## Pebble in a Pond Wave Equation - Instructor Guide

- Since this is only a 25 minutes module, the CPU multicore/multithread and GPU hardware architectures should already be covered in previous modules. The focus of this module should only be on adding OpenMP to existing MPI code implementations and its use cases in different scientific applications.
- Instructors should review the materials covered in the presentation slide set and do further readings of the concepts being presented.
- This module will start with presentation slides that covers multi-core and multi-threaded CPU's, concepts, and data array movement to/from memory cache.
- Instructors should use simple random array addition examples to demonstrate primary MPI subroutines for copying the data from node to node, and vice versa. Then, this program can be expanded to use OpenMP for testing compiler directives.
- The acoustic wave equation example is used as a more scientific application use case. This example will also demonstrate good and bad practices in MPI/OpenMP programming that students should be aware of.
- Understand and present the idea of a post processing step to develop visual graphics animations and know how to use ImageMagik.

## Common Pitfalls

- Depending on when this module is used for teaching or learning, MPI/OpenMP implementation might have changed. Therefore, both instructors and students are encouraged to check the HPC community for updates.
- Watch out for IO, as MPI outputs data writes in random order. Moving data between nodes could potentially reduce application performance tremendously.
- Beware of making many changes and then timing results. Use an incremental approach – make one change – then time entire program. See what runtime improvements happened. Rerun several times to determine if actual times are consistent.
- In the beginning, do a paper design of memory layout and decide either a one dimension or two-dimensional compute node structure is best.