Homework 4
APPM 4600 Numerical Analysis, Fall 2025

Due date: Friday, September 19, before midnight, via Gradescope. **Instructor**: Prof. Becker
Revision date: 9/14/2025
Theme: root finding, fixed point theorems, contraction mapping theorem, cobweb plots, scalar optimization.

Instructions Collaboration with your fellow students is OK and in fact recommended, although direct copying is not allowed. The internet is allowed for basic tasks (e.g., looking up definitions on wikipedia) but it is not permissible to search for proofs or to *post* requests for help on forums such as `http://math.stackexchange.com/` or to look at solution manuals. Please write down the names of the students that you worked with. Please also follow our AI policy.

An arbitrary subset of these questions will be graded.

Turn in a PDF (either scanned handwritten work, or typed, or a combination of both) to **Gradescope**, using the link to Gradescope from our Canvas page. Gradescope recommends a few apps for scanning from your phone; see the Gradescope HW submission guide.

We will primarily grade your written work, and computer source code is *not* necessary (and you can use any language you want). You may include it at the end of your homework if you wish (sometimes the graders might look at it, but not always; it will be a bit easier to give partial credit if you include your code). For nicely exporting code to a PDF, see the APPM 4600 HW submission guide FAQ.

Problem 1: Write a function `bisect` which implements the bisection method, and takes as input a function `f`, numbers specifying the interval $[a, b]$, and a tolerance `tol` which controls how far the approximate root is from the true root.

Make sure to notify the user in some way (e.g., raising an error, or returning an informative exit code) is the given function does *not* change signs on $[a, b]$. Optionally, you can also specify a maximum number of iterations.

The deliverable for this problem is your **code** for this function `bisect`. You may use any programming language, though Python and Matlab are encouraged.

Note: In Matlab, the input function might be defined in another file, like in `myfun.m`. In this case, to call your bisection function, use it like `bisect(@myfun,...)`. The `@` tells Matlab to make a function handle. Alternatively, if you defined `myfun` as an anonymous function, then you don't need the `@`, e.g., `fcn=@(x)x^2; bisect(fcn, ...)`.

Problem 2: Consider the equation $2x - 1 = \sin x$. Recall that on Homework 2 we found an interval $[a, b]$ which was guaranteed to have a root, and then in fact proved that there was only one root, so it is unique. For this problem, we'll use the same a, b that you found in Homework 2.

a) Use your function from Problem 1 to approximate r to eight correct decimal places. The **deliverable** is a list of the approx solution at every step.

b) The function $f(x) = (x - 5)^9$ has a root (with multiplicity 9) at $x = 5$ and is monotonically increasing for $x > 5$ and monotonically decreasing for $x < 5$, and should thus be a suitable candidate for your function above. Use `a=4.82` and `b=5.2` and `tol = 1e-4` and use `bisection` with:

   i. $f(x) = (x - 5)^9$.

Copyright © 2025 CU Applied Math Dept.

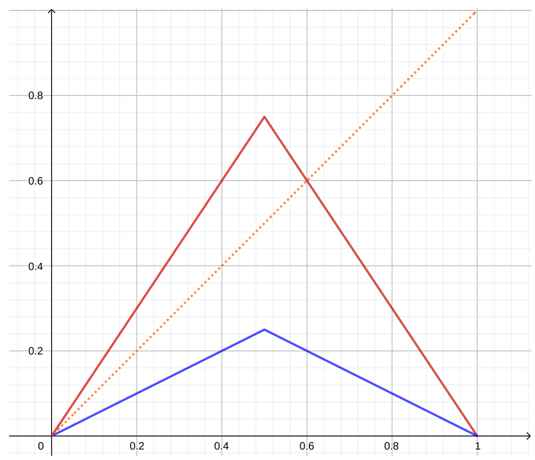

Figure 1: Tent function for $\mu = .5$ (blue) and $\mu = 1.5$ (red), as well as the line $y = x$ in dashed orange

ii. The expanded expanded version of $(x-5)^9$, that is, $f(x) = x^9 - 45x^8 + \ldots - 1953125$.[1] You may use `polyval` or `numpy.polyval` (old) or `numpy.polynomial.Polynomial` (new)

The **deliverables** for this problem are (1) a graph of the error produced from both variants discussed above, and (2) a discussion of what you think is happening.

Problem 3: Tent Map. Consider the "tent function"

$$g_\mu(x) = \mu \min(x, 1-x)$$

on the interval $[0,1]$, where $0 \le \mu \le 2$. This is like an upside down and shifted absolute value function (suggestion: graph it, for both $\mu < 1$ and $\mu > 1$; and see Fig. 1). We're interested in fixed points of the tent map, so solutions to $x = g_\mu(x)$. This is a favorite example used when teaching dynamical systems. *Suggestion:* Try making a "cobweb plot" of this function using, e.g., geogebra.org/m/uvsfvNDt. The software may not like functions defined with a "max" function, so you can instead use the fact that

$$\min(x, 1-x) = -\left|x - \frac{1}{2}\right| + \frac{1}{2}. \tag{1}$$

a) Show that for $\mu \in [0,2]$ that $g_\mu(x) \in [0,1]$ for all $x \in [0,1]$.

b) Let $0 \le \mu < 1$. *Hint: For this problem, triangle inequality or reverse triangle inequality may be useful*[2]

 i. Prove there is a unique fixed point of g_μ in $[0,1]$.

 ii. Given an arbitrary starting point $x_0 \in [0,1]$, would the fixed point iteration necessarily find the fixed point?

 iii. What is the fixed point?

[1]Note: you can get these coefficients using Matlab's `poly` or Python's `numpy.poly` and specifying that 5 is a root with multiplicity 9; this saves you having to type them in. The newer Python way is `from numpy.polynomial import Polynomial; Polynomial.fromroots(9*[5]).coef`

[2]The triangle inequality says that you cannot get from point a to point b any faster than taking a direct path. Mathematically, it is $|a-b| \le |a| + |b|$ (as well as $|a+b| \le |a| + |b|$). In one dimension, you can prove this by considering all the possible cases for the sign of a and b. In higher dimensions, the triangle inequality is always true by definition for any *norm*, since norms are required to satisfy the triangle inequality. The *reverse triangle inequality* is just the triangle inequality with a $a = a + 0$ trick, i.e., $|a| = |a+0| = |a-b+b| \le |a-b| + |b|$ so rearranging gives $\boxed{|a-b| \ge |a| - |b|}$.

c) What are the fixed points if $\mu = 1$?

d) Prove that for $1 < \mu \leq 2$ there are two fixed points in $[0, 1]$, and find these fixed points. Would the fixed point iteration find either of these fixed points? Does the contraction mapping theorem apply?

e) Run the fixed point iteration starting at $x = \pi/6$ for both $\mu = 1.1$ and $\mu = 1.5$. Do this for about 10 iterations; do you get a rough feel for what is happening? Then run 10^6 iterations and make a histogram (still for both values of μ). Include the histogram plots with your homework. Does the iteration converge to one of the fixed points?

Problem 4: **Word problem** Layla is a good college student and also likes to ski. Her happiness level for taking d credits this semester is given by the function

$$F(d) = \underbrace{d^2}_{\text{joy of learning}} - \underbrace{.001\left(e^d - 1\right)}_{\substack{\text{less free time} \\ \text{for skiing}}}.$$

How many credits should she sign up for in order to maximize her happiness? (For simplicity, assume that fractional credits are allowed). Specifically, show how to convert this into a root-finding problem, solve it with your own root-finding implementation (include some output from every iteration), and report the final answer.