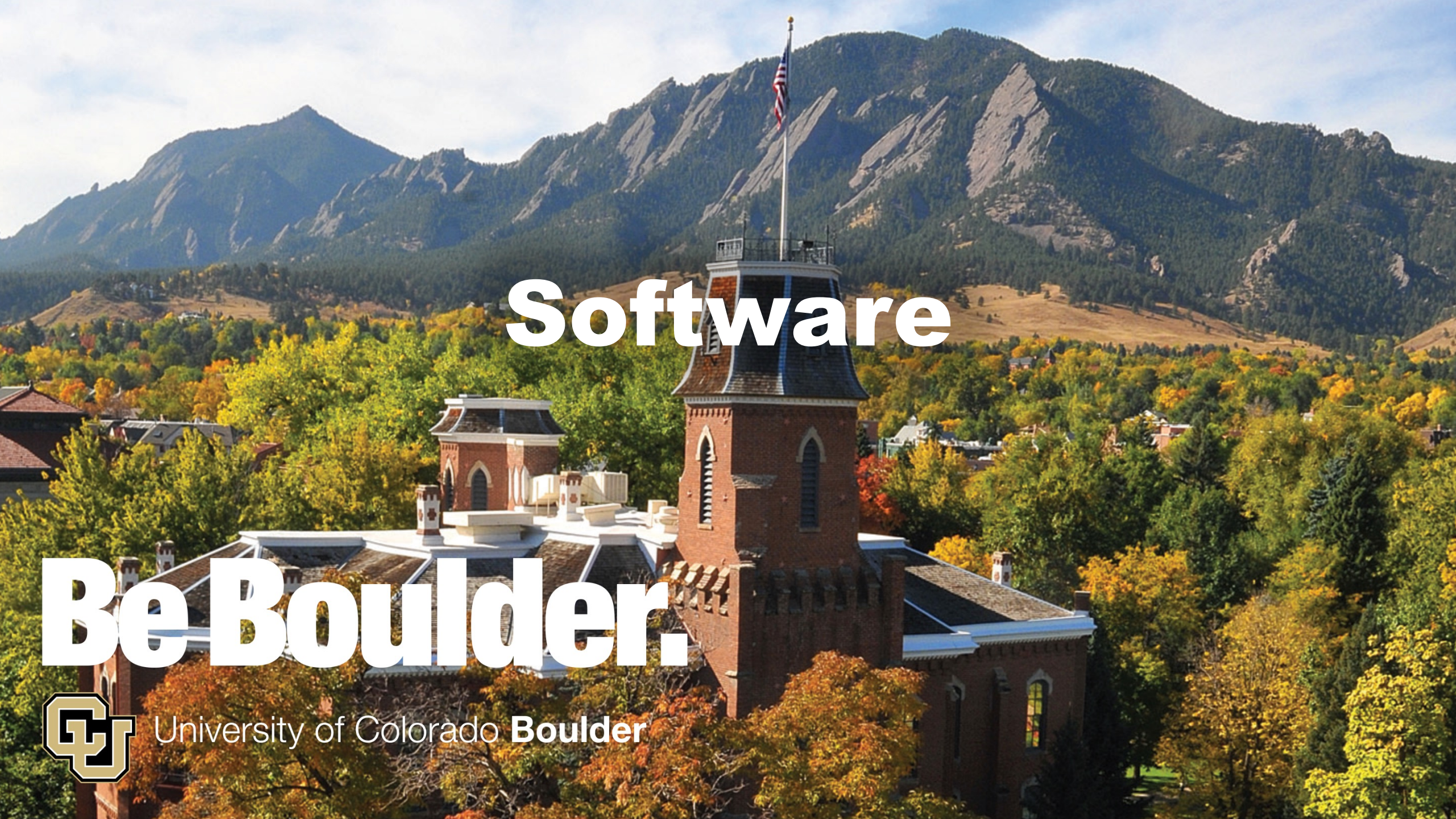
Software
Be Boulder.
University of Colorado Boulder

Software installation

- Managing loaded software can be a headache
  - Make sure that correct versions are available
  - Make sure that software dependencies for package A don't interfere with Package B
- If simply load software in a directory can run into these issues on a shared system
- Want to use a package manager
  - However usually a combination of both on HPC systems

Modules

- Environment modules allow centers to provide multiple versions of software and load dependencies seamlessly
- A module is a package that contains all of the files required to run the software, including libraries
  - Will load required dependencies
- Users can access software using a few simple commands

Working with Modules

- See a list of available modules

  `module avail`

- Load a module
  - Adds software to your $PATH
  - May also load dependencies
  - May also unload other versions or dependencies that would conflict
  - **`module load <name_of_module>/<version>`**
    - Example: **`module load hdf5`**

Working with Modules

- See a list of available modules
  `module avail`
- Unload a module
  `module unload`

List of loaded modules
`module list`

Unload all modules
`module purge`

- Discover information about module
  - **`module spider <name_of module>/<version>`**
    - Example: **`module spider mpich`**
    - Tells you about dependencies, the package, etc.

Installing Your Own Software

- Sometimes the cluster you are working on does not have the software you need
- General process:
  - Download software
  - Install software
    - Read instructions
    - Install dependencies
  - Compile
  - Use

Installing Your Own Software

- What might this look like?
  - Clone some files from Git
  - Download a docker or singularity image
  - Install from a file
  - Olden days – install from a disk
  - Just get the files on the compute system you're installing on
- Install additional software
- Run `make`
- `./install_file`