Week 8 Part 3: Euler's Method

Numerical solutions to ODEs are all about approximating a derivative and using that to approximate the solution. We already did that in Week 7, so lets use those approximations.

Consider, $y' = f(t, y)$. Recall

$$y' = \frac{y(t+h) - y(t)}{h}$$

Let $t = t_n$ and $t + h = t_{n+1}$. Further, we write $y(t_n) = y_n$ and $y(t_{n+1}) = y_{n+1}$. Then

$$\frac{y_{n+1} - y_n}{h} \approx f(t_n, y_n),$$

and therefore

$$y_{n+1} = y_n + hf(t_n, y_n); \qquad y_0 = y(t_0) \tag{1}$$

This is called the Forward Euler method because we use the forward difference.

We can also do a backward approximation.

$$y' = \frac{y(t) - y(t-h)}{h}$$

Let $t = t_{n+1}$ and $t + h = t_n$. Further, we write $y(t_n) = y_n$ and $y(t_{n+1}) = y_{n+1}$. Then

$$\frac{y_{n+1} - y_n}{h} \approx f(t_{n+1}, y_{n+1}),$$

and therefore

$$y_{n+1} = y_n + hf(t_{n+1}, y_{n+1}); \qquad y_0 = y(t_0) \tag{2}$$

This is called the Backward Euler method because we use the forward difference. Notice here we have to solve for y_{n+1} since it is implicitly in $f()$.

If you didn't get a chance to take a look a the Week 7 lectures, here is an alternate, but equivalent derivation of the Forward Euler's method:

$$y'(t) = f(t, y) \Rightarrow f(t, y) \approx \frac{\Delta y}{\Delta t} = \frac{y - y_0}{t - t_0}.$$

Now lets evaluate f at t_1, y_1, then we get,

$$f(t_1, y_1) \approx \frac{y_1 - y_0}{t_1 - t_0} \Rightarrow y_1 - y_0 \approx (t_1 - t_0)f(t_0, y_0) \Rightarrow y_1 \approx y_0 + (t_1 - t_0)f(t_0, y_0).$$

Look at that! We just developed a formula to approximate y at t_1 by using the information we had for the system at t_0. If we can approximate the data at t_1 by using the previous time (i.e. t_0), why can't we do this for any time? That is we can approximate y at t_{n+1} via the formula, $y_{n+1} \approx y_n + \Delta t f(t_n, y_n)$. The standard way to write this however is with, $h = \Delta t$, basically a renaming and we usually use $y_0 = y(t_0)$, i.e. the initial condition, and we also drop the $\approx$ and us $=$. So our general formula is,

$$y_{n+1} = y_n + hf(t_n, y_n); \; y_0 = y(t_0). \tag{3}$$

When debugging your codes use the following example, and make sure your values are close to mine. Your values might be ever so slightly off, but not more than say `1e-8`.

(1) $f(t,y) = 3 + t - y$, which gives us the equation $y_{n+1} = y_n + h \cdot (3 + t_n - y_n)$ where $y_0 = 1$.

(a) Here we have $h = 0.1$, so we have the following t's. We get them just by starting at t_0 and incrementing. $t_0 = 0$, $t_1 = 0.1$, $t_2 = 0.2$, $t_3 = 0.3$, $t_4 = 0.4$. Then we have, $y_1 = y_0 + h \cdot (3 + t_0 - y_0) = 1 + (0.1)(3 + 0 - 1) = 1.2$, $y_2 = y_1 + h \cdot (3 + t_1 - y_1) = 1.39$, $y_3 = y_2 + h \cdot (3 + t_2 - y_2) = 1.571$, and $y_4 = y_3 + h \cdot (3 + t_3 - y_3) = 1.7439$. Lets put this in a table to make it look pretty,

n	0	1	2	3	4
t_n	0	0.1	0.2	0.3	0.4
y_n	1	1.2	1.39	1.571	1.7439

(b) Hopefully part a gave you a good idea of how we do these problems, so I'll just give the table of values I received when running my code on matlab (remember $h = 0.05$):

n	0	1	2	3	4	5	6	7	8
t_n	0	0.05	0.1	0.15	0.2	0.25	0.3	0.35	0.4
y_n	1	1.1	1.1975	1.2926	1.3855	1.4762	1.5649	1.6517	1.7366

(c) Here $h = 0.025$,

n	0	1	2	3	4	5	6	7	8
t	0	0.025	0.05	0.075	0.1	0.125	0.15	0.175	0.2
y	1	1.05	1.0994	1.1481	1.1963	1.2439	1.2909	1.3374	1.3833

n	9	10	11	12	13	14	15	16
t	0.225	0.25	0.275	0.3	0.325	0.35	0.375	0.4
y	1.4288	1.4737	1.5181	1.562	1.6055	1.6484	1.6910	1.7331

(d) Next we solve the equation via integrating factors to get $y = 2 + t - e^{-t}$, and calculating the points gives us the following comparison,

h	$t =$	0.1	0.2	0.3	0.4
0.1	$y(t) =$	1.2	1.39	1.571	1.7439
0.05	$y(t) =$	1.1975	1.3855	1.5649	1.7366
0.025	$y(t) =$	1.1963	1.3833	1.562	1.7331
Exact	$y(t) =$	1.19516	1.38127	1.55918	1.72968