Department of Scientific Computing

Written Preliminary Examination

Spring 2024

May 20 – May 23, 2024

Instructions:

- Write your Student ID (which is the Employee ID, a 9 digits string) on each page of your answer sheets. Do *not* write your name on your answer sheets. When turning in your exam, include a cover page with your name and Student ID.
- Solve only 10 of the 11 questions as completely as you can. All questions are weighted equally.
- If you use web sources, please list them clearly.
- The exam starts at 1:00 pm on May 20, 2024 and the answers are due no later than 1:00 pm on May 23, 2024; no exceptions allowed. If you have a 50% time extension, the deadline is 1:00 am on May 25, 2024. Your answers can be prepared electronically (saved in PDF format), or you can take photos of your answers (please name the photos properly).
- To submit the answers, create a folder ∼/prelim2024/ in your home directory on `pamd.sc.fsu.edu`. Inside ∼/prelim2024/, create one folder for each question, and the folders should be named as "Q#-studentID/", for example, "Q1-studentID" for question 1 and "Q2-studentID/" for question 2. Upload your answers and code in the corresponding folders. Some software for uploading files are WinSCP (Windows), FileZilla (Windows, Mac, and Linux), and scp (Linux).
- If you have any questions related to this exam as you work on it, please send an e-mail to the person responsible, *and* Dr. Chen Huang (chuang3@fsu.edu).

Q1. **Fast Fourier Transformation** (Dr. Meyer-Baese, ameyerbaese@fsu.edu)

For a prime factor FFT the following 2D DFT is used:

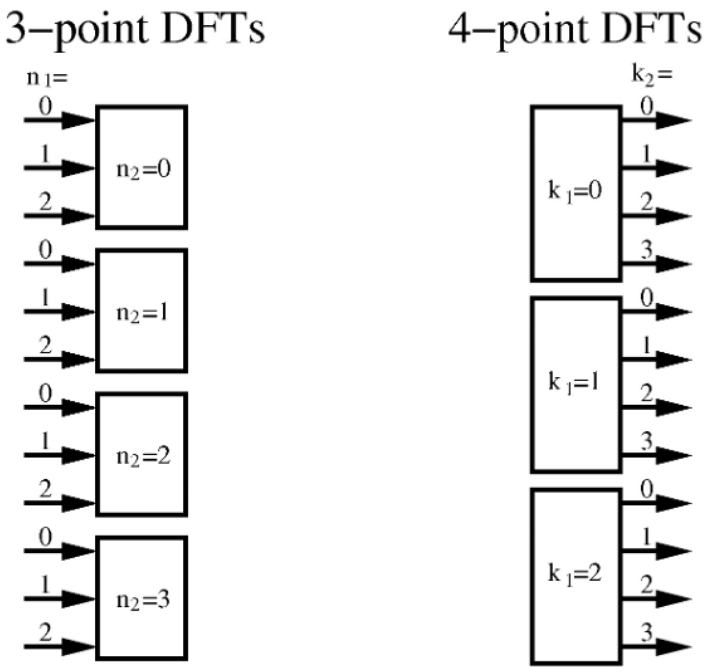
$$X[k_1,k_2]=\sum_{n_2=0}^{N_2-1} W_{N_2}^{n_2k_2}\left(\sum_{n_1=0}^{N_1-1} x[n_1,n_2]W_{N_1}^{n_1k_1}\right)$$

1. Complete the following table for the index map for a $N=12$ with $N_1=3$ and $N_2=4$ FFT, with $n=(4n_1+3n_2)$ mod 12, and $k=(4k_1+9k_2)$ mod 12 (40 points):

n_1	n_2			
	0	1	2	3
0				
1				
2				

k_1	k_2			
	0	1	2	3
0				
1				
2				

2. Complete the SFG (for $x[n]$, $X[k]$ and connection between first and second stage) for the FFT: (60 points)



Q2. **Linear Algebra** (Dr. Quaife, bquaife@fsu.edu)

Given a set of data points $(x_i, y_i) \in \mathbb{R}^2$, $i = 1, \ldots, N$, and a collection of basis functions $\phi_j : \mathbb{R} \to \mathbb{R}$, $j = 1, \ldots, L$, the method of least squares seeks to solve the optimization problem

$$\min_{\mathbf{w} \in \mathbb{R}^L} \left(\sum_{i=1}^{N} \left(y_i - \sum_{\ell=1}^{L} w_\ell \phi_\ell(x_i) \right)^2 \right),$$

where $\mathbf{w} = (w_1, w_2, \ldots, w_L)^T$.

a) The optimization problem can be written as

$$\min_{\mathbf{w} \in \mathbb{R}^L} \|\mathbf{y} - X\mathbf{w}\|^2,$$

where $\mathbf{y} = (y_1, \ldots, y_N)^T$, and X is an $N \times L$ matrix. Write down the matrix X. (15 points)

b) It is often a good idea to control the size of the weights w_ℓ. This can be done by solving the regularized optimization problem

$$\min_{\mathbf{w} \in \mathbb{R}^L} \left(\|\mathbf{y} - X\mathbf{w}\|^2 + \lambda \|\mathbf{w}\|^2 \right),$$

where $\lambda > 0$ is a parameter. One way to solve this optimization problem is to solve the normal equations

$$(X^T X + \lambda I)\mathbf{w} = X^T \mathbf{y},$$

where I is the $L \times L$ identity matrix. Using the normal equations, write a code that takes the data points (x_i, y_i) and the regularization parameter λ as inputs and solves the least squares problem. Your code should output (1) the error of the model ($\|\mathbf{y} - X\mathbf{w}\|^2$), and (2) the norm of the coefficients ($\|\mathbf{w}\|^2$). You can solve the normal equations using any direct solver for linear systems that you want. Use the basis functions

$$\phi_j(x) = \exp\left(-1000(x - c_i)^2\right), \quad i = 1, \ldots, 100,$$

where the points $c_i = (i-1)/99$ consist of the 100 evenly spaced points in $[0, 1]$. (40 points)

c) Apply your code to solve the least squares problem with the data $y_i = \sin(2\pi x_i) + 0.1 \sin(20\pi x_i)$, where x_i and $i = 1, \ldots, N$, are sampled from the uniform distribution in $[0, 1]$. Complete the second and third columns of the table. Discuss the relationship between λ and the results in each of these columns. Your discussion must include the general trends and the reason the trends are observed. (30 points)

λ	$\|\mathbf{y} - X\mathbf{w}\|^2$	$\|\mathbf{w}\|^2$	GMRES Iterations
0			
10^{-8}			
10^{-6}			
10^{-4}			
10^{-2}			
10^{0}			
10^{2}			

d) We are solving a small problem, so a direct solver for the normal equations is possible. However, if L were too large, this would be inefficient. Use unpreconditioned GMRES to solve the normal equations and complete the fourth column of the table by reporting the number of iterations required to achieve a tolerance of 10^{-12}. Use `gmres` (Matlab) or `scipy.sparse.linalg.gmres` (Python). Discuss the relationship between λ and the number of required GMRES iterations. Your discussion must include the general trend and the reason the trend is observed. (15 points)

Q3. **Ordinary Differential Equation** (Dr. Quaife, bquaife@fsu.edu)

Consider the system of initial value problems (IVPs)

$$\begin{aligned}\frac{dL}{dt} &= f_1(L,X,Y) = -4\alpha X - 4800L,\\ \frac{dX}{dt} &= f_2(L,X,Y) = -5L(Y-0.01) - 330X,\\ \frac{dY}{dt} &= f_3(L,X,Y) = 5LX - 330(Y-0.1),\end{aligned}$$

where $\alpha > 0$ is a parameter. We call a point (L_0, X_0, Y_0) a fixed point if $f_i(L_0, X_0, Y_0) = 0$, for $i = 1, 2, 3$. They are called fixed points because if the solution reaches one, it will remain there indefinitely. This system of IVPs has three fixed points at $(L_0, X_0, Y_0) = (0, 0, 0.1)$ and

$$(L_0, X_0, Y_0) = \left(\tilde{L},\ -\frac{1200}{\alpha}\tilde{L},\ 0.1 - \frac{200}{11\alpha}\tilde{L}^2\right), \quad \text{where } \tilde{L} = \pm 66\sqrt{\frac{\alpha}{880,000} - 1}.$$

a) Write a solver for this system of IVPs in the time interval $t \in [0, 1]$ using `ode45` (Matlab) or `solve_ivp` (Python). Use the initial condition $(L(0), X(0), Y(0)) = (0.01, 0.01, 0.1)$. Note that this initial condition is near one of the fixed points. (30 points)

b) Use your code to solve the system of IVPs for $\alpha = 1 \times 10^6$, $\alpha = 3 \times 10^6$, and $\alpha = 2 \times 10^7$. Plot the solution for each value of α, and indicate where in the plot each of the three fixed points are located. These should be three-dimensional plots with each axis representing one of the dependent variables. Describe the general behavior of the solution for each value of α. (10 points)

c) Write down the Jacobian for this problem. The Jacobian is the matrix of first-order derivatives of the right-hand side of the system of IVPs (30 points)

$$J = \begin{bmatrix} \frac{\partial f_1}{\partial L} & \frac{\partial f_1}{\partial X} & \frac{\partial f_1}{\partial Y} \\ \frac{\partial f_2}{\partial L} & \frac{\partial f_2}{\partial X} & \frac{\partial f_2}{\partial Y} \\ \frac{\partial f_3}{\partial L} & \frac{\partial f_3}{\partial X} & \frac{\partial f_3}{\partial Y} \end{bmatrix}.$$

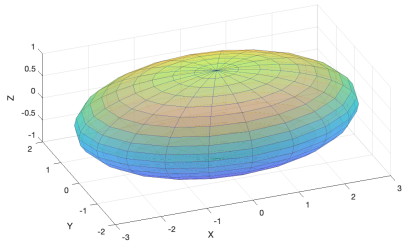
d) When the solution is near one of the fixed points, its behavior can be analyzed by considering the eigenvalues of the Jacobian evaluated at each of the fixed points. Write a code that computes the three eigenvalues of the Jacobian for each value of α, and report them in your writeup. (15 points)

e) Describe how the eigenvalues explain what you observe in part b). In particular, how do the eigenvalues explain the observed behavior when the solution is near a fixed point? (15 points)

Q4. **Numerical Integration** (Dr. Huang, chuang3@fsu.edu)

Problems can be programmed with any languages. Make sure to send your code with detailed instructions to chuang3@fsu.edu.

a) An ellipsoid is defined by the equation $x^2/a^2 + y^2/b^2 + z^2/c^2 = 1$. Write a program to calculate the volume of the ellipsoid shown below ($a = 3$, $b = 2$, $c = 1$) using the accept-reject Monte Carlo method. Note that a point is inside the ellipsoid if



$x^2/a^2 + y^2/b^2 + z^2/c^2 < 1$. The accuracy of the Monte Carlo method depends of the number of samplings (N). Perform calculations for different N values and demonstrate that the error decays as $1/\sqrt{N}$. The exact volume of an ellipsoid is $\frac{4}{3}\pi abc$. (50 points)

b) Use Gauss-Legendre quadrature to calculate the following integration:

$$\int_{-5}^{0} (x^2 + 1)dx \tag{1}$$

Note that we need to first transform the integration domain to [-1,1] for using the Gauss-Legendre quadrature. Report the results for different numbers of nodes. You can use any programs to generate the weights and nodes. (30 points)

c) Gauss-Hermite quadrature can be used to calculate integrals of the form $\int_{-\infty}^{\infty} e^{-x^2} f(x)dx$. Use the change of variables technique to transform the following integral in that form, and calculate it using Gauss-Hermite quadrature:

$$\int_{-\infty}^{\infty} e^{-(2x-1)^2}(x^2 + 1)dx \tag{2}$$

The accuracy increases as the number of nodes are increased. Report the results for several different nodes. You can use any programs to generate the weights and nodes. (20 points)

Q5. **Approximation and Differentiation** (Dr. Huang, chuang3@fsu.edu)

Problems can be programmed with any languages. Make sure to send your code with detailed instructions to chuang3@fsu.edu.

Let's expand $f(x) = e^{-x^2}\sin(x-1)+1$ in the domain [-1,1] with Legendre polynomials $P_i(x)$ as $f(x) \approx \sum_{i=0}^{n} c_i P_i(x)$. The coefficients $\{c_i\}$ can be obtained by minimizing the objective function

$$W = \int_{-1}^{1} [f(x) - \sum_{i=0}^{n} c_i P_i(x)]^2 dx$$

a) Derive the expression for $\partial W/\partial c_i$. (15 points)

b) Since Legendre polynomials of different degrees are orthogonal over [-1,1], we can derive c_i based on $\partial W/\partial c_i = 0$. Show your derivations. (15 points)

c) Based on the results from (b), calculate c_i for the first five Legendre polynomials: $P_0(x) = 1$, $P_1(x) = x$, $P_2(x) = (3x^2-1)/2$, $P_3(x) = (5x^3-3x)/2$, and $P_4 = (35x^4 - 30x^2 + 3)/8$. You can use scipy, MATLAB, or any other packages for the calculations involving integration. Submit the code and report $\{c_i\}$. Compare $f(x)$ and $\sum_{i=0}^{n} c_i P_i(x)$ by plotting them in one figure. (20 points)

d) $\{c_i\}$ can also be obtained by directly minimizing W. Based on the gradients $\partial W/\partial c_i$ derived from (a), write a program to minimize W. You can use the optimization packages from scipy, MATLAB, or other programs. Make sure to employ the gradients in the optimization. Submit your code and $\{c_i\}$. (30 points)

e) In ACS I, we learned that the complex step differentiation is a powerful scheme that can take very small step size. Modify your code to calculate $\partial W/\partial c_i$ using the complex step differentiation. Minimize W using the new gradient method. Submit your code and report $\{c_i\}$. (20 points)

Q6. **Data Structures** (Dr. Erlebacher, gerlebacher@fsu.edu)

Problem Statement: Quadtree for image compression

1. Notes

You may use AI to solve the problem. However, you *must* provide links to your conversations and provide all your sources. Remember that AI can be used to translate from one language to another. Use Copilot with your favorite IDE (e.g., Visual Studio Code) to help you write and document your code with lower error rate.

2. Background

Quadtrees are highly efficient for spatial data compression, spatial indexing, and more. In this problem, you will construct a datastructure to compress large images. You will write code in Python and C++ to accomplish this, and display the results graphically using either C++ or Python (not both). You are provided with three input images (included in the zipped file sent to you) to process and display with the quadtree.

3. Problem Description

Develop a quadtree structure to store an input image given in jpeg format. You will perform the following tasks:

1. Create a function to read an image in jpeg format, and convert it to grayscale. Consider reading the image using Python, saving it to an floating point array after converting the image to black and white. Consider PIL in Python. To read jpeg from C++, consider storing the image in some intermediate binary format with Python, and reading it from C. A portable method would be to output the data with one float per pixel from Python, and read this array from C++. Each pixel would have a single grayscale value between 0 and 1. At least one of the images is in color. The RGB values are converted to grayscale using the NTSC formula:

$$\text{gray} = 0.299 \times \text{Red} + 0.587 \times \text{Green} + 0.114 \times \text{Blue}$$

   where R, G, B are scaled between 0 and 1.

2. In each leaf, store lower-left corner (xl, yl), width and height (w, h) (as integers) of the region encompassed by the leaf. If the image is of size $M \times N$, the root leaf would store the lower left-hand corner of the region as (0,0) and the weight/height as $(w, h) = (M, N)$. Thus, one is storing the number of pixels in the horizontal and vertical axes of the enclosed region.

3. The first input parameter is the a threshold T for the pixels enclosed by a node. If the standard deviation of the pixels within the enclosed region is less than T, further node subdivision is not required. Consider values for T in the range [.0001 to 0.01].

4. The second input parameter is the maximum depth, D, of the quadtree. Consider D in the range 4 through 12. What is the impact of D on the resolution of the finest (smallest) regions stored in leaves of the graph?

5. Each node (whether a leaf node or not) will store the average gray value (between 0 and 1), and standard deviation of the pixels the region represents. Each node should also store the depth of the node (the root has depth zero).

6. All functions should be documented (both in Python and C++), with function arguments described. Add inline comments as necessary to make it easier to read your code.

4. Tasks to Perform

1. Create a quadtree function that takes as input, the given image, and generates as output the quadtree that stores a compressed representation of the image. (C++ and Python) (15 points)

2. During the construction of the quadtree, construct a doubly-linked list that stores pointers to the leaf nodes. As the quadtree grows in size, the doubly-linked list expands as well. (C++ and Python) (15 points)

3. Implement a function to display the image using each of the data structures (quad-tree and linked list):

   (a) running through the quadtree and displaying the leaves, and

   (b) traversing the linked list and displaying its nodes.

   Display the image from the mean values, and from the standard deviation of colors represented by the leaves. (10 points)

4. Subtract the compressed image from the original image, and plot an image of the absolute value of this difference, labeled E. (5 points)

5. Create a plot of this error as a function of the threshold T for a fixed value of the depth D. (10 points)

6. Create a plot of this error as a function of the depth D for a fixed value of the threshold T. (10 points)

7. Repeat the two plots, but plot the compression ratio rather than the error. Define compression ratio as the number of pixels divided by the number of leaves. Therefore, if the are 10^6 pixels in the image and 10^3 leaves, the compression ratio would be $1000:1$ (1000 to 1). (5 points)

8. Add a timer and compare the times taken to construct and display the quadtrees (with the quad-tree and the linear list). (5 points)

9. Provide a description of how the two data structures will be used along with an estimate of asymptotic complexity for both memory (amount of storage in bytes) and time (number of floating point operations). State any approximations you are using to arrive at an answer. (15 points)

10. Finally, in your own words, discuss what you have learned, and propose a series of optimization to lower computational and/or reduce memory usage. (10 points)

Q7. **Maximum Likelihood Estimation** (Dr. Chipilski, hchipilski@fsu.edu)

Problem setup and deliverables

The topic of maximum likelihood estimation (MLE) was covered in the first part of your ACS II course and will be further practiced in this question. You can find the iid[1] sample $\{X_i\}_{i=1}^{1000}$ from `lognormal_sample.txt` in the zipped file sent to you. You will be asked to provide MLE parameter estimates for a lognormal distribution with a probability density function (pdf)

$$p(x_i;\mu,\sigma^2) = \frac{1}{x_i(2\pi\sigma^2)^{1/2}} \exp\Big(-\frac{(\ln x_i-\mu)^2}{2\sigma^2}\Big). \tag{3}$$

Analytical solutions

In some cases, such as the lognormal model considered in this problem, it is possible to obtain closed-form MLE solutions. You will see how this can be done by following the steps below.

a) Write down the likelihood function $L(\boldsymbol{\theta};x_1,\ldots,x_n)$. *Hint: Use the independence property and Eq.* (3) *to factorize the joint pdf* $p(x_1,...,x_n;\mu,\sigma^2)$. (10 points)

b) Show that the associated log likelihood function is given by

$$l(\mu,\sigma^2;x_1,\ldots,x_n) \coloneqq \ln L = -\frac{n}{2}\ln(2\pi) - \frac{n}{2}\ln(\sigma^2) - \frac{1}{2\sigma^2}\sum_{i=1}^{n}\Big[\ln(x_i)-\mu\Big]^2 - \sum_{i=1}^{n}\ln(x_i).$$

Hint: The process of deriving $l(\mu,\sigma^2;x_1,\ldots,x_n)$ *is very similar to the Gaussian case – refer to ACS II lecture notes and other web resources.* (10 points)

c) Find MLE expressions for $\hat{\boldsymbol{\theta}} = [\hat{\mu} \quad \hat{\sigma}^2]^\top$ by solving the problem

$$\max_{\mu,\sigma^2} \; l(\mu,\sigma^2;x_1,\ldots,x_n).$$

Hint: To find $\hat{\boldsymbol{\theta}}$*, you need to impose first-order conditions for achieving a maximum in* $l(\boldsymbol{\theta};x_1,\ldots,x_n)$*. Note that it is more convenient to compute* $\hat{\mu}$ *before* $\hat{\sigma}^2$*.* (30 points)

d) Use a scripting language (e.g., Python or MATLAB) to apply the MLE expressions from part c) to the data in `lognormal_sample.txt`. Report your MLE estimates $\hat{\mu}$ and $\hat{\sigma}^2$ with accuracy up to 5 decimal points. Plot a histogram of `lognormal_sample.txt` and overlay the pdf from Eq. (3) using the calculated MLE estimates $\hat{\mu}$ and $\hat{\sigma}^2$. What do you observe? (20 points)

[1] independent and identically distributed

Numerical solutions

In other cases, the MLE solutions are not analytically tractable and must be obtained numerically via gradient-based optimization techniques. For our lognormal model in Eq. (3), this can be done by defining the negative log likelihood function $\tilde{l} \coloneqq -l$ and solving the following problem

$$\min_{\mu,\sigma^2} \ \tilde{l}(\mu, \sigma^2; x_1, \ldots, x_n).$$

After setting the initial parameter values to $\boldsymbol{\theta}_0 = [\mu_0 \ \sigma_0^2]$, a numerical solution to the MLE problem is obtained by iteratively updating the parameter estimates according to

$$\boldsymbol{\theta}_{i+1} = \boldsymbol{\theta}_i - \alpha \ \nabla_{\boldsymbol{\theta}=\boldsymbol{\theta}_i} \tilde{l},$$

where the first component of the 2D gradient vector $\nabla_{\boldsymbol{\theta}=\boldsymbol{\theta}_i} \tilde{l}$ is $\frac{\partial \tilde{l}}{\partial \mu}\Big|_{\mu=\mu_i}$, whereas the second one is $\frac{\partial \tilde{l}}{\partial \sigma^2}\Big|_{\sigma^2=\sigma_i^2}$ (*Hint: These derivatives can be readily obtained from your results in part c).* The parameter α is the step size and determines how far the gradient we descend.

e) Implement the numerical method described above using $\mu_0 = 10$, $\sigma_0^2 = 5$, and $\alpha = 0.001$. How many iterations are necessary to converge to the analytical MLE estimates calculated in part d)? To help answer this question, plot the values of $\tilde{l}_i$, μ_i and σ_i^2 at each iteration number i. (30 points)

Instructions

You are allowed to utilize your favorite scripting language (e.g., Python or MATLAB), but make sure to submit all relevant code together with your analytical calculations and written discussions. I expect you to disclose all resources used to solve this MLE problem, including any interactions with AI tools like ChatGPT (please attach them as part of your report).

Q8. **Parallel Programming** (Dr. Zavala, osz09@fsu.edu)

Parallel integration of $h(x) = e^x$ over the interval [0,1]

- (OpenMP) Using Simpson's rule, write a parallel program to integrate $h(x) = e^x$ over the interval $[0, 1]$. Divide the interval into 10^3 sections and parallelize the computation using OpenMP. Plot your computation time versus the number of threads used, ranging from 2 to 10 (indicate the number of CPUs/threads your computer has). Submit your code and explain your results. (30 points)

- (MPI) Using Simpson's rule, write a parallel program to integrate $h(x) = e^x$ over the interval $[0, 1]$. Evenly divide the interval into $2N_p$ subintervals, where N_p is the number of processes. Each process i $(i = 0, 1, \ldots, N_p - 1)$ is responsible for two function evaluations at x_{2i}, x_{2i+2} and computes the area A_i using Simpson's rule:

$$A_i = \frac{\Delta x}{6} \left(h(x_{2i}) + 4h(x_{2i+1}) + h(x_{2i+2}) \right)$$

  The process i should only evaluate the function $h(x)$ at x_{2i} and x_{2i+1} and get $h(x_{2i+2})$ from the process $i+1$. In this way, for an expensive evaluation of h, the computational time could be reduced almost by one third. In this algorithm, the processes send data to each other in a ring-like fashion, except for the last process, which calculates h at $x_{2(N_p-1)}, x_{2(N_p-1)+1}$, and $x_{2(N_p-1)+2} = x_{2N_p}$. Write an MPI program implementing this algorithm. Run your code simulating 10 processes and write the individual areas obtained by each process A_i as well as the full approximation. (70 points)

Since we are living in a world with ChatGPT 4o and have not changed our policies, I will *carefully consider* any errors.

Q9. **Statistics** (Dr. Beerli, pbeerli@fsu.edu)

Analyze the two data set in the directory on *pamd:/research/pbeerli/prelim2024/bacteria*. The data are two time series of bacterial growth; in biology, estimating a growth parameter and considering the maximum number of individuals that can be sustained for a long time is common. The start population was of size 100 bacteria for each of the experiments. I expect working code and explanations of your findings; I expect the result in a notebook that could be used as a tutorial; the language is up to you, Python, Julia, R, or others if you can ensure I can run the notebook. Do not use any statistical packages for your analysis code; you can use any package for the plotting routines.

1. Discuss and show the model that you want to use (I expect at least two parameters) (10 points)
2. Plot a histogram of the data (into one plot to compare the raw data – explain the differences) (10 points)
3. Write a Markov-Chain-Monte Carlo code to analyze the data and estimate your parameters; do not use any prepackaged statistics or Monte Carlo package; I expect you to code the Metropolis algorithm at a minimum, although I prefer the Metropolis-Hastings. Make sure that I understand your code (with Python, you can use numpy). Your output should show the posterior distribution of your parameters; this posterior distribution can then be used to define support intervals. (30 points)
4. Code an alternative approach, for example, use expectation-maximization or variational Bayes, rejection-sampling, or others; again, I want to see code; do not use calls to packages like scipy or stats packages. (20 points)
5. Show the results of your two approaches, discuss them, and plot them with the raw data. Maximum points will need support intervals and an extensive description of the results (30 points)

Instructions:

- I do not expect that you write all the code by yourself, but be creative in what you use; copying/pasting code is sometimes more cumbersome than writing your own.
- You can use any tool (including AI) you find on the internet to get the answers (but do not ask your friend!)
- YOU MUST GIVE A LIST OF ALL THE TOOLS, WEBSITES, ARTICLES YOU USED TO INVESTIGATE THE ANSWERS AND CODES.
- I expect a zip file with a working Jupyter notebook that includes all codes, results, and discussions submitted to me at (pbeerli@fsu.edu).
- Grading: Total 100 Points: 10 points for defining and describing the model; 10 points for a description of the data using a plot; 30 Points for coding a correct MCMC algorithm; 20 points for coding a correct alternative algorithm; 30 points for the analysis of the data using your tools, including a description of your results (plot and potentially table).

Q10. **Numerical Integration** (Dr. Sachin, sshanbhag@fsu.edu)

Consider the continuous function,

$$f_n(x) = \sum_{j=1}^{n} \frac{|\sin(j\pi x)|}{j}$$

with $n = 30$, defined over the domain $x \in [0, 1]$. Answer the following questions:

a) Find all the locations at which the *derivative* $f_n'(x)$ is discontinuous. Use this information to plot $f_n(x)$ so that its key features are visible. (20 points)

b) Now consider the integral,

$$I = \int_0^1 f_n(x)\, dx.$$

Choose any numerical method to approximate I with sufficient accuracy. Explain your choice. For this part, you are allowed to use software from any numerical library. (60 points)

c) It is straightforward to find an analytical formula for I. Compare the analytical result with your numerical estimate. **Hint**: Consider each term in the summation separately. You may use symbolic software for this question. (20 points)

Q11. **Parallel Programming** (Dr. Wang, wwang3@fsu.edu)

Goldbach's conjecture is one of the oldest and best-known unsolved problems in the number theory of mathematics. Every even integer greater than 2 can be expressed as the sum of two primes.

Please write a parallized C or C++ code with OpenMP to verify Goldbach's conjecture for all the even integers less than ten million (10,000,000).

Try your code with different number of cores and plot scalability, i.e., the performance of your code vs #cores.

Grading: Correctness of the serial code (30 pts); correctness running of the paralized code (30 pts); load balancing consideration (20 pts); good scalability of your code (20 pts). 5 to 50 pts may be deducted for each bug found in your code.