Assignment 7

Mathematics: Introduction to Differential Equation, Taylor Series
Programming: Epsilon or eps or ϵ of machine

A differential equation is a relationship between a quantity and its derivative, or rate of change. So, first we review derivatives. You remember from calculus 1 that $f'(x) = \lim_{h\to 0} \dfrac{f(x+h)-f(x)}{h}$, and you learned a variety of techniques for finding $f'(x)$. If you have learned anything in this class it is that life (math) on a computer is not a simple extension of math on blackboard. With that in mind, it is better to check the notion of derivative in practice and see what happens. Pick a function, say $f(x) = \sin(x)$, pick a point at which to compute the derivative, say $\pi/6$ (remember derivative formulas for trig functions are good only if you use radians), and try to evaluate the derivative. Theoretically we expect to get $\cos(\pi/6)$ or $\sqrt{3}/2$. (By the way, can you prove this? If not, and you are a math major, call the Ghost Busters! $\cdots$ But I digress.) So how do we calculate the derivative numerically, well you approximate it as a difference quotient, just copying $\dfrac{f(x+h)-f(x)}{h}$, and evaluating it for different values of h. The snag is how do you do $\lim_{h\to 0}$? That is where the trap is. You will say let me take smaller and smaller values of h, I am supposed to come infinitely close to the derivative, that is what the definition says. Here we part ways. To see, just try it. Do a do loop, `for i=1:20 h=`10^{-i} `...` and print the error to full accuracy. (Some of you are still calculating to 2 decimal places. Wake up! We have 16 decimal places on standard Matlab or a typical software.) Which value of h gives the best answer? Let's call this $h^\star$. Definitely $h^\star$ is not the smallest h value you used.

So what gives? There are two issues we need to learn. Here is an overly brief explanation. First issue: a floating number on computer has finite precision (other than certain numbers, as in $M/2^n$, which are exactly reproduced). In high school you used to write $1/3 = 0.33$, of course that is not correct, you need an infinite string of threes. Well a typical computer is like you but it cuts numbers off at the 16-th decimal place, almost. A typical floating number x on the computer memory has an *error of representation* that is about $\epsilon \cdot x$, where ϵ is software/hardware dependent and is about 10^{-16} for us, indicating that the information past the 16-th decimal place is lost. You can type `eps` at Matlab to see its exact value. Or you can type `>>help eps`. By definition ϵ is the smallest number such that $1+\epsilon > 1$ on your machine. Type $1+10^{-20} > 1$ and $1+10^{-12} > 1$ to see the difference.

Second issue: If you are my favorite Martian you remember from calculus 2 that $f(x+h) = f(x)+f'(x)h+f''(x)h^2/2+\cdots$. So $\dfrac{f(x+h)-f(x)}{h} - f'(x) = f''(x)h/2+\cdots$. So the formula you use for the derivative has a mathematical *truncation error* of about $f''(x)h/2$. More over your difference quotient has an error of about $2f(x)\epsilon/h$, (you have to add the errors, they do not cancel, and for a generic h values, not too small, this is a good approximation, $\cdots$, we are skipping some details here) so your total error is $T(h) = \dfrac{2|f(x)|\epsilon}{h} + \dfrac{h|f''(x)|}{2}$.

Now here is a standard problem from calculus 1, for what value of h is the total error $T(h)$ smallest? Compute this and compare it to the $h^\star$ you found above.