Installing MPI libraries for Intel Macs

- These instructions are for Intel Macs
  - Apple Silicon M1 macs use a new architecture which is only slowly developing support
- You will need to install some MPI libraries and place suitable binaries in your path.
  - The simplest method to install unix-like programs on Mac is to install a package manager such as Homebrew or Macports.
  - See https://brew.sh/ for more details on homebrew.
  - A one line install command for homebrew is
    /bin/bash -c "$(curl -fsSL https://raw.githubusercontent.com/Homebrew/install/HEAD/install.sh)“
  - Install the openmpi package with brew install openmpi

Installing MPI libraries for Linux

- These instructions are for linux
- You need to install MPI libraries and binaries
- The easiest way is using the appropriate package manager for your chosen distro, e.g.
  - Ubuntu/Debian :-
    `sudo apt install libopenmpi-dev`
  - Centos/Redhat :-
    `sudo yum  install openmpi-devel`

Installing MPI libraries for M1 Macs

- Apple Silicon M1 macs are new and only gradually getting support.
- You can try the native installation of Homebrew but it is not always guaranteed to work (see https://brew.sh/ for more details).
  - A one line install command for homebrew is
    `/bin/bash -c "$(curl -fsSL https://raw.githubusercontent.com/Homebrew/install/HEAD/install.sh)“`
  - Install the openmpi package with `brew install openmpi`
- If this misbehaves, then the most likely solution is to follow the suggestions here to install the homebrew manager under Rosetta.
- If this still doesn’t work, get in touch and we’ll set you up an Azure Labs linux VM to SSH into using VS code.

Building and running the codecode

- The easiest way to ensure that header files are in your search path and that the correct libraries are linked is to use the mpi compiler wrappers you've just installed.
- In brief:
  - Instead of cc/gcc/clang use mpicc
  - Instead of c++/g++/clang++ use mpic++
- On centos (e.g. the DUG HPC) you'll need to run module load mpi first
- It's safe to build most non-MPI libraries with the mpi wrappers, so you can usually just make a like for like switch.
- Like windows, use "mpiexec" to run the code over multiple cores
  - mpiexec –n *#cores <executable name>*