Phys 512 Problem Set 1

Jan Viatteau (260857910)

Problem 1

1. We start by doing the same Taylor expansion that gave us the estimate with just $x \pm \delta$ (but we should pull out a few more terms since more will vanish). We still have :

$$\begin{aligned} f(x+\delta) &= f(x) + \delta f'(x) + \frac{1}{2}\delta^2 f''(x) + \frac{1}{6}\delta^3 f^{(3)}(x) \\ &+ \frac{1}{24}\delta^4 f^{(4)}(x) + \frac{1}{120}\delta^5 f^{(5)}(x) + ... \\ f(x-\delta) &= f(x) - \delta f'(x) + \frac{1}{2}\delta^2 f''(x) - \frac{1}{6}\delta^3 f^{(3)}(x) \\ &+ \frac{1}{24}\delta^4 f^{(4)}(x) - \frac{1}{120}\delta^5 f^{(5)}(x) + ... \\ f(x+\delta) - f(x-\delta) &= 2\delta f'(x) + \frac{1}{3}\delta^3 f^{(3)}(x) + \frac{1}{60}\delta^5 f^{(5)}(x) + ... \end{aligned}$$

and we also have the $x \pm 2\delta$ terms, which we can get by just replacing δ by 2δ in the term above :

$$f(x+2\delta) - f(x-\delta) = 4\delta f'(x) + \frac{8}{3}\delta^3 f^{(3)}(x) + \frac{8}{15}\delta^5 f^{(5)}(x) + ...$$

Now, to cancel the $f^{(3)}$ term, we have to multiply the first equation by 8 and subtract the other :

$$8[f(x+\delta) - f(x-\delta)] - f(x+2\delta) + f(x-\delta) \simeq 12\delta f'(x) - \frac{2}{5}\delta^5 f^{(5)}(x)$$

so our best estimate for $f'(x)$ will be :

$$f'(x) \simeq \frac{8[f(x+\delta) - f(x-\delta)] - f(x+2\delta) + f(x-2\delta)}{12\delta}$$

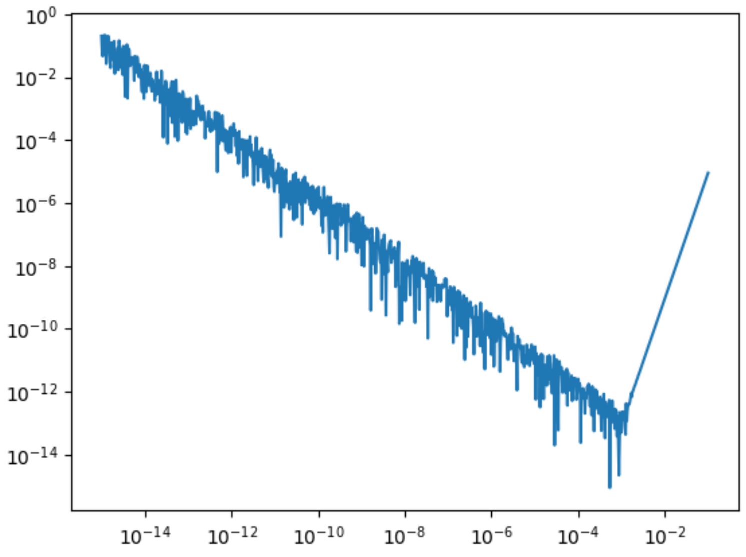
2. The leading order of the leftover part is then $\frac{1}{30}\delta^4 f^{(5)}(x)$. The roundoff error is still the same we determined in class, $f\epsilon/\delta$. The total error (that we differentiate to find the δ that minimizes total error) is then :

$$\begin{aligned}\text{total error} &= \frac{f\epsilon}{\delta} + \frac{f^{(5)}\delta^4}{30}\\ 0 &= -\frac{f\epsilon}{\delta^2} + \frac{2f^{(5)}\delta^3}{15}\\ \delta &= \left(\frac{15f\epsilon}{2f^{(5)}}\right)^{1/5}\end{aligned}$$

which is roughly the optimal δ, where f and $f^{(5)}$ are the function and its fifth derivative, and ϵ is the fractional accuracy discussed in class (10^{16} for double precision).

Let's check if our optimal δ is actually roughly correct. Taking $f(x) = e^x$ and then $f(x) = e^{0.01x}$, we scan a logspace of δs and compare the actual derivative to the obtained estimate to get the error. The results are showed in the plots below.

As can be seen in the plots, for $f(x) = e^x$ the optimal δ is close to about 10^{-3}. Using the formula above, we get that the optimal δ should be at $\left(\frac{15\times e^x\times 10^{-16}}{2\times e^x}\right)^{1/5} \simeq 9.44\times 10^{-4}$, which is pretty much agrees with what we can see in the plot. For $f(x) = e^{0.01x}$, we see the optimal δ is around 0.1, when what we get from the formula is $\left(\frac{15\times e^{0.01x}\times 10^{-16}}{2\times 0.01^5 e^{0.01x}}\right)^{1/5} \simeq 0.0944$. In this case as well, the estimate is pretty good.



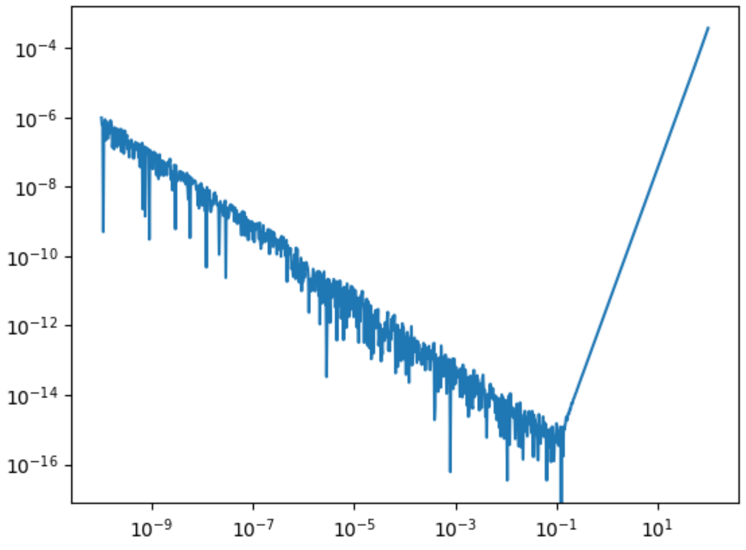


Figure 1: Results obtained when trying out a range of δ for $f(x) = e^x$ (top) and $f(x) = e^{0.01x}$ (bottom). Horizontal axis is δ, vertical axis shows the error between estimate and analytically computed derivative.

Problem 2

For my routine, I followed the equations given in Numerical recipes. To estimate the optimal dx, the reasoning that we also followed in class gave an estimate of $dx \simeq \left(\frac{\epsilon_f \times f}{f^{(3)}}\right)^{1/3} = \epsilon_f^{1/3} \times x_c$, where x_c is the curvature scale. Since we're working with an arbitrary function and have no more additional information, we can estimate $x_c \sim x$ and use that to estimate the ideal dx. However, to avoid dividing by 0, we replace x_c by 1 when it is near 0. The derivative is then given by, as suggested,

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

An estimate of the error is then given in equation 5.7.9 in Numerical recipes, $e_r + e_f = \epsilon_f^{2/3}|f'|$. An example of result obtained is for example the derivative of the exponential at $x = 1$, obtained to be 2.7182818284437933 (when it is really 2.718281828459045), with an estimated error of 5.856e-11 (when it is actually 1.525e-11). The estimate is then decent, but the error is overestimated. A different result I tested is also the sine function at $x = 12$. This time, the result was accurate down to 4.415e-10, but the estimated error was lower (1.818e-11). Underestimating errors is more serious than overestimating them, but I am not sure how to produce a better estimate.

Problem 3

For this problem, since we are given a measurement the derivative dV/dT for every point and given how the plot looks, a linear interpolation at each point should be good enough. Actually, what we do is first find the nearest point to the input voltage, then get the slope at that nearest point and calculate what would be the temperature at the input voltage if the curve was linear. Then, the rough estimate for the error could be the difference between the temperature of a nearby measurement and the temperature at that voltage, estimated by the linear interpolation. The results are shown in the two plots below.

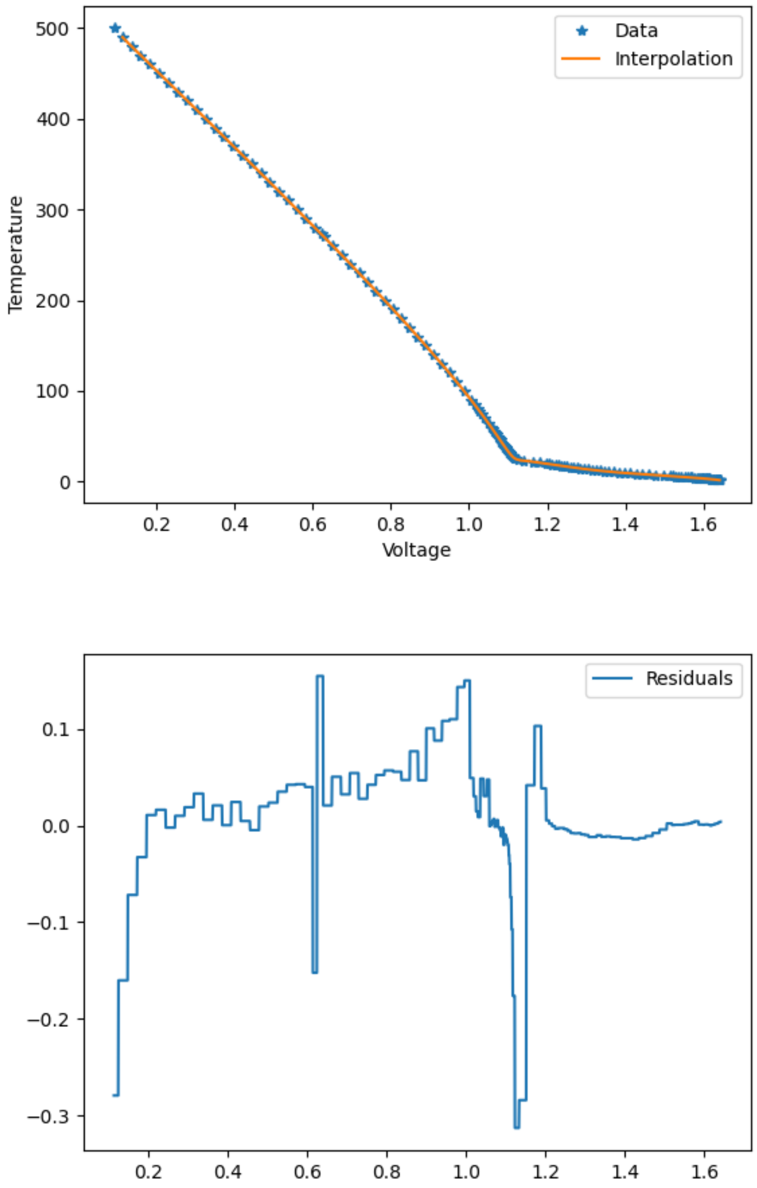


Figure 2: Interpolation and estimated residuals obtained using the Lakeshore 670 data

As we can see, for temperatures in the hundreds, the straightforward linear interpolation is good enough that we don't really get errors greater than 0.3 for most points.

Problem 4

The results for all interpolations for cos between $-\frac{\pi}{2}$ and $\frac{\pi}{2}$ are shown in the plot below. Since the plot showing the actual curve only showed that the curves all overlapped, I only kept the differences between each interpolation and the actual true function. Pretty much all code used was adapted from the code studied in lectures.

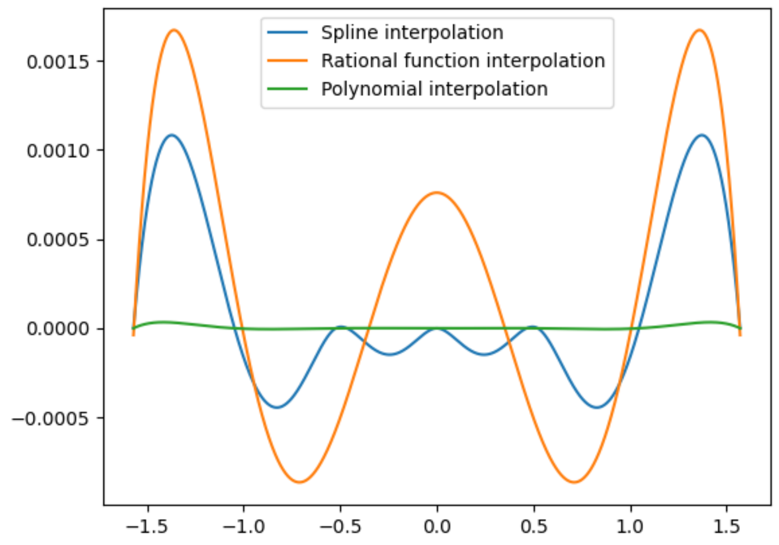


Figure 3: Figure showing the differences between the actual cos function and the spline, rational function and polynomial interpolations. The points used for interpolation are 7 points spread linearly on the interval. Interestingly, the polynomial interpolation seems to be giving the best result. That could perhaps be due to the fact that on this interval, the cos function resembles most closely a simple polynomial.

For the Lorentzian, the code was kept pretty much the same, the only changes being the function and studied interval. The number of points was also kept at 7 points. The result using np.linalg.inv is shown below.

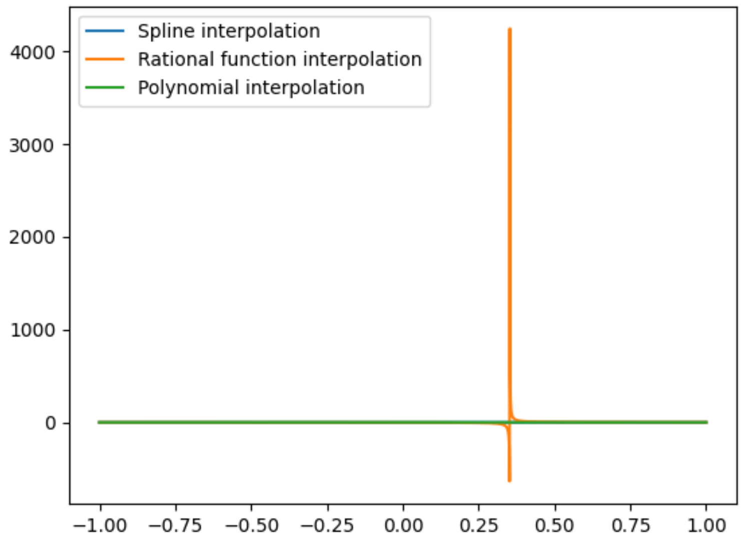


Figure 4: The interpolations for a Lorentzian. Given that the Lorentzian goes to 0 at $\pm\infty$, we could have expected that the rational function interpolation would have yielded the best result given its ability to reproduce that behavior (whereas polynomials such as those used for spline and polynomial interpolation have to diverge at some point). However, here we see that at some spot, the error for the rational function interpolation goes massive, following a kind of hyperbolic shape. This must be happening because we try to do a division by 0 somewhere.

Then, the change after using np.linalg.pinv instead is shown in the plot below.

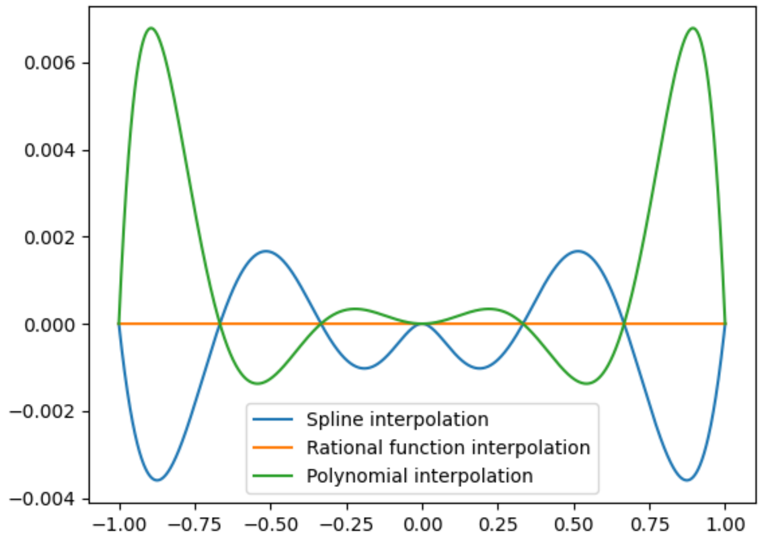


Figure 5: Same plot as above, but we switch np.linalg.inv in the code by np.linalg.pinv and now, as we expected, the rational function interpolation yields the best results.