Department of Scientific Computing

Written Preliminary Examination

Summer 2023

May 22 – 25, 2023

Instructions:

- Solve only 10 of the 11 questions as completely as you can.
- All questions are weighted equally.
- All parts of a question are weighted equally unless stated otherwise.
- You must score 75% or more on 7 or more questions to pass the written portion of the preliminary exam.
- If you use web sources, please list them clearly.
- The exam is due back to Karey Fowler no later than 12 noon on Thursday May 25, 2023; no exceptions allowed.
- If you have questions related to this exam as you work on it, please send an email to the person responsible, **and** Dr. Xiaoqiang Wang (wwang3@fsu.edu). The person responsible is listed at the beginning of each question.
- If a question asks for code or other attachments, please email both the person responsible and Dr. Xiaoqiang Wang.
- Write your Student ID on each of your answer sheets. When turning in your exam, include a cover page with your name and Student ID.

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

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

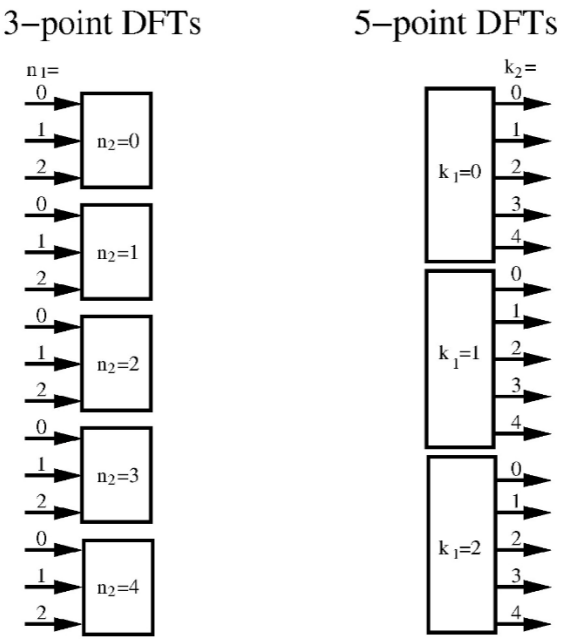
$$X[k_1,k_2]=\sum_{n_2=0}^{N_2-1}W_{N_2}^{n_2k_2}\left(\sum_{n_1=0}^{N_1-1}x[n_1,n_2]W_{N_1}^{n_1k_1}\right)$$

(a) (40 pts) Complete the following table for the index map for a $N=15$ and $N_1=3$ and $N_2=5$ FFT with: $n=5n_1+3n_2 \bmod 15$, and $k=10k_1+6k_2 \bmod 15$

n_1	n_2				
	0	1	2	3	4
0					
1					
2					

k_1	k_2				
	0	1	2	3	4
0					
1					
2					

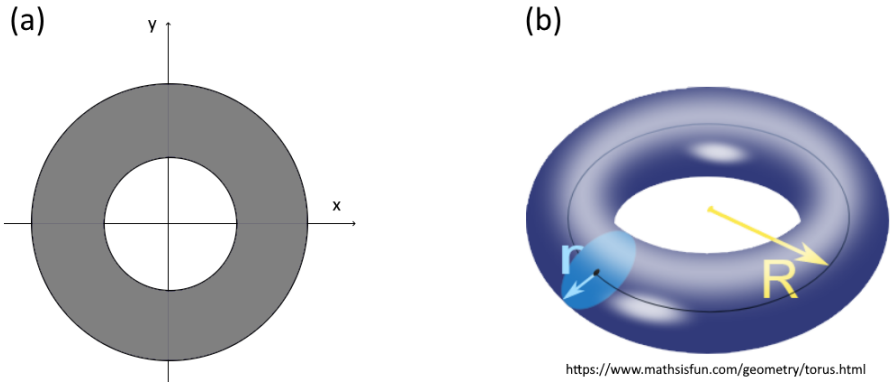
(b) (60 pts) Complete the SFG (for $x[n]$, $X[k]$ and connection between first and second stage) for the FFT:



Q2. **Monte Carlo Integration and Parallel Programming** (Dr. Huang, chuang3@fsu.edu)

Problems (a) and (c) can be programmed with any languages, such as C, FORTRAN, Python, and MATLAB. Problem (b) should be programmed with either C or FORTRAN using MPI. Make sure to send your code with detailed instructions on how to run it.

(a) (30 pts) Write a program to use the Monte Carlo method to calculate the area of the ring in Figure (a). The radius of the inner circle is 10, and the radius of the outer circle is 20. The exact value is 300π, based on which we can calculate the error E for a given number of samplings N. Make a plot to demonstrate the decay rate $E \sim 1/\sqrt{N}$.



(b) (30 pts) Please use OpenMP to parallelize the code above. Plot the performance of your code with regard to 1, 2, 3, 4 cores.

(c) (40 pts) A torus is defined by two radii, a major radius R and a minor radius r, as shown in Figure (b). Its volume exact value is $(\pi r^2)(2\pi R)$. A point is inside the torus if $(R - \sqrt{x^2 + y^2})^2 + z^2 < r^2$. Write a Monte Carlo program to estimate the volume of a torus with $R = 5$ and $r = 2$. Compare your results with the exact value.

Q3. **Ordinary Differential Equation** (Dr. Shanbhag, sshanbhag@fsu.edu)

(a) (25 pts) Consider the initial value problem (IVP),

$$\frac{dy}{dt} = -|y|, \qquad y(0) = 1,$$

which has the solution $y(t) = e^{-t}$. The presence of the absolute sign does not affect the theoretical solution. However, it affects the numerical solution as we shall see.

Using any adaptive explicit solver (like ode45 in Matlab or similar) with default settings, compute the numerical solution to this IVP for $t \in [0, 40]$ and show that the solution becomes negative and eventually blows up. Explain why this happens.

Use the same IVP solver for all the parts below.

(b) (15 pts) A standard trick to strictly enforce non-negativity is to employ the substitution, $y = e^Y$, which transforms the ODE to,

$$Y' = -|e^Y|/e^Y = -1, \qquad Y(0) = 0,$$

Solve for $Y(t)$ numerically, and hence plot $y(t) = e^{Y(t)}$.

(c) (25 pts) Enforcing non-negativity is important in modeling chemical kinetics, where negative concentrations are unphysical, and can lead to solutions that blow up. Consider the system,

$$\begin{aligned} \frac{dc_1}{dt} &= 0.5c_1(1 - 0.05c_1) - 0.1c_1c_2, & c_1(0) &= 25 \\ \frac{dc_2}{dt} &= 0.01c_1c_2 - 0.001c_2, & c_2(0) &= 5. \end{aligned}$$

Solve and plot $c_1(t)$ and $c_2(t)$ for $t \in [0, 870]$. Explain your observations.

(d) (35 pts) Apply the standard trick mentioned in the second part with $c_1 = e^{y_1}$ and $c_2 = e^{y_2}$, and show that a non-negative solution can be obtained for $t \in [0, 1000]$ without any blow up.

Q4. **Approximation** (Dr. Wang, wwang3@fsu.edu)

Approximate the function

$$f(x) = \frac{\tanh(10x-5)) - \tanh(10x+5))}{2},$$

defined only on the interval $[-1, 1]$ using a set of basis functions $\{e_0(x), e_1(x), e_2(x), \ldots, e_n(x)\}$, by minimizing the L_2 norm:

$$\int_{-1}^{1} [f(x) - p_n(x)]^2 \, dx, \quad \text{where } p_n(x) = \sum_{i=0}^{n} c_i e_i(x).$$

We need to calculate the coefficients c_i.

(a) (40 pts) Suppose we use Legendre polynomials as a basis, so that:

$$p_n(x) = \sum_{i=0}^{n} c_i P_i(x),$$

where $P_i(x)$ are Legendre polynomials, for which the inner-product,

$$\langle P_i, P_j \rangle = \int_{-1}^{1} P_i(x) P_j(x) dx = \frac{2}{2i+1} \delta_{ij}.$$

Using the idea of inner-products, write an algorithm to project $f(x)$ onto the Legendre polynomial basis, and determine the coefficients c_i for $n = 4$. Use numerical quadrature for the calculation if needed.

Note: The first few Legendre polynomials are,

$$\{1, x, \frac{1}{2}(3x^2-1), \frac{1}{2}(5x^3-3x), \frac{1}{8}(35x^4-30x^2+3)\}$$

(b) (10 pts) Plot $f(x)$ and compare it with $p_4(x)$. Plot $|f(x) - p_4(x)|$ as well.

(c) (30 pts) Suppose we use sin, cos functions as the basis, i.e., $\{1, \sin(\pi x), \cos(\pi x), \sin(2\pi x), \cos(2\pi x), ..., \sin(n\pi x), \cos(n\pi x)\}$, please use the same idea to calculate the approximation of $f(x)$ for $n = 4$.

(d) (10 pts) Plot $f(x)$ and compare it with the approximation.

(e) (10 pts) Compared to the first method using Legendre polynomials, to apply the second method using the $\sin, \cos$ functions, there is a special requirement for the $f(x)$, what is it? Does our function $f(x)$ satisfy it?

Q5. **Linear Programming** (Dr. Dexter, ndexter2@fsu.edu)

This question concerns convex optimization. Recall that a function $f : \mathbb{R}^n \to \mathbb{R}$ is *convex* if for any $0 \leq t \leq 1$ and $\boldsymbol{x}, \boldsymbol{y} \in \mathbb{R}^n$ then

$$f(t\boldsymbol{x} + (1-t)\boldsymbol{y}) \leq tf(\boldsymbol{x}) + (1-t)f(\boldsymbol{y}).$$

We say that a set $S \subseteq \mathbb{R}^n$ is *convex* if $t\boldsymbol{x} + (1-t)\boldsymbol{y} \in S$ for all $0 \leq t \leq 1$ and $\boldsymbol{x}, \boldsymbol{y} \in S$.
A *convex optimization problem* is a problem of the form

$$\text{minimize} \quad f(\boldsymbol{z}) \quad \text{over} \quad \boldsymbol{z} \in S,$$

where $f : \mathbb{R}^n \to \mathbb{R}$ is a convex function and S (known as the *feasible set*) is a convex set. For example, for the **basis pursuit problem**

$$\min_{\boldsymbol{z} \in \mathbb{R}^n} \|\boldsymbol{z}\|_1 \quad \text{subject to} \quad \boldsymbol{A}\boldsymbol{z} = \boldsymbol{b}, \tag{1}$$

where $\|\boldsymbol{z}\|_1 = \sum_{i=1}^n |z_i|$ for $\boldsymbol{z} \in \mathbb{R}^n$, the feasible set $S = \{\boldsymbol{z} \in \mathbb{R}^n : \boldsymbol{A}\boldsymbol{z} = \boldsymbol{b}\}$. Using this definition for S, equation (1) can be re-written as

$$\text{minimize} \quad f(\boldsymbol{z}) \quad \text{over} \quad \boldsymbol{z} \in S$$

where $f(\boldsymbol{z}) = \|\boldsymbol{z}\|_1$.

(a) (30 pts) Linear programs are some of the most fundamental types of convex optimization problems. A *linear program* is a special type of optimization problem taking the form

$$\min_{\boldsymbol{z} \in \mathbb{R}^p} \boldsymbol{c}^\top \boldsymbol{z} \quad \text{subject to} \quad \tilde{\boldsymbol{A}}\boldsymbol{z} = \boldsymbol{b},\ \boldsymbol{z} \geq \boldsymbol{0}. \tag{2}$$

Here $\boldsymbol{c} \in \mathbb{R}^p$, $\tilde{\boldsymbol{A}} \in \mathbb{R}^{m \times p}$, and $\boldsymbol{b} \in \mathbb{R}^m$. The inequality $\boldsymbol{z} \geq \boldsymbol{0}$ means that every component of $\boldsymbol{z}$ is nonnegative. Show this is a convex optimization problem and that the basis pursuit problem, equation (1), is a linear program. For the proof that basis pursuit is a linear program, see the Wikipedia page for Basis Pursuit, `https://en.wikipedia.org/wiki/Basis_pursuit#Equivalence_to_Linear_Programming`.

(b) (70 pts) This problem involves estimating the performance of solving the linear program derived in the previous step for different values of number of linear measurements $m \in M := \{10, 20, 30, 40, 50, 60, 70, 80, 90, 100\}$ and with varying levels of noise $\sigma \in \Sigma := \{10^{-4}, 10^{-3}, 10^{-2}, 10^{-1}\}$.

The setup for this programming problem is as follows. Let $T = 10$, $s = 10$, and $N = 128$. For each $m \in M$ and $\sigma \in \Sigma$ and each trial $t = 1, \ldots, T$, generate a new standard Gaussian random matrix $\boldsymbol{A}$, i.e., $A_{i,j} \sim \mathcal{N}(0, 1)$ for each $i = 1, \ldots, m$ and $j = 1, \ldots, N$, and set $\bar{\boldsymbol{A}} := \boldsymbol{A}/\sqrt{m} \in \mathbb{R}^{m \times N}$. Also for each trial, generate an s-sparse random vector $\boldsymbol{x} \in \mathbb{R}^N$ with nonzero elements uniform randomly distributed among its components, i.e., $\|\boldsymbol{x}\|_0 = s$ where $\|\boldsymbol{x}\|_0 := \#\{i : x_i \neq 0\}$ (the number of nonzero elements of $\boldsymbol{x}$). This can be achieved using the following MATLAB code.

```
% Generate a sparse vector x
x = zeros(n,1); % initialize x
x(1:s) = randn(s,1); % set the first s entries as unit normal
x = x(randperm(n)); % randomly permute the entries of x
```

Then for each trial, set $\boldsymbol{b} = \bar{\boldsymbol{A}}\boldsymbol{x} + \sigma\boldsymbol{e}/\|\boldsymbol{e}\|_2$ where $\boldsymbol{e} \in \mathbb{R}^m$ is a Gaussian random noise vector, i.e., $e_i \sim \mathcal{N}(0,1)$ for $i = 1, \ldots, m$ (also generated for each trial).

For each m, σ, run T trials solving problem (1) for $\boldsymbol{x}$ with the above computed $\boldsymbol{b}$ and $\bar{\boldsymbol{A}}$ using MATLAB's `linprog` method. Here you will use the linear program of the form (2) derived in the previous step to solve (1). Record the relative ℓ_2 error

$$\frac{\|\boldsymbol{x} - \hat{\boldsymbol{x}}\|_2}{\|\boldsymbol{x}\|_2}$$

for the solution $\hat{\boldsymbol{x}}$ obtained from `linprog` for each $m \in M$ and $\sigma \in \Sigma$ and trial t, and average over all trials T. Then plot the relative error with respect to m with a different line for each value of σ. How does the minimum value of the relative error relate to σ for each line? After approximately how many measurements m does each line achieve this minimum value?

Q6. **Optimization** (Dr. Dexter, ndexter2@fsu.edu)

(a) (25 pts) Let $f_1, \ldots, f_k : \mathbb{R}^n \to \mathbb{R}$ be convex functions. Show that the set

$$C = \{\boldsymbol{x} \in \mathbb{R}^n : f_i(\boldsymbol{x}) \leq 0,\ i = 1, \ldots, k\},$$

is a convex set.

(b) (40 pts) A *convex optimization problem* is a problem of the form

$$\text{minimize}\ \ f(\boldsymbol{z}) \quad \text{over} \quad \boldsymbol{z} \in S,$$

where $f : \mathbb{R}^n \to \mathbb{R}$ is a convex function and S (known as the *feasible set*) is a convex set.

Let $\boldsymbol{Q}_0, \ldots, \boldsymbol{Q}_k \in \mathbb{R}^{n \times n}$ be nonnegative definite matrices, $\boldsymbol{r}_0, \ldots, \boldsymbol{r}_k \in \mathbb{R}^n$, $s_0, \ldots, s_k \in \mathbb{R}$, $\boldsymbol{A} \in \mathbb{R}^{m \times n}$ and $\boldsymbol{b} \in \mathbb{R}^m$. A *quadratically-constrained quadratic program* takes the form

$$\begin{aligned} \text{minimize}\ & \frac{1}{2}\boldsymbol{x}^\top \boldsymbol{Q}_0 \boldsymbol{x} + \boldsymbol{r}_0^\top \boldsymbol{x} + s_0 \\ \text{subject to}\ & \frac{1}{2}\boldsymbol{x}^\top \boldsymbol{Q}_i \boldsymbol{x} + \boldsymbol{r}_i^\top \boldsymbol{x} + s_i \leq 0,\ i = 1, \ldots, k, \\ & \boldsymbol{A}\boldsymbol{x} = \boldsymbol{b}. \end{aligned} \tag{3}$$

Show that this is a convex optimization problem. Hint: use part (a) to show that the feasible set is convex.

(c) (35 pts) Using a suitable change of variables, show that the *quadratically constrained basis pursuit* (QCBP) problem

$$\text{minimize}\ \|\boldsymbol{x}\|_1 \quad \text{subject to} \quad \|\boldsymbol{A}\boldsymbol{x} - \boldsymbol{b}\|_2 \leq \eta, \tag{4}$$

can be converted to a problem of the form (3). Hint: see the proof of problem **Q5** part (a) from Wikipedia, and try performing some algebraic manipulation to convert the constraint of (4) into the desired form.

Q7. **Statistics** (Dr. Beerli, pbeerli@fsu.edu)

Analyze the data set in the file on *pamd:/research/pbeerli/prelim2023/gaussdata*. The data comes from a Gaussian Mixture distribution with k components, and different means μ_i, but have all the same standard deviation σ:

(a) (10 pts) Plot a histogram of the data

(b) (20 pts) Analyze the data using a standard package (for example, use those in Python or Julia), deliver an estimate of k and all the μ_i.

(c) (20 pts) Use the non-parametric bootstrap approach to analyze and give the estimates.

(d) (20 pts) Use a Bayesian approach the deliver the result. There are a few methods you could use; pick one!

(e) (20 pts) Describe the differences/commonalities of the three different estimators in not more than a page of text; which one do you trust most? Why?

(f) (10 pts) Give a list of all the tools, websites, and articles you used to investigate the answers and codes.

You are not expected 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 you find on the internet to get the answers (but do not ask your friend!).

A zip file that includes a working Jupyter notebook that includes all codes, results, and the discussion is expected to be submitted to Dr. Beerli and Dr. Wang.

Q8. **Linear Algebra** (Dr. Quaife, bquaife@fsu.edu)

Consider an underdetermined linear system $A\mathbf{x} = \mathbf{b}$ where $A \in \mathbb{R}^{m\times n}$, $\mathbf{x} \in \mathbb{R}^n$, $b \in \mathbb{R}^m$, and $m < n$. Assume that A has full row rank which means that $\text{rank}(A) = m$. This guarantees that $A\mathbf{x} = \mathbf{b}$ has infinitely many solutions for any right hand side $\mathbf{b}$. Here you will investigate several techniques to find a solution of $A\mathbf{x} = \mathbf{b}$ for two different matrices A. The matrices are

$$A_1 = \begin{pmatrix} 5 & 4 & 1 & 8 & 4 & 8 \\ 7 & 4 & 9 & 1 & 7 & 2 \\ 7 & 10 & 10 & 3 & 2 & 7 \end{pmatrix} \quad \text{and} \quad A_2 = \begin{pmatrix} 5 & 4 & 1 & 8 & 4 & 8 \\ 7 & 4 & 9 & 1 & 7 & 2 \\ -2 & 10^{-5} & -8 & 7 & -3 & 6 \end{pmatrix}.$$

For the right hand side, let

$$\mathbf{b} = \begin{pmatrix} -1 \\ 0 \\ 5 \end{pmatrix}.$$

You must submit your code for full credit. Use built-in routines or libraries to compute the necessary singular value decompositions and QR factorizations. You may use any language, but it is probably easiest to use an interpreted language such as Matlab or Python.

a) (10 pts) The singular value decomposition (SVD) of $A \in \mathbb{R}^{m\times n}$ is $A = U\Sigma V^T$, where $U \in \mathbb{R}^{m\times m}$ and $V \in \mathbb{R}^{n\times n}$ are orthogonal matrices (ie. $U^TU = UU^T = I_m$ and $V^TV = VV^T = I_n$), and $\Sigma \in \mathbb{R}^{m\times n}$ is diagonal with the singular values contained along its diagonal. For the examples you are considering, Σ has the form

$$\Sigma = \begin{pmatrix} \sigma_1 & 0 & 0 & 0 & 0 & 0 \\ 0 & \sigma_2 & 0 & 0 & 0 & 0 \\ 0 & 0 & \sigma_3 & 0 & 0 & 0 \end{pmatrix}.$$

The null space of a matrix A is the set of vectors $\mathbf{x}$ that satisfy $A\mathbf{x} = \mathbf{0}$. Because the ranks of A_1 and A_2 are both 3, each of their null spaces are spanned by $6 - 3 = 3$ vectors. Using a pencil and paper calculation, show that the last 3 columns of V form a basis of the null space of A. This part requires no code.

b) (10 pts) One way to find a solution of $A\mathbf{x} = \mathbf{b}$ is to define the pseudo-inverse $A^\dagger = A^T(AA^T)^{-1}$, and let

$$\mathbf{x} = A^\dagger\mathbf{b}.$$

For this choice of $\mathbf{x}$, compute the residual $\|A\mathbf{x} - \mathbf{b}\|$ for $A = A_1$ and $A = A_2$. Also, compute the condition number of AA^T. Explain why one residual is larger than the other.

c) (25 pts) Another way to find a solution of $A\mathbf{x} = \mathbf{b}$ is to compute the economy-size SVD $A = U\Sigma V^T$ where $U \in \mathbb{R}^{m\times m}$, $V \in \mathbb{R}^{n\times m}$, and $\Sigma \in \mathbb{R}^{m\times m}$ is diagonal with the singular values contained along its diagonal. In this decomposition, U is an orthogonal

matrix, and V satisfies $V^TV = I_m$, but $VV^T \neq I_n$. For a general underdetermined system, use a pencil and paper calculation to show that

$$\mathbf{x} = V\Sigma^{-1}U^T\mathbf{b}$$

is a solution of $A\mathbf{x} = \mathbf{b}$. For this choice of $\mathbf{x}$, compute the residual $\|A\mathbf{x} - \mathbf{b}\|$ for $A = A_1$ and $A = A_2$.

d) (25 pts) Another way to find a solution of $A\mathbf{x} = \mathbf{b}$ is to use the QR decomposition of A with partial pivoting, $A = QRP^T$, where $Q \in \mathbb{R}^{m\times m}$ is an orthogonal matrix, $R \in \mathbb{R}^{m\times n}$ is an upper triangular matrix, and P is a permutation matrix. Compute the solution

$$\mathbf{x} = P\begin{bmatrix}\mathbf{y}\\ \mathbf{0}\end{bmatrix},$$

where $\tilde{R}\mathbf{y} = Q^T\mathbf{b}$, $\tilde{R}$ is the matrix containing the first m columns of R, and the vector $\mathbf{0}$ is the zero vector with $n - m$ elements. For this choice of $\mathbf{x}$, compute the residual $\|A\mathbf{x} - \mathbf{b}\|$ for $A = A_1$ and $A = A_2$.

e) (25 pts) A final way to find an approximate solution is to let $\mathbf{x}$ be the minimum of the regularized problem

$$\min_{\mathbf{x}\in\mathbb{R}^n} \left(\|A\mathbf{x} - \mathbf{b}\|^2 + \lambda\|\mathbf{x}\|^2\right),$$

where $\lambda > 0$ is a parameter. The solution of this minimization problem is

$$\mathbf{x} = (A^TA + \lambda I)^{-1}A^T\mathbf{b},$$

where I is the $n \times n$ identity matrix. For this choice of $\mathbf{x}$, compute the residual $\|A\mathbf{x}-\mathbf{b}\|$ for $A = A_1$ and $A = A_2$ and the three values $\lambda = 0.1, 0.01, 0.001$. Therefore, you should report six residuals for this part.

f) (5 pts) You have now used four methods to find a solution of the underdetermined system $A\mathbf{x} = \mathbf{b}$ for two different matrices. Based on the residuals $\|A\mathbf{x} - \mathbf{b}\|$ discuss the pros and cons of each method. Be sure to discuss the role of the condition number of A.

Q9. **Partial Differential Equation** (Dr. Quaife, bquaife@fsu.edu)

Let u be an infinitely differentiable function.

a) (30 pts) Use Taylor series to compute the order of accuracy p of the following approximations

$$\begin{aligned} u(x) &= \frac{1}{3}\left(u(x+h) + u(x) + u(x-h)\right) + \mathcal{O}(h^p), \\ u'(x) &= \frac{1}{2h}\left(u(x+h) - u(x-h)\right) + \mathcal{O}(h^p), \\ u'''(x) &= \frac{1}{2h^3}\left(u(x+2h) - 2u(x+h) + 2u(x-h) - u(x-2h)\right) + \mathcal{O}(h^p). \end{aligned}$$

b) (30 pts) Perform three convergence studies by applying the three approximations in a) to the function $u(x) = \cos(2\pi x)$ when discretized at equispaced points in $[0, 1)$. Calculate the errors using an appropriate vector norm. Use enough values of h to observe the asymptotic order of convergence. You can report the errors using tables or using loglog plots. Describe how the convergence study verifies the calculations from part a).

For this part, you will have to use the periodicity of u to adjust the approximations for points close to 0 and 1. For example, when approximating $u'''(h)$, the value $u(-h)$ is required, but $-h$ is outside the domain $[0, 1)$. However, by the periodicity of u, $u(-h)$ can be replaced with $u(1-h)$, and $1-h$ is inside the domain $[0, 1)$.

c) (10 pts) Consider the partial differential equation with periodic boundary conditions

$$\begin{aligned} &u_t + uu_x + \delta^2 u_{xxx} = 0, \quad x \in [0, 2\pi),\ t > 0, \\ &u(x, 0) = \cos(2\pi x), \end{aligned}$$

where $\delta > 0$ is a parameter. Write down a numerical method by discretizing the time derivative with forward Euler, and the other three terms (u, u_x, and u_{xxx}) with the approximations in part a). Be sure to use the periodic boundary conditions.

d) (30 pts) Implement the numerical method in part c) with $\delta = 0.02$, $h = 0.01$, time step size $\Delta t = 10^{-5}$, and time horizon $T = 0.5$. Submit your code along with plots of the solution at times $t = 0, 0.1, 0.2, 0.3, 0.4, 0.5$.

Q10. **Partial Differential Equation** (Dr. Plewa, tplewa@fsu.edu)

The aim of this problem is to examine the convergence of a numerical method applied to a one-dimensional steady-state heat transport equation using a method of manufactured exact solutions (MES).

Background Recall that in the case no exact solution is available, one can assess the solution accuracy and its convergence properties using either the self-convergence method, in which a sequence of models is obtained on meshes with different sizes, or by comparing a discrete solution obtained with the computer code to that obtained by applying analytically a differential operator to a prescribed analytic solution, which is a basis of the MES method. Note that in the MES case, the exact solution is known, which offers unique advantages over the self-convergence method, in which case one only has access to a highly resolved model that may or may not represent the actual exact solution.
Provided that a numerical implementation of the PDE solver is available, the MES method requires:

1. an analytic solution;
2. the analytic result of applying the differential operator to the analytic solution;
3. provision of adding a distributed source term to the PDE solver.

Here we will denote the analytic solution as $T(x,t)$ and the differential operator as $\mathcal{L}(x,t)$.

Problem Description In certain situations, the energy evolution of a physical system can be described as a heat diffusion process. If one makes additional simplifying assumptions and uses appropriate thermodynamic relations, the heat (internal energy) can be expressed as a linear function of temperature. If, in addition, sinks and sources of heat remain in balance, the problem becomes time-independent. In this case, the resulting partial differential equation is,

$$\frac{\partial^2 T}{\partial x^2} = 0.$$

Using the operator notation, $\mathcal{L}(x) = (T)_{\rm xx}$, the above equation can be written as,

$$\mathcal{L}(x) = 0.$$

Next, we need to define the analytic solution, and our choice is,

$$T_{\rm a}(x) = 100 + x^3.$$

Finally, we need to define a computation domain, which is a region $0 \le x \le 5$. We will assume Dirichlet boundary conditions, and the required values can be obtained by evaluating $T_{\rm a}(x)$ at the boundary locations.
For the MES method, the analytically differentiated exact solution, $\mathcal{L} \circ T_{\rm a}(x)$, must be computed at each mesh cell and compared to the solution.

Deliverables

a) (30 pt) Write a finite difference solver for a steady-state linear diffusion problem in C, C++, or Fortran. Provide a provision to include an arbitrary source term. In your implementation allow for a user-defined size of the computational domain and mesh size. Include procedures that evaluate the exact solution, the source term, and boundary conditions.

b) (50 pt) Obtain a series of models on meshes with 8, 16, 32, 64, 128, and 256 zones. Graph solution residuals for each mesh resolution, compute L_1, L_2, and L_∞ norm, and corresponding estimates of the order of convergence. Show the norms and convergence orders in a table with data accurate to 4 decimal places. Discuss the results from the point of view of the theoretical order of convergence.

c) (20 pt) Repeat computations and analysis from point (b) for,

$$T_{\mathrm{a}}(x) = 1 + (x+1)^3 + \tanh(x-5).$$

In your report please include the code necessary in (b) and (c).

Q11. **Data Structures** (Dr. Erlebacher, gerlebacher@fsu.edu)

Feel free to use any technology (i.e., ChatGPT, GPT4, Copilot, etc.) and software you wish to solve this problem. Disclose the software and any prompts used. You are not allowed to use any libraries in which quad-trees are implemented. (All questions are equal-weighted in grade points.)

a) Consider the domain $D \in [0,1]\times[0,1]$. Create a routine that generates N line segments whose coordinates are distributed uniformly within D.

b) Write a program that implements a quad-tree to store this collection of lines randomly distributed in D. Define a class structure that includes efficient routines for line insertion, line removal, and querying the lines that intersect a subdomain of the 2D plane. The query routine should return the list of lines that intersect this domain. The code must be written in C++.

c) Create a synthetic dataset of 1000 random lines, and insert them into the quadtree.

d) Create a graph of the quadtree (using any language or approach you want), and overlay the lines on the quadtree. Python is recommended.

e) Create the quadtree for $N = 10^2, 10^3, 10^4, 10^5$ lines and create a table with the timings. Plot the graph of time versus N. What scaling does the graph represent? Using your results, what is the asymptotic scaling of the time to create the quadtree as a function of N? Use big-O notation.

f) Create 20 line segments fully contained in the subdomain $S = [.1, .4] \times [.3, .7]$. Insert these 20 lines into a new quadtree and query for the lines contained in the subdomain S. Plot the quadtree with lines overlaid, as well as the subdomain. Again, use any language you want for the plotting. Python is recommended.

g) Now assume that all the lines have their endpoints outside the domain. Explain any problems you might anticipate and the solutions you might consider. No programming is required. Be as complete as possible. There is no unique answer.