method, numerical solvers started relatively simple and have evolved into the more complex higher order Taylor methods and into the efficient Rung-Kutta methods. And the search for more efficient and accurate methods have lead to the more complicated variable step solvers.

References

1. Arnold, Dave *Technical Help and Advice*
2. Brown, Greg *Program Inspiration*
3. Burden, Richard L., J. Douglas Faires **Numerical Analysis: Fourth Edition**, PWS-KENT Publishing Company, Boston: 1993
4. Dormand, John R., **Numerical Methods for Differential Equations: A Computational Approach**CRC Press, Boca raton: 1996
5. Romero, Chris *C++ Technical Help and Advice*