AM205: Assignment 1

Due: 27 September at 5pm

Question 1 [8 points] [Written question, no code required.]

A binary floating point number $\hat{x} \in \mathbb{F}$ is represented as

$$\hat{x} = (-1)^s \times \left(1 + \sum_{k=1}^{p} d_k 2^{-k}\right) \times 2^E = \pm(1.\underbrace{d_1 d_2 \ldots d_p}_{p \text{ mantissa bits}})_2 \times 2^E. \tag{1}$$

Here each d_k is a binary digit (or "bit"), hence $d_k \in \{0, 1\}$ for $k = 1, 2, \ldots, p$. Similarly $s = \{0, 1\}$ is the sign bit, and the exponent E is an integer in the interval $L \leq E \leq U$. Different choices of p, L and U lead to different floating point number systems.

(a) [3 points] Consider a "toy" 7-bit floating-point number system with $p = 3$, $L = -2$, and $U = 3$. Hence 3 bits are required to represent E, and we suppose that two values of E are reserved to indicate exceptional values like zero, `NaN` and `Inf`. It is generally assumed that $d_0 = 1$. However, the IEEE floating point standard uses the concept of "subnormal" numbers, in which d_0 is allowed to be 0 in order to enable smaller numbers to be represented. A floating point number in which $d_0 = 1$ is referred to as "normalized", whereas a floating point number in which $d_0 = 0$ is referred to as "subnormal".

Ignoring exceptional values, what is the largest positive number that can be represented in this "toy" number system? What is the smallest positive normalized number that can be represented? What is the smallest positive subnormal number that can be represented? Make a line plot of the distribution of the normalized floating-point numbers and a second plot showing the distribution if subnormal numbers are allowed. Make plots covering the full range of floating-point numbers as well as plots that are zoomed to the region $[-1, 1]$. Also, make sure you indicate $\hat{x} = 0$ in each plot.

(b) [3 points] Let $x = 11.73$, $y = 1.631$, $z = x + y$. Calculate $\hat{x} = \text{round}(x)$, $\hat{y} = \text{round}(y)$, and $\hat{z} = \hat{x} \oplus \hat{y}$ using the "toy" number system from (a) with "round-to-nearest". Report your result for $\hat{z}$ using both the notation from (1) and standard decimal notation. What is the relative error $|\hat{z} - z|/z$?

(c) [2 points] Recall that we proved the following inequality for the "round-to-nearest" rule in Unit 0:

$$\left|\frac{\text{round}(x) - x}{x}\right| \leq \frac{\epsilon}{2}. \tag{2}$$

This result is in fact only valid for x in the normalized range. Construct a counterexample using the "toy" number system from (a) which shows that (2) can be violated. You should choose x to be between the smallest positive subnormal number and the smallest positive normalized number.

Question 2 [6 points] [Written question, no code required.]

(a) [4 points] Derive the interpolating polynomial for the data $(-1, 1.7)$, $(-0.1, 0.53)$ and $(1, 1.3)$ using the monomial basis and the Lagrange basis. Show that these two representations are equivalent. (You can use MATLAB to solve any linear systems that you construct.)

(b) [2 points] Derive the interpolating polynomial for the data $(-3, -3.4)$, $(1.5, -1.06)$ and $(7, -2.6)$ by appropriately mapping and scaling the interpolant from (a). (Hint: Use the affine map $\hat{x} = 5x + 2$.)

Question 3 [38 points]

In this question we consider polynomial interpolation. (Unlike in lectures, in this question interpolation point indexing starts at 1 rather than 0 since this is more natural in MATLAB.)

(a) [6 points] Write a MATLAB function `lagrange_interp` that uses Lagrange polynomials, L_k, and the formula $p_{n-1}(x) = \sum_{k=1}^{n} y_k L_k(x)$ to interpolate data $\{(x_k, y_k),\ k = 1, 2, \ldots, n\}$. `lagrange_interp` should take three vector arguments: the interpolation points $\{x_k\}$, the corresponding interpolation values $\{y_k\}$, and the points at which to evaluate the interpolant $\{x_{\text{eval},\ell}\}$; it should return the vector $\{p_{n-1}(x_{\text{eval},\ell})\}$.

Let $f(x) = \sin(5.5x)$. Use your `lagrange_interp` function to interpolate f at n evenly spaced points in the interval $x \in [-1, 3]$. Plot $\|f - p_{n-1}\|_\infty$ vs. n on "semilog-y" axes for $n = 5, 10, 15, \ldots, 55, 60$; here you should estimate the "infinity norm" by sampling at 1000 equispaced points in $[-1, 3]$. In a separate plot, superimpose f and p_{59}.

(b) [6 points] Write a function `chebyshev_pts` that returns n Chebyshev points in the interval $[a, b]$. Use the formula $x_j = \cos((2j-1)\pi/2n)$, $j = 1, \ldots, n$ (you will need to map from $[-1, 1]$ to $[a, b]$). Repeat (a) using Chebyshev points instead of equispaced points.

(c) [8 points] Recall the interpolation error formula (from lectures)

$$f(x) - p_{n-1}(x) = \frac{f^n(\theta)}{n!}(x - x_1)\ldots(x - x_n), \quad \text{for some } \theta \in (a, b).$$

Use this formula to derive an upper bound for $\|f - p_{n-1}\|_\infty$, where p_{n-1} is the degree $n-1$ interpolant of $f(x) = \sin(5.5x)$ at n Chebyshev points on an arbitrary interval $[a, b]$. (Your bound should be a mathematical formula; it should **not** rely on sampling to approximate the infinity norm.) Use a figure to compare your bound for $\|f - p_{n-1}\|_\infty$ with the Chebyshev interpolation error from (b) (*i.e.* on the interval $[-1, 3]$) for $n = 5, 10, 15, \ldots, 55, 60$ and explain why the error bound "fails" for sufficiently large n.

(d) [4 points] Plot the Lebesgue constants for the interpolation points in (a) and (b) for $n = 5, 10, \ldots, 55, 60$. (Sample at 1000 evenly spaced points to approximate the maximum in the definition of the Lebesgue constant.) How does this relate to your answers to (a) and (b)?

(e) [7 points] On a tensor-product grid in 2D, a Lagrange polynomial is given by the product of two 1D Lagrange polynomials:

$$p_{n-1,m-1}(x, y) = \sum_{j=1}^{n}\sum_{k=1}^{m} f(x_j, y_k) L_{jk}(x, y) \quad \text{where} \quad L_{jk}(x, y) = L_j(x) L_k(y).$$

Use this formula to interpolate the function $f(x, y) = \sin(1.8x)\cos(1.7xy)\exp(-0.4xy)$ on a 10×8 grid of Chebyshev points on the domain $[-2, 2] \times [-1, 1]$. In your report, specify the error $|f(x_{\text{eval}}, y_{\text{eval}}) - p_{9,7}(x_{\text{eval}}, y_{\text{eval}})|$ at the point $(x_{\text{eval}}, y_{\text{eval}}) = (1.23, -0.68)$. Also, make a contour plot with 35 contour levels and a colorbar by evaluating $f - p_{9,7}$ on a uniform 50×50 grid of points that cover $[-2, 2] \times [-1, 1]$. Superimpose (using contrasting colors) the 10×8 grid of interpolation points and the evaluation point $(x_{\text{eval}}, y_{\text{eval}})$ on your contour plot.

(f) [7 points] Let $\mathcal{C} \subset \mathbb{R}^2$ denote an 8×8 Chebyshev grid in $[-1,1] \times [-1,1]$. Let $g_1(x,y) = x$, $g_2(x,y) = y + \sin\left(\frac{2\pi x}{4}\right) - 1$, and let $\hat{\mathcal{C}} = \{(g_1(x,y), g_2(x,y) \,|\, (x,y) \in \mathcal{C}\} \subset \mathbb{R}^2$.

Interpolate the function $f(x,y) = 2x^2 + \cos(2.7xy)$ on $\hat{\mathcal{C}}$. Report the error $|f(x_{\text{eval}}, y_{\text{eval}}) - p_{7,7}(x_{\text{eval}}, y_{\text{eval}})|$ at $(x_{\text{eval}}, y_{\text{eval}}) = (0.69, 0.05)$ and make the same type of contour plot of the error as in (e) (*i.e.* again use a 50×50 grid of points for plotting purposes, and superimpose interpolation points and the evaluation point). Note that in this case the domain of your contour plot should be the curvilinear domain covered by the interpolation points $\hat{\mathcal{C}}$.

Question 4 [5 points] [Written question, no code required.]

A square matrix $B \in \mathbb{R}^{n \times n}$ is positive definite if $v^T B v > 0$ for all nonzero $v \neq 0$. Suppose $A \in \mathbb{R}^{m \times n}$, where $m \geq n$. Give a necessary and sufficient condition on A for $A^T A$ to be positive definite, and provide a proof that $A^T A$ is positive definite if and only if your condition is satisfied.

Question 5 [14 points]

A planet follows an elliptical orbit, which can be represented in a Cartesian (x, y) coordinate system by the equation

$$b_0 + b_1 x + b_2 y + b_3 xy + b_4 y^2 = x^2.$$

(a) [11 points] Use "backslash" in MATLAB to find the linear least-squares fit for the parameters b_0, b_1, b_2, b_3, b_4, given the following observations of the planet's position:

x	1.02	0.95	0.87	0.77	0.67
y	0.39	0.32	0.27	0.22	0.18
x	0.56	0.44	0.30	0.16	0.01
y	0.15	0.13	0.12	0.13	0.15

(This data is provided in the file q5a_data.txt on the AM205 website.) Plot the resulting orbit with a continuous curve (you should plot the entire ellipse) and the observations (marked with $\times$ symbols) on the same plot. What is the 2-norm of the residual for your best fit?

(b) [3 points] The observation data is nearly rank-deficient, which implies that the matrix $A^T A$ is nearly singular and hence the parameter fit will be sensitive to perturbations in the data (*i.e.* the least-squares fit is poorly conditioned in this case). To show this, compute the best fit to the perturbed data $\hat{x} = x + \Delta x$ and $\hat{y} = y + \Delta y$, where Δx, Δy are given below.

Δx	-0.0029	0.0007	-0.0082	-0.0038	-0.0041
Δy	-0.0033	0.0043	0.0006	0.0020	0.0044
Δx	0.0026	-0.0001	-0.0058	-0.0005	-0.0034
Δy	0.0009	0.0028	0.0034	0.0059	0.0024

(This data is provided in the file q5b_data.txt on the AM205 website.) Superimpose the two sets of observations and corresponding orbits on the same plot.

Question 6 [24 points]

As part of an aircraft manufacturing process, a rectangular metal plate needs to be curved into a segment of a cylinder and then placed in the appropriate position on the aircraft's frame so that it can be welded into place. To reduce costs, the aircraft company wants to use a robotic arm for this procedure.

The metal plate has a set of "alignment points" on it, which need to be matched with corresponding "target points" on the aircraft's frame so that the plate can be welded correctly. The target points are labeled with reflectors so that they can be located with an automated laser-based alignment system. Note that the alignment system has some inherent measurement error (*i.e.* noise).

The metal plate is initially in the $z = 0$ plane, and the alignment points on the plate are initially at (coordinates measured in meters):

$$\begin{array}{cccc} (-1,6,0) & (-0.75,6,0) & (0.75,6,0) & (1,6,0) \\ (-1,4,0) & (-0.75,4,0) & (0.75,4,0) & (1,4,0) \\ (-1,-4,0) & (-0.75,-4,0) & (0.75,-4,0) & (1,-4,0) \\ (-1,-6,0) & (-0.75,-6,0) & (0.75,-6,0) & (1,-6,0) \end{array} \tag{3}$$

The laser alignment system indicates that the target points are at:

$$\begin{array}{cccc} (-7.85,2.22,2.17) & (-7.80,2.42,1.98) & (-7.81,3.13,0.69) & (-7.86,3.19,0.41) \\ (-5.87,1.78,1.88) & (-5.89,1.95,1.69) & (-5.90,2.65,0.42) & (-5.89,2.77,0.18) \\ (1.82,-0.01,0.87) & (1.86,0.15,0.70) & (1.83,0.89,-0.61) & (1.80,0.94,-0.81) \\ (3.73,-0.50,0.57) & (3.78,-0.29,0.44) & (3.74,0.42,-0.88) & (3.76,0.50,-1.07) \end{array} \tag{4}$$

The robotic arm is able to do the following steps in the order specified below:

1.) Bend the plate about the y-axis into the $-z$ half-space so that the plate is curved into a segment of a cylinder with a specified radius; the range of radii that the robot arm can impose is $[2.9\text{ m}, 3.1\text{ m}]$. Note that this bending procedure is an "identity mapping" for the $x = 0$ slice of the plate.

2.) Rotate the curved plate around the x-axis by θ_x radians using the matrix $R_x = \begin{pmatrix} 1 & 0 & 0 \\ 0 & \cos\theta_x & -\sin\theta_x \\ 0 & \sin\theta_x & \cos\theta_x \end{pmatrix}$.

3.) Rotate the curved plate around the y-axis by θ_y radians using the matrix $R_y = \begin{pmatrix} \cos\theta_y & 0 & \sin\theta_y \\ 0 & 1 & 0 \\ -\sin\theta_y & 0 & \cos\theta_y \end{pmatrix}$.

4.) Rotate the curved plate around the z-axis by θ_z radians using the matrix $R_z = \begin{pmatrix} \cos\theta_z & -\sin\theta_z & 0 \\ \sin\theta_z & \cos\theta_z & 0 \\ 0 & 0 & 1 \end{pmatrix}$.

5.) Translate the curved plate to its final destination.

(a) [6 points] Specify a nonlinear geometric mapping function that determines the "plate curvature" mapping from step 1. Apply your mapping to the plate in its "initial state" in order to produce two separate plots of curved plates with radii of curvature of 2.9 m and 3.1 m. (MATLAB's `surf` function can be helpful for this type of plot.)

(b) [8 points] Using equations wherever possible, explain your approach for computing the 7 parameters required by the robot arm to perform steps 1-5 from above.

(c) [10 points] Use a nonlinear least squares algorithm that can impose constraints on parameters to find the "best-fit" values for the 7 parameters. For example, you may use `lsqnonlin` from the Matlab Optimization Toolbox. Note that by default `lsqnonlin` will numerically approximate the Jacobian required by the nonlinear least-squares algorithm.

List the best-fit parameters and the norm of the residual associated with your solution. Plot the plate with its alignment points superimposed for the sequence of six different configurations: the initial configuration and after each of step 1 through 5. Also, superimpose the target points in the sixth figure.

Question 7 [5 points]

Write "AM205" by interpolating control points with cubic splines. For full credit, your letters/numbers should be clearly readable. Include two spline plots in your write-up: one that shows the spline curve and control points and another that just shows the spline curve. (You may use MATLAB's `spline` function if you wish.)