Rayleigh Hackathon 2018 Final report

Introduction

To further develop the spherical convection/dynamo code Rayleigh and to grow and foster its user community, 18 users and developers of Rayleigh worked side-by-side over a 3 day period close to Boulder, Colorado in September 2018.

The Rayleigh community made significant progress along various lines of project development. Some efforts begun at the workshop remain in their initial stages. This includes a multi-month effort to revamp Rayleigh's linear solve to take advantage of a newly-developed sparse Cheby/Gaelerkin scheme, as well as a separate effort to provide a generic spectral-input interface for the code. Moreover, several code additions were brought to fruition at the workshop, such as the addition of new axial (z-axis-aligned) diagnostics, new internal spectral-indexing capability, and numerous bug-fixes to the turbulent-kinetic-energy diagnostics package. In addition to code development, substantial progress was made on non-code aspects of the Rayleigh project. An initial testing framework was established, an github.io page was created, and several improvements to the documentation were developed over the course of the workshop.

A portion of the workshop was also devoted to discussing future directions for the Rayleigh code. Several potential future developments are enumerated at tinyurl.com/Rayleigh2018Future . Two topics which were heavily discussed included further optimization of numerical algorithms and creation of interface that facilitate the interoperability of Rayleigh with other commonly-used spectral convection codes. Additional post-processing capability, particularly with regards to 3-D output, is now being planned as well.

During the course of the hackathon, every participant contributed source code to the project. Together, users and developers added a total of more than 4,000 lines of code, arising from 176 individual contributions, and including 40 new tests.

Below is the timeline and a log of the individual contributions. Many of these contributions are discussed in greater detail following the table of participants' interests.

Timeline

Day	**Scheduled items**
Monday, 09/17	9:00A: Participant Introductions The open source philosophy Future Directions 12:30P: Data Discussion 1:30P:

	Github tutorial. Continuous Integration Creating first pull requests - working on Summit, code structure, and making changes to diagnostics codes
Tuesday, 09/18	8:30A: Parallel structure of Rayleigh Cookbooks, initial projects Noon: lunch 1P: Tinker Time
Wednesday, 09/19	8:30A: Morning rounds, discussion about authors of the project, forums, Tinker TIme 2P: Wrap-up discussion, future hacks

Participants and areas of interest

Name, affiliation, email	**Goals and interests for this hackathon**
Rene Gassmoeller, UC Davis, rene.gassmoeller@mailbox.org	1. Help others achieve their goals 2. Review pull requests 3. Set up tester
Lorraine Hwang UC Davis ljhwang@ucdavis.edu	1. Logistics 2. Documentation 3. Setting up forum 4. Reporting
Nick Featherstone CU Boulder nicholas.featherstone@colorado.edu	1. Introducing people to Rayleigh's design 2. Helping others
Cian Wilson Carnegie Institution DTM cwilson@carnegiescience.edu	1. Learn core of Rayleigh 2. Discuss feasibility of wish list 3. Start to implement some of wish list
Sebastian Glane *Berlin Institute of Technology* glane@tu-berlin.de	1. Learn how to implement new items to Rayleigh 2. Document numerical and time-stepping schemes and improve time-stepping 3. Implement output to vtk-format for visualization using paraview 4. Implement Gaussian filtering of the data

Ryan Orvedahl CU Boulder ryan.orvedahl@colorado.edu	1. Learn details of parallelization 2. Learn how Diagnostics are done 3. Add more outputs to the Diagnostics
Brad Hindman CU/JILA hindman@lcd.colorado.edu	1. Learn how to contribute to the Rayleigh project in an organized fashion 2. Learn how to add unique physics modules to Rayleigh (e.g, turbulent diagnostics) 3. Initiate collaborative efforts
Loren Matilsky CU/JILA loren.matilsky@colorado.edu	1. Listen in on accurate/efficient Chebyshev implementation 2. Listen in on work on Cartesian mode 3. Learn how to implement various custom diagnostics--third derivatives? 4. Learn about OpenMP implementation (no experience currently.)
Maria Weber University of Chicago, Adler Planetarium maweber@uchicago.edu	1. Learn what's 'under the hood' of Rayleigh 2. Help with improved documentation 3. Improved and documented visualization pipelines 4. Learn more about creating custom diagnostics
Ben Miquel CU Boulder benjamin.miquel@colorado.edu	1. Understand better the core and the architecture of Rayleigh 2. Implement the Chebyshev QI method for accurate high-order radial derivatives (even with high resolution)
Krista Soderlund UTIG krista@ig.utexas.edu	1. Learn how to contribute to Rayleigh, especially wrt diagnostics and inclusion of new physics (e.g., conducting inner core) 2. Help with output plotting routines 3. Best practices for data sharing
Kyle Augustson CEA-Saclay kyle.augustson@cea.fr	1. Consider how to generalize the Rayleigh core for additional PDEs that can utilize the mathematical/numerical structure of it. 2. Assist with implementation of quasi-inverse methods. 3. Consider how to implement alternative time-stepping routines, e.g. Gauss-Legendre fully implicit Runge-Kutta methods. Need for alternative parallelization. 4. Assess ability to have flexible geometry, e.g. those with smooth mapping between

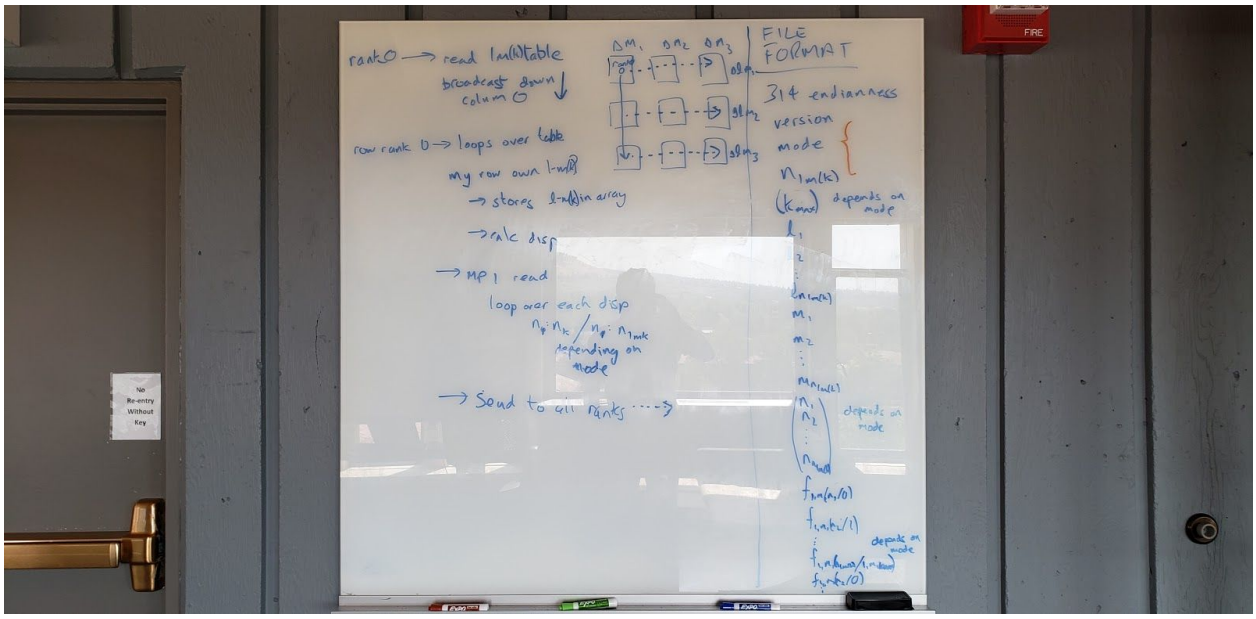
	Cartesian, cylindrical, or spherical geometry. (Ties into point 1). 5. Examine possibility of modular output, runtime-compiled routines that have access to internal data and operators (in physical or spectral space) to build custom output.
Connor Bice CUB connor.bice@colorado.edu	1. Understand Rayleigh's data structures / math layer / parallelization better 2. Learn how to interact with the repository correctly 3. Fix the issues with angular momentum conservation for deep shells
Jon Aurnou UCLA aurnou@ucla.edu	1. Learn how to run Rayleigh and output data from the simulations 2. Learn how to discuss changes and to contribute to Rayleigh 3. Implement new diagnostics especially relevant to low Rossby number flows that exist in planetary cores & deep atmospheres 4. Building of active development community 5. Focus on how best for CIG to continue to develop Rayleigh into the future
Valentin Skoutnev Princeton U. skoutnev@princeton.edu	1. Learn how to build/run Rayleigh 2. Learn how to add diagnostics and submit pull requests. 3. Understand a bit about the solve part of the code so that I can go back and try working with it.
Hiro Matsui CIG, UC Davis hematsui@ucdavis.edu	1. Learn data format of Rayleigh to lead the Rayleigh data output into Calypso

Report on projects the participants worked on

Began Implementing Generic Spectral Input Interface

Nick Featherstone, Cian Wilson

We began development of an interface via which users can supply spectral coefficients for use in specifying custom initial conditions and boundary conditions to Rayleigh. After deciding on a standard file format (recorded on whiteboard below), Cian began work on the Python-side writing routine, and Nick began implementing the Rayleigh-side reading routine.



Modified the Rayleigh tester to test for changes in the output

Rene Gassmoeller, Tyler Esser

We improved the Rayleigh tester to not only check if Rayleigh compiles correctly, but also to check if the one test case that we have at the moment creates the same result as before. This also includes setting up a directory structure to add more test cases in the future.

Update the README.md file to link to CONTRIBUTING.md

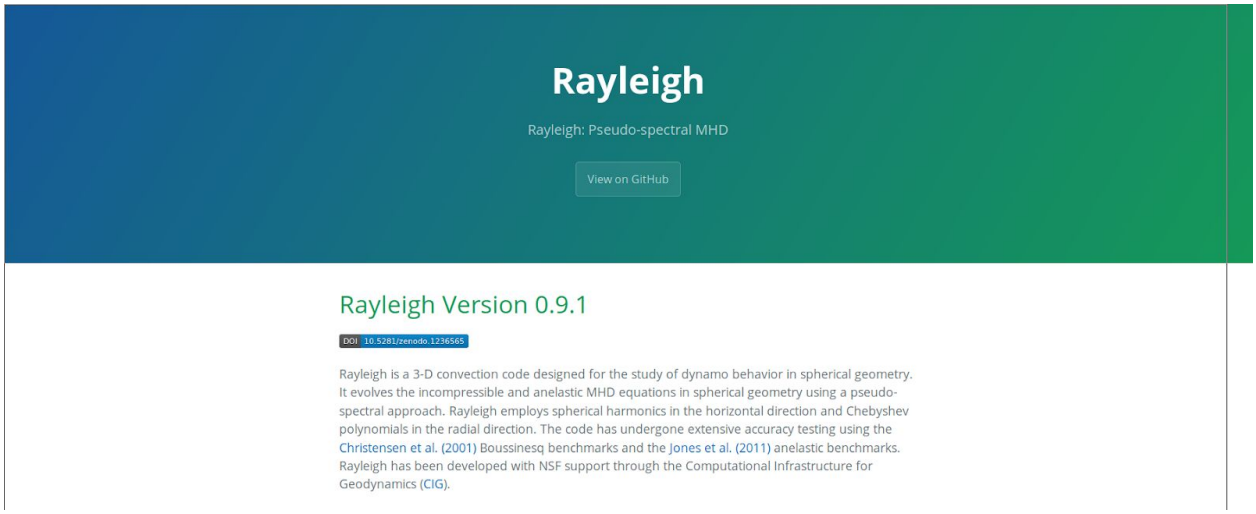
Rene Gassmoeller

The readme now also links to the contributing guidelines, to make them more visible.

Create a github pages website for Rayleigh

Rakesh Yadav, Nick Featherstone, Rene Gassmoeller

We created a preliminary webpage for Rayleigh, visible at
https://geodynamics.github.io/Rayleigh/



Implementation of the QI method

Ben Miquel, Kyle Augustson

For those curious about the Quasi-Inverse method for Chebyshev polynomials, here are the notes that Ben Miquel sent to me as prep work for understanding the work of Keith Julien and Mike Watson and how one might mesh it with the inner workings of Rayleigh. As a quick summary, the Quasi-Inverse formalism is a means by which the currently dense matrices used in the implicit solve may be recast into a block-banded sparse solve. This is done by expanding the non-constant coefficients and the radial derivatives of the mode-wise spherical-harmonically projected PDEs on Chebyshev polynomials and noting that the Kronecker product of the derivative operators and the Chebyshev coefficient matrices of the coefficients lead to a LHS matrix that then will have a sparse block-banded structure (as seen in Figures 7, 9, and 11 in the 2009 Julien & Watson paper).

We have begun tracing through Rayleigh to see where, at least as a test case, the Z equation may be cast into the QI formalism. The WPS equations, being coupled, require a more subtle consideration wherein the equation may need to be reformulated to have order-consistent boundary conditions by eliminating the third order radial derivative on W. This is nice in another way as it changes the spatial structure of the equations to being more elliptic and thus more akin to the underlying implicit Poisson solve for the instantaneously equilibrated pressure field. In any case, it looks like a fair amount of code modification within the equation sets, linear solve, and parallel framework will be necessary to accommodate the qi_pack library that Ben Miquel has written.

So far, we have implemented the data structures necessary to use Ben's routines and developed the coefficient transforms to build the sparse implicit matrices. The next step is for Ben to change which matrices are returned from his library so that they may be more directly used in the implicit solve. Once those matrices are available, we may implement the wrapping routine to the QI solving call in Ben's library and begin to test the QI implementation. If this is successful, we will tackle the implementation of the WPS equations so that more full scale numerical tests can be conducted.

Started on Cookbooks

Maria Weber

I have added 'cookbook' examples for the four Rayleigh main input files for the 'minimal' benchmark cases. I'm currently working on finishing up the cookbook example for the benchmark diagnostics output run with some example Python generated figures for verification. The latter part is work in progress.

Started on Cookbooks

Sebastian Glane

I fixed the table formatting in the Diagnostics_Plotting.pdf document. The use of the longtable package allows to now split the tables over several pages which yields a more compact formatting, see document in repository. Furthermore, I have started to write a documentation on the representation of the fields in Rayleigh and on the solution process discussed in the meetings. The overall picture flow of the solution is clear and needs to be sketched up in LaTeX. However, some details about the states of the buffer and variables still need to be clarified. As a minor contribution I replaced all tab stops in the src-folder by spaces. Tab stops and blank at the end of line were also deleted. At this stage, the Fortran compiler does not accept lines longer than ~130 characters. Therefore I had to manually insert some line continuations.

Fixing a boundary condition bug

Connor Bice

I fixed the boundary condition strict_L_conservation, which plugs an angular momentum leak but introduced a discontinuity at the top boundary which was most prominent for the models in which the leak is fast (deep shells). Verification run still in progress.

Fixed a variety of bugs in the turbulent and viscous diagnostics

Brad Hindman

1) I submitted a small set of bug corrections for sign errors in the viscous force diagnostics.
2) A math error was fixed in the spherical geometry in the viscous transport of turbulent kinetic energy.
3) The turbulent kinetic fluxes were all modified to correct an overall sign error.
4) An accuracy in the computation of the flow divergence was enhanced in the turbulent kinetic energy budget diagnostics.

Fixed various bugs

Cian Wilson

1) Fix for sed whitespace removal on macs
2) Prevent segfaulting when no fields are specified for an output type by checking if arrays are allocated before deallocating them
3) Spherical_3D output needed the access='stream' attribute to be set

Added 3D output diagnostic routines

Cian Wilson

Added a class to rayleigh_diagnostics.py that can read the Spherical_3D output from Rayleigh. Also added a script that takes this output and converts it to vtus. This currently leaves conical holes at the poles and we discussed potential fixes for this. Seemed to conclude that interpolating/averaging to a value at the pole and adding prismatic elements would be a reasonable way forward.

Added Kinetic Helicity and Axial Field Diagnostics

Ryan Orvedahl, Brad Hindman,Jon Aurnou,Valentin Skoutnev, Krista Soderlund

Added new quantity codes relating to the various components of the kinetic helicity and tested the outputs on a benchmark. Added a new Diagnostics Module for Axial Fields and completed coding the velocity and helicity parts.

Added a function to fetch local address of spectrum data from spherical harmonics mode

Hiro Matsui

Add a function to obtain local address of the spectrum data array (p1a, pab) from imput spherical harmonics mode Y_{l}^{m}. This function will be in the module Load_Balance.
The function will be
j = Find_local_sph_mode_address(l, m).

If process does not have the requested mode, Find_local_sph_mode_address returns 0.
If process has the requested mode Find_local_sph_mode_address returns local address for spherical harmonics mode.

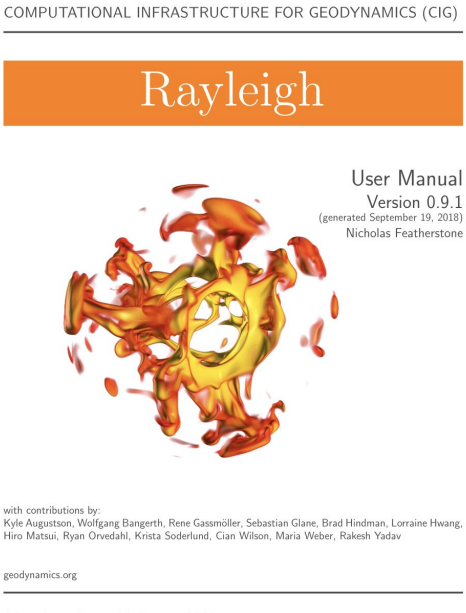
Documentation Improvements

Lorraine Hwang

1. Fixed miscellaneous typos and corrected path name to files that were changed due to the directory restructuring prior to the hack as well as removal of some files during the hack. More cleanup and proofreading needs to be done.

2. Added a cover page with image to the User Manual. This adds as contributors everyone who is listed as a contributor to the repository and all those with pending pull requests.

3. Added a .bib file with the obvious references in the manual. However, this has not been fully implemented. Will need future work.



Create Discourse Forum

Lorraine Hwang

Initiated new forum for discussions at community.geodynamics.org. The Rayleigh developers group will be our beta testers for configuring this new tool for CIG.

Statistics about Rayleigh's growth during the hackathon

The following contains a number of statistics about how much Rayleigh has grown during the hackathon (until Oct 3rd, 2018):

- Number of source files in Rayleigh before/after: 94 -> 189 +95
- Lines of code in Rayleigh before/after: 82,217 -> 84,643 +2,426
- Number of merged pull requests before/after: 2 -> 46 +44
- Commits in github before/after: 78 -> 128 +50
- Number of tests before/after: 0 -> 1 +1

These statistics were generated through the following commands:

- find src/ | egrep '\.(F|F90|c|py)$' | wc -l
- cat `find src/ | egrep '\.(F|F90|c|py)$'` | wc -l
- git log --format=oneline | grep "Merge" | wc -l
- git log --format=oneline | grep -v "Merge" | wc -l
- ls -l tests/* | wc -l