# Euler's Method

---

- Consider the differential equation

$$\frac{dy}{dx} = f(x, y). \tag{0.1}$$

  - ◇ If $y(x)$ is a solution, then

$$y(x_n + h) = y(x_n) + hy'(x_n) + O(h^2)$$

    is the Taylor series expansion of $y(x_n + h)$ near $x_n$.

  - ◇ If we already know that the solution at $x_n$ is approximately $y(x_n) \approx y_n$, then the truncated Taylor series suggests that

$$y(x_n + h) \approx y_{n+1} = y_n + hf(x_n, y_n).$$

- To obtain the solution to equation (0.1) at $x = b$ with initial condition $y(a) = y_0$, we do

  1. Select a step size $h = \frac{b-a}{N}$ where $N$ is a specified positive integer.
  2. Initialize $x_0 = a$.
  3. Do the loop for $n = 0, \ldots, N-1$

$$\begin{aligned} x_{n+1} &= x_n + h \\ y_{n+1} &= y_n + hf(x_n, y_n). \end{aligned}$$

  - ◇ You must provide two essential pieces of information: the stepsize and the initial condition.
  - ◇ The larger the stepsize is, the larger the *truncation error*.
  - ◇ The smaller the stepsize is, the more work needs to be done (and possibly the more built-up of floating-point errors.)

# Examples Using Maple and Matlab

---

- Consider the initial value problem

$$\begin{aligned} y' &= \frac{y}{x} + x^2 \sin x \\ y(\frac{\pi}{2}) &= 0. \end{aligned}$$

- On Maple (using 4-th order Runge-Kutta method):

$with(DEtools);$
$eq := diff(y(x), x) - y(x)/x - x^2 * sin(x);$
$DEplot(eq, y(x), x = Pi/2..50, [[y(Pi/2) = 0]], stepsize = .5);$

- On Matlab (using ODE45)

  - ◇ You first prepare a user-supplied problem description. For example, we call it *problem34.m*, which should of the form

```
function yprime = problem34(x,y);
yprime &=& y/x+x^2*sin(x);
```

  - ◇ From within the MATLAB session, type in the following commands:

```
t0 = pi/2;
y0=0;
tspan=[1,50];
options=odeset('RelTol',1.e-6,'AbsTol',1.e-6);
[t,y] = ode23('problem34',tspan,y0,options);
plot(t,y)
```

- Note that these packages, by default, are using higher order numerical methods as a black box. We don't get to see how these methods are programmed. These examples however should illustrate how the methods should be called.

- A prototype program of Euler's method written in MATLAB:

```
function [tout,yout] = myeuler(ypfun,t0,tfinal,y0,N);
h = (tfinal-t0)/N;
tout=t0;yout=y0;
tc=t0;yc=y0;
for step = 1:N
    yc = yc + h*feval(ypfun,tc,yc);
    tc = tc + h;
    tout = [tout;tc];
    yout = [yout;yc];
end
```

  - ◇ Note that this is a fixed stepsize method. The stepsize is determined internally by the integer $N$. The error is *not* directly determined by the user.
  - ◇ A call to the method should look like

```
t0 = pi/2;
y0=0;
tfinal = 50;
N = 500;
[t,y] = myeuler('problem34',t0,tfinal,y0,N); \\
plot(t,y)
```