Whenever calculations are needed to solve a problem, those calculations must be submitted as part of the homework assignment.

Homework must be submitted electronically. Unless express permission has been given in advance by the instructor for a late homework submission, a 30% percent penalty will be deducted for each late day (or part of a late day).

In printing non-integers, be sure to use scientific notation and to show at least 7 decimal digits of precision.

Exercise 1.1. (Sequences and rates of convergence.)
(a) Let $\gamma = 0.96$. Compute and print the first 10 elements of the sequence $\{x_k\}, k = 0, \ldots$, where $x_k = \gamma^{2^k}$. (Note that the exponent is $2^k$!) What is the limit of this sequence? Perform a mathematical analysis that confirms the rate of convergence of the sequence and also gives the asymptotic error constant. Explain your logic.

(b) Compute and print out the first 20 elements of the sequence $\{x_k\}$, where $x_k = 1/(k^k)$, starting with $k = 1$. Determine the limit of the sequence. Show that the sequence satisfies both tests for superlinear convergence given in Lecture Notes 2.

Exercise 1.2. (Bisection–I.) Consider a nonlinear scalar-valued twice-continuously differentiable function $f(x)$ and assume that you are given two points $a_0$ and $b_0$, the endpoints of an interval such that $f(a_0)f(b_0) < 0$. Write a generic code that applies the bisection algorithm to $f$ in this interval. The input to this code should include a routine that calculates $f(x)$ for any $x$, and the two points $a_0$ and $b_0$.

Your program should stop in one of the following three ways: (i) your code finds $\bar{x}$ where $f(\bar{x}) = 0$, (ii) the length of the latest interval of uncertainty is less than or equal to a specified tolerance $\text{xtol}$, or (iii) $\text{maxit}$ bisection steps have been performed, where the value of $\text{maxit}$ is specified.

Keep in mind that Matlab numbers arrays starting with 1 rather than 0! At each bisection step, print the iteration index $k$, $a_k$, $f(a_k)$, $b_k$, $f(b_k)$, and $|a_k - b_k|$.

Consider the function
$$\tilde{f}(x) = f = x^{11} - 11x^{10} + 55x^9 - 165x^8 + 330x^7 - 462x^6 + 462x^5 - 330x^4 + 165x^3 - 55x^2 + 11x - 1, \quad (1.1)$$
which is algebraically equivalent to $(x - 1)^{11}$.

(a) Show mathematically that $\tilde{f}$ has only one zero, at $x^* = 1$.

(b) Let $a_0 = 0.9$, $b_0 = 1.2$, and $\text{maxit} = 5$. Run your bisection program, evaluating $\tilde{f}$ in exactly the form given in (1.1)—do not simplify the algebra!

Does $x^*$ lie in the final bisection interval $[a_k, b_k]$? Explain whether or not the results are what you expected. If not, explain why. What do these results tell us about the practical reliability of the bisection algorithm?

(c) Run your bisection program a second time, this time applied to $\tilde{f}(x) = (x - 1)^{11}$, evaluated in this form, using the same $a_0$, $b_0$, and $\text{maxit}$ as in part (ii). Was your bisection routine successful? Comment on any differences between the results of (ii) and (iii).
Exercise 1.3. (Bisection–II.)\footnote{Taken from Exercise 10, Chapter 4, Greenbaum and Chartier, Numerical Methods.} The function
\[ f(x) = \frac{x^2 - 4x + 4}{x^2 - 2x - 3} \]
has exactly one zero, at \( x^* = 2 \), in the interval \([1, 4]\). Let \( a_0 = 1 \) and \( b_0 = 4 \), and confirm that \( f(a_0)f(b_0) < 0 \). Then run your bisection program on \( f(x) \), with \([a_0, b_0]\) as an initial interval of uncertainty, stopping after 12 iterations or when the interval of uncertainty is less than or equal to \( 10^{-3} \). Does bisection appear to be converging to \( x^* \)? Explain, based on the numbers from your program and any other evidence you may think of, why your answer is “yes” or “no”. If “no”, explain whether or not this non-convergence is consistent with the theory of bisection.

Exercise 1.4. (Special conditions for a pure Newton method.) Consider using a pure Newton method to solve for a zero of \( f(x) \), where \( f \) is a twice-continuously differentiable function for which it is known that \( f''(x) > 0 \) for all \( x \). Suppose in addition that we are given a finite interval \([a, b]\) such that \( f(a) > 0 \) and \( f(b) < 0 \).

(a) Show that: (i) \( f(a) < 0 \); (ii) \( f'(x) \neq 0 \) for any \( x \in [a, b] \); and (iii) there exists only one zero of \( f(x) \) in \([a, b]\). [You may want to use one or more versions of the Mean Value Theorem and Rolle’s Theorem in your proofs.]

(b) Suppose that the point \( x_0 \) is in \([a, b]\) and that \( f(x_0) > 0 \). Show that, for every Newton iterate \( x_k \) with \( k \geq 1 \), (i) \( x_k < x_{k-1} \) and (ii) \( f(x_k) > 0 \).

(c) Pick a value \( c > 0 \) and a value of \( x_0 \) that is much larger than \( \sqrt{c} \). Given that a single iteration of a pure Newton method is performed to find a zero of \( \tilde{f} \), what are the interesting properties of the Newton iterate \( x_1 \) with respect to the interval of uncertainty \([0, x_0]\)? Explain.

(d) Consider applying Newton’s method to \( \tilde{f}(x) = x^2 - c \) with \( c = 0 \), starting with \( x_0 = 1 \). What sequence of iterates will be generated? What rate of convergence do they display? Explain.

Exercise 1.5. A point of historical interest: on computers lacking floating-point division, Newton’s method can be used to compute the reciprocal of a positive real number \( d \) without performing division.

(a) Give the mathematical form of the pure Newton iteration for the function \( \tilde{f}(x) = d - 1/x \), where \( d > 0 \). Explain why computing the Newton iterates does not require explicit calculation of \( 1/x \).

(b) Write a special-purpose program that generates pure Newton iterates for the function \( \tilde{f} \) (given a value \( d > 0 \)) for \( k = 0, \ldots, \) stopping after \texttt{maxit} steps for a specified value \texttt{maxit}. At each iteration, print \( k, x_k, \tilde{f}(x_k), \) and \(|x_k - 1/d|\), using format \texttt{long e} for the last three values. Run your program with \( d = 4 \) and \texttt{maxit} = 7, trying the three starting points \( x_0 = 0.4, x_0 = 0.5, \) and \( x_0 = 1.0 \). In each case, comment on the behavior of the Newton iterates—are they converging to \( 1/d \)? If so, how rapidly? If not, what is happening to the Newton iterates?

(c) What happens if \( x_0 = 2/d \)? Show why the Newton iterates will converge if \( 0 < x_0 < 2/d \).

Exercise 1.6. Suppose that \( f \) is a nonlinear scalar-valued twice-continuously differentiable function such that \( f''(x) > 0 \) for all \( x \). Assume that two points \( x_0 \) and \( x_1 \) are given such that \( f(x_0)f(x_1) < 0 \), and that the regula falsi method is applied to find \( x^* \) such that \( f(x^*) = 0 \), using \( x_0 \) and \( x_1 \) to define an initial interval. Show that one of the initial points will be used in the linear fit in every regula falsi iteration.

Exercise 1.7. (Implementing Newton and secant.) Suppose that we wish to compute a point \( x^* \) for which \( f(x^*) = 0 \), where \( f \) is a scalar-valued twice-continuously differentiable function of a single real variable.

(a) Write two programs (pure Newton and pure secant) that implement the associated zero-finding methods.

The codes written for this first homework assignment will be unrealistic because the value of \( x^* \), the exact zero, is assumed to be known, so that the error in each iterate can be calculated exactly.
For later assignments, your codes will need to be modified so that they do not use the explicit value of $x^*$; the logic of each iteration does not require $x^*$.

Each of your two codes should stop (i) if an exact zero $\bar{x}$ is found, i.e., a point such that $f(\bar{x}) = 0$, (ii) after `maxit` iterations, and (iii) if $|f|$ is less than `ftol`, for specified values of `maxit` and `ftol`.

The input to the Newton code will include a routine that calculates $f(x)$ and $f'(x)$ for a given value of $x$, plus specification of an initial point $x_0$.

The input to the secant code will include a routine that calculates $f(x)$, plus specification of two starting points, $x_0$ and $x_1$.

**Keep in mind that Matlab starts the numbering of arrays with 1, not with 0!**

- At iteration $k$ in the Newton code program, print $k$, $x_k$, $f_k$, $f'_k$, and $x_k - x^*$, using scientific notation and showing at least 7 digits of accuracy.
- At iteration $k+1$ in the secant code, print $k+1$, $x_k$, $f_k$, $x_{k+1}$, $f_{k+1}$, $x_{k+1} - x^*$, and the ratio $|x_{k+1} - x^*|/|x_k - x^*|$. You will use these Newton and secant programs later in the course.

(b) Run your Newton code for $f(x) = x^3 + 4x - 5$, with $x_0 = 10$, taking `ftol = 10^{-14}` and `maxit = 15`. Comment on and explain the behavior of the Newton iterates: Are they converging to $x^*$? Does the iterates appear to be converging quadratically? Does the sequence of errors satisfy the expected properties of a quadratically convergent Newton sequence? Are the iterates converging from one side of $x^*$? Are they converging monotonically, i.e., each iterate is closer to $x^*$ than the previous iterate?

(c) Run your secant code for $f(x) = x^3 + 4x - 5$ with two different sets of starting points.

(a) $x_0 = 10$ and $x_1 = 5$;
(b) $x_0 = 10$ and $x_1 = -5$.

For both pairs, comment on and explain the behavior of the secant iterates. Are they converging to the solution? Does convergence appear to be superlinear? Do the iterates define an interval of uncertainty? Are the iterates converging from one side of $x^*$? Are they converging monotonically, i.e., each iterate is closer to $x^*$ than the previous iterate? Comment on and explain whether, and when, the “waltz” behavior of the secant iterates can be observed in these iterates.