Numerical Methods

This Lecture gives a very brief overview of numerical methods that computers use when they plot solution curves for differential equations. For more details you can read Sections 2.7 and 8.1-4 in the Boyce & DiPrima textbook or Chapter 8 of the Matlab book or Chapter 7 of the Maple book, but this is optional.

Summary.

(1) Euler's Method. Simple. Error proportional to h^2. (h is the step size.)
(2) Improved Euler's Method. Less simple. Error Proportional to h^3.
(3) Modified Euler's Method. Error Proportional to h^3.
(4) Runge-Kutta. Complicated. Error Proportional to h^5.
(5) ode45. A multistep method. Step size varies. Used in most CAS's.

Euler's Method.

Suppose y is a function of t such that $\frac{dy}{dt} = f(t, y)$ and $y(t_0) = y_0$, where f is given. Let $t_1 > t_0$. We want to estimate $y(t_1)$ without actually solving for $y(t)$ in general. We will do this by constructing a tangent line to the solution curve $y(t)$ at the point (t_0, y_0) using $y'(t_0) = f(t_0, y_0)$ as the slope. This tangent line is

$$y - y_0 = f(t_0, y_0)(t - t_0).$$

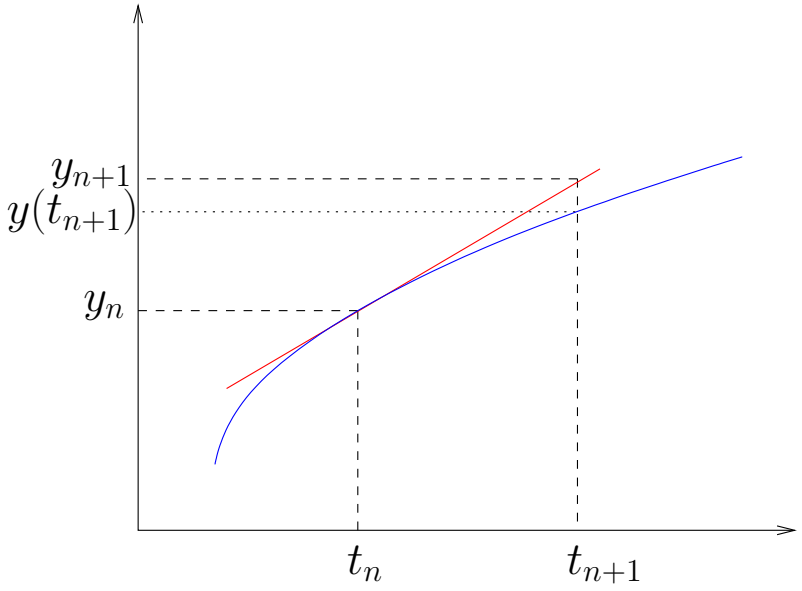
We evaluate is at $t = t_1$ and solve for y, which we now call y_1.

$$y_1 = y_0 + f(t_0, y_0)(t_1 - t_0).$$

We can continue this process for $t_2 > t_1$, $t_3 > t_2$, and so on. If we assume the step size is always the same, $h = t_{n+1} - t_n$, we get

$$y_{n+1} = y_n + h\, f(t_n, y_n).$$

Then $y(t_n) \approx y_n$ for each n. The magnitude of the difference is called the **error**. There are means to estimate the error. See textbook.



Improved Euler's Method. [Featured in the movie Hidden Figures: `https://www.youtube.com/watch?v=RK8xHq6dfAo`.]

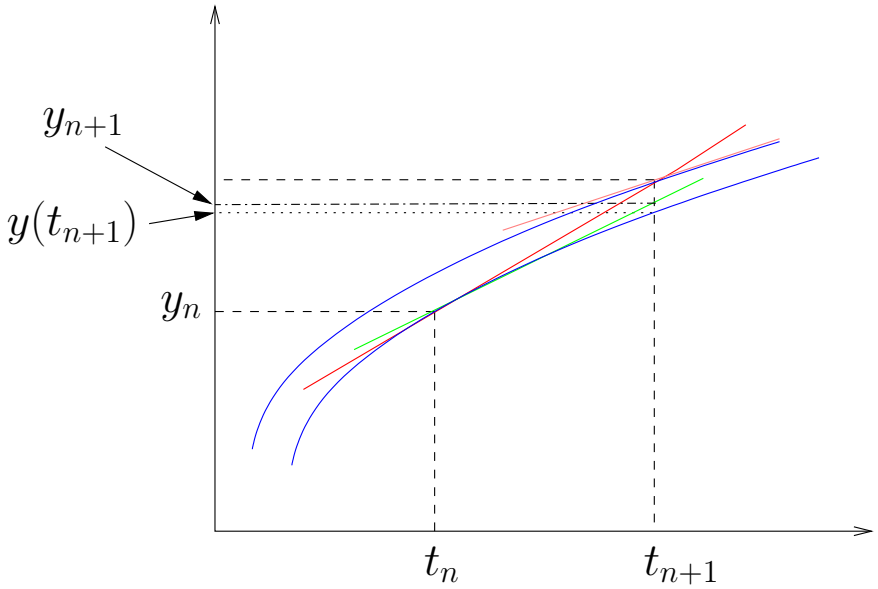
As we move along a solution curve, $y(t)$, the slope changes and thus error builds up. One idea is to measure the slope at t_{n+1} and average it with the slope at t_n.

$$y(t_{n+1}) \approx y_n + h\,\frac{y'(t_n) + y'(t_{n+1})}{2} \qquad ??$$

The problem is we cannot evaluate $y'(t_{n+1}) = f(t_{n+1}, y_{n+1})$ because we have not computed y_{n+1}. To get around this, at least approximately, we use Euler's Method to estimate y_{n+1}. This gives

$$y_{n+1} = y_n + h\,\frac{f(t_n, y_n) + f(t_{n+1}, y_n + hf(t_n, y_n))}{2}.$$

The error from each step in Euler's Method is proportional to h^2, while for the Improved Euler's Method it is proportional to h^3. See Section 8.2, #14.



The slope of the green line is the average of the slopes of the red and pink lines.

Modified Euler's Method. Here an estimate of the slope of $y(t)$ midway between t_n and t_{n+1} is used. This gives

$$y_{n+1} = y_n + h \cdot f\left(t_n + \frac{h}{2}, y_n + \frac{h}{2} f(t_n, y_n)\right).$$

See Section 8.2 #22.

Runge-Kutta. We use

$$y_{n+1} = y_n + h \cdot \left(\frac{m_1 + 2m_2 + 2m_3 + m_4}{6}\right)$$

where

$$\begin{aligned} m_1 &= f(t_n, y_n) \\ m_2 &= f(t_n + h/2, y_n + hm_1/2) \\ m_3 &= f(t_n + h/2, y_n + hm_2/2) \\ m_4 &= f(t_{n+1}, y_n + hm_3) \end{aligned}$$

See Section 8.3 of the textbook.

ode45. This allows for the step size to change as we follow the solution curve. Roughly, if the concavity is large we need smaller steps and if the curve flattens out we can take bigger steps. This is usually the default option in Computer Algebra Systems. For more on this see the *Differential Equations with Matlab* textbook.