grade: check (upgrade to check + by addressing comments in blue)

Activity 3

Comparing Floating Pt #s

1) This file gives $x_1 = \cos(\pi/4)/\sin(\pi/4) = 1$ and $x_2 = 1$. Our if statement says if our x_1 and x_2 are both represented in the computer the same way, it will output the phrase "x_1 and x_2 are equal"! If they are not equal, the code returns "x1 and x2 are not equal"

2) The issue for the code must be that cos(pi/4) is not exactly represented as sin(π/4). Both should give √2/2 or, .7071. But, because these are not represented in the computer exactly the same, we do not get cos(π/2)/sin(π/2) to equal exactly 1.0. Instead, we probably have something very close to 1.

After adding

```
{
cout<<fixed<<setprecision(16)<<x1<<"  "<<x2<<endl;
}
```

I could see that x1 = .9999999403953552
and x2 = 1.0000000000000000

which is what I expected.

3) Instead of checking that they are exactly equal, I'd compare them as a fraction. If the fraction is within a certain (very close) range from 1.0, we could say they were "very close to being equal", or "basically equal"

Numerical Derivatives: Pass 1

1) I'm confused on the "function prototypes" section does and why it's written the way it is written.

The relative errors are going to be printed.

a. log10(h)

b. log10(fabs((diff_fd-exact)/exact))

c. log10(fabs((diff_cd-exact)/exact))

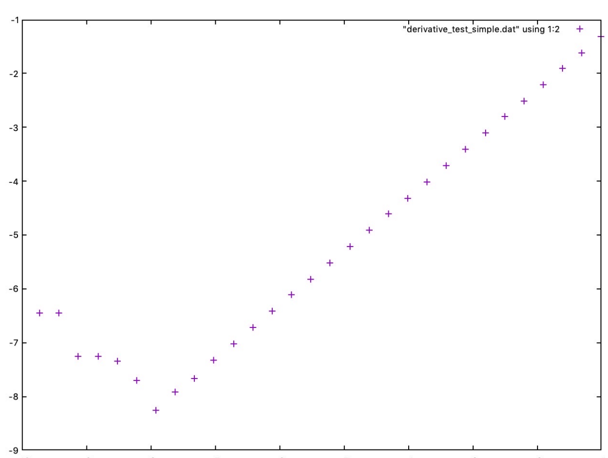
I'm now also confused on why we need all of the other parts that come after the "main program". Why don't they go before?

2) #include <cmath>

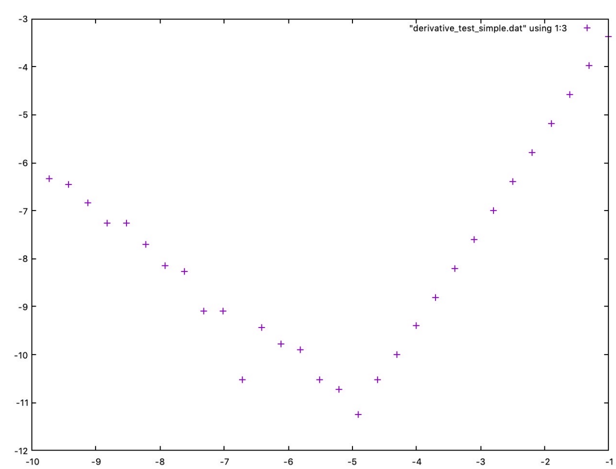
- my Gnuplot worked

3)

forward Diff.



central diff.



4) For both, m was equal to 1 to the right of the optimal h value, and negative/not linear to the left of the opt. h value. Forward is better. It allows for a smaller opt. h, which will give better results.

5) solve for h opt using 3.9 and 3.13

D_f | D_c | $\epsilon_m = 10^{-16}$

$f = e^{-\alpha x}$ $\alpha = 1$

$h = \left(\frac{2\epsilon_m}{f^{(2)}}\right)^{1/2}$ $\qquad$ $h = \left(\frac{24\epsilon_m}{f^3}\right)^{1/3}$

$h = 2.3 \cdot 10^{-8}$ $\qquad$ $h = -1.86 \cdot 10^{-5}$ $\qquad$ $f' = -e^{-x}$

when $x = 1$

$f'' = e^{-x}$

$f''' = -e^{-x}$

6) Yes, Em would change. Garrit said "$6 \cdot 10^{-3}$ = central 10^{-4} forward"

ϵ_m: 10^{-6}, which is a much larger error.

the optimal h would be larger and therefor have a worn approximation for the derivative.

The slope does not change. Can you see why?
(Hint: the value of epsilon plays no role in determining the slope, though it will shift the curves up for single precision)

Makefiles for multiple

1) Yes, I would guess that it is faster to edit the separate file because you'd only edit one spot. In routines, there'd be more edits to make if you try editing in many places.

2) Yes, we have 3 columns. Each is a different method for approximating the integral for e^x for various N values.

3) ✓

4) The trapezoid error plot scaled linearly for most N values, Simpson stopped scaling linearly after N 13 values and the Gaussian never scaled linearly.

5) I couldn't figure this out

Hint: What is Log(a)? What is Log(a^2)? What is Log(a^3)? etc..

Finding the Approx. Error

1) ✓

2) trapezoid: $m \approx -2$

Simpson: $m \approx -4$

3) Yes, my data and slopes are consistent with the notes

What about the slopes in the roundoff region?