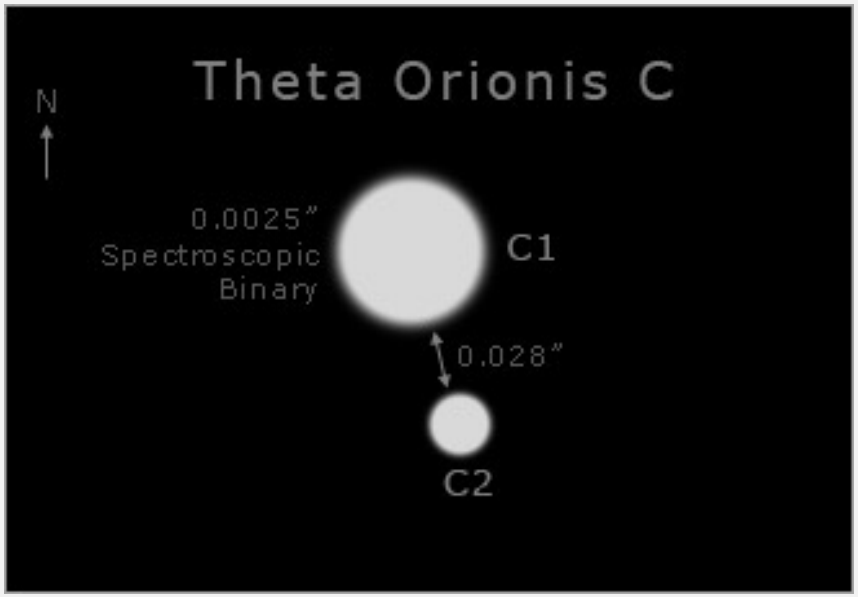
Final Presentation Astrophysics Project

Finn Schmidhuber, Ethan Jaegers,
Emily Miller, & Katherine McCarthy

Agenda

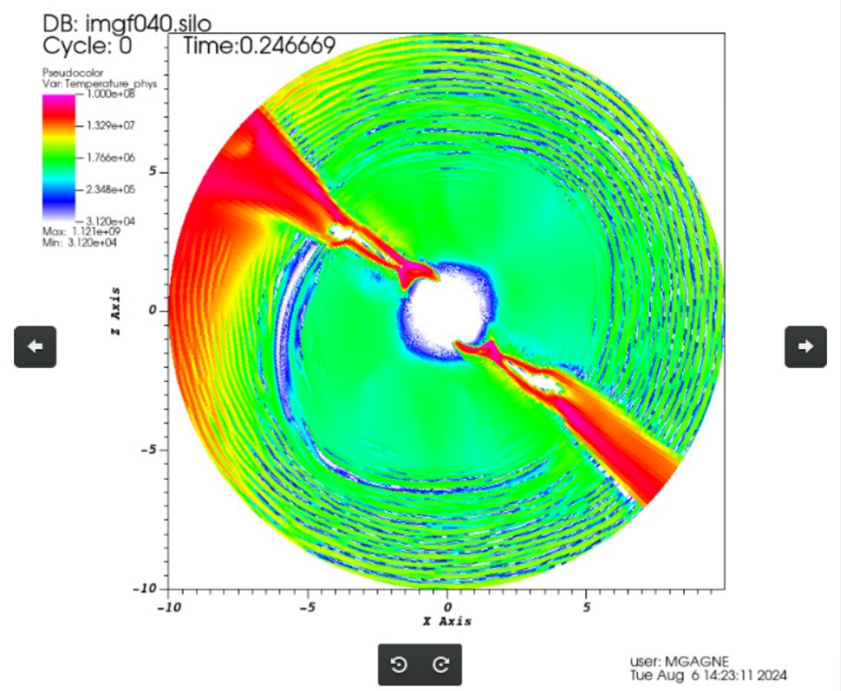
- Review of Goals & Project Base
- Project Progress
  - Makefile
  - Compiling & Optimization
  - Post-Processing
- Unexpected Events
- Minimizing Risk
- Testing
- Final Remarks



Background & Goals

Background Refresher

- Visualization of star atmospheres for researchers.
- Fortran based.
- Currently uses NASA supercomputer.
- Large scale.
  - Complete run takes 4-7 days
- Not initially made for public use.
- Emphasis:
  - X-rays
  - Electromagnetic fields



General Goals

- Emphasis on **exploration,** not **completion.**
- Understanding of codebase and current weaknesses / areas of improvement.
  - "Messing with the hornet's nest" metaphor.
- Create space for future groups to update and work on software.

01	02	03	04	05
1.) **Reduce learning curve for users**	2.) **Reduce potential for user error**	3.) **Increase usability for various systems**	4.) **Increase replicability for future growth**	5.) **Improve post-processing usage**

Project Progress

Makefile Successes

- Streamlined the makefile.
  - Removed unnecessary components.
  - Updated Fortran_Sources.
    - Reflects source code files directly.
  - Gave the best output out of all versions.

```
# Compilers and flags
CXX = mpic++
FC = mpif90
CFLAGS = -Wall -g -O2
FFLAGS = -fdefault-real-8 -mcmodel=medium -O2
LIBS = -lm -lstdc++

# Silo library
SILO_LIBS = -L/path/to/libsilo -lsilo
SILO_INCLUDES = -I/path/to/libsilo/include

# Source files
FORTRAN_SOURCES = Apps.F Riemann.F Main.F Ader.F \
                  BuildGeomesh.F onedRS.F twodRS.F MatSolv.F CAK.f
CXX_SOURCES = qgm_parallelio.cc
FORTRAN_OBJS = $(FORTRAN_SOURCES:.F=.o)
CXX_OBJS = $(CXX_SOURCES:.cc=.o)
TARGET = xray_project

# Default rule
all: mpi

# MPI executable
mpi: $(FORTRAN_OBJS) $(CXX_OBJS)
    $(FC) $(FFLAGS) -o $(TARGET) $(FORTRAN_OBJS) $(CXX_OBJS) \
        $(SILO_LIBS) $(LIBS) -lmpi_cxx

# Pattern rules
%.o: %.F
    $(FC) $(FFLAGS) -c $< -o $@

%.o: %.cc
    $(CXX) $(CFLAGS) -c $< -o $@

# Clean up
clean:
    rm -f *.mod *.o *.out $(TARGET) *.silo *00*

.PHONY: all clean mpi
```

Makefile Successes Cont.

```
# Source files
FORTRAN_SOURCES = Apps.F Riemann.F Main.F Ader.F \
                  BuildGeomesh.F onedRS.F twodRS.F MatSolv.F CAK.f
CXX_SOURCES = qgm_parallelio.cc
FORTRAN_OBJS = $(FORTRAN_SOURCES:.F=.o)
CXX_OBJS = $(CXX_SOURCES:.cc=.o)
TARGET = xray_project
```

Makefile Progress

- Progress barred by errors.
- Run 6 and 4.
  - .f -> .o
- Errors are not from makefile.
  - "No rule to make target"
  - "Command not found"
  - "Miconfigured flags..."

```
Error: Symbol ‘rad’ at (1) already has basic type of REAL
Apps.F:3557:16:

 3557 |       REAL :: d0, u0, p0, gammaa, d, uu, p, a, b,
      |               1
Error: Symbol ‘d0’ at (1) already has basic type of REAL
Apps.F:3561:19:

 3561 |       REAL :: alpha, beta, tau, r0, omega, mach
      |                   1
Error: Symbol ‘alpha’ at (1) already has basic type of REAL
Apps.F:3563:16:

 3563 |       REAL :: Ar, Br, Br0, Bphi, Bphi1, Bphi2, eps, tempaa,
      |               1
Error: Symbol ‘ar’ at (1) already has basic type of REAL
Apps.F:3569:72:

 3569 |       REAL :: V_CAK ( 3), cak_rhoc, xs, ys, zs, Rs
      |
Error: Unexpected data declaration statement at (1)
Apps.F:4344:9:

 4344 |       END SUBROUTINE PROBLEM_GENERATOR_U_A
      |         1
Error: Expecting END PROGRAM statement at (1)
Apps.F:4400:20:

 4400 |       IMPLICIT NONE
      |                    1
Error: Duplicate IMPLICIT NONE statement at (1)
riemann.com:4:23:

    4 |       LOGICAL do_on_gpu
      |                        1
Error: Symbol ‘do_on_gpu’ at (1) already has basic type of LOGICAL
riemann.com:6:19:
```

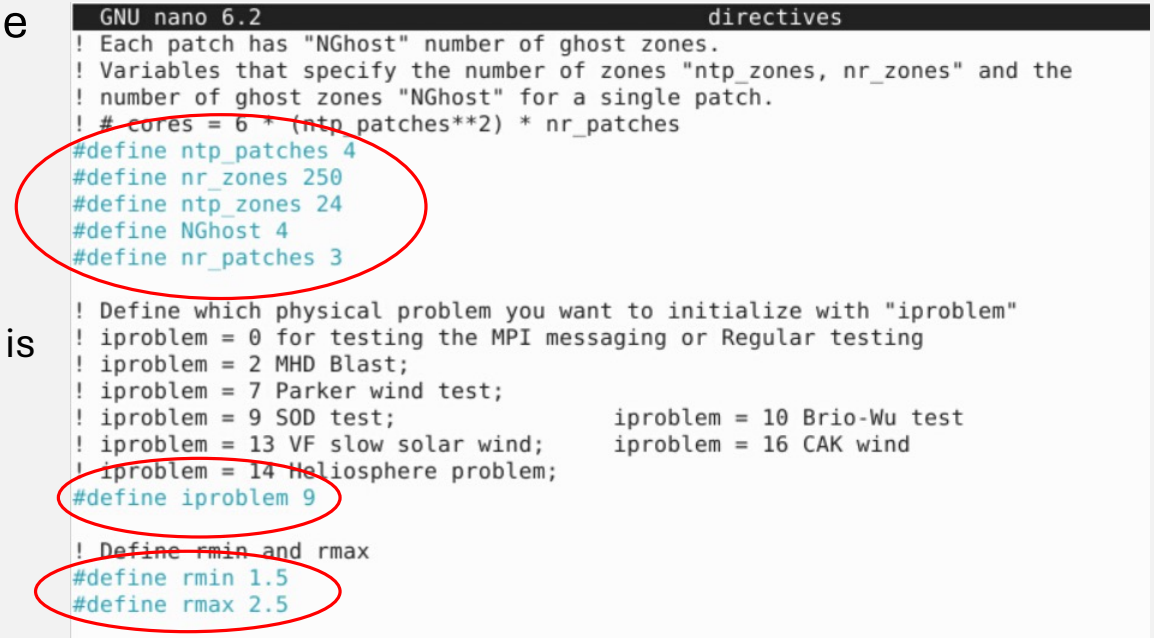
```
      |                            1
Error: Rank mismatch between actual argument at (1) and actual argument at (2) (rank-1 and scalar)
Main.F:8864:23:

 8779 |       CALL MPI_Bcast   ( gl_err_U, n_cc_components, MPI_REAL8, 0,
      |                           2
......
 8864 |       CALL MPI_Bcast ( gl_err_Efx, 1, MPI_REAL8, 0, MPI_COMM_WORLD,ierr)
      |                         1
Error: Rank mismatch between actual argument at (1) and actual argument at (2) (rank-1 and scalar)
Main.F:8865:23:

 8779 |       CALL MPI_Bcast   ( gl_err_U, n_cc_components, MPI_REAL8, 0,
      |                           2
......
 8865 |       CALL MPI_Bcast ( gl_err_Efy, 1, MPI_REAL8, 0, MPI_COMM_WORLD,ierr)
      |                         1
Error: Rank mismatch between actual argument at (1) and actual argument at (2) (rank-1 and scalar)
Main.F:8866:23:

 8779 |       CALL MPI_Bcast   ( gl_err_U, n_cc_components, MPI_REAL8, 0,
      |                           2
......
 8866 |       CALL MPI_Bcast ( gl_err_Efz, 1, MPI_REAL8, 0, MPI_COMM_WORLD,ierr)
      |                         1
Error: Rank mismatch between actual argument at (1) and actual argument at (2) (rank-1 and scalar)
Main.F:12354:27:

11269 |       CALL MPI_ALLREDUCE ( dtcurp1_reduce,   ! Input for reudction
      |                            2
......
12354 |       CALL MPI_ALLREDUCE ( V_min_local,          ! Input for reudction
      |                            1
Error: Rank mismatch between actual argument at (1) and actual argument at (2) (scalar and rank-1)
Main.F:12359:27:
```

Makefile Wrap-Up

- Testing phase.
  - Detailed errors in documentation for future group(s).
  - More work to be done for full functionality.
    - Error analysis and fixes within source code.
- Multiple makefile versions.
  - Placed in Github doc folder.

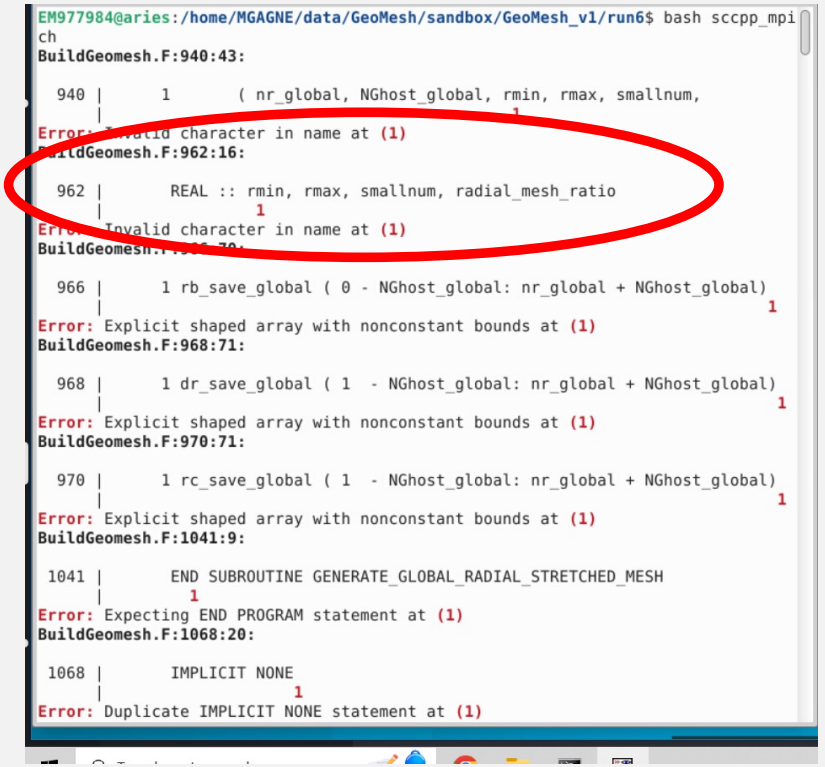
Compiling and Optimization

- Successfully globalized some essential variables for easy access to users.
  - Successfully compiles after removing repeated mentions within Fortran programs.
  - Process to globalize variables is repeatable and documented.
  - Big step towards end goal.

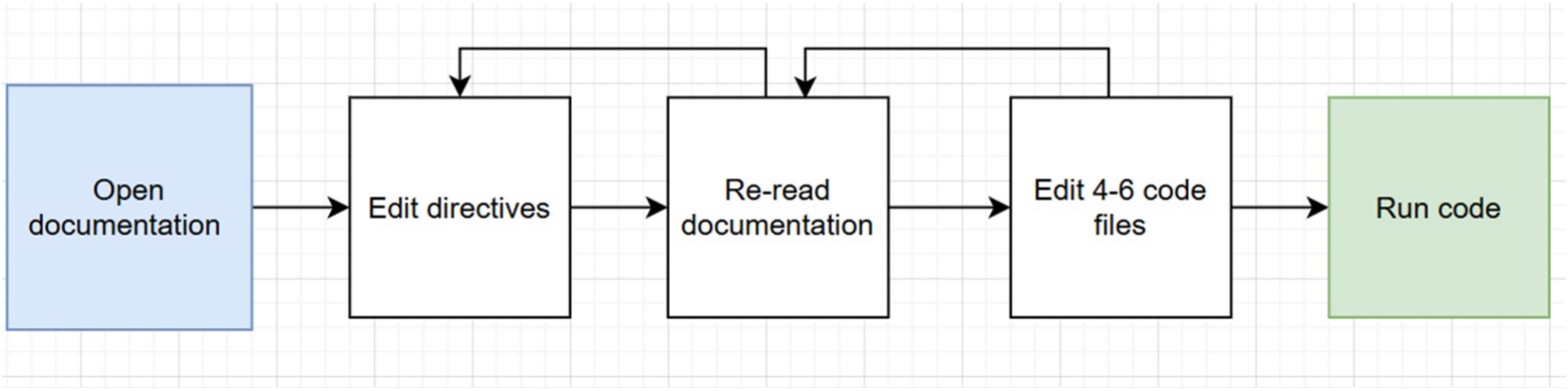


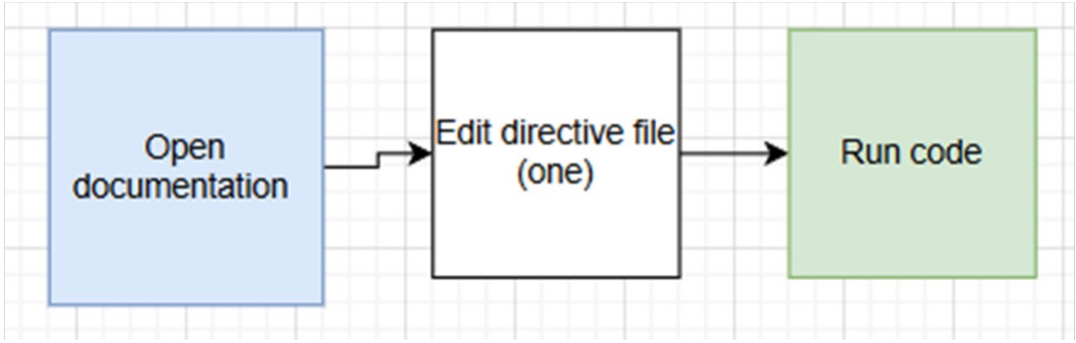
Compiling and Optimization

- Developed process for navigating code base and errors in directives file updates.
  - Lots of redundancies in variable declaration throughout code base.
  - Migrating 1 variable to directives resulted in ~200 errors.
- Timeframe for variable migration reduced from 1.5 hours to 25 minutes.



Flowchart Representation

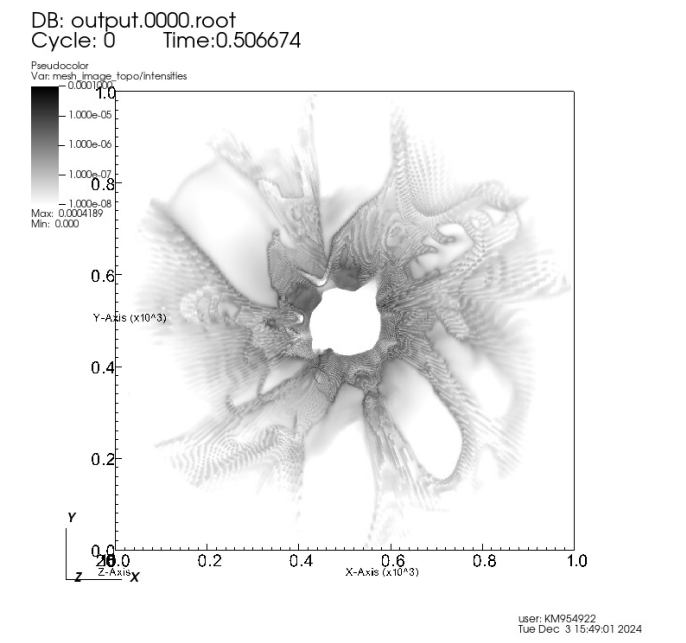




Post-Processing

- Updated both qotLxb.py & xray_image_3D.py.
- Ran and tested 4 sets of zoom & quality .
  - Issue with testing and saving both 2048 quality tests.
  - Abandoned in favor of zoom 1 & 2 for 1024 quality.
- Organized testing.
  - 4 separate directories for each test of quality and zoom.
- Created thorough and explanatory documentation.

```
KM954922@aries:/data/CooMesh/sandbox/run19/xray$ ls
hdf5_z1_1024  hdf5_z2_1024  lnsilo.sh          visit0000.session.gui  xray_image_3D.py
hdf5_z1_2048  hdf5_z2_2048  visit0000.session  visitlog.py
```

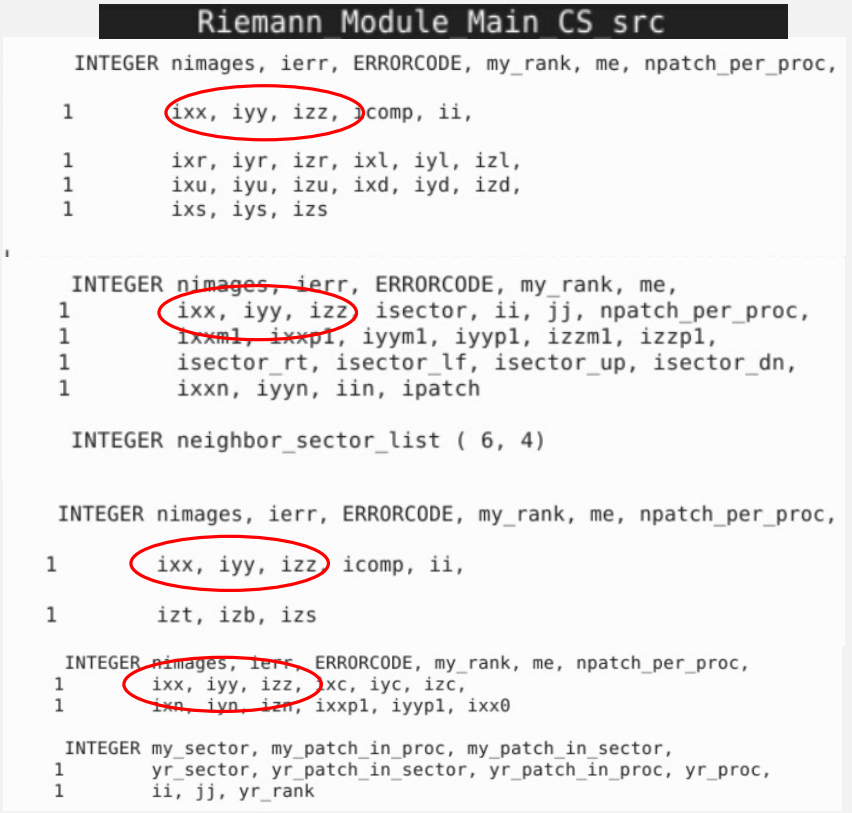


VisIt output of zoom 1, 1024 quality.

Overarching Aspects

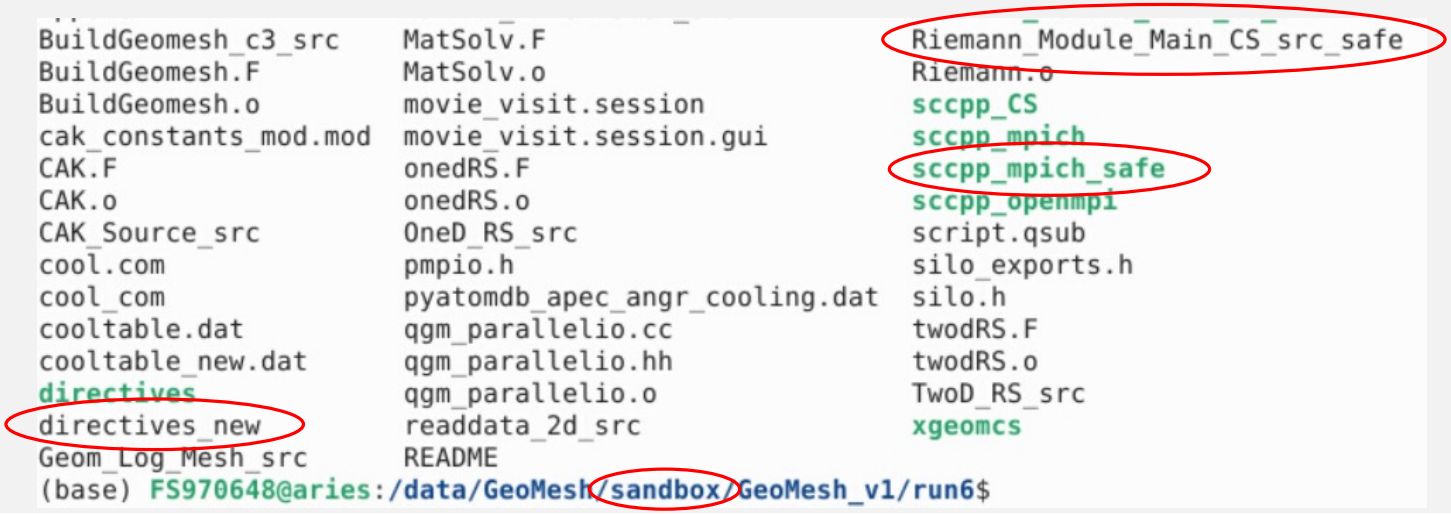
Unexpected Events

- Loss of storage space due to continuous runs and files in the system.
  - Rendered VNC unusable until our client deleted large unneeded files.
- Code is made in a convoluted and circuitous manner, making updates difficult.
  - Results in tedious and repetitive fixes.
  - Very prone to errors within the compiling of the entire program.



Minimizing Risk

- All code edits completed in a sandbox environment.
- File editing completed with back-up of each file in the run.
- Separate directories were made for each testing case.
- Comprehensive documentation for all changes made.



Testing Phase

- Compile code without errors and use silo files to ensure validity and consistency in results.
  - Currently on run 7, compiled last on Tues. evening.
- Compare compilation time to previous runs.
- Post-processing aspect: Examine visuals using various filters or visualization options to ensure accuracy.

Final Remarks

Goals for Future Groups

- What still needs to be done:
  - Creating and formatting documentation within directives file.
  - Globalizing variables for user-friendly experience.
  - Continued documentation on GitHub for users to follow step-by-step

01	02	03	04	05
1.) Code refactoring.	2.) Creating & Formatting Documentation for Directives file	3.) Globalizing Variables	4.) Detailed documentation for Future Users	5.) Improve and complete GitHub Repository.

Deliverables for the Future Groups

Github	Makefile	Optimization	Post Processing
• Functional sandbox environment. • Documentation folder.	• Multiple versions created.	• Modified compilers. • Create directives file.	• Thorough Documentation for VisIt process. • Separate directories for testing & reference.


Q&A