

Assignment - 1

Assigned: February 19, 2024

Due: March 4, 2024

Max. marks: 100

General Instructions

- Submit a hard-copy of the solutions either in-class or in TAs office NAC-1, Room no. 322.
- Upload a scanned copy of the solutions and the codes you have developed as a single zipped folder on Moodle.
- Do not email your solutions/codes. Emailed submissions will not be graded.

Hand Calculations

1. (10 points) Solve the following system of linear equations using recursive-doubling algorithm by clearly showing all the required steps. You may use the recursive expressions that we have developed or solve it directly using the philosophy of the method.

$$\begin{aligned} 3x_1 - x_2 &= 2 \\ -x_1 + 3x_2 - x_3 &= 1 \\ -x_2 + 3x_3 - x_4 &= 1 \\ -x_3 + 3x_4 &= 2. \end{aligned}$$

Note: The answer to this trivial question can be obtained easily. You will get points only if you present your solution using recursive-doubling algorithm that we learnt in the class.

Programming Questions

2. (30 points) Consider the calculation of the derivative of the following function,

$$f(x) = \sin(5x), \quad 0 \leq x \leq 3 \quad (1)$$

using four-order accurate Padé scheme for the interior and third-order accurate one-sided Padé scheme near the boundaries, given as follows,

$$f'_{j+1} + 4f'_j + f'_{j-1} = \frac{3}{h}(f_{j+1} - f_{j-1}) \quad (2)$$

where j is any interior point ($j = 1, 2, \dots, n-1$) and

$$f'_0 + 2f'_1 = \frac{1}{h} \left(-\frac{5}{2}f_0 + 2f_1 + \frac{1}{2}f_2 \right) \quad (3)$$

$$f'_n + 2f'_{n-1} = \frac{1}{h} \left(\frac{5}{2}f_n - 2f_{n-1} - \frac{1}{2}f_{n-2} \right) \quad (4)$$

where h is the grid spacing and n is the number of grid points in the x direction.

- (a) Develop a serial program to compute the derivative using the tridiagonal LU decomposition. Plot the analytical solution and the numerical solution obtained for $n = 25$.
- (b) Develop an OpenMP program to compute the derivative using the recursive-doubling algorithm. Plot the analytical solution and the numerical solution for $n = 100$ and number of threads $p = 2$. Plot the time-taken by the solver for $n = 1000$ for number of threads $p = 2, 4$ and 8 .

3. (60 points) Consider the solution of the following Poisson equation,

$$\nabla^2 \phi = -q; \quad q = 2(2 - x^2 - y^2); \quad \phi(\pm 1, y) = 0; \quad \phi(x, \pm 1) = 0; \quad (5)$$

using Gauss-Seidel method. The discretized equation using this method can be written as follows,

$$\phi_{i,j}^{(k+1)} = \frac{1}{4} \left[\phi_{i+1,j}^{(k)} + \phi_{i-1,j}^{(k+1)} + \phi_{i,j+1}^{(k)} + \phi_{i,j-1}^{(k+1)} \right] + \frac{\Delta^2}{4} q_{i,j} \quad (6)$$

where $\Delta = \Delta x = \Delta y$. Consider an initial guess of $\phi(x, y) = 0$ everywhere. The exact solution to this problem is given by $\phi = (x^2 - 1)(y^2 - 1)$. Use double precision arithmetic.

- (a) Develop a serial Gauss-Seidel program to solve the discretized equations and solve the system using $\Delta = \Delta x = \Delta y = 0.1$ (that is 21 points along each of the directions). Report the number of iterations required to bring the numerical solution to within 1% of the exact solution. Plot the numerical and the analytical solutions of ϕ vs x for $y = 0.5$.
 - (b) Develop an OpenMP program for Gauss-Seidel method using the (a) diagonal approach (b) red-black coloring approach.
 - (c) For each of the parallel programs developed above, verify that your parallel program is giving the same result as that of the serial program that you developed above for a grid of $\Delta = 0.1$ by plotting the two results on the same graph. Upon successful verification, run this program for $\Delta = 0.1, 0.01$, and 0.005 using 8 threads. Plot the time-taken by the parallel solvers and the serial solver as a function of the grid size (Δ). Do you see any improvement in performance? Which parallel method is better?
 - (d) For a grid size of $\Delta = 0.005$, plot the time-taken by each of the parallel solvers as a function of the number of threads. Consider the number of threads to be $p = 2, 4, 8$ and 16 . Which parallel method is better?
-