Due before: **11:59 PM on Wed Oct 20 2021**

CSE-443/543: High Performance Computing

Homework #5

Part #1 Due: Wed Oct 20 2021 before 11:59 PM (15 points)

Part #2 Due: Wed Oct 27 2021 before 11:59 PM (10 points)

Submission Instructions

This homework assignment must be turned-in electronically via Canvas CODE plug-in. Ensure your program compiles successfully, without any warnings or style errors. Ensure you have documented the methods. Ensure you have tested operations of your program as indicated. Once you have tested your implementation, upload just the following via Canvas CODE plug-in:

1. The C++ header and source file(s) you modified for this project.
2. The PDF file with the performance report for this document.

General Note: Upload each file associated with homework individually to Canvas. Do not upload archive file formats such as zip/tar/gz/7zip/rar etc.

Objective

The objective of this homework is to improve the performance of a Neural Network program:

- Work with the Linux Perf profiler to identify aspects of the program to improve
- Propose and implement performance enhancements to the program.
- Benchmark the performance gains due to the proposed enhancement.

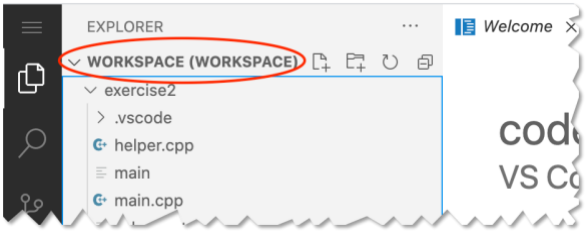
Grading Rubric:

- Your program should compile without any warnings.
- **NOTE:** Violating CSE programming style guidelines is a compiler error! Your program should not have any style violations reported.
- **Do not use global variables –** that is not good programming practice.
- You may not use OpenMP for this part of the project.
- Part #1 & Part 2: Implement a profile driven **substantive** performance improvement to the supplied program and provide quantitative data to substantiate the performance improvement. (**10 points each part**)
- **Delayed submission: Only 80% points:** Submission delayed **by no more than 24-hours** will be accepted for a partial credit of maximum 80% of the points.

Due before:	11:59 PM on Wed Oct 20 2021

Task #1: Setting up VS-Code project

Estimated time to complete: 5 minutes



1. Log into OSC's OnDemand portal via https://ondemand.osc.edu/. Login with your OSC id and password.
2. Startup a VS-Code server and connect to VS-Code. Ensure you switch to your workspace. Your VS-Code window should appear as shown in the adjacent screenshot.
3. Next, create a new VS-Code project in the following manner:
   a. Start a new terminal in VS-Code
   b. In the VS-Code terminal use the following commands:

```
$ # First change to your workspace directory
$ cd ~/cse443
$ # Use ls to check if workspace.code-workspace file is in pwd
$ # Next copy the basic template for a C++ project
$ cp -r /fs/ess/PMIU0184/cse443/templates/basic homework5
$ # Copy the starter code for this project
$ cp /fs/ess/PMIU0184/cse443/homework/homework5/* homework5
```

4. Add the newly created project to your VS-Code workspace to ease studying and modifying the source code.

Project overview

In this project you are given an artificial Neural Network (*i.e.*, `NeuralNet`) program that is built around the `Matrix` class developed will the previous project. The Neural Network implementation closely follows the implementation by Michael Nielsen (http://neuralnetworksanddeeplearning.com/) to recognize or "classify" handwritten digits shown in the adjacent image. Note that neural networks essentially map to matrix operations. The learning or training phase implemented in `NeuralNet::learn` method uses several of the matrix operations that we developed in the previous project. Similarly, the classification or "recognition" feature in `NeuralNet::classify` method uses some of the matrix operations. The supplied source code for this project consists of the following 3 sets of files:

1. `Matrix.h/.cpp` : This pair of source files contain the implementation for a simple Matrix class.
2. `NeuralNet.h/.cpp`: This pair of source files contains implementation for a simple neural network based on the implementation from Michael Nielsen (http://neuralnetworksanddeeplearning.com/).
3. `main.cpp`: This is a top-level source file that works with handwritten example images from the MNIST benchmark.

Project requirements

In this project, you are expected to improve the performance (*i.e.*, make the program run faster) of the supplied neural network classification program. The reference performance is compiling the program with `-O3` and running the program without any command-line arguments. See the supplied SLURM script for the compiling and running the program. The performance improvements will be conducted in the following 2 phases:

1. Phase #1: The first phase requires you to improve the performance for the given starter code by suitably modifying the source code (you may modify any of the supplied files as you see fit). You may not use OpenMP constructs.
2. Phase #2: Continue to build (no, you cannot use the starter code for this phase unless you chose not to submit anything for Phase #1) on the improvements from Phase #1 to further improve the performance. You may not use OpenMP constructs.

Notes for each phase:

- For each phase, the results from the program should be consistent and correct – that is, you should be preserving the functionality!
- For each phase, for full points your modification should yield statistically significant runtime improvements.
- You are expected to submit the modified source code and required performance report for each phase.

Approach for improving performance

1. In each phase, first record the starting/reference runtime for the supplied program. You will be trying to further reduce the runtime from this starting point.
2. Next, profile the program using `perf` to identify the aspects of the program you would like to address to improve performance. You may select any aspect associated with computational aspects, memory management aspects, or I/O.
3. Ensure you include your perf report information in your performance report.
4. In the report, document the change(s) you are planning to incorporate.
5. Perform the necessary code changes that you identified.
6. Measure and record the runtime of the revised version of the program to show the performance improvement realized by your changes.

Turn-in:

This homework assignment must be turned-in electronically via Canvas CODE plug-in. There are two separate turn-ins that are required as per the deadline(s) specified for each phase of this project.

1. Phase #1: Upload the source code and performance report.
2. Phase #2: Upload the source code and performance report.