

Scientific Computing, Fall 2020

Assignment I: Numerical Computing

Aleksandar Donev

Courant Institute, NYU, donev@courant.nyu.edu

Sample Solution Key

Here is an example solution of a **variation of HW1 problem 4**, where we do a first derivative instead of a second. I do not quite expect this kind of report from everyone, but the key things to take are the figures to present the results, and the explanations of results.

1 Sample Problem

The derivative of a function $f(x)$ at a point x_0 can be calculated using finite differences, for example the first-order *one-sided difference*

$$f'(x = x_0) \approx \frac{f(x_0 + h) - f(x_0)}{h}$$

or the second-order *centered difference*

$$f'(x = x_0) \approx \frac{f(x_0 + h) - f(x_0 - h)}{2h},$$

where h is sufficiently small so that the approximation is good.

- [10pts] Consider a simple function such as $f(x) = \cos(x)$ and $x_0 = \pi/4$ and calculate the above finite differences for several h on a logarithmic scale (say $h = 2^{-m}$ for $m = 1, 2, \dots$) and compare to the known derivative. For what h can you get the most accurate answer?
- [10pts] Obtain an estimate of the *truncation error* in the one-sided and the centered difference formulas by performing a Taylor series expansion of $f(x_0 + h)$ around x_0 . Also estimate what the *roundoff error* is due to cancellation of digits in the differencing. At some h , the combined error should be smallest (optimal, which usually happens when the errors are approximately equal in magnitude). Estimate this h and compare to the numerical observations.

1.1 Sample Solution

Comments from me on what is important are in square brackets and italic.

1.1.1 Part 1: Writing and Executing the Code

The first derivative is computed in the Matlab code `FirstDeriv.m` with default double precision and in the Matlab code `FirstDerivSP.m` with single precision arithmetic. The results are shown in Fig. 1. [*Crucial here is to use **log-log scaling** of both axes because both vary over many orders of magnitude, to show results with symbols and theory with lines, with different colors and shading and a clear legend so one can tell what is what.*]

[*In addition to just showing the figure crucial is to **interpret the results**, i.e., tell us what you see/learn from the figure. This will always count for at least one half of the points – just writing code and plotting things does not constitute scientific computing though it is an essential part of it!*] We see that for the one-sided difference we obtain the smallest error when $h \approx 10^{-4}$ for single precision (relative error of 10^{-4} , i.e., 4 digits of accuracy), and when $h \approx 10^{-8}$ for double precision (8 digits of accuracy). For the two-sided difference we obtain minimal error for $h \approx 10^{-2}$ for single precision (relative error of 10^{-6} , i.e., 6 digits of accuracy), and when $h \approx 10^{-5}$ for double precision (11-12 digits of accuracy).

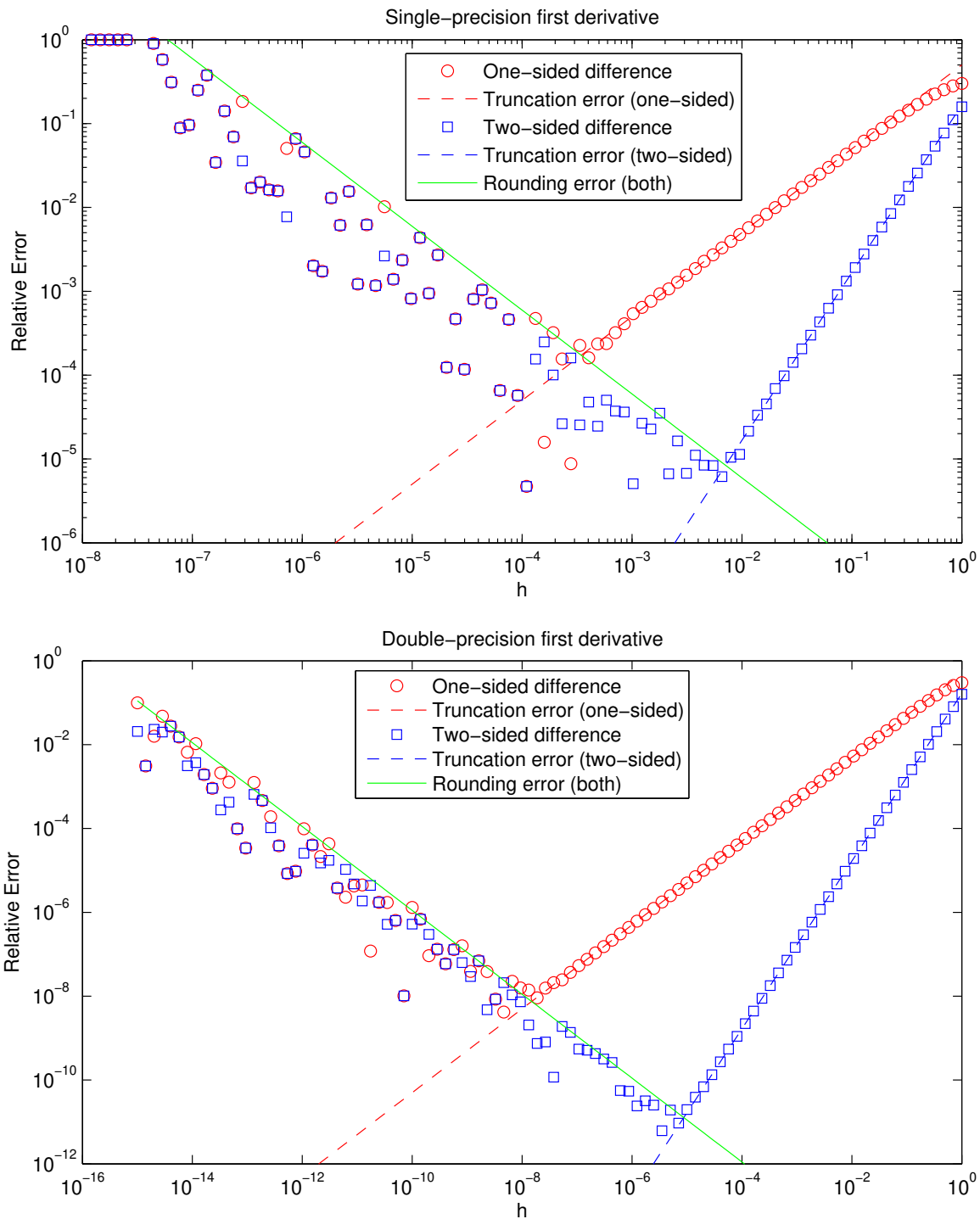


Figure 1: Results for the accuracy of the one-sided and centered-difference numerical approximation to the first derivative. The top panel is for single-precision and the bottom panel is for double precision. The symbols show numerical results (circles for one-sided, squares for two-sided), the dashed lines of the same color show theoretical estimates of the truncation error, and the solid line shows the estimate of the roundoff error.

1.1.2 Part 2: Analysis of the Results

[Important here to name the types of errors to demonstrate you understand what they are] There are two sources of numerical here: truncation error of replacing the limit in the definition of the derivative, and roundoff error from performing the calculation with finite precision arithmetic.

The truncation error here is obtained from a Taylor series expansion, to get, for the one-sided difference:

$$\begin{aligned}
f_1 &= \frac{f(x_0 + h) - f(x_0)}{h} = \frac{h f'(x_0) + h^2 f''(x_0)/2 + O(h^3)}{h} \\
&= f'(x_0) - f''(x_0) \frac{h}{2} + O(h^2).
\end{aligned}$$

The magnitude of the relative error can be estimated to be of $O(h)$,

$$|\epsilon_t| = \frac{|f_1 - f'(x_0)|}{|f'(x_0)|} \approx \frac{|f''(x_0)| h}{|f'(x_0)| 2} = \frac{h}{2}.$$

and the absolute error is of the same order. For the two-sided difference, the truncation error is much smaller,

$$\begin{aligned}
f_2 &= \frac{f(x_0 + h) - f(x_0 - h)}{2h} =, \\
&= f'(x_0) - f'''(x_0) \frac{h^2}{6} + O(h^3),
\end{aligned}$$

so now the relative error can be estimated to be of $O(h^2)$,

$$|\epsilon_t| = \frac{|f_2 - f'(x_0)|}{|f'(x_0)|} \approx \frac{|f'''(x_0)| h^2}{|f'(x_0)| 6} = \frac{h^2}{6}.$$

A rough estimate [*Remember that factors of order 2 or so are not important here, just an estimate is good enough*] of the roundoff error is obtained by noting that the numerator is computed with absolute error of about u , where u is the machine precision or roundoff unit ($\sim 10^{-16}$ for double precision). The actual value of the numerator is close to $h f'(x = x_0)$, so the magnitude of the relative error in the numerator is on the order of

$$\epsilon_r \approx \left| \frac{u}{h f'(x = x_0)} \right| \approx \frac{u}{h},$$

since $|f'(x = x_0)| \approx 0.7$ is of order unity. [*Crucial here to show that you understood what the source of the problem is: **cancellation of digits***] We see now that due to the cancellation of digits in subtracting nearly identical numbers, we can get a very large relative error when h is small. The relative truncation error of the whole calculation is thus dominated by the relative error in the numerator, and is close to ϵ_r .

Let's consider first the one-sided difference. [*I will not give the details for the centered one but your report of course should.*] The magnitude of the overall relative error is approximately the sum of the truncation and roundoff errors,

$$\epsilon \approx \epsilon_t + \epsilon_r = \frac{h}{2} + \frac{u}{h}.$$

[*The important lesson here is that different errors dominate in different regimes, and to show you understood which one*] We see that for large h the truncation error dominates (first term) but for small h the roundoff error (second term) dominates. So we cannot take h either too large or too small. For double precision, $u \sim 10^{-16}$. The minimum error is achieved for (just minimize ϵ w.r.t. h by taking first derivative and setting to zero)

$$h \approx h_{\text{opt}} = \sqrt{2u} \approx \sqrt{2 \cdot 10^{-16}} \approx 10^{-8},$$

and the actual value of the smallest possible relative error is

$$\epsilon_{\text{opt}} = \frac{h_{\text{opt}}}{2} + \frac{u}{h_{\text{opt}}} = \sqrt{2u} = h_{\text{opt}} \sim 10^{-8}.$$

These estimates agree with the numerical results shown in Fig. 1. Just replace $u \approx 6 \cdot 10^{-8}$ for single precision to get that $h_{\text{opt}} = \epsilon_{\text{opt}} \approx 10^{-4}$, in agreement with our numerical results.