

MATH 440A/540A Parallel Scientific Computing
Homework Assignment 2
Last Submission Date: March 22 (8:00am)

This assignment must be entirely your own work. If any part of your assignment has been copied, then your mark may be reduced to zero. Any consultation (given/taken) must be acknowledged. Late assignments will be penalized at the rate of five marks per day. In fact, you should aim to submit the assignment several days before the due date because extensions will not be granted for reasons such as busy Sayers Lab or MIO nodes or computer breakdowns.

Submit all Assignment3 computer program files (with a readme.txt file containing details required to reproduce your reported output) and any PDF/WORD documents of your reports (both of the form `xxxx_*.*` : replace `xxxx` with your last name) to `mganesh@mines.edu` and to the grader `breyes@mymail.mines.edu`. Submit written/printed copies (of your reports, tables, figures, codes etc.) to your lecturer.

The main task in this assignment is to use parallel (Fortran or C or C++) programming with MPI to evaluate highly-oscillatory integrals by letting each allocated processing core in a communicator to compute only mildly-oscillatory integrals, for all chosen wavenumbers.

Let K be a fixed wavenumber. Let $f(K, \cdot) : [a, b] \rightarrow \mathbb{R}$ be a highly-oscillatory function. (Think of, or plot, $f(K, x) = \cos(Kx)$, $x \in [0, \pi]$, for $K = 1000$.)

Let

$$H(K) = \int_a^b f(K, x) \, dx.$$

Identify K naturally parallel tasks $A(K, i)$ (through constants a_i, b_i) for $i = 1, \dots, K$ so that

$$H(K) = \int_a^b f(K, x) \, dx = \sum_{i=1}^K \int_{a_i}^{b_i} f(K, x) \, dx =: \sum_{i=1}^K A(K, i)$$

and that for each $i = 1, \dots, K$, the partial integral $A(K, i)$ is an integral of $f(K, \cdot)$ on $[a_i, b_i]$ of width $(b - a)/K$. Write down a_i, b_i , for $i = 1, \dots, K$.

Partition each interval $[a_i, b_i], i = 1, \dots, K$, using a fixed number $n + 1$ of (quadrature) points:

$$a_i = x_{i,0} < x_{i,1} < x_{i,2} < \dots < x_{i,n} = b_i.$$

Let

$$\xi_{i,j} = \frac{x_{i,j-1} + x_{i,j}}{2}, \quad h_{i,j} = x_{i,j} - x_{i,j-1}, \quad i = 1, \dots, K, \quad j = 1, \dots, n.$$

(For example, if each $[a_i, b_i], i = 1, \dots, K$, is subdivided into $n + 1$ equally spaced quadrature points, write down, $x_{i,j}$ and $\xi_{i,j}$, and $h_{i,j}$ for $j = 0, \dots, n$.)

For each fixed K and $i = 1, \dots, K$, consider three distinct approximations to $A(K, i)$, denoted by $A_{M,n}(K, i)$, $A_{T,n}(K, i)$, $A_{S,n}(K, i)$, where the mid-point (M), trapezoidal (T), and Simpson's (S) rule approximations with the $n + 1$ quadrature points are defined by

$$\begin{aligned} A_{M,n}(K, i) &= \sum_{j=1}^n f(K, \xi_{i,j}) h_{i,j}, & A_{T,n}(K, i) &= \sum_{j=1}^n \frac{1}{2} [f(K, x_{i,j-1}) + f(K, x_{i,j})] h_{i,j}, \\ A_{S,n}(K, i) &= \frac{2}{3} A_{M,n}(K, i) + \frac{1}{3} A_{T,n}(K, i). \end{aligned}$$

1. Write a procedure with dummy input variables being

- an external integrand function, say, fn (which is assumed to be a function of two variables; wavenumber is the first variable)
- a wavenumber, say, w_n ,
- a vector of quadrature points, say, pts (with starting index 0)
- and the total number of quadrature pts, say $q_n + 1$

and the output variables being M, T, S that are computed using the formulas:

$$S = \frac{2}{3}M + \frac{1}{3}T,$$

$$M = \sum_{j=1}^{q_n} fn(w_n, mpts(j))len(j), \quad T = \sum_{j=1}^{q_n} \frac{1}{2}[fn(w_n, pts(j-1)) + fn(w_n, pts(j))]len(j),$$

where $len(j) = pts(j) - pts(j-1)$, $mpts(j) = (pts(j) + pts(j-1))/2$ $j = 1, \dots, q_n$.

2. Write a procedure with dummy input variables being a vector, say, $points$, its size $ptsiz$, a wavenumber, say, k_w , and the output being a vector fun_val_points , defined by

$$fun_val_points = \cos(100 * points - k_w * \sin(points))$$

3. Using the procedures in Q.1 and Q.2, write a main (Fortran or C or C++) program with MPI to compute, for each of the 9901 integer wavenumbers $K \in [100, 10000]$, (with total number of processing cores P determined by the `MPI_COMM_SIZE` command and parallelized in a load balanced way), three distinct approximate values for

$$H(K) = \int_0^\pi f(K, x) dx, \quad f(K, x) = \cos(100x - K \sin(x)),$$

with 101 *equally spaced* quadrature points on each $[a_i, b_i]$, for $i = 1, \dots, K$, denoted by $H_{M,100}(K), H_{T,100}(K), H_{S,100}(K)$, where (using the notation introduced earlier)

$$H_{M,100}(K) = \sum_{i=1}^K A_{M,100}(K, i), \quad H_{T,100}(K) = \sum_{i=1}^K A_{T,100}(K, i), \quad H_{S,100}(K) = \sum_{i=1}^K A_{S,100}(K, i).$$

Your parallel main program with MPI (parallelizing the naturally parallel tasks for each K) should:

- Print out, in a four column table, ten wavenumbers $K = 1000, 2000, \dots, 10000$ and associated approximations $H_{M,100}(K), H_{T,100}(K)$ and $H_{S,100}(K)$, with appropriate headings including the total number P of cores used in computation. (Submit the tabulated values obtained using $P = 8$ and $P = 16$ cores.)
- You may use the following high-order approximate values to make sure that your computed values match at least a few digits of these given values:
 $H(100) \approx 3.027448328771664E-01$, $H(1100) \approx -3.675617781869349E-02$.
 Use the format in these numbers and the table below as a sample for your print out format and headings.

- Save the approximate values of $H(K)$ for $K = 100, 101, \dots, 10,000$ in a file to facilitate plotting the functions, $H_{M,100}$, $H_{T,100}$ and $H_{S,100}$ on $[100, 10000]$
4. Plot your approximate functions $H_{M,100}$, $H_{T,100}$, $H_{S,100}$, computed using $P = 16$ cores and submit the three plots. (A sample approximate plot of H is below.)

The following approximations to oscillatory integrals $H(K)$ were computed using $P = 16$ cores			
K	Midpoint_H(K)	Trapezoidal_H(K)	Simpson's_H(K)
1000.0	x.xxxxxxxxxxxxxxxxxExxx	x.xxxxxxxxxxxxxxxxxExxx	x.xxxxxxxxxxxxxxxxxExxx
2000.0	x.xxxxxxxxxxxxxxxxxExxx	x.xxxxxxxxxxxxxxxxxExxx	x.xxxxxxxxxxxxxxxxxExxx
3000.0	x.xxxxxxxxxxxxxxxxxExxx	x.xxxxxxxxxxxxxxxxxExxx	x.xxxxxxxxxxxxxxxxxExxx
4000.0	x.xxxxxxxxxxxxxxxxxExxx	x.xxxxxxxxxxxxxxxxxExxx	x.xxxxxxxxxxxxxxxxxExxx
5000.0	x.xxxxxxxxxxxxxxxxxExxx	x.xxxxxxxxxxxxxxxxxExxx	x.xxxxxxxxxxxxxxxxxExxx
6000.0	x.xxxxxxxxxxxxxxxxxExxx	x.xxxxxxxxxxxxxxxxxExxx	x.xxxxxxxxxxxxxxxxxExxx
7000.0	x.xxxxxxxxxxxxxxxxxExxx	x.xxxxxxxxxxxxxxxxxExxx	x.xxxxxxxxxxxxxxxxxExxx
8000.0	x.xxxxxxxxxxxxxxxxxExxx	x.xxxxxxxxxxxxxxxxxExxx	x.xxxxxxxxxxxxxxxxxExxx
9000.0	x.xxxxxxxxxxxxxxxxxExxx	x.xxxxxxxxxxxxxxxxxExxx	x.xxxxxxxxxxxxxxxxxExxx
10000.0	x.xxxxxxxxxxxxxxxxxExxx	x.xxxxxxxxxxxxxxxxxExxx	x.xxxxxxxxxxxxxxxxxExxx

