IN THIS MODULE:

• Code requirements/porting considerations
• Compile the code
• Conduct an accuracy test
BEFORE WE BEGIN:

• Open up a terminal window and navigate to ~/Desktop/Rayleigh_Tutorial

• All exercises will take place within this directory

• Create a subdirectory named module1
CODE REQUIREMENTS:
(INSTALL THESE BEFORE BUILDING RAYLEIGH)

• Fortran 2003 and C++ compilers
• BLAS
• LAPack
• FFTW version 3.x or later
• MPI

* Intel’s MKL provides interfaces to BLAS, LAPack, and FFTW
BUILDING THE CODE:

RUN THIS:

```bash
$ cd rayleigh
$ ./build_rayleigh.bsh CIG
```

- Copies all source files (*.F90) to rayleigh/build
- Copies Makefiles/Makefile_CIG to build/Makefile
- Builds Rayleigh:
  
  executable located in /build/rayleigh

NOTE: ./build_rayleigh works when csh installed
PORTING TO NEW MACHINES:

• Create a new Makefile_NAME and place it in the rayleigh/Makefiles directory

```bash
$ ./build_rayleigh.bsh NAME
```

• See examples in Rayleigh/Makefiles:
  • Intel Compiler  : Makefile_Pleiades
  • IBM Compiler   : Makefile_Mira
  • GNU Compiler  : Makefile_CIG
MAKEFILE CUSTOMIZATION

From rayleigh/Makefiles/Makefile_CIG

```bash
F90 = mpiF90
CC = gcc
#
# Flags for the LAPack Libraries
LIBFLAGS = -L/usr/lib/x86_64-linux-gnu -lfftw3 -L/usr/lib -lblas -llapack -lstdc++

ifeq ($(RAYLEIGH_OPT),debug)
    F90FLAGS = -O1 -fbounds-check -fbacktrace -ffixed-line-length-132 -I/usr/include
else
    F90FLAGS = -O3 -ffixed-line-length-132 -I/usr/include
endif
```

- RAYLEIGH_OPT1 is passed through build_rayleigh.bsh
- Also RAYLEIGH_OPT2 and RAYLEIGH_OPT3

Link BLAS, LAPack, FFTW

"Include" directories

Compiler Specific Flags
USING BUILD_RAYLEIGH FLAGS:

TRY THIS:

```
$ ./build_rayleigh.bsh CIG debug
```

Sets $RAYLEIGH_OPT1 to debug

Enables debugging flags in Makefile_CIG

NO DEBUGGING FOR NOW PLEASE!

RERUN THIS:

```
$ ./build_rayleigh.bsh CIG
```
OUR FIRST RUN: PREPWORK

- Each simulation gets a dedicated directory.
- Change to run directory.

$ cd ~/Desktop/rayleigh_Tutorial/module1

- A copy or link to Rayleigh should reside in the run directory.
- Softlink the executable.

$ ln -s ../rayleigh/build/rayleigh .
OUR FIRST RUN: PREPWORK

• Each simulation requires an input file (run parameters)
• Grab this file from input_examples:

```
$ cp ../rayleigh/input_examples/c2001_case0_minimal_input .
```

• Rayleigh expects its input to be named “main_input”
• Rename the file to “main_input”

```
$ mv c2001_case0_minimal_input main_input
```

• Let’s edit main_input

```
$ gedit main_input ..etc..
```
MAIN_INPUT

- Instructions from the user
- Flags override defaults
- Grouped into namelists
- Namelists control different aspects of the simulation.
- Main topic of this tutorial!
Modify these values ...

n_r = 32
n_theta = 48
n_prow = 2
n_pcol = 2
benchmark_report_interval = 500
max_iterations = 2500

... and save.
OUR FIRST RUN

- Run the code...

```
$ mpiexec --np 4 ./rayleigh
```

- You will see:

---

**Startup:**

Preamble

---

**Middle:**

Timestep Info

---

**Completion:**

Timing Info

---

...while we’re waiting...
IN-SITU BENCHMARKING

- Fully nonlinear, but low-Re
- Steady-state with rotating pattern
- Predefined set of analyses
- When porting: run a benchmark!

**Benchmark Inputs**

- Anelastic: Jones et al., 2011, Icarus, 216, 120
  - c2001_case0_minimal (hydro)
  - c2001_case1_minimal (MHD)
- Boussinesq: Christensen et al. 2001, PEPI, 128, 25
  - input_examples/j2011_hydro_steady_minimal
  - input_examples/j2011_mhd_steady_minimal
CHECK YOUR RESULTS

$ ls
$ more Benchmark_Reports/00002500

### RAYLEIGH ACCURACY BENCHMARK SUMMARY

**Benchmark:** Christensen et al. 2001 (Non-MHD, Case 0)

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Radial Resolution N_R</td>
<td>32</td>
</tr>
<tr>
<td>Angular Resolution N_theta</td>
<td>48</td>
</tr>
<tr>
<td>Averaging Interval</td>
<td>0.040000</td>
</tr>
<tr>
<td>Beginning Iteration</td>
<td>2100</td>
</tr>
<tr>
<td>Ending Iteration</td>
<td>2500</td>
</tr>
<tr>
<td>Number of Samples</td>
<td>5</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Observable</th>
<th>Measured</th>
<th>Suggested</th>
<th>% Difference</th>
<th>Std. Dev.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Kinetic Energy</td>
<td>58.219893</td>
<td>58.348000</td>
<td>-0.219557</td>
<td>0.074600</td>
</tr>
<tr>
<td>Temperature</td>
<td>0.426441</td>
<td>0.428120</td>
<td>-0.392224</td>
<td>0.000220</td>
</tr>
<tr>
<td>Vphi</td>
<td>-10.105877</td>
<td>-10.157100</td>
<td>-0.504312</td>
<td>0.003859</td>
</tr>
<tr>
<td>Drift Frequency</td>
<td>0.185113</td>
<td>0.182400</td>
<td>1.487441</td>
<td>0.007528</td>
</tr>
</tbody>
</table>

- Normally % Difference will be well under 1%
- This is not great. Really need ~ 30,000 time steps
- For tutorial, compare against pre-verified code output …
CHECK YOUR RESULTS

$ more Benchmark_Reports/00002500

$ more ../accuracy_check

Do they match?

Questions?