Please write your family and given names and **underline** your family name on the front page of your paper.

All answers **must** be typed (preferably in latex), and compiled to a single pdf. All code and output of Q2 and Q3 should be *embedded* in latex/pdf. Code and output should be embedded with `fixed-width fonts`, e.g. Courier. Font size of all fonts must be 12. Linespacing set to 1.1 or close.
What to submit:
(1) The single pdf file 00A1.pdf (with embedded code and output).
(2) Any source code you wrote (for computation, etc).
(3) Any plot file embedded in the latex/pdf.
Thus, the code and plot will be available within latex/pdf, *as well as* separately.
See course website for example of latex, embedding plots, code, using fixed width fonts, etc.
Some points will be given for the quality of presentation.

1. Let $f(x) = \dfrac{xe^x - x}{x^2}$. (Keep f as given.)

(a) [*13 points*] What is the (relative) condition number of $f(x)$ as a function of x? For what values of x does it get very large? Explain.

(b) [*12 points*] Consider that x is a positive number close to 0. Indicate how you would compute $f(x)$ (or an approximation to it) in a numerically stable way. Justify your answer. What is the condition number of the approximation to $f(x)$ you indicated for $x \to 0$?

(c) [*10 points*] Consider that x is a negative number close to 0. Indicate how you would compute $f(x)$ (or an approximation to it) in a numerically stable way. Justify your answer.

2. In this question, we study the quality of two approximations to the first derivative of a function and of one approximation to the second derivative of a function.

By an instance of Taylor's theorem, $f(x+h) = f(x) + hf'(x) + \dfrac{h^2}{2} f''(\xi)$ for some ξ between x and $x+h$. Thus $f'(x) = \dfrac{f(x+h) - f(x)}{h} - \dfrac{h}{2} f''(\xi)$, and the approximation $f'(x) \approx \dfrac{f(x+h) - f(x)}{h} \equiv g_a(x; h; f)$ involves discretization (truncation) error $-\dfrac{h}{2} f''(\xi)$.

Note: The approximation $f'(x) \approx \dfrac{f(x+h) - f(x)}{h}$ is said to be of *first order*, because the truncation error is $O(h)$.

(a) [*6 points*] Derive an upper bound for the (absolute value of the) truncation error in $f'(x) \approx \dfrac{f(x+h) - f(x)}{h}$, assuming $|f''(x)| \leq M_2$. (The bound should be in terms of h and M_2.)

Assume that the (absolute value of the) rounding error in evaluating $\dfrac{f(x+h) - f(x)}{h}$ is at most $5\,\dfrac{\varepsilon_{\text{mach}}}{h}$.

Assuming M_2 is independent of h, study the bound for the (absolute value of the) total computation error in g_a, as a function of h.

(b) [*12 points*] Use Taylor's theorem to derive a more accurate approximation to $f'(x)$ using the values $f(x+h)$ and $f(x-h)$. Let $g_b(x; h; f)$ be the approximation to $f'(x)$.

Derive an expression for the truncation error in g_b and an upper bound for the (absolute value of the) truncation error assuming $|f'''(x)| \leq M_3$. (The bound should be in terms of h and M_3.)

Note: The approximation to $f'(x)$ derived in (b) should be of *second order*, with truncation error of $O(h^2)$.

Assume that the (absolute value of the) rounding error in evaluating g_b is at most $5\,\dfrac{\varepsilon_{\text{mach}}}{2h}$.

Assuming M_3 is independent of h, study the bound for the (absolute value of the) total computation error in g_b, as a function of h.

(c) [*12 points*] Use Taylor's theorem to derive a second-order approximation to $f''(x)$ using (a linear combination of) the values $f(x+h)$, $f(x)$ and $f(x-h)$. Let $g_c(x; h; f)$ be the approximation to $f''(x)$.

Derive an expression for the truncation error in g_c and an upper bound for the (absolute value of the) truncation error, assuming $|f''''(x)| \leq M_4$. (The bound should be in terms of h and M_4.)

Note: The truncation error should be of $O(h^2)$.

Assume that the (absolute value of the) rounding error in evaluating g_c is at most $6\,\dfrac{\varepsilon_{\text{mach}}}{h^2}$.

Assuming M_4 is independent of h, study the bound for the (absolute value of the) total computation error, in g_c, as a

function of h.

(d) [*15 points*] Let $f(x) = \ln(x)$, $x = 1$, and $h > 0$. Write a MATLAB program, which, for $h = 10^{-16}, 10^{-15}, \cdots, 10^{-1}$, computes g_a, g_b and g_c, and the respective errors. (To calculate the errors in g_a, g_b and g_c, use the exact values of $\ln'(1)$ and $\ln''(1)$.) Also estimate approximately M_2, M_3 and M_4 for the given function $f(x) = \log(x)$ and $x = 1$, and compute the bounds for the total computational errors as derived in (a), (b) and (c). In one plot, in log-log scale, plot the bounds of the errors in g_a (solid line), g_b (dashed line) and g_c (dotted line), versus h, for $h = 10^{-16}, 10^{-15}, \cdots, 10^{-1}$. Indicate the data points in the plot by "x", i.e. use

```
loglog(h, abs(erra), '-',  h, bounda, 'x-', ...
       h, abs(errb), '--', h, boundb, 'x--', ...
       h, abs(errc), ':',  h, boundc, 'x:')
```

where `h = 10.^[-16:-1];`. Use `axis tight;` after the plot. Include a legend to distinguish between lines plotted. Also output the points (stepsize, error) where the minimum error occurs in each case (a), (b) and (c), using a %9.2e format. Comment on the results.

Note: In (a), (b) and (c), when you study the bounds of the total computation errors as functions of h, indicate where minima or maxima occur.

To approximately estimate M_2, M_3 and M_4, differentiate (by hand) the function, and find (by hand and approximately) the upper bound of the absolute value of the respective derivatives in the interval of interest. Then hard-code this into the MATLAB program. If you find that M_3 and M_4 are dependent on h, you are not doing anything wrong, but consider that the dependence is mild, so that you can ignore it in the studies of the total computational errors (a), (b) and (c) as functions of h.

For a value of machine epsilon ($\varepsilon_{\text{mach}}$), use the MATLAB predefined value eps.

3. [*20 points*] We know from calculus that

$$\lim_{n\to\infty}(1 + \frac{1}{n})^n = e = \exp(1).$$

Write a programme or script that approximately computes e using $(1 + \frac{1}{n})^n$, for $n = 10^i$, $i = 0, \cdots, 12$. Tabulate i, the "exact" e (taken from the built-in exp(1) in double precision), the approximate e, the relative error, and the quantity $1 + \frac{1}{n}$. Use a format compatible with

```
fprintf('%13.0f %13.10f %13.10f %11.3e %14.11f\n', ...
        an, eapprox, e, (e-eapprox)/e, 1 + 1/an);
```

Run the programme once in single precision (all floats, except the "exact" e), and once in double precision (all doubles). (Thus you need some conventional programming language such as C, python, Fortran, etc). Comment on the results.

General note: Do no use any symbolic package; MATLAB may allow you to find derivatives, integrals, Taylor series, etc, of functions symbolically, but these features should not be used in this course, unless explicitly requested.