PHYS 512
Fall 2022

student: Patrick H. (260931238)
prof: Jon Sievers

Assignment 1

1

Evaluate our function f @ $(x \pm \delta)$ & $(x \pm 2\delta)$

a) What should our estimate of the first derivative at x be?

Use Taylor series expansion @ $(x \pm \delta)$ & $(x \pm 2\delta)$:

$$f(x \pm \delta) = f(x) \pm f^{(1)}(x)\delta + \frac{1}{2} f^{(2)}(x)\delta^2 \pm \frac{1}{6} f^{(3)}(x)\delta^3 + \frac{1}{24} f^{(4)}(x)\delta^4 \pm \frac{1}{120} f^{(5)}(x)\delta^5 + \mathcal{O}(\delta^6)$$

$$f(x \pm 2\delta) = f(x) \pm 2f^{(1)}(x)\delta + 4f^{(2)}(x)\delta^2 \pm \frac{4}{3} f^{(3)}(x)\delta^3 + \frac{2}{3} f^{(4)}(x)\delta^4 \pm \frac{4}{15} f^{(5)}(x)\delta^5 + \mathcal{O}(\delta^6)$$

From numerical recipes § 5.7 p. 230, we have a symmetrized form

$$f'(x) \approx \frac{f(x+h) - f(x-h)}{2h}$$

For $h = \delta$: $$f_1'(x) \approx \frac{f(x+\delta) - f(x-\delta)}{2\delta} \qquad (1)$$

for $h = 2\delta$: $$f_2'(x) \approx \frac{f(x+2\delta) - f(x-2\delta)}{4\delta} \qquad (2)$$

① : $2\delta f_1^{(1)}(x) \approx f(x+\delta) - f(x-\delta)$

$$= \cancel{f(x)} + f^{(1)}(x)\delta + \cancel{\frac{1}{2} f^{(2)}(x)\delta^2} + \frac{1}{6} f^{(3)}(x)\delta^3 + \cancel{\frac{1}{24} f^{(4)}(x)\delta^4} + \frac{1}{120} f^{(5)}(x)\delta^5$$

$$- \left(\cancel{f(x)} - f^{(1)}(x)\delta + \cancel{\frac{1}{2} f^{(2)}(x)\delta^2} - \frac{1}{6} f^{(3)}(x)\delta^3 + \cancel{\frac{1}{24} f^{(4)}(x)\delta^4} - \frac{1}{120} f^{(5)}(x)\delta^5\right)$$

$$= 2f^{(1)}(x)\delta + \frac{1}{3} f^{(3)}(x)\delta^3 + \frac{1}{60} f^{(5)}(x)\delta^5$$

$$\Rightarrow 2\delta f_1^{(1)}(x) \approx 2f^{(1)}(x)\delta + \frac{1}{3} f^{(3)}(x)\delta^3 + \frac{1}{60} f^{(5)}(x)\delta^5 \qquad ③$$

② : $4\delta f_1^{(1)}(x) \approx f(x+2\delta) - f(x-2\delta)$

Much like with $h = \delta$, here the even n^{th}-derivative terms cancel out; the odd n^{th}-derivative terms are doubled

$$\Rightarrow 4\delta f_2^{(1)}(x) \approx 4f^{(1)}(x)\delta + \frac{8}{3} f^{(3)}(x)\delta^3 + \frac{8}{15} f^{(5)}(x)\delta^5 \qquad ④$$

How to combine ③ & ④ to cancel δ^3 terms?
From their coefficients, take:

$$8(③) - (④) \Rightarrow 8\left(2\delta f_1^{(1)}(x)\right) - \left(4\delta f_2^{(1)}(x)\right)$$

$$=$$

$$8\left(2f^{(1)}(x)\delta + \frac{1}{60} f^{(5)}(x)\delta^5\right)$$

$$- \left(4f^{(1)}(x)\delta + \frac{8}{15} f^{(5)}(x)\delta^5\right)$$

$$\Rightarrow 16\delta f_1^{(1)}(x) - 4\delta f_2^{(1)}(x)$$

$$=$$

$$16 f^{(1)}(x)\delta + \frac{2}{15} f^{(5)}(x)\delta^5 - 4f^{(1)}(x)\delta - \frac{8}{15} f^{(5)}(x)\delta^5$$

$$\Rightarrow 16\delta f_1^{(1)}(x) - 4\delta f_2^{(1)}(x) = 12 f^{(1)}(x)\delta - \frac{2}{5} f^{(5)}(x)\delta^5$$

$\Rightarrow$ We can isolate $f^{(1)}(x)$:

$$12 f^{(1)}(x)\delta = 16\delta f_1^{(1)}(x) - 4\delta f_2^{(1)}(x) + \frac{2}{5} f^{(5)}(x)\delta^5$$

$$\Rightarrow f^{(1)}(x) = \frac{1}{12\delta}\left[16\delta f_1^{(1)}(x) - 4\delta f_2^{(1)}(x) + \frac{2}{5} f^{(5)}(x)\delta^5\right]$$

$$= \frac{1}{12\delta}\left(16\delta\left[\frac{f(x+\delta)-f(x-\delta)}{2\delta}\right] - 4\delta\left[\frac{f(x+2\delta)-f(x-2\delta)}{4\delta}\right]\right) + \frac{1}{30} f^{(5)}(x)\delta^4$$

$$f^{(1)}(x) = \frac{1}{12\delta}\Big(8\big(f(x+\delta)-f(x-\delta)\big) - \big(f(x+2\delta)-f(x-2\delta)\big)\Big) + \frac{1}{30} f^{(5)}(x)\delta^4$$

We hence have an estimate of $f^{(1)}(x)$:

$$\boxed{f^{(1)}(x) = \frac{1}{12\delta}\left[8\big(f(x+\delta)-f(x-\delta)\big) - \big(f(x+2\delta)-f(x-2\delta)\big)\right] + \frac{1}{30} f^{(5)}(x)\delta^4 + \mathcal{O}(\delta^5)}$$

1

b) What should δ be in terms of the machine precision & properties of the function?

For an function of the estimate of the derivative found in a),
the leading order truncation error is now $\varepsilon_t \sim \delta^4 f^{(5)}(x)$

From Numerical Recipes, round off error is $\varepsilon_r \sim \varepsilon_f \left| \frac{f(x)}{h} \right|$

for ε_f = fractional accuracy with which f is computed
$\sim \varepsilon_m$ [machine's floating point format]
$= 2^{-52}$ for 64-bit machine

h = step-size $= \delta$ here

$\Rightarrow \varepsilon_r \sim \varepsilon_m \left| \frac{f(x)}{\delta} \right|$

So the variance of the derivative of f is

$$\varepsilon_t^2 + \varepsilon_r^2 = \left(\delta^4 f^{(5)}(x) \right)^2 + \left(\varepsilon_m \left| \frac{f(x)}{\delta} \right| \right)^2$$

$$= \delta^8 \left(f^{(5)}(x) \right)^2 + \varepsilon_f^2 \frac{f^2(x)}{\delta^2}$$

We want that variance to be minimum. Take $\frac{\partial}{\partial \delta} = 0$:

$$\frac{\partial \left(\delta^8 \left(f^{(5)}(x) \right)^2 \right)}{\partial \delta} + \frac{\partial \left(\varepsilon_f^2 \frac{f^2(x)}{\delta^2} \right)}{\partial \delta} = 0$$

$$\Rightarrow 8 \delta^7 f^{(5)}(x)^2 + \varepsilon_f^2 f^2(x) \left(\frac{-2}{\delta^3} \right) = 0 \qquad \times \delta^3$$

$$\Rightarrow 8 \delta^{10} f^{(5)}(x)^2 + \varepsilon_f^2 f^2(x)(-2) = 0$$

$$\Rightarrow 8 \delta^{10} f^{(5)}(x)^2 = 2 \varepsilon_f^2 f^2(x)$$

$$\Rightarrow \delta^{10} = \frac{2 \varepsilon_f^2 f^2(x)}{8 f^{(5)}(x)^2} \Rightarrow \delta^5 = \frac{1}{2} \frac{\varepsilon_f f(x)}{f^{(5)}(x)} \Rightarrow \boxed{\delta \sim \sqrt[5]{\frac{\varepsilon_f f(x)}{f^{(5)}(x)}}}$$

☆ For our exponential function $f(x) = \exp(x)$,

the optimal δ value is found to be $\delta \sim \sqrt[5]{\frac{\varepsilon_f f(x)}{f^{(5)}(x)}}$

$$\boxed{\delta \sim \sqrt[5]{\varepsilon_f}}$$

As $f(x) = f^{(5)}(x)$ in our case

☆ For $f(x) = \exp(0.01x)$, we can do the same:

$\delta \sim \sqrt[5]{\frac{\varepsilon_f f(x)}{f^{(5)}(x)}}$ where $f^{(5)}(x) = 0.01^5 f(x)$

$$\delta \sim \frac{1}{0.01} \sqrt[5]{\varepsilon_f}$$

$$\boxed{\delta \sim 100 \sqrt[5]{\varepsilon_f}}$$

2

We would like to write a numerical differentiator which, for any function f, computes the first derivative f' at some point x:

$$f' \approx \frac{f(x+dx)-f(x-dx)}{2dx}$$

where the step size dx has to be chosen optimally.
(here let's denote it δ)

From Numerical Recipes, using this symmetrized form yields an optimal step size:

$$\delta \sim \left(\frac{\epsilon_f f(x)}{f^{(3)}(x)}\right)^{\frac{1}{3}}$$

Let's try to find an estimator for $f^{(3)}(x)$.
From #1 a), we have eq^{ns} 3 & 4:

$$③ \quad 2\delta f_1^{(1)}(x) \approx 2f^{(1)}(x)\delta + \frac{1}{3}f^{(3)}(x)\delta^3 + \frac{1}{60}f^{(5)}(x)\delta^5$$

$$④ \quad 4\delta f_2^{(1)}(x) \approx 4f^{(1)}(x)\delta + \frac{8}{3}f^{(3)}(x)\delta^3 + \frac{8}{15}f^{(5)}(x)\delta^5$$

for $f_1^{(1)}(x) \simeq \dfrac{f(x+\delta)-f(x-\delta)}{2\delta}$

$$f_2^{(1)}(x) \approx \frac{f(x+2\delta)-f(x-2\delta)}{4\delta}$$

We can combine 3 & 4 to get rid of $f^{(1)}(x)$ terms, & we'll have additional $O(\delta^5)$ term:
Take $2\times$③
$-1\times$④ :

$$4\delta f_1^{(1)}(x) - 4\delta f_2^{(1)}(x) \approx \frac{2}{3}f^{(3)}(x)\delta^3 + O(\delta^5) - \frac{8}{3}f^{(3)}(x)\delta^3 + O(\delta^5)$$

$$\Rightarrow 4\delta f_1^{(1)}(x) - 4\delta f_2^{(1)}(x) \simeq -2f^{(3)}(x)\delta^3 + \underbrace{O(\delta^5)}_{\text{ignore}}$$

$$\Rightarrow f^{(3)}(x) \approx \frac{4\delta f_1^{(1)}(x) - 4\delta f_2^{(1)}(x)}{-2\delta^3}$$

$$\Rightarrow f^{(3)}(x) \approx \frac{4\delta\left[\frac{f(x+\delta)-f(x-\delta)}{2\delta} - \frac{f(x+2\delta)-f(x-2\delta)}{4\delta}\right]}{-2\delta^3} \Rightarrow$$

$$f^{(3)}(x) = \frac{[f(x+2\delta)-f(x-2\delta)] - 2[f(x+\delta)-f(x-\delta)]}{2\delta^3}$$

Which gives an expression for an optimal δ:

$$\delta \sim \left(\frac{2\delta^3 \varepsilon_f f(x)}{[f(x+2\delta)-f(x-2\delta)] - 2[f(x+\delta)-f(x-\delta)]} \right)^{\frac{1}{3}}$$

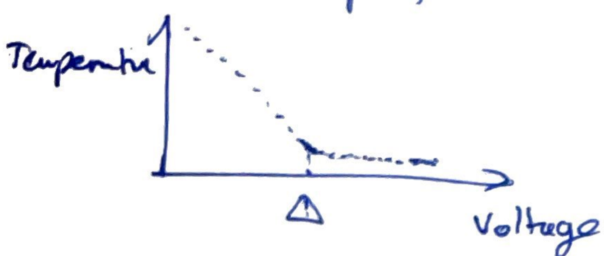
We can start with an initial ballpark accurate guess for δ:

$$\delta \sim x \varepsilon_f^{\frac{1}{3}} \quad \text{[from Numerical Recipes]}$$

& input it in our optimized equation & iterate a few times to get a sensibly optimized estimate for δ.

[3]

Given the shape of the data [which looks approx. like]:



I am using a cubic polynomial to interpolate the data, not requiring smooth derivatives given the behavior of the data at $\triangle$ where the derivative may be acting up.

To do so I used Jon's snippets of codes from his cubic_interp.py file. The idea is, for some x value [here voltage] for which we want to know y [here the temperature], choose a neighborhood of four points [from the data] around your x. Over those points, perform a cubic polynomial fit with numpy's polyfit. With the equation [technically, the coefficients found], evaluate at the initial x the interpolated y value.