Accelerating Scientific Applications - 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 OpenACC 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 different OpenACC concepts and implementations.
- Instructors should use simple vector addition examples to demonstrate primary OpenACC directives for copying the data from host to device, vice versa, and parallelizing loops.
- 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 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, OpenACC implementation might have changed. Therefore, both instructors and students are encouraged to check the PGI community for updates.
- OpenMP is rapidly changing and adopting OpenACC techniques for accelerating applications on GPUs. There is also a growing consensus that OpenMP and OpenACC should merge as one. Instructors and students should be aware of the changes and make necessary modification to the examples and exercises solutions.
- Watch out for IO. Moving data between CPU and GPU 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.
- Look only for work sections of code, find the nested loop. Code maintenance is best when the fewest lines of code are under the influence of directives.