Ordinary Differential equations continued

Eugeni E. Mikhailov
The College of William & Mary

Lecture 21
Recall Euler’s method

Let’s for simplicity consider simple first order ODE (notice lack of vector)

\[ y' = f(x, y) \]

There is an exact way to write the solution

\[ y(x) = \int_{x_0}^{x} f(x, y) \, dx \]

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

\[ \tilde{y}(x_{i+1}) = \tilde{y}(x_i + h) = \tilde{y}(x_i) + \tilde{f}(x_i, \tilde{y}_i)h + O(h) \]
The second-order Runge-Kutta method

Using multi-variable calculus and Taylor expansion, it can be shown
\[ \vec{y}(x_{i+1}) = \vec{y}(x_i + h) = \vec{y}(x_i) + C_0 \vec{f}(x_i, \vec{y}_i) h + C_1 \vec{f}(x_i + ph, \vec{y}_i + qh \vec{f}(x_i, \vec{y}_i)) h + O(h^3) \]

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 \(O(h^3)\))

\[ k_1 = h \vec{f}(x_i, \vec{y}_i) \]
\[ k_2 = h \vec{f}(x_i + \frac{h}{2}, \vec{y}_i + \frac{1}{2} k_1) \]

\[ \vec{y}(x_i + h) = \vec{y}_i + k_2 \]
The fourth-order Runge-Kutta method

truncation error $O(h^5)$

\[
\begin{align*}
k_1 &= hf(x_i, \tilde{y}_i) \\
k_2 &= hf(x_i + \frac{h}{2}, \tilde{y}_i + \frac{1}{2}k_1) \\
k_3 &= hf(x_i + \frac{h}{2}, \tilde{y}_i + \frac{1}{2}k_2) \\
k_4 &= hf(x_i + h, \tilde{y}_i + k_3) \\
\tilde{y}(x_i + h) &= \tilde{y}_i + \frac{1}{6}(k_1 + 2k_2 + 2k_3 + k_4)
\end{align*}
\]
Matlab ODEs solver

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 other

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