# Regression Testing for Athena++

# Code testing

- Reasons
  - Keep developers from breaking each other's code
  - Or their own code
  - Or users' expectations
- Styles
  - Unit testing: make sure each component works individually
    - Hard $\rightarrow$ not implemented
  - Regression testing: make sure the code as a whole does not lose functionality
    - Easy $\rightarrow$ covered below

# Directory structure

# How to run regression tests

- Go to regression test directory

```
cd tst/regression
```

- Run all tests

```
python run_tests.py
```

- Or run suites of tests

```
python run_tests.py sr gr
```

- Or run individual tests

```
python run_tests.py sr/hydro_shocks_hlle
```

# What tests tell you

- Hopefully "passed" for each test
- Sometimes "failed"
  - No further message: test script returned failure
  - "unexpected failure in . . . "
    - `prepare()`: configuration/compilation
    - `run()`: Athena++ ran but aborted with error
    - `analyze()`: problem reading output data
- Final summary at end ("25 out of 25 tests passed")

# Writing tests: Location

- Directories under `regression/scripts/tests/` correspond to suites
  - Examples: GR, viscosity, MPI
  - All tests must be in a suite
  - Suites cannot be nested
- Each test is a single Python file in such a directory
- If creating a new directory, must include `__init__.py` (empty file)

# Writing tests: Making a new test

- Follow `regression/scripts/tests/example.py`
- Three functions must be defined: `prepare()`, `run()`, `analyze()`
  - Reason: unexpected catastrophic errors can be traced better
- Use functions in `regression/scripts/utils/athena.py` to interface with Athena++

# Writing tests: Compiling/configuring

```
import scripts.utils.athena as athena

def prepare():
  athena.configure('g', 't',
      prob='shock_tube_rel',
      coord='minkowski')
  athena.make()
```

Equivalent to

```
python configure.py -gt \
    --prob=shock_tube_rel \
    --coord=minkowski
make clean
make
```

# Writing tests: Running Athena++

```
import scripts.utils.athena as athena

def run():
  arguments = [
      'job/problem_id=gr_shock_tube',
      'output1/file_type=vtk',
      'output1/variable=cons',
      'output1/dt=0.4',
      'time/cfl_number=0.4',
      'time/tlim=0.4',
      'mesh/nx1=400']
  athena.run('hydro_sr/athinput.mb_1', arguments)
```

Equivalent to

```
cd bin
./athena -i ../inputs/hydro_sr/athinput.mb_1 \
    job/problem_id=gr_shock_tube ...
```

# Writing tests: Checking the output

```
import sys
sys.path.insert(0, '../../vis/python')
import athena_read

def analyze():

  ref_file = 'data/sr_hydro_shock1_hlle.vtk'
  x_ref,_,_,data_ref = athena_read.vtk(ref_file)
  mx_ref = data_ref['mom'][0,0,:,0]

  new_file = \
      'bin/gr_shock_tube.block0.out1.00001.vtk'
  x_new,_,_,data_new = athena_read.vtk(new_file)
  mx_new = data_new['mom'][0,0,:,0]

  ...
```

# Writing tests: Checking the output

```
import numpy as np
import scripts.utils.comparison as comparison

def analyze():
  ...

  error_abs_mx = comparison.l1_diff(x_ref, mx_ref,
      x_new, mx_new)
  error_rel_mx = error_abs_mx \
      / comparison.l1_norm(x_ref, mx_ref)
  if error_rel_mx > 0.01 or np.isnan(error_rel_mx):
    return False
  return True
```

Must return `True` (test passes) or `False`

# Writing tests: Notes

- `regression/bin/` is deleted before and after each test
- Tests should not (permanently) interfere with other directories
- Static data can be stored in `regression/data/`
  - Part of repository – do not make files too large
- Python utility scripts for analyzing datasets in `regression/scripts/utils/`
- Varieties of regression tests
  - Output matches precomputed values
  - Convergence tests
  - Compilation-only

# Discussion

- What tests do we need?
  - The problem of permutations
- How portable should tests be?
  - Currently runs on any machine
  - Should tests cover `icc` or multiple nodes?
  - If so, should they be part of the default test suite?
- Should code be committed that breaks tests?