AMS 562 Homework 3

Due: 10/19/2022, 11:59 PM

Task description:

Part 1

For the first part, you need to write a program that computes the derivative of the $\sin$ function at point $x = \frac{\pi}{4}$, i.e., compute $y\prime|_{\pi 4}$, where $y = \sin(x)$. A typical way to do this is to use the finite difference formula, i.e., given a smooth function $y = f(x)$, its approximated derivative at point x is

$$y'(x) = \frac{f(x+h) - f(x)}{h} \tag{1}$$

where h is a measure of some local finite difference. This approximation converges to the true derivative for smooth functions in the limit $h \to 0$. Furthermore a more accurate approximation can be obtained by the following approximated also known as *center-difference*

$$y'(x) = \frac{f(x+h) - f(x-h)}{2h}. \tag{2}$$

Your program needs to take the parameter h as the command line argument. To convert a C-string to a floating point number, you will need to use the `std::atof` defined in `<cstdlib>`. Sanity check of `argc` and `argv` is required to make sure that h is provided and valid. Compute the derivative of $\sin$ at point $x = \frac{\pi}{4}$, using both approximation methods and compare the error against the true result i.e. $\cos(\frac{\pi}{4})$. Record the results for $h = 1 \times 10^{-1}, 1 \times 10^{-2}, 1 \times 10^{-3}, 1 \times 10^{-4}$. What happens when $h = 1 \times 10^{-16}$.

Part 2

For the second part, you need to write a program that determines the extreme arc lengths among a collection of random sample points on the unit sphere, i.e., $x^2 + y^2 + z^2 = 1$. The definition of arc length between two points is the length of the arc that passes through the *great circle* defined by the two points. The arc length can be computed by the inverse cosine function, i.e.,

$$l = \cos^{-1}(\mathbf{n}_1 \cdot \mathbf{n}_2) \tag{3}$$

where $\mathbf{n}$ are the **unit** outward normal vectors on the spherical surface, and $(\cdot)$ is the inner product. Notice the outward normal vectors have the same coordinate values as the points on the unit sphere. Similar to Part I, the number of the sample points is passed in as the first command line argument,say N. Then you need to allocate three dynamic arrays, `float *x, *y, *z;`, of size N and pass them to the black-box function to get a collection of random points on the sphere. Next, write an iterative loop to find the maximal and minimal arc lengths with respect to the **first point** (`x[0]`,`y[0]` and `z[0]`) and their corresponding indices in the point collection. Finally, print out the results (the actual minimal and maximal arc length, and the corresponding points, including the first point) in the terminal.

Hints:

- For $\sin$ and $\cos^{-1}$ functions, they are declared in the *standard math library* — `<cmath>`. Similarly, you can access the value of π with M_PI therein. Notice that you can pre-define the value of π since it's a known constant, i.e., `const double my_pi = 3.14159265358979323846;`.
- For dynamical memory allocation, don't forget to deallocate the memory at the end.

Submission guidelines:

- Make sure that you create a separate branch (under the name `hw3`) for this hw, checkout to that branch, and make all your commits there.
- Make sure that you don't put all your work in one single commit.
- Create a pull request (under the name `HW3 Submission`) before the deadline, but **do not** merge it.