Ordinary Differential equations continued

Eugeniy E. Mikhailov

The College of William & Mary

Lecture 20

Notes

Recall Euler's method

$$\vec{y}' = \vec{f}(x, \vec{y})$$

There is an exact way to write the solution

$$\vec{y}(x) = \int_{x_0}^{x} \vec{f}(x, \vec{y}) dx$$

However for small interval of $x, x+h$ we assume that $\vec{f}(x, \vec{y})$ is constant

$$\vec{y}(x_{i+1}) = \vec{y}(x_i + h) = \vec{y}(x_i) + \vec{f}(x_i, \vec{y}_i)h + \mathcal{O}(h)$$

Notes

The second-order Runge-Kutta method

Using multi-variable calculus and Taylor expansion, it can be shown

$$\begin{aligned} \vec{y}(x_{i+1}) &= \vec{y}(x_i + h) = \\ &= \vec{y}(x_i) + C_0 \vec{f}(x_i, \vec{y}_i)h + C_1 \vec{f}(x_i + ph, \vec{y}_i + qh\vec{f}(x_i, \vec{y}_i))h + \mathcal{O}(h^3) \end{aligned}$$

When

$$C_0 + C_1 = 1, \quad C_1 p = 1/2, \quad C_1 q = 1/2$$

There is a lot of possible choices of parameters C_0, C_1, p, and q which has no advantage over the others.
One of popular choices is $C_0 = 0$, $C_1 = 1$, $p = 1/2$, and $q = 1/2$ for

Modified Euler's method or midpoint method (error $\mathcal{O}(h^3)$)

$$\begin{aligned} k_1 &= h\vec{f}(x_i, \vec{y}_i) \\ k_2 &= h\vec{f}(x_i + \frac{h}{2}, \vec{y}_i + \frac{1}{2}k_1) \\ \vec{y}(x_i + h) &= \vec{y}_i + k_2 \end{aligned}$$

Notes

The forth-order Runge-Kutta method

truncation error $\mathcal{O}(h^5)$

$$\begin{aligned} k_1 &= h\vec{f}(x_i, \vec{y}_i) \\ k_2 &= h\vec{f}(x_i + \frac{h}{2}, \vec{y}_i + \frac{1}{2}k_1) \\ k_3 &= h\vec{f}(x_i + \frac{h}{2}, \vec{y}_i + \frac{1}{2}k_2) \\ k_4 &= h\vec{f}(x_i + h, \vec{y}_i + k_3) \\ \vec{y}(x_i + h) &= \vec{y}_i + \frac{1}{6}(k_1 + 2k_2 + 2k_3 + k_4) \end{aligned}$$

Notes

Matlab ODEs solvers

Have a look in help files for ODEs in particular

- `ode45` - adaptive explicit 4th order Runge-Kutta method (good default method)
- `ode23` - adaptive explicit 2nd order Runge-Kutta method
- `ode113` - "stiff" problem solver
- and others

Adaptive stands for no need to chose '*h*', algorithm will do it by itself. But do remember the rule of not trusting computers.
Also run `odeexamples` to see some of the demos for ODEs solvers

Notes

Notes

Notes

Notes