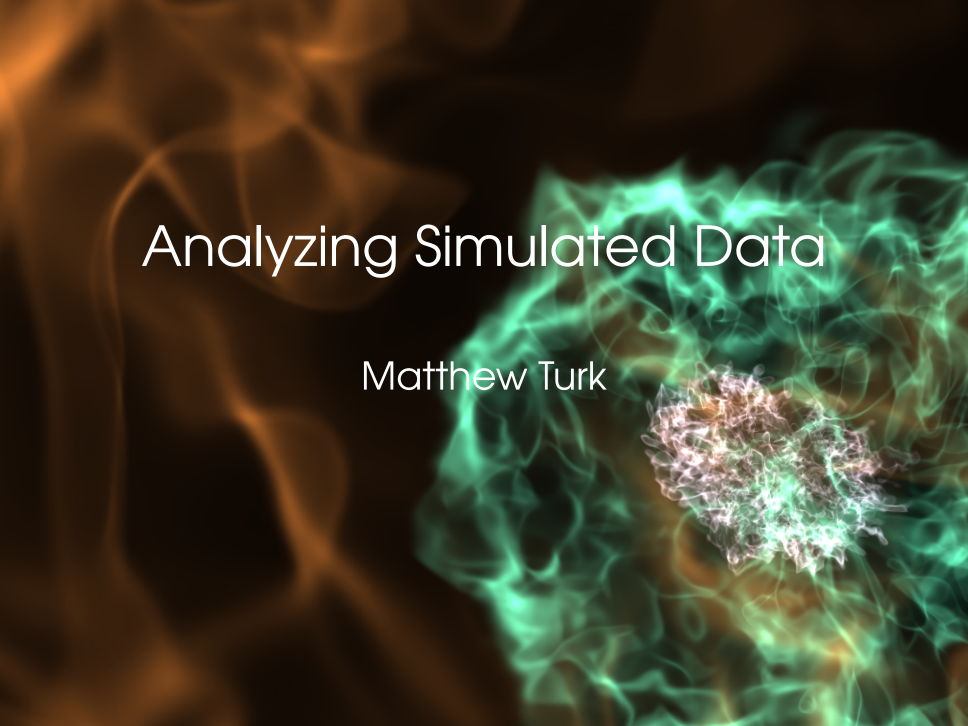
# Analyzing Simulated Data

Matthew Turk



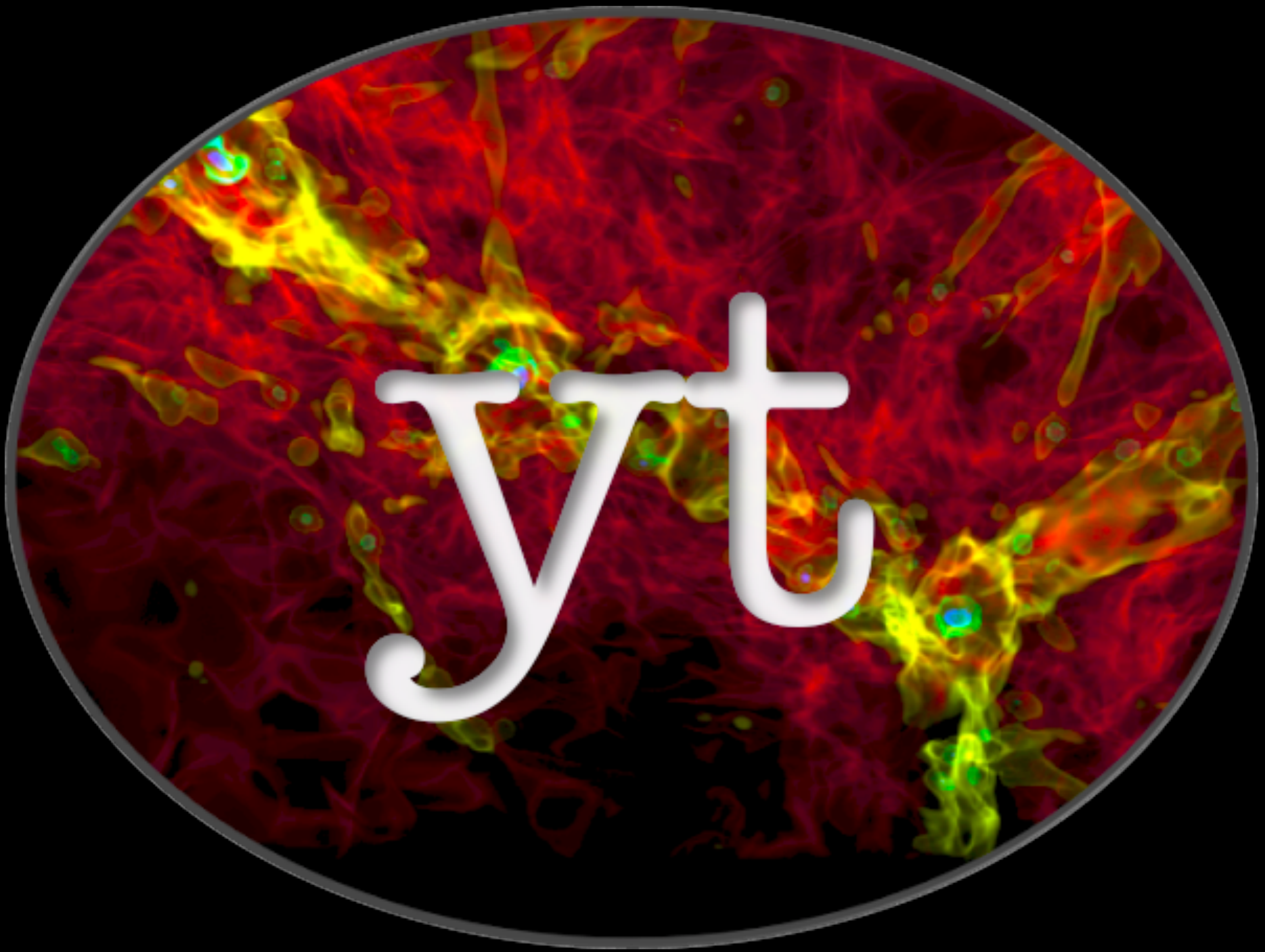
# There is only one sky.

(but there are many simulation codes)

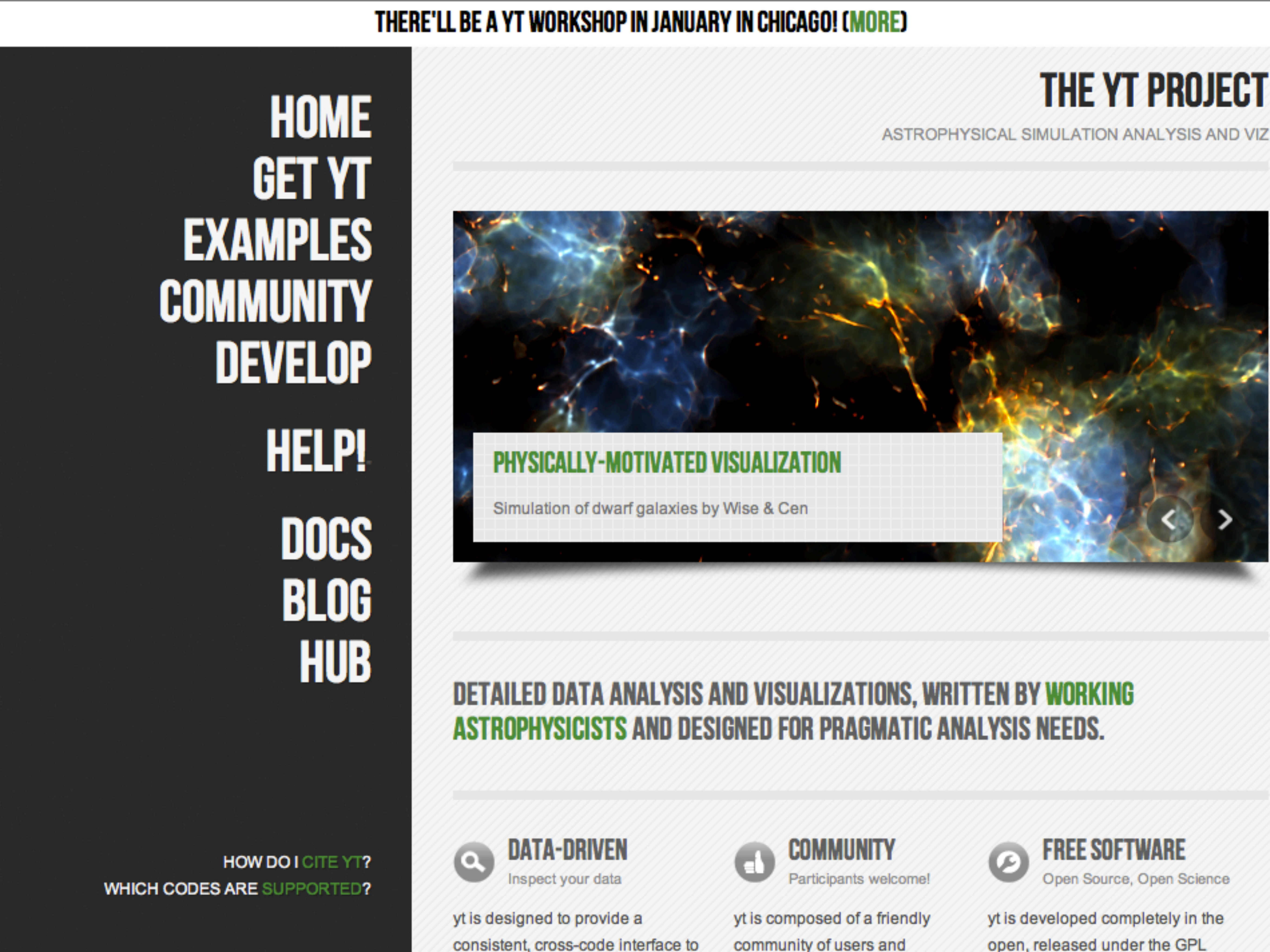
Different methods,
data structures,
assumptions,
IO methods,
units,
variable names,
...

aimeric@acapulco

# Analysis.



astro-ph/1011.3514
yt-project.org



THERE'LL BE A YT WORKSHOP IN JANUARY IN CHICAGO! (MORE)

HOW DO I CITE YT?

WHICH CODES ARE SUPPORTED?

# THE YT PROJECT

ASTROPHYSICAL SIMULATION ANALYSIS AND VIZ

PHYSICALLY-MOTIVATED VISUALIZATION

Simulation of dwarf galaxies by Wise & Cen

## DETAILED DATA ANALYSIS AND VISUALIZATIONS, WRITTEN BY WORKING ASTROPHYSICISTS AND DESIGNED FOR PRAGMATIC ANALYSIS NEEDS.

### DATA-DRIVEN

Inspect your data

yt is designed to provide a consistent, cross-code interface to

### COMMUNITY

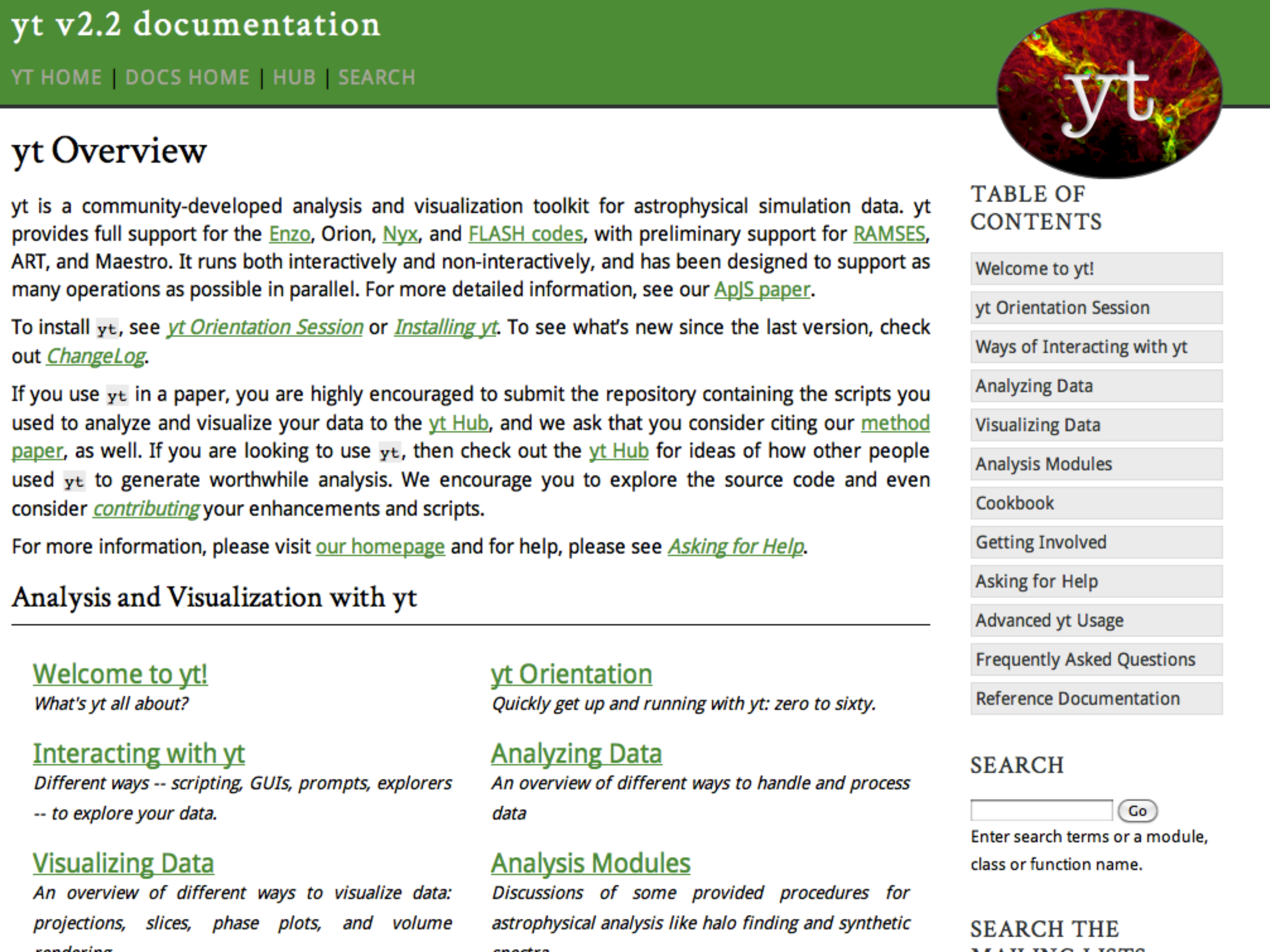
Participants welcome!

yt is composed of a friendly community of users and

### FREE SOFTWARE

Open Source, Open Science

yt is developed completely in the open, released under the GPL



# yt Overview

yt is a community-developed analysis and visualization toolkit for astrophysical simulation data. yt provides full support for the Enzo, Orion, Nyx, and FLASH codes, with preliminary support for RAMSES, ART, and Maestro. It runs both interactively and non-interactively, and has been designed to support as many operations as possible in parallel. For more detailed information, see our ApJS paper.

To install `yt`, see *yt Orientation Session* or *Installing yt*. To see what's new since the last version, check out *ChangeLog*.

If you use `yt` in a paper, you are highly encouraged to submit the repository containing the scripts you used to analyze and visualize your data to the yt Hub, and we ask that you consider citing our method paper, as well. If you are looking to use `yt`, then check out the yt Hub for ideas of how other people used `yt` to generate worthwhile analysis. We encourage you to explore the source code and even consider *contributing* your enhancements and scripts.

For more information, please visit our homepage and for help, please see *Asking for Help*.

## Analysis and Visualization with yt

### Welcome to yt!
*What's yt all about?*

### yt Orientation
*Quickly get up and running with yt: zero to sixty.*

### Interacting with yt
*Different ways -- scripting, GUIs, prompts, explorers -- to explore your data.*

### Analyzing Data
*An overview of different ways to handle and process data*

### Visualizing Data
*An overview of different ways to visualize data: projections, slices, phase plots, and volume rendering*

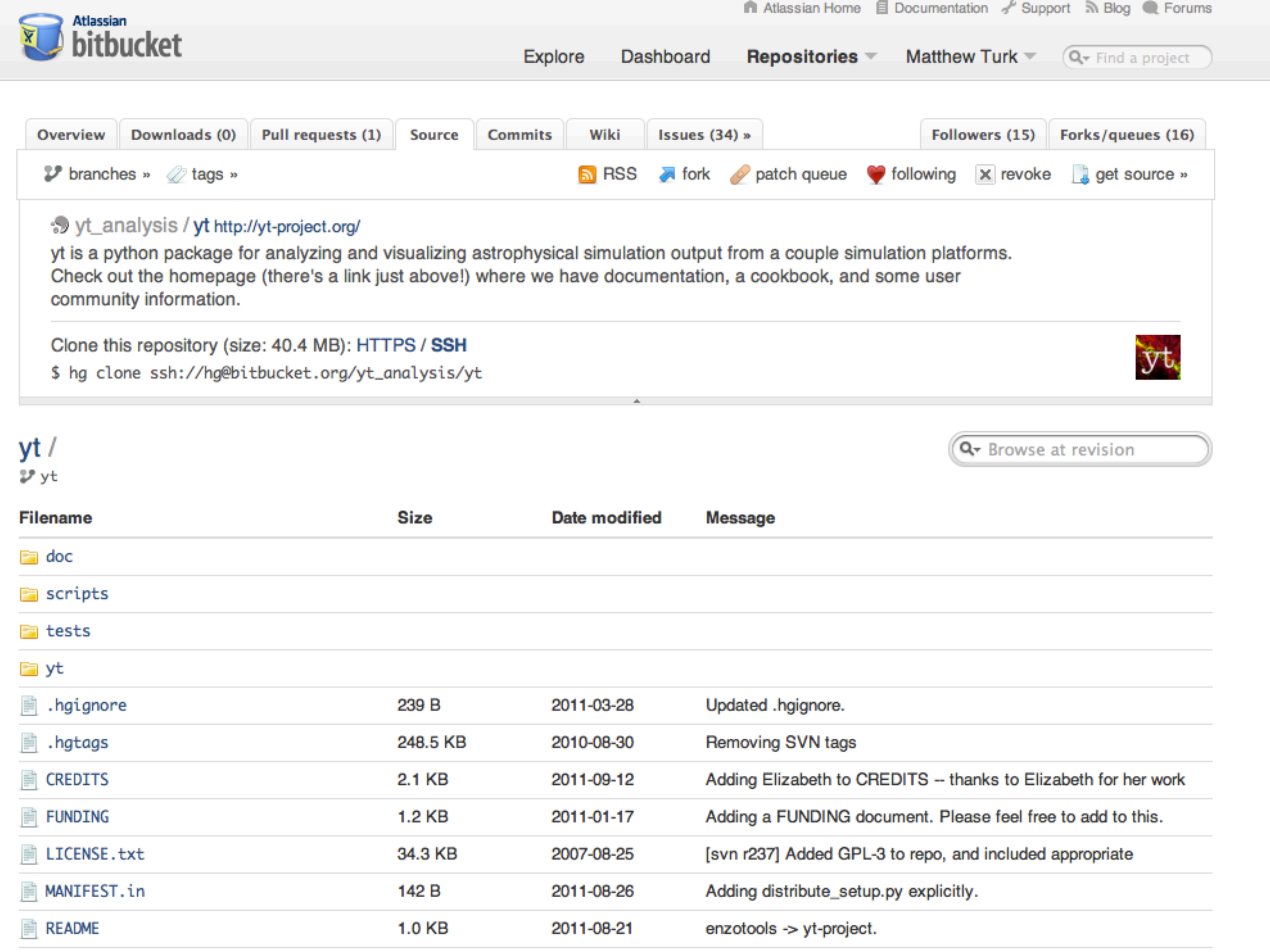
### Analysis Modules
*Discussions of some provided procedures for astrophysical analysis like halo finding and synthetic spectra*

TABLE OF CONTENTS

SEARCH

Enter search terms or a module, class or function name.

SEARCH THE



yt_analysis / yt http://yt-project.org/

yt is a python package for analyzing and visualizing astrophysical simulation output from a couple simulation platforms. Check out the homepage (there's a link just above!) where we have documentation, a cookbook, and some user community information.

Clone this repository (size: 40.4 MB): HTTPS / SSH

```
$ hg clone ssh://hg@bitbucket.org/yt_analysis/yt
```

yt /

yt

| Filename | Size | Date modified | Message |
|---|---|---|---|
| doc | | | |
| scripts | | | |
| tests | | | |
| yt | | | |
| .hgignore | 239 B | 2011-03-28 | Updated .hgignore. |
| .hgtags | 248.5 KB | 2010-08-30 | Removing SVN tags |
| CREDITS | 2.1 KB | 2011-09-12 | Adding Elizabeth to CREDITS -- thanks to Elizabeth for her work |
| FUNDING | 1.2 KB | 2011-01-17 | Adding a FUNDING document. Please feel free to add to this. |
| LICENSE.txt | 34.3 KB | 2007-08-25 | [svn r237] Added GPL-3 to repo, and included appropriate |
| MANIFEST.in | 142 B | 2011-08-26 | Adding distribute_setup.py explicitly. |
| README | 1.0 KB | 2011-08-21 | enzotools -> yt-project. |

# install script:

Full dependency stack

Source code

Development environment

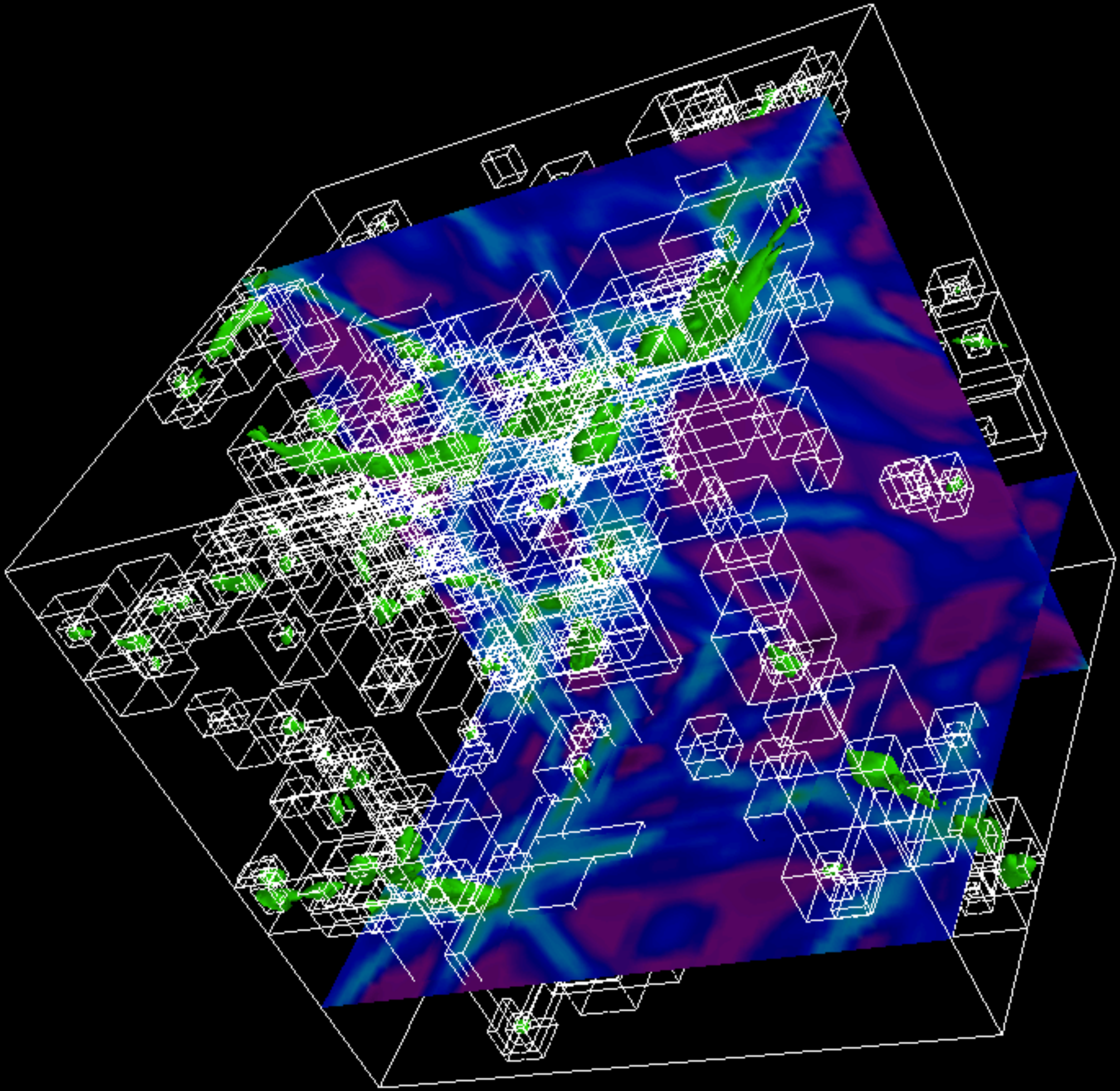
GUI

Sample data

yt has been designed to address physical, not computational, entities.

# Process

- Read
- Correlate
- Process
- Visualize



The Universe is full of gas, dark matter and stars. yt makes it easy to access that material.

Transparent IO, masking of overlapping data, load-on-demand, geometric and non-geometric selection, field generation, and common interfaces to different datatypes.

Enzo, Orion, CASTRO, FLASH

Chombo, Tiger, Athena, ART, RAMSES

yt is designed to be the *lingua franca* of astrophysical codes.

# Objects

(conceptual, uniformly accessible NumPy stores)

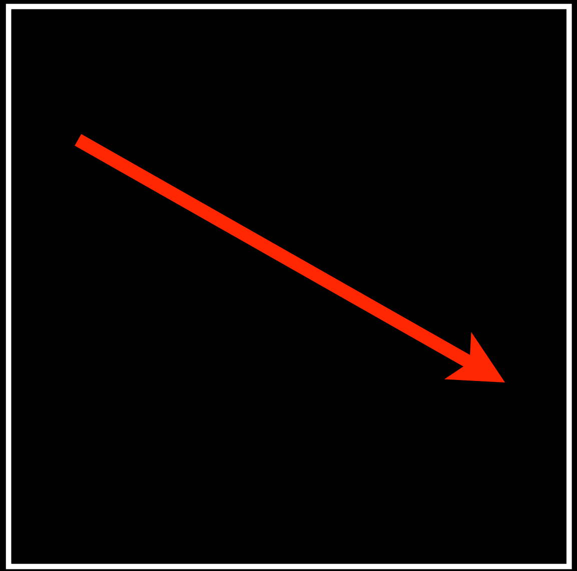
# Objects

| | |
|---|---|
| Orthogonal Rays | 1D |
| Non-orthogonal Rays | |
| Slices | 2D |
| Oblique Slices | |
| Projections | |
| Spheres | 3D |
| Rectangular Prisms | |
| Disks/Cylinders | |
| Inclined Boxes | |
| Clumps | |
| Extracted Regions | |
| Boolean combinations | |

# Objects

All respect unified interface:

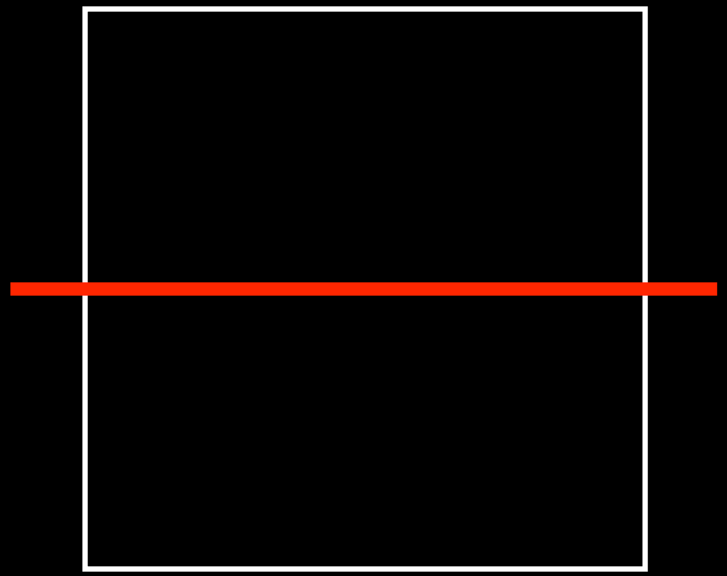
```
from yt.mods import *
pf = load(“DataDump0155.dir/DataDump0155”)
ray = pf.h.ray([0.1, 0.2, 0.5],
               [0.4, 0.9, 0.1])
print ray[“Density”]
```



# Objects

All respect unified interface:

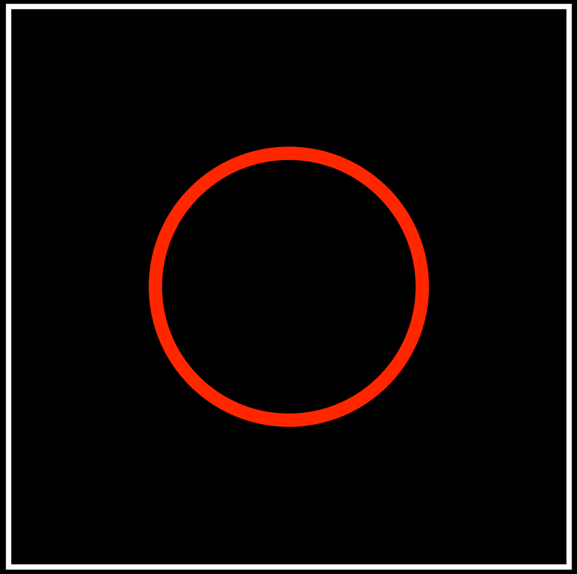
```
from yt.mods import *
pf = load(“DataDump0155.dir/DataDump0155”)
sl = pf.h.slice(0, 0.5)

print sl[“Density”]
```



# Objects

All respect unified interface:

```
from yt.mods import *
pf = load(“DataDump0155.dir/DataDump0155”)
sp = pf.h.sphere(100.0/pf[‘au’], ‘max’)

print sp[“Density”]
```



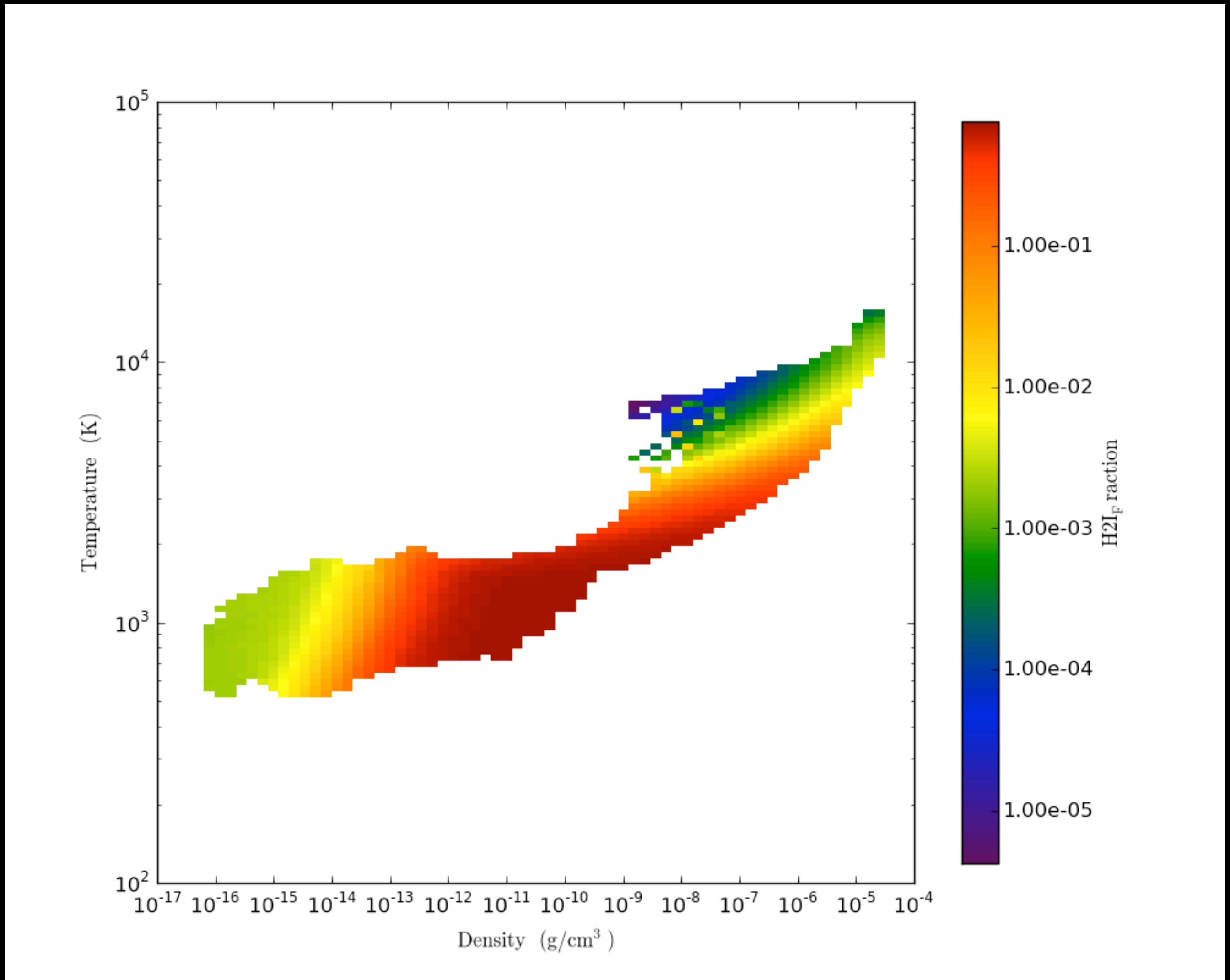
```
from yt.mods import *
pf = load(“DataDump0155.dir/DataDump0155”)
v, c = pf.h.find_max(“Density”)
```

Adding new fields should be easy.

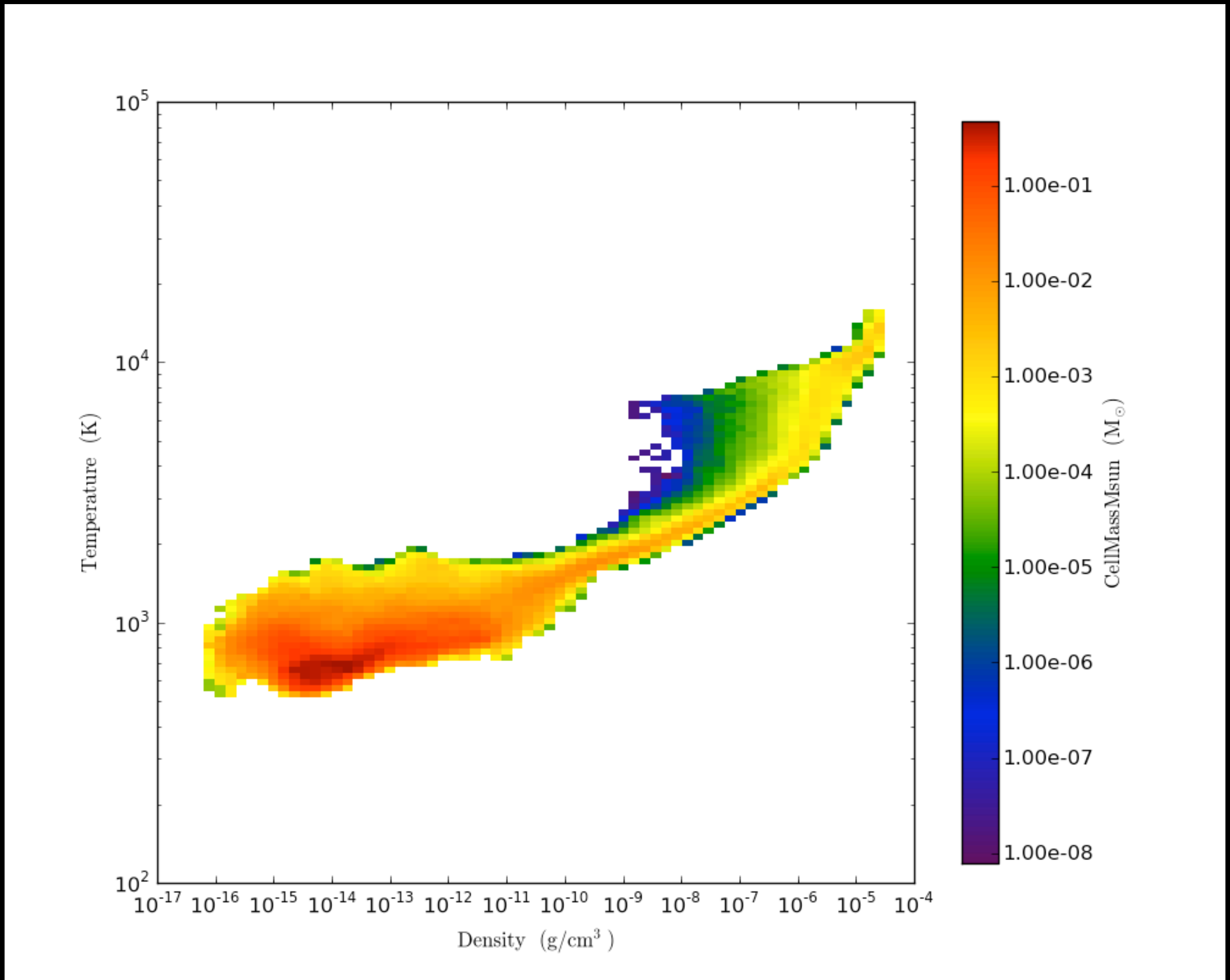
```
from yt.mods import *

@derived_field(“Pressure”)
def Pressure(field, data):
    return (data.pf[”Gamma”] - 1.0) * \
       data[”Density”]*data[”ThermalEnergy”]
```

Scripts should be simple and clear.

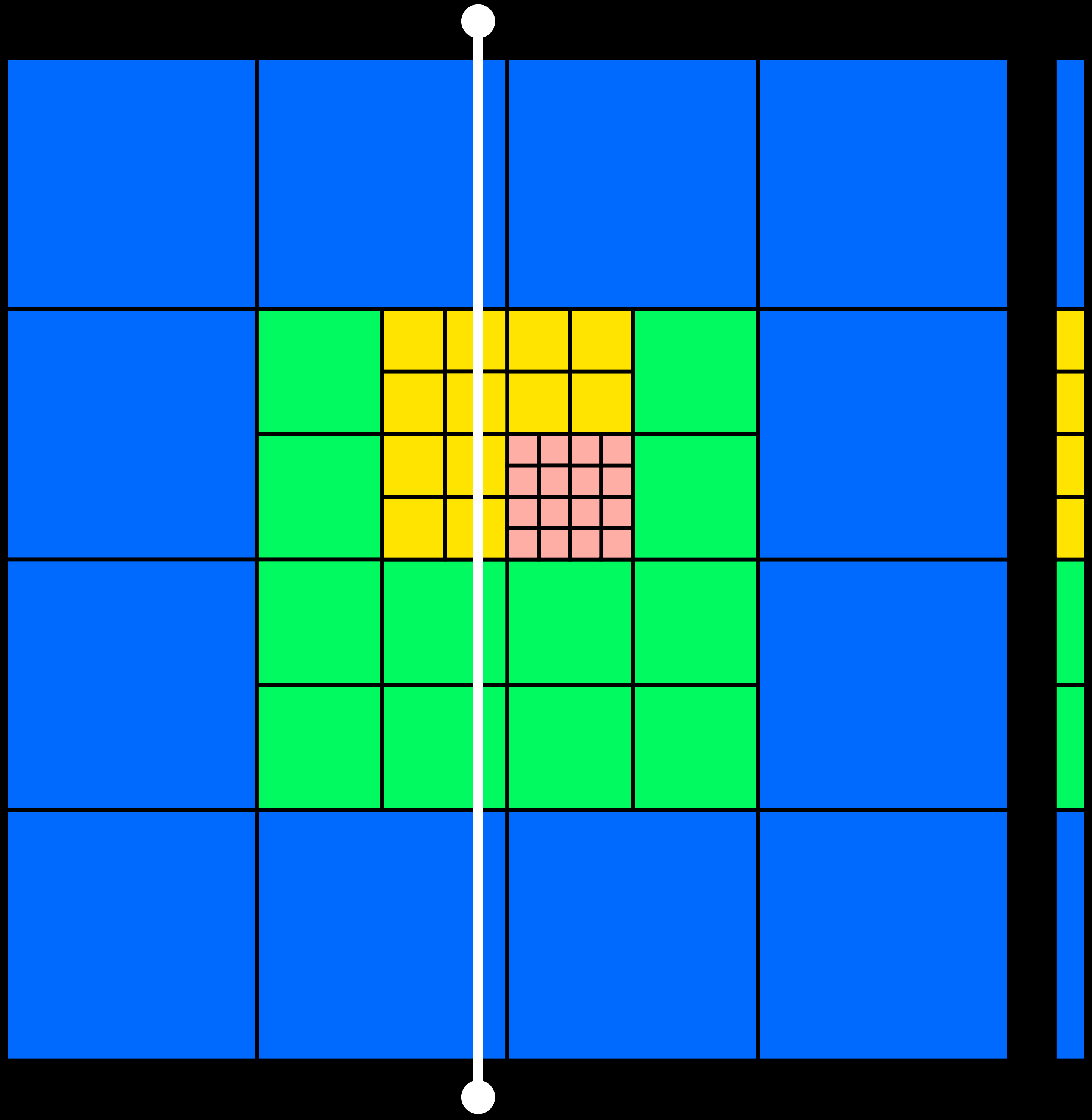
```
from yt.mods import *
pf = load(“DataDump0155.dir/DataDump0155”)
pc = PlotCollection(pf)
pc.add_phase_sphere(1000.0, ‘au’,
 [“Density”, “Temperature”, “H2I_Fraction”])
pc.save()
```

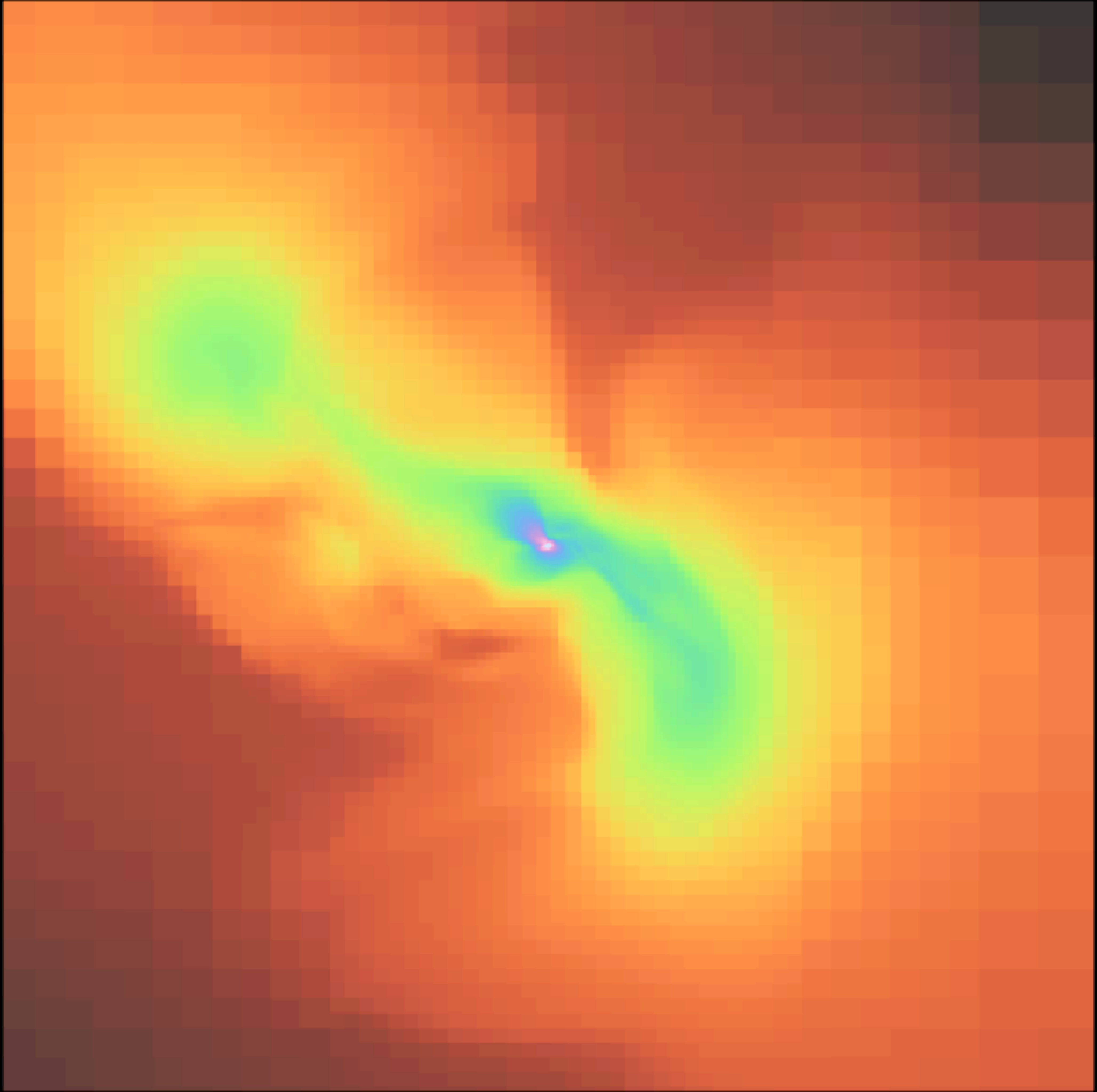


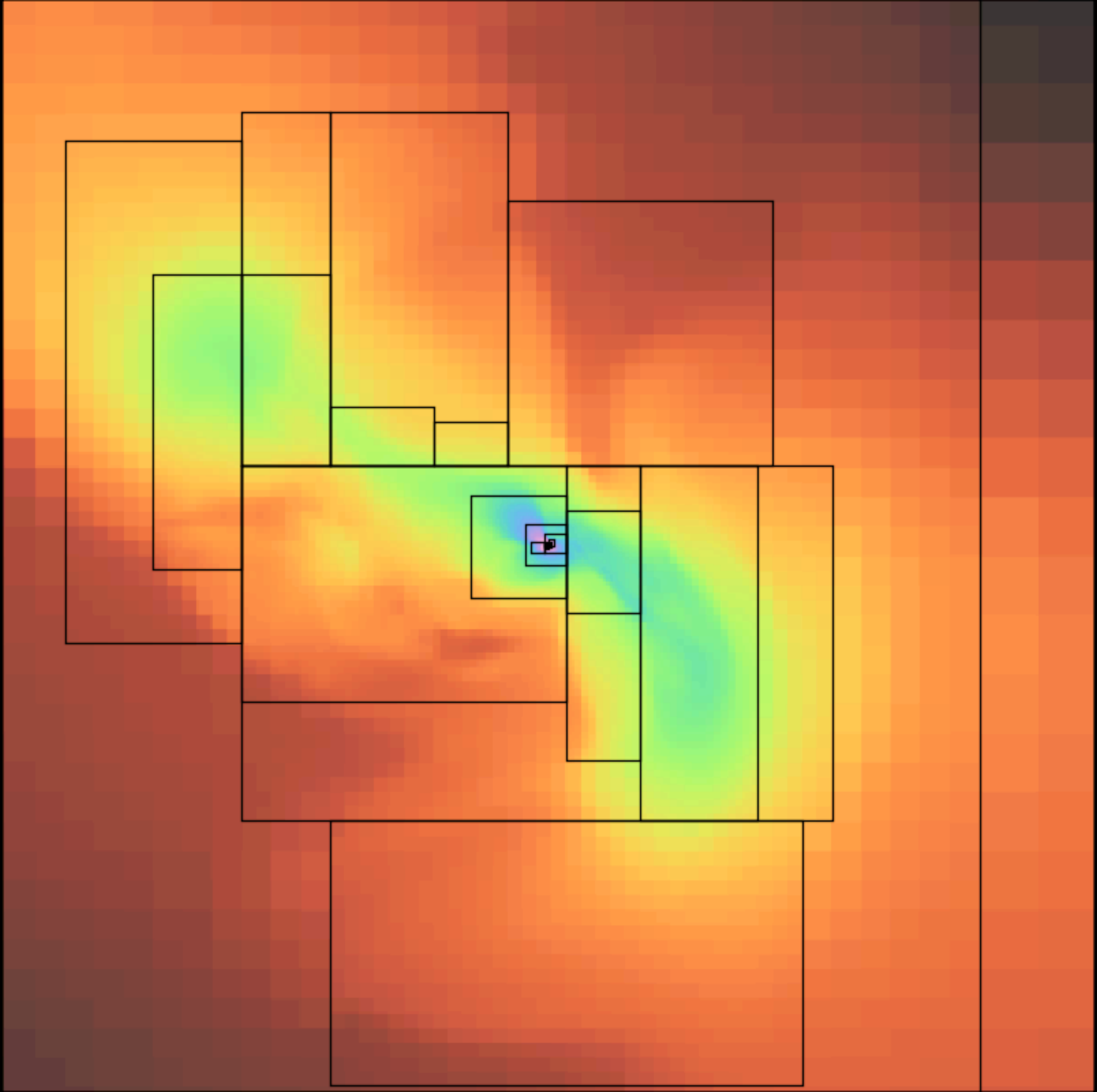
```
from yt.mods import *
pf = load(“DataDump0155.dir/DataDump0155”)
pc = PlotCollection(pf)
pc.add_phase_sphere(1000.0, ‘au’,
 [“Density”, “Temperature”,
  “CellMassMsun”], weight = None)
pc.save()
```



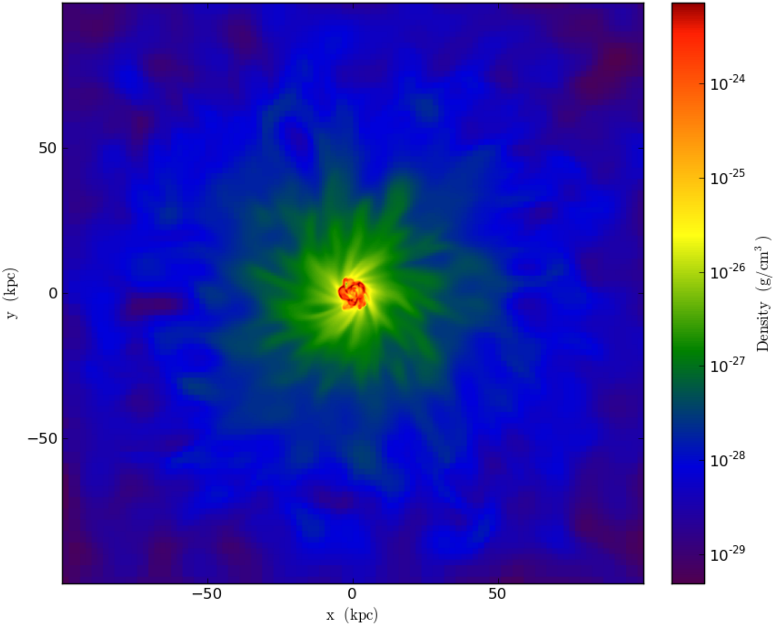
# Slices



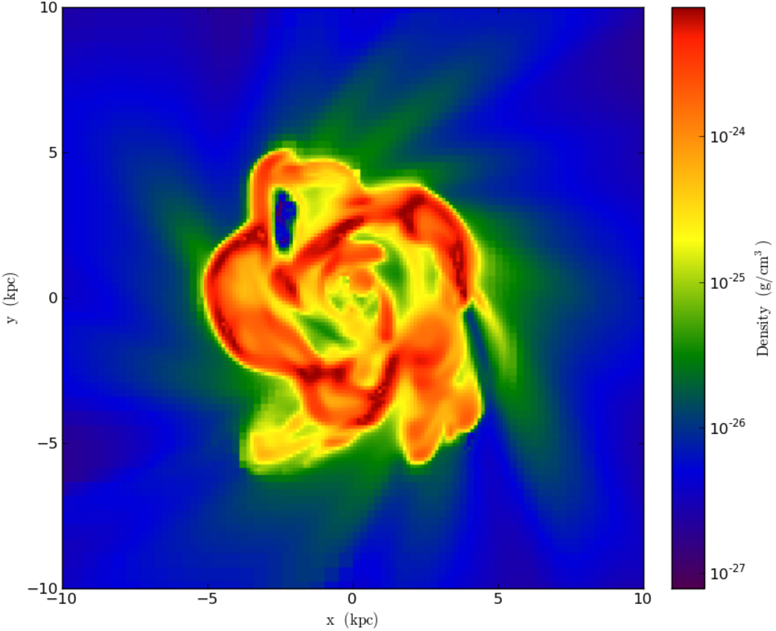




```
from yt.mods import *
pf = load(’galaxy0030/galaxy0030’)
p = SlicePlot(pf, 2, ’Density’, ’c’, (200,’kpc’))
p.save(’Galaxy’)
```

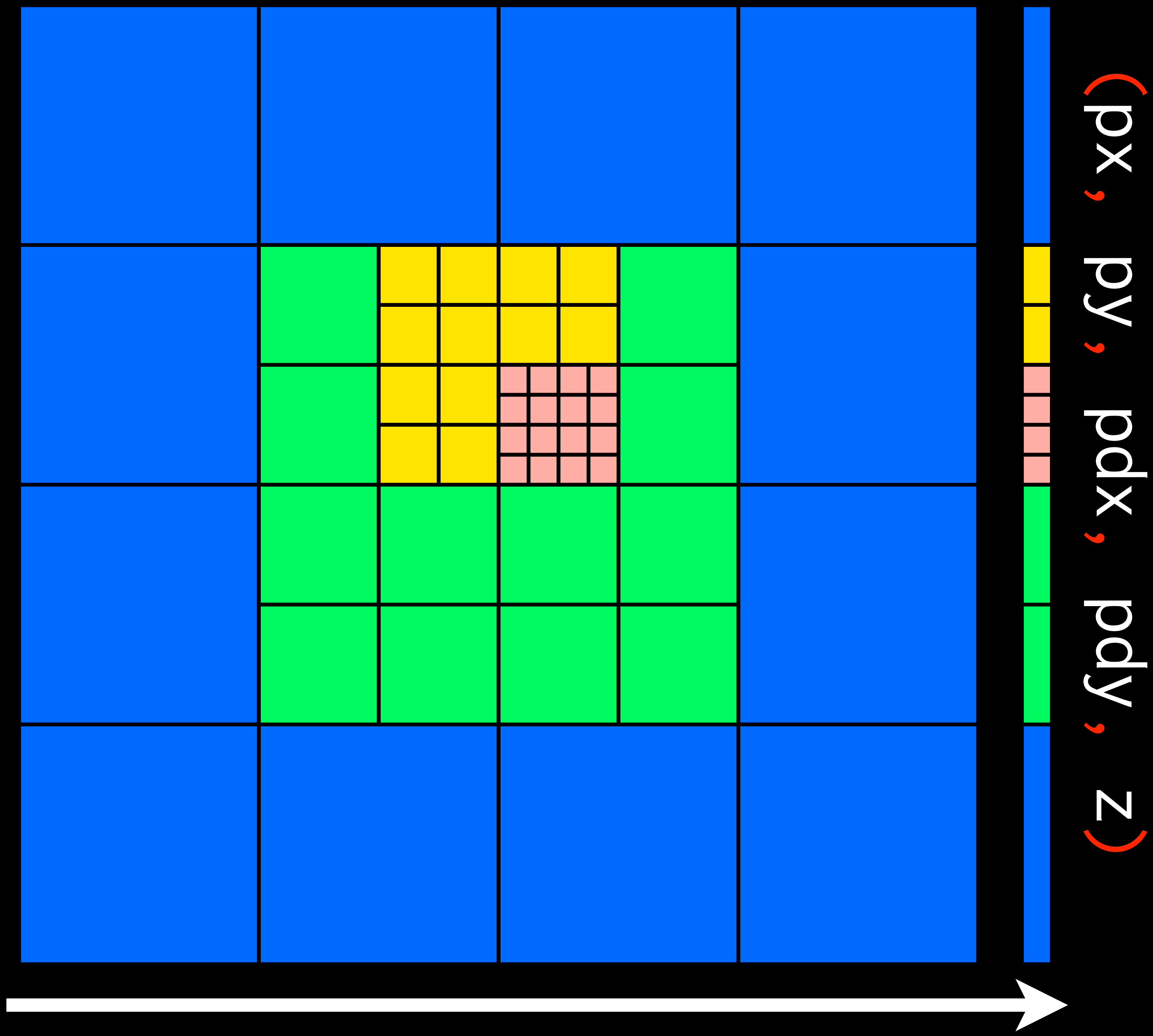

y (kpc)
50
0
−50
x (kpc)
−50
0
50
Density (g/cm$^3$)
$10^{-24}$
$10^{-25}$
$10^{-26}$
$10^{-27}$
$10^{-28}$
$10^{-29}$

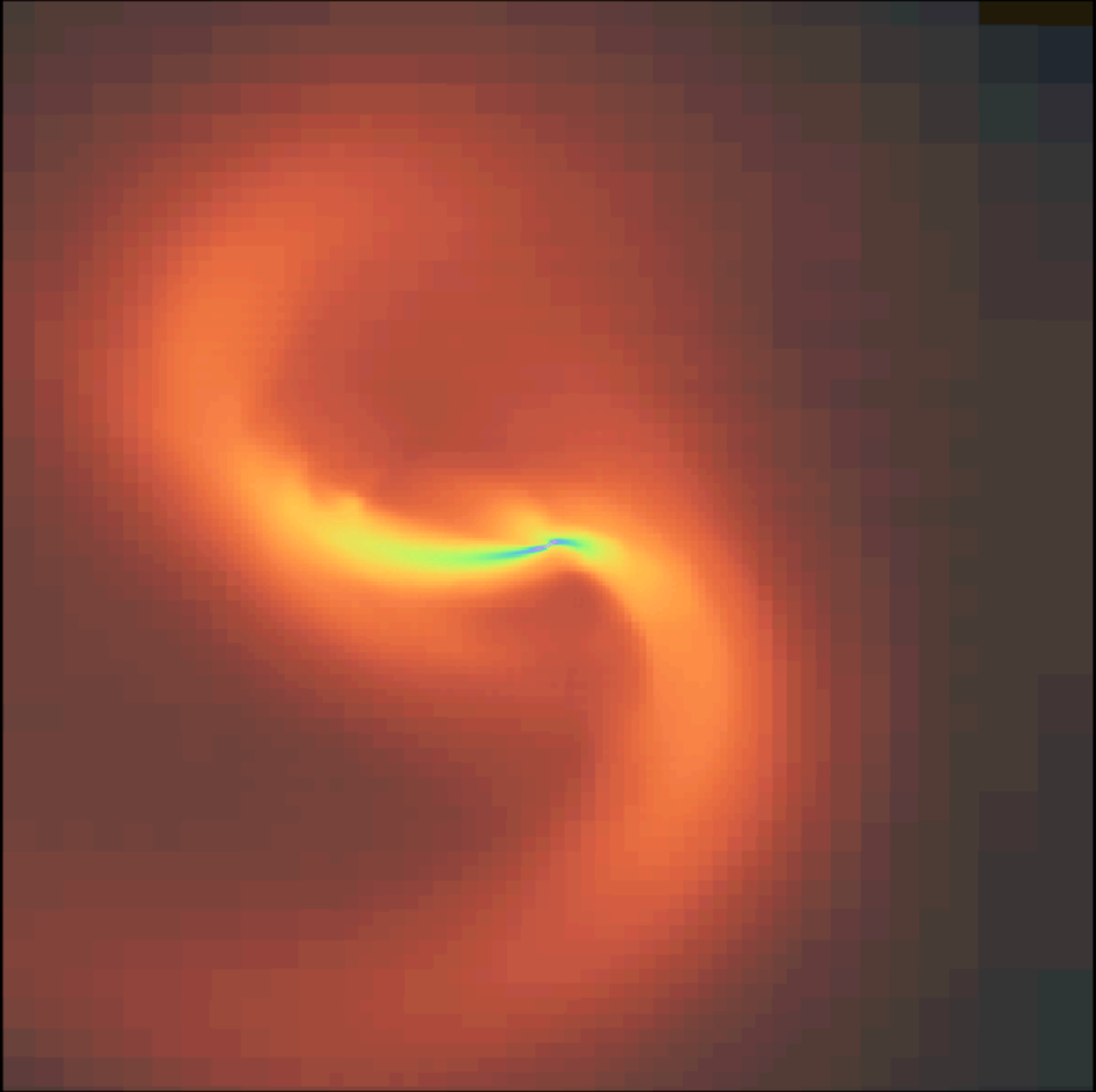
```
from yt.mods import *
pf = load(’galaxy0030/galaxy0030’)
p = SlicePlot(pf, 2, ’Density’, ’c’, (200,’kpc’))
p.set_width(20, ’kpc’)
p.save("GalaxyZoom")
```



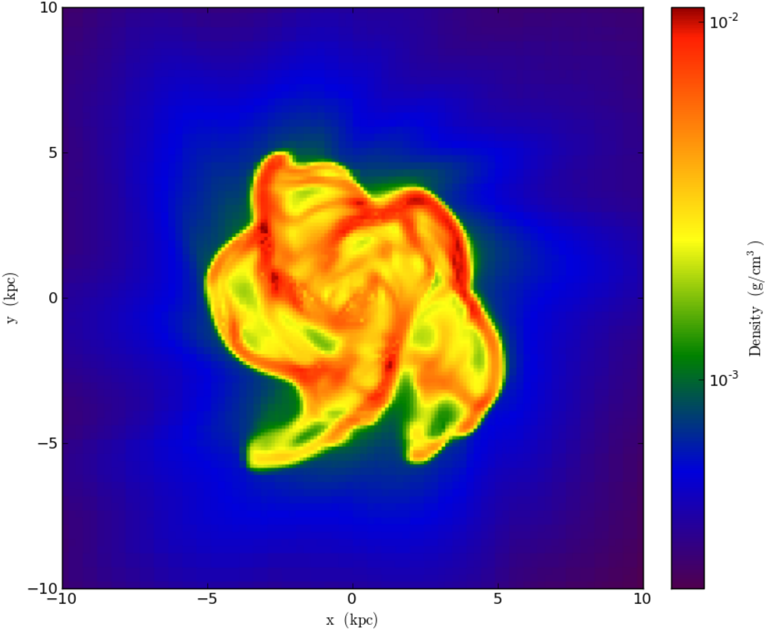
10
5
0
−5
−10
−10
−5
0
5
10
y (kpc)
x (kpc)
$10^{-24}$
$10^{-25}$
$10^{-26}$
$10^{-27}$
Density (g/cm$^3$)

# Projections



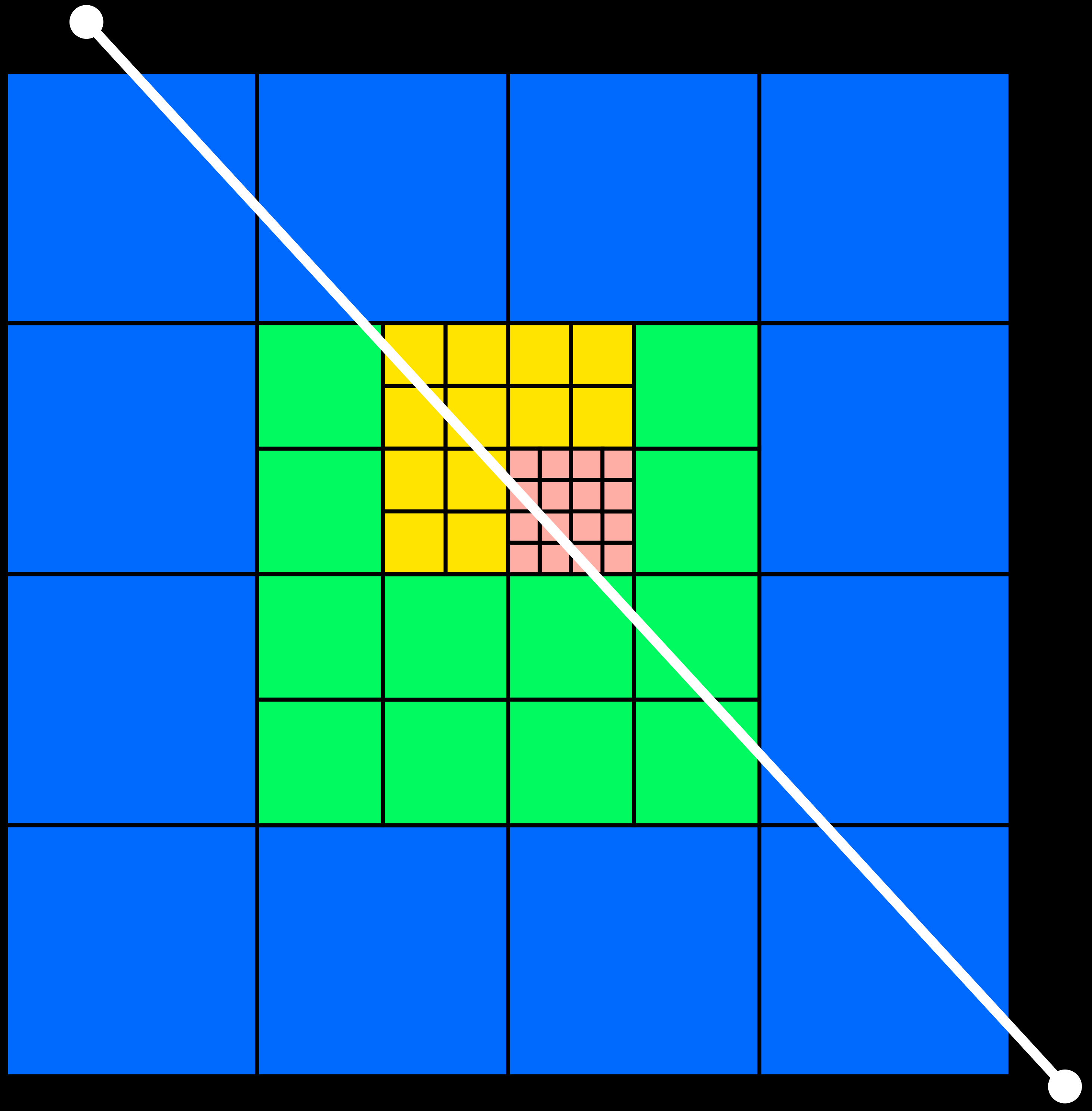


```
from yt.mods import *
pf = load(’galaxy0030/galaxy0030’)
p = ProjectionPlot(pf, 2, ’Density’, ’c’, (20,’kpc’))
p.save(’Galaxy’)
```

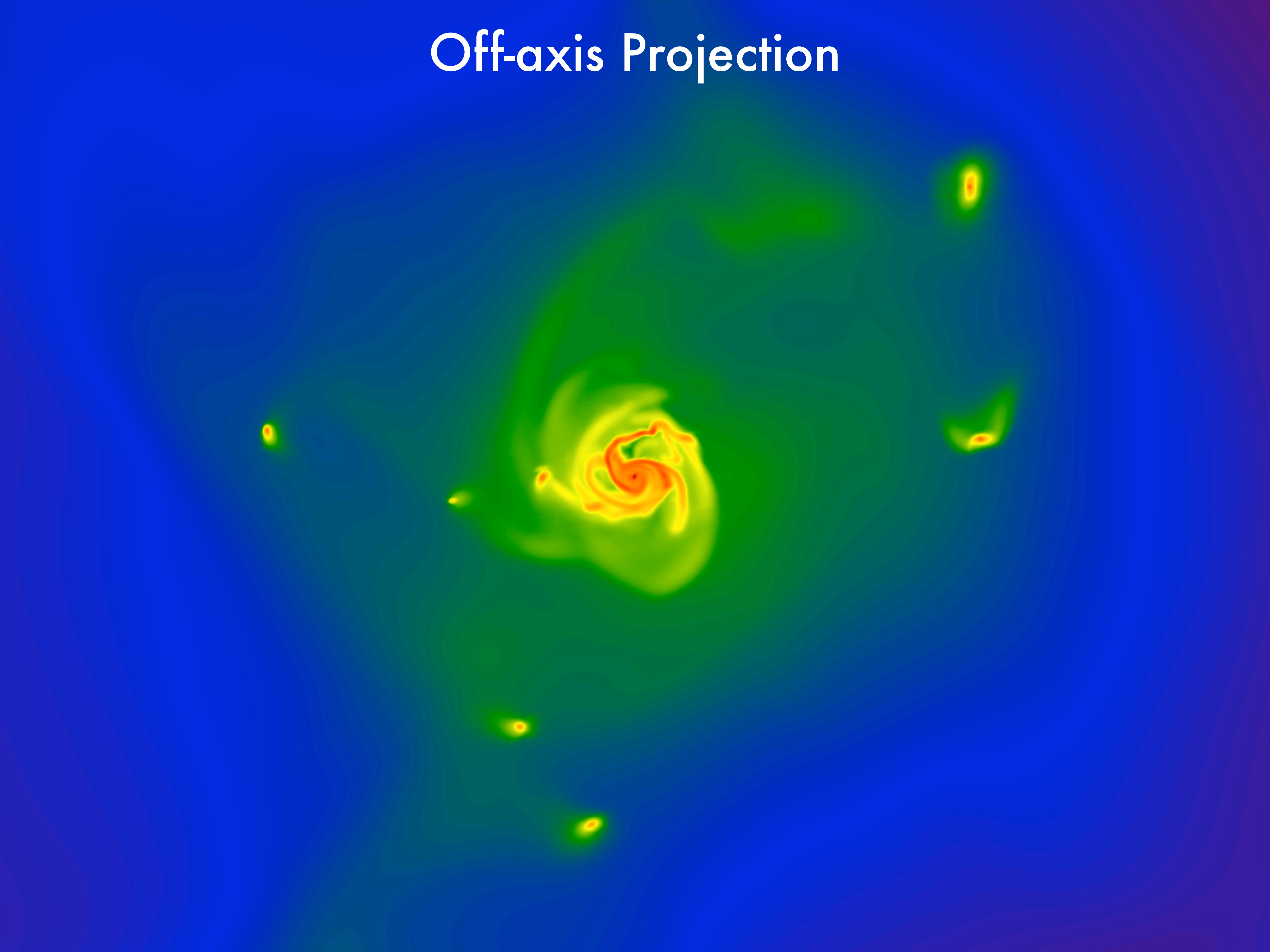


10
5
0
−5
−10
y (kpc)
−10
−5
0
5
10
x (kpc)
$10^{-2}$
$10^{-3}$
Density (g/cm$^3$)

# Oblique Slices



# Off-axis Projection



# Project once,
pixelize many

`(px, py, pdx, pdy, z)`

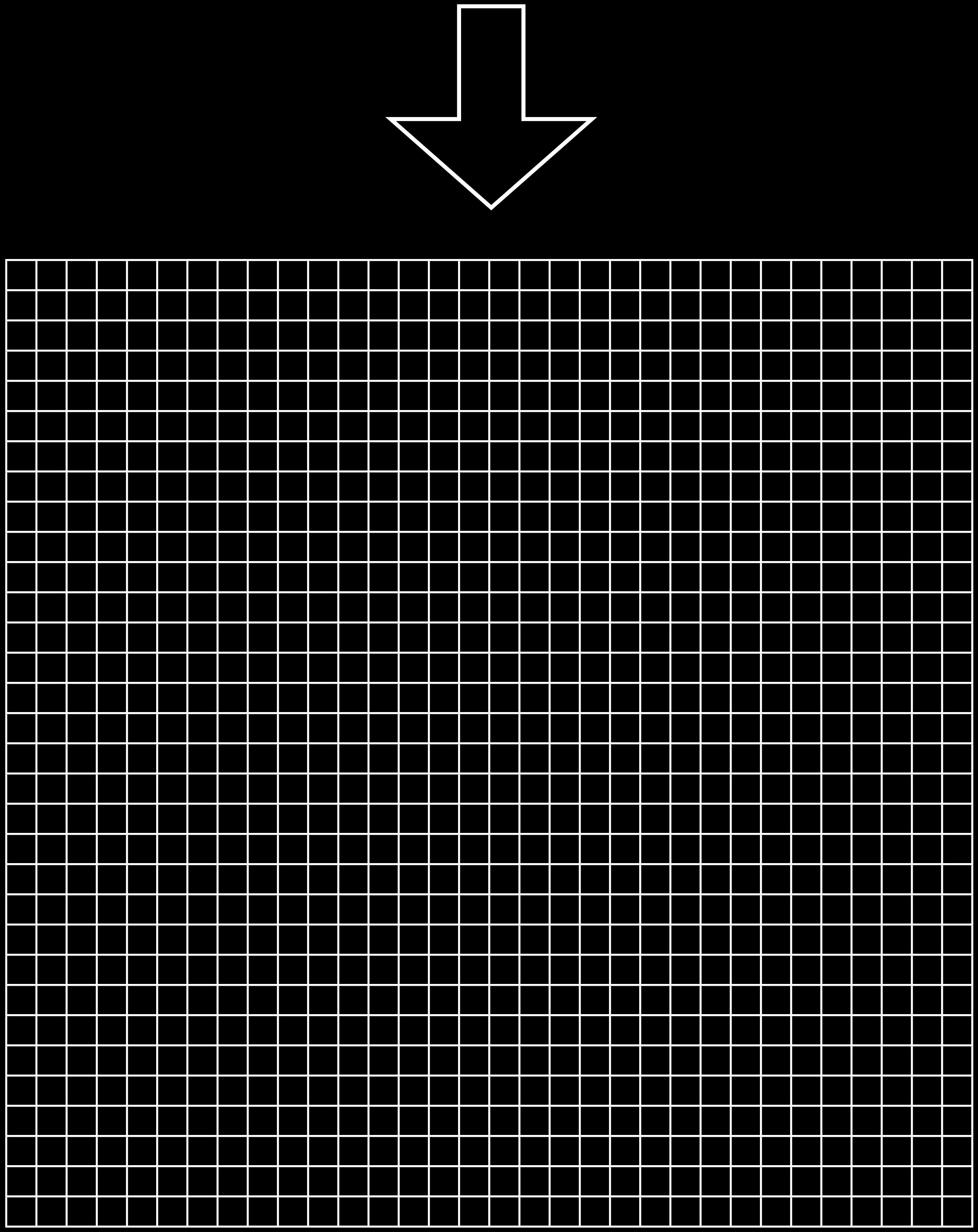


Image Buffer

(px, py, pdx, pdy, z)

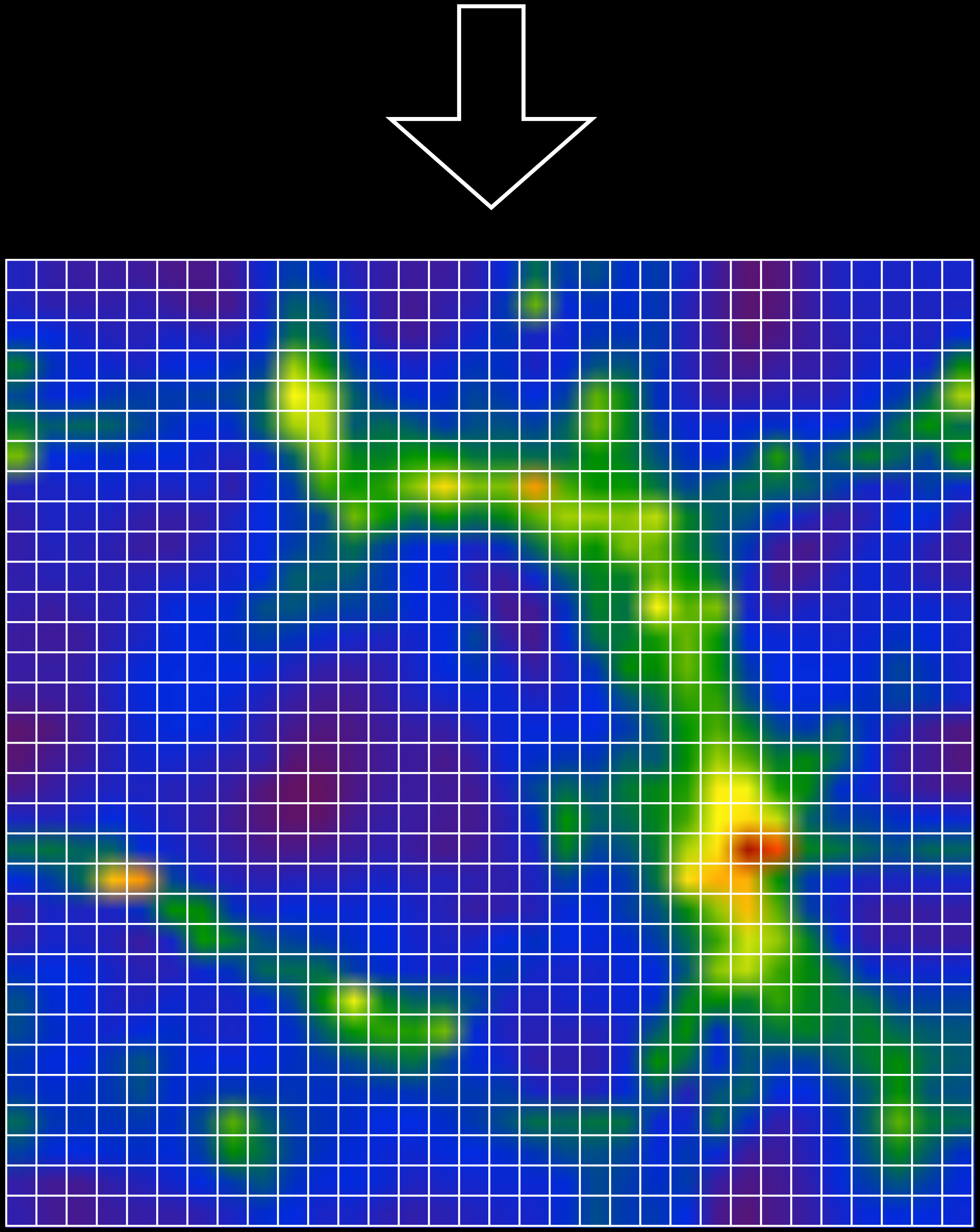


Image Buffer

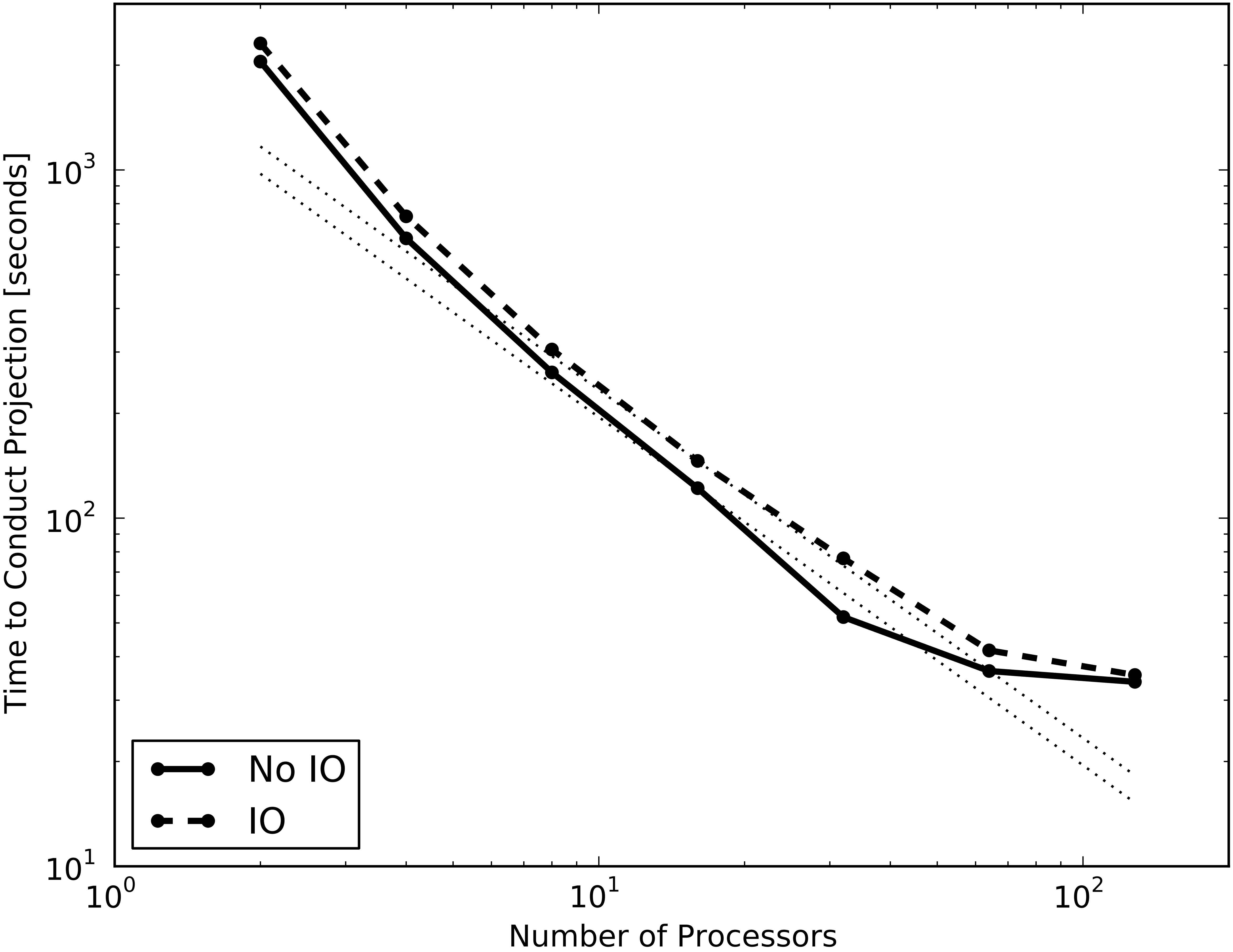
# Parallelism

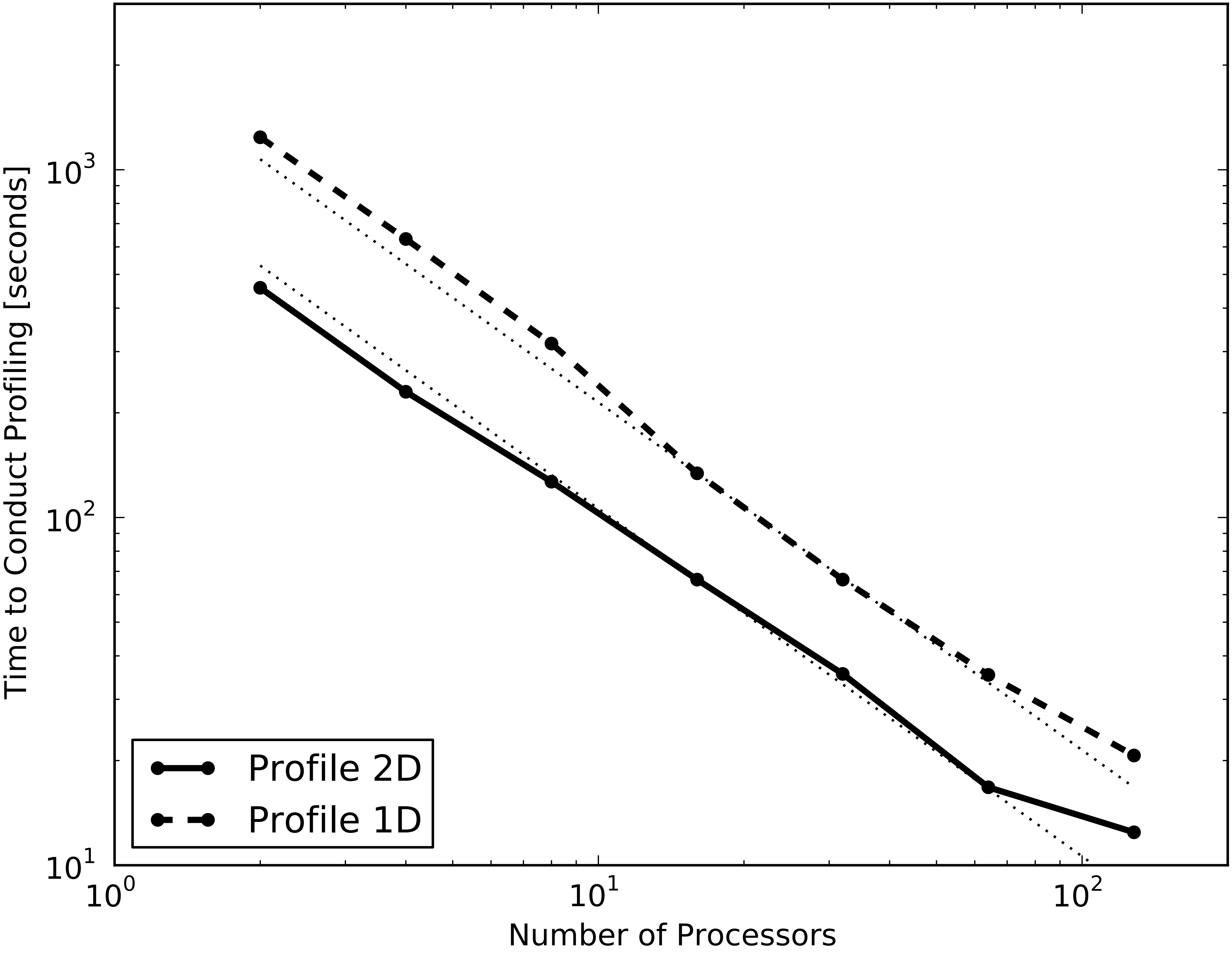
# Parallelism

| Embarassingly Parallel | Spatial Decomposition |
|---|---|
| Decomposed by load or IO characteristics | Helper functions to decompose the domain |

# Parallelism

| Embarassingly Parallel | Spatial Decomposition |
|---|---|
| Quantities | |
| Profiles | |
| Slices | Halo Finding |
| Projections | |
| Volume Rendering | |

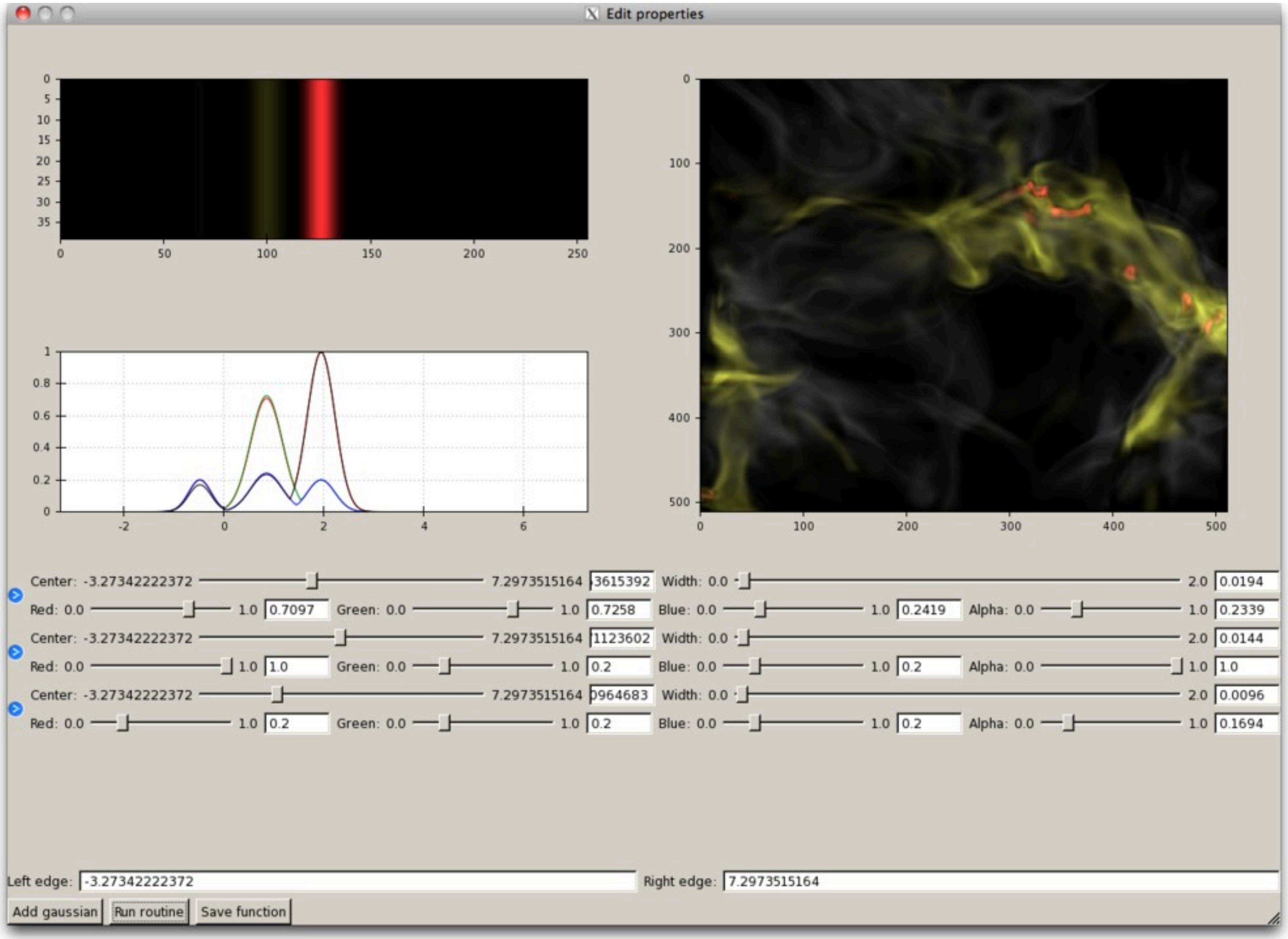

Time to Conduct Projection [seconds]
Number of Processors
$10^1$
$10^2$
$10^3$
$10^0$
$10^1$
$10^2$
No IO
IO


Time to Conduct Profiling [seconds]
$10^3$
$10^2$
$10^1$
$10^0$
$10^1$
$10^2$
Number of Processors
Profile 2D
Profile 1D

# Multi-level parallelism: dynamic workgroups, communicators, subgroups and task queues

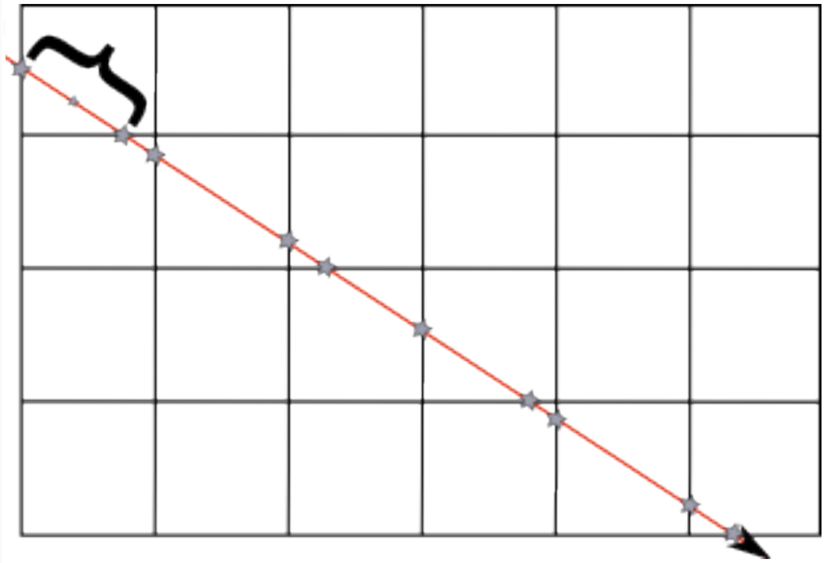
# Volume Rendering

# Designed around integrating through a volume: visualization is a side effect.

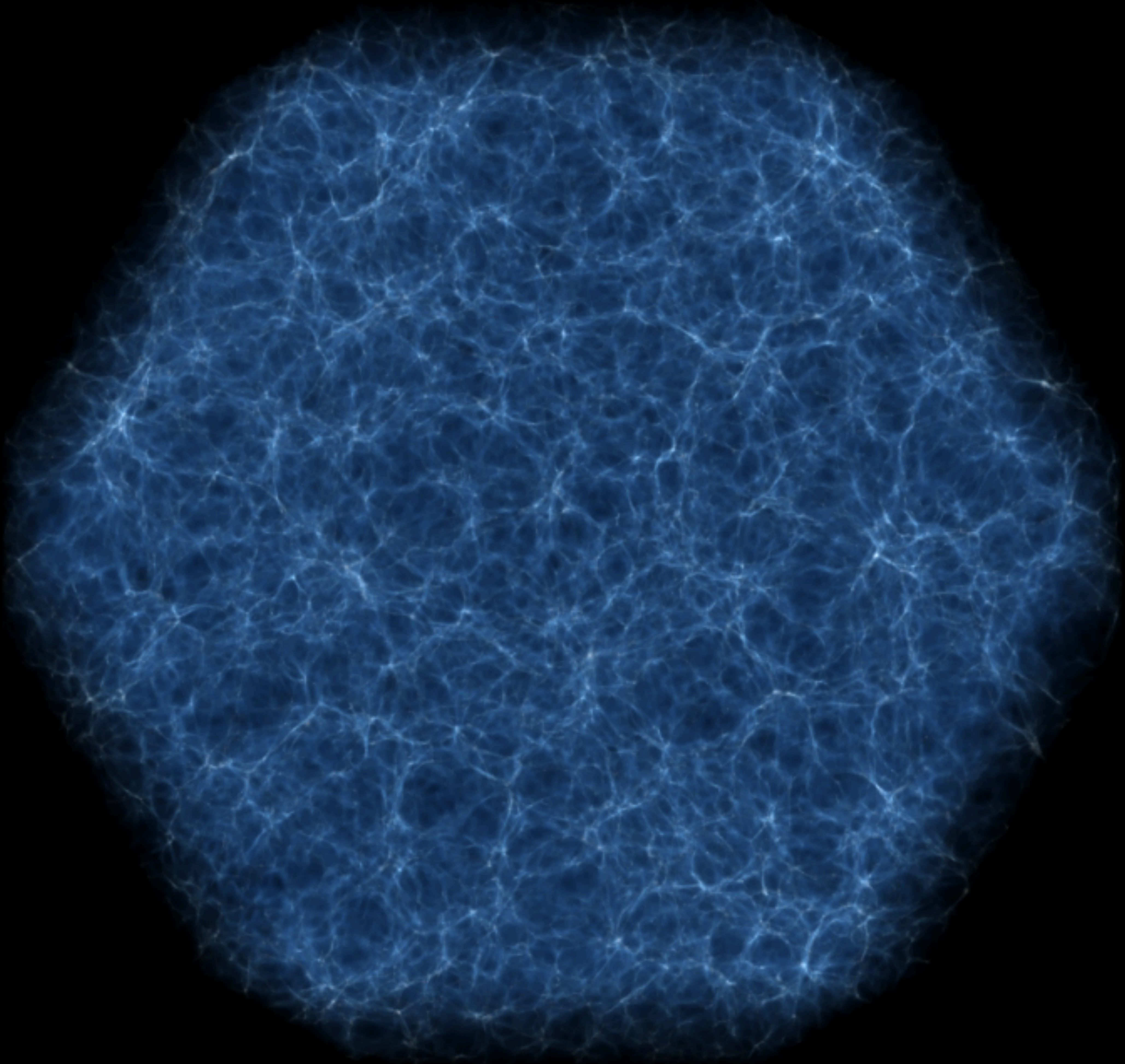


Center: -3.27342222372 7.2973515164 3615392 Width: 0.0 2.0 0.0194
Red: 0.0 1.0 0.7097 Green: 0.0 1.0 0.7258 Blue: 0.0 1.0 0.2419 Alpha: 0.0 1.0 0.2339
Center: -3.27342222372 7.2973515164 1123602 Width: 0.0 2.0 0.0144
Red: 0.0 1.0 1.0 Green: 0.0 1.0 0.2 Blue: 0.0 1.0 0.2 Alpha: 0.0 1.0 1.0
Center: -3.27342222372 7.2973515164 0964683 Width: 0.0 2.0 0.0096
Red: 0.0 1.0 0.2 Green: 0.0 1.0 0.2 Blue: 0.0 1.0 0.2 Alpha: 0.0 1.0 0.1694

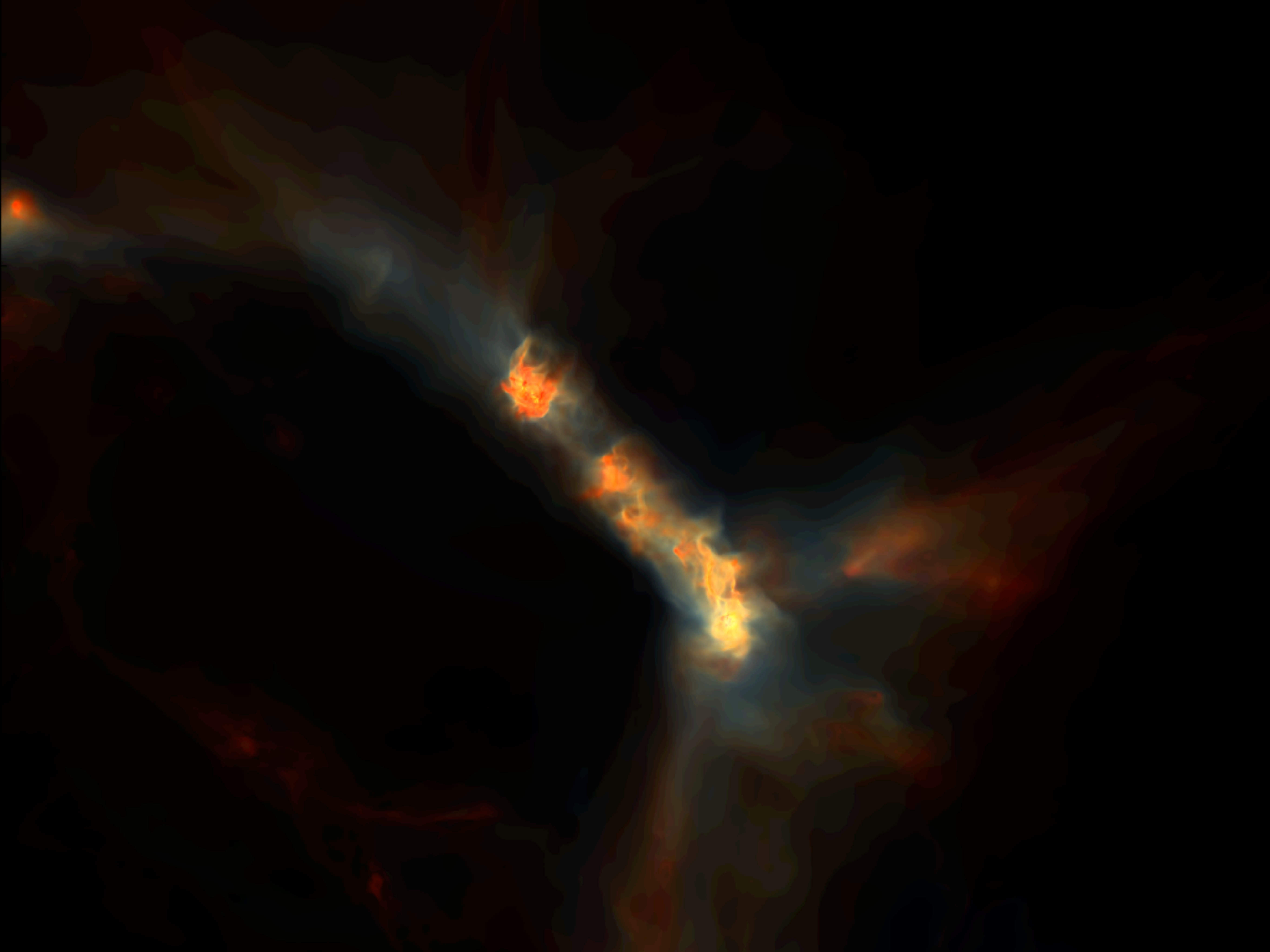
Left edge: -3.27342222372
Right edge: 7.2973515164

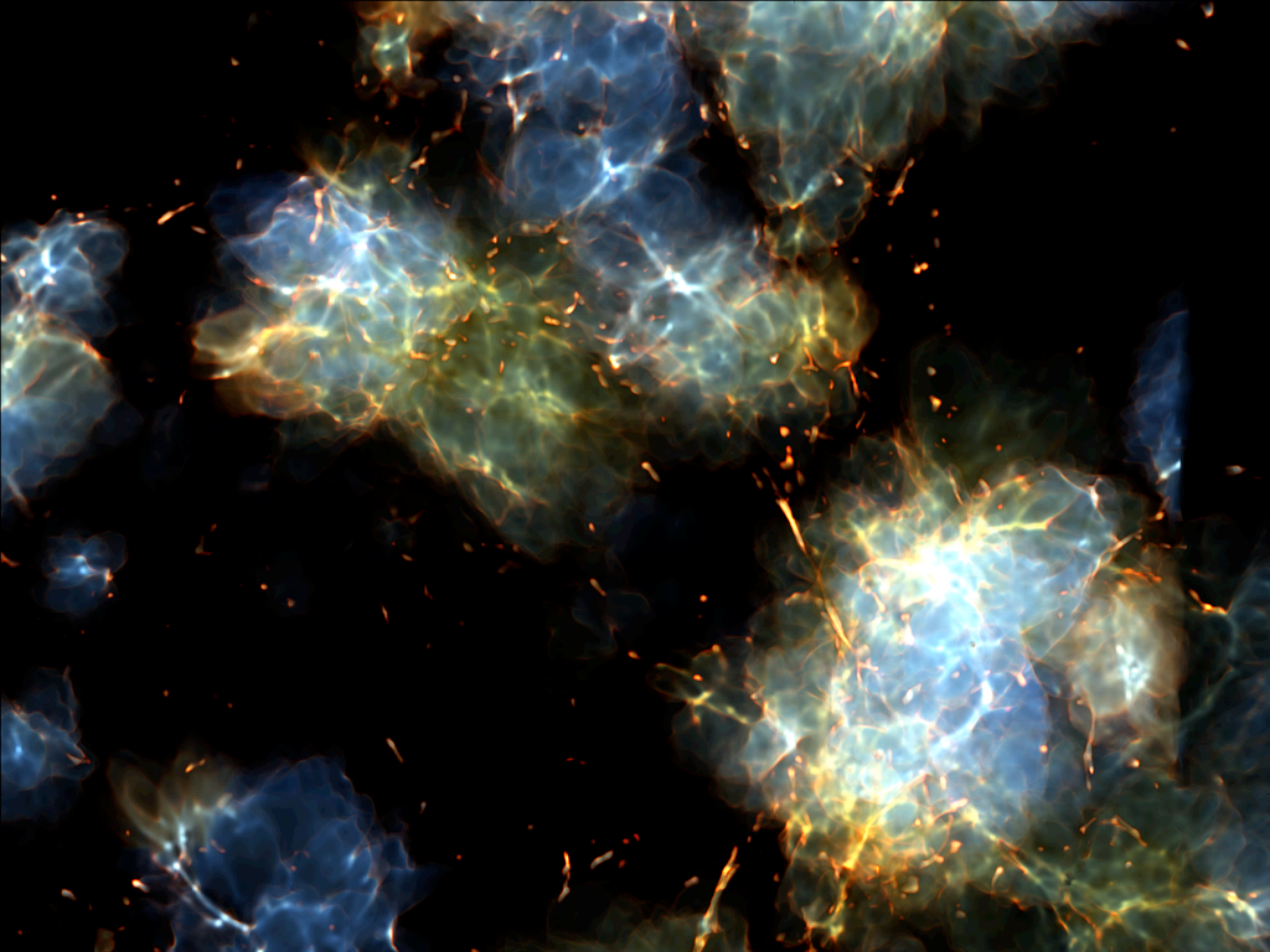


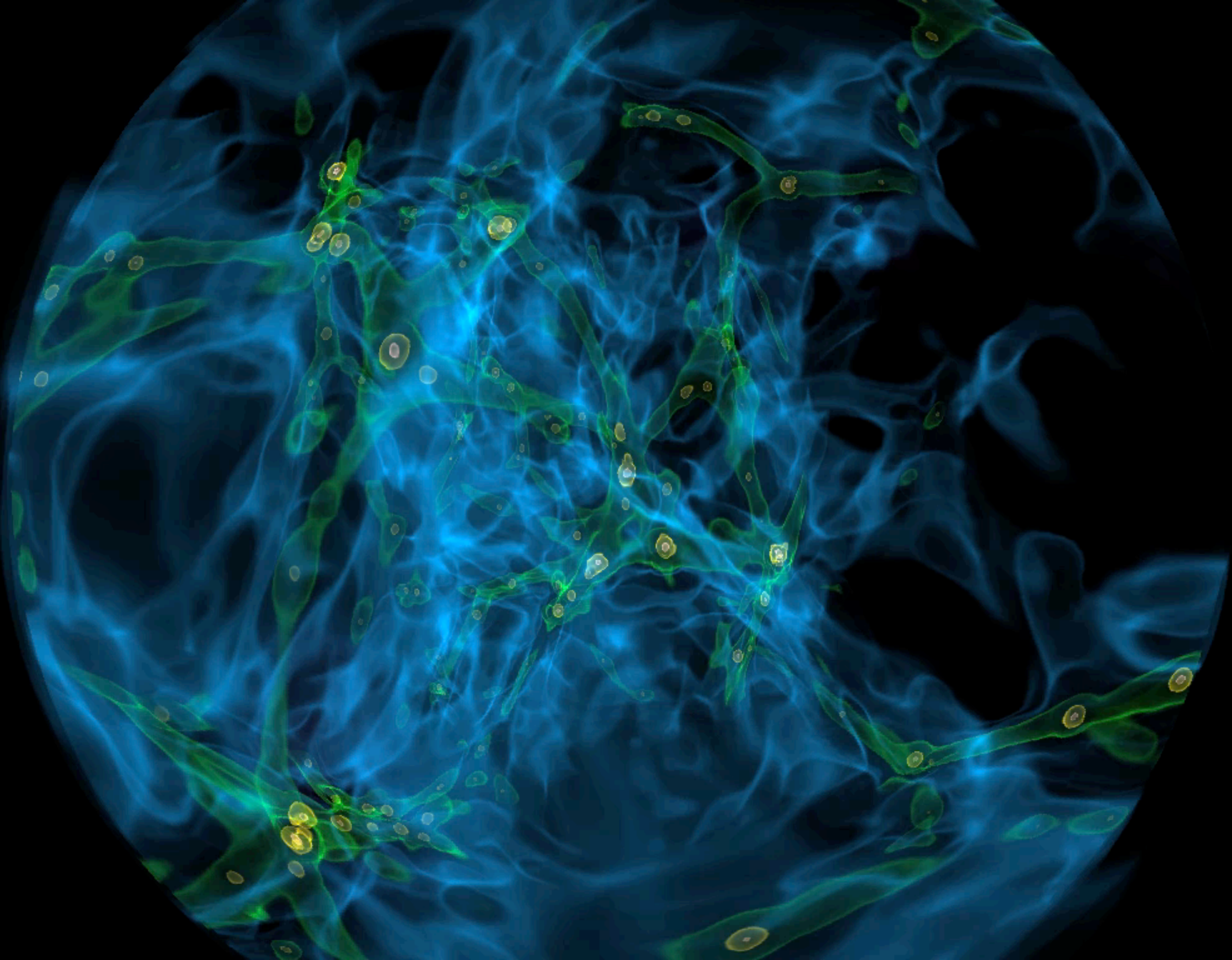
$$\frac{dI_\nu}{ds} = j_\nu - \alpha_\nu I_\nu$$



```
from yt.mods import *
pf = load(“DD1701/DD1701”)
v, c = pf.h.find_max(”Density”)
L = [1.0, 1.0, 1.0]
W = 100.0/pf[’mpc’]

tf = vr.PlanckTransferFunction()
cam = pf.h.camera(c, L, W, 1024, tf)
cam.snapshot()
```



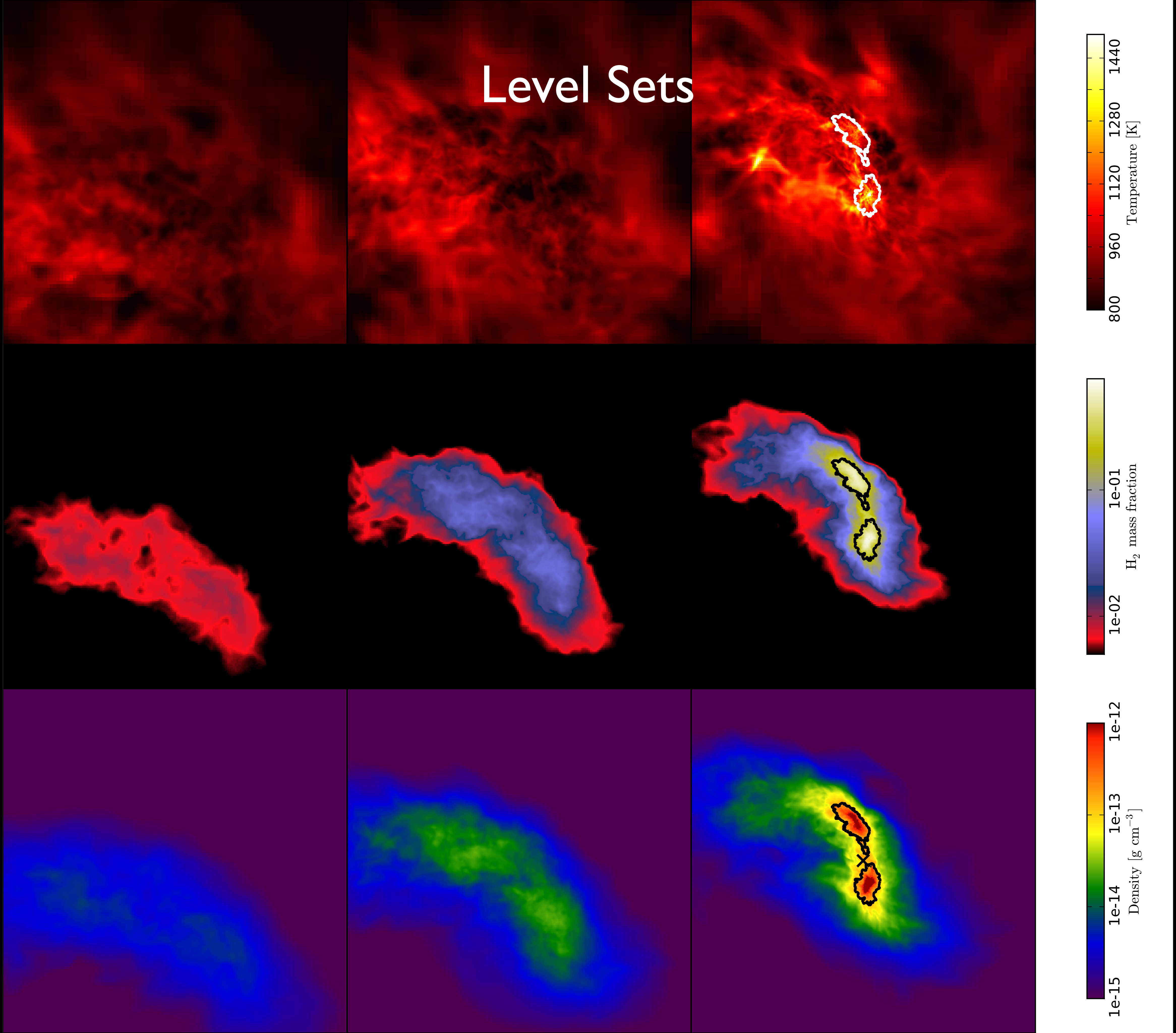




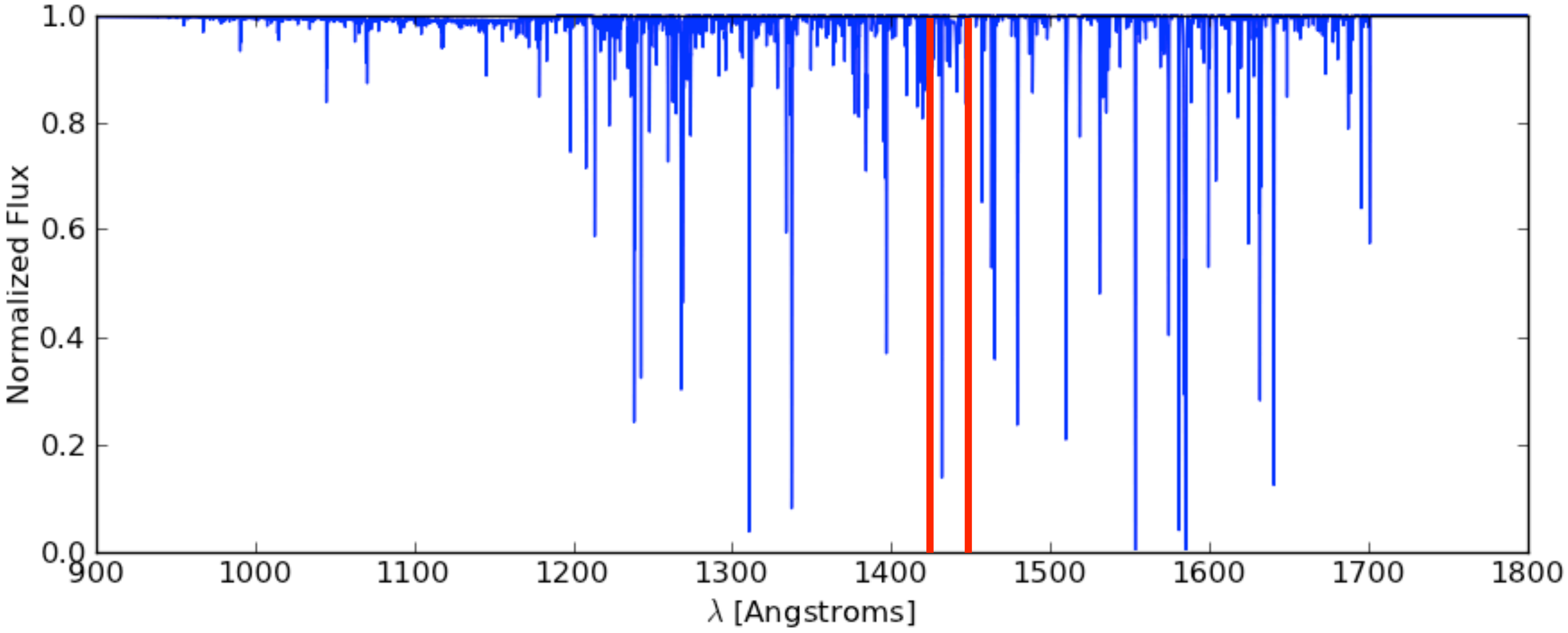
# Canned Analysis Tasks

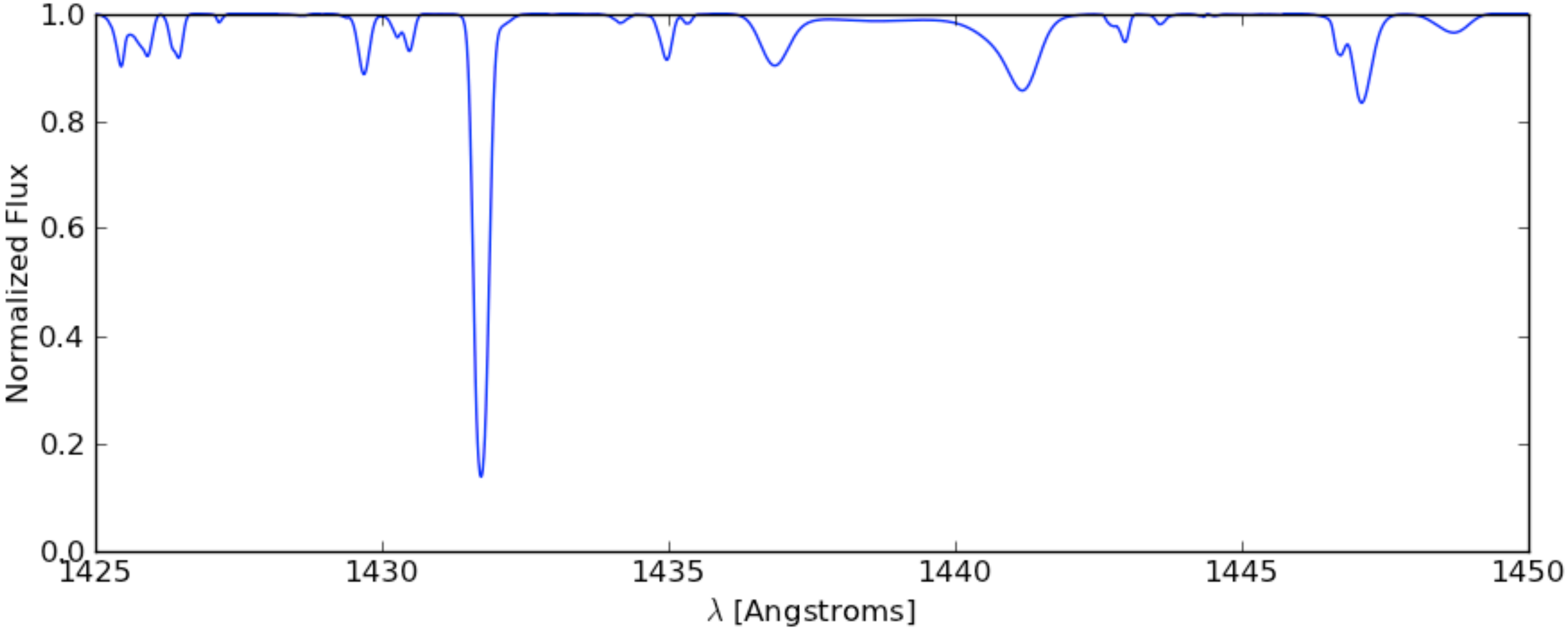
Absorption Spectrum
Coordinate Transformations
Halo Finding
Mass Functions
Merger Trees
Halo Profiling
Level Sets
Light Cones
Light Rays
Time Series
Star Analysis
Two-Point Analysis

# Level Sets

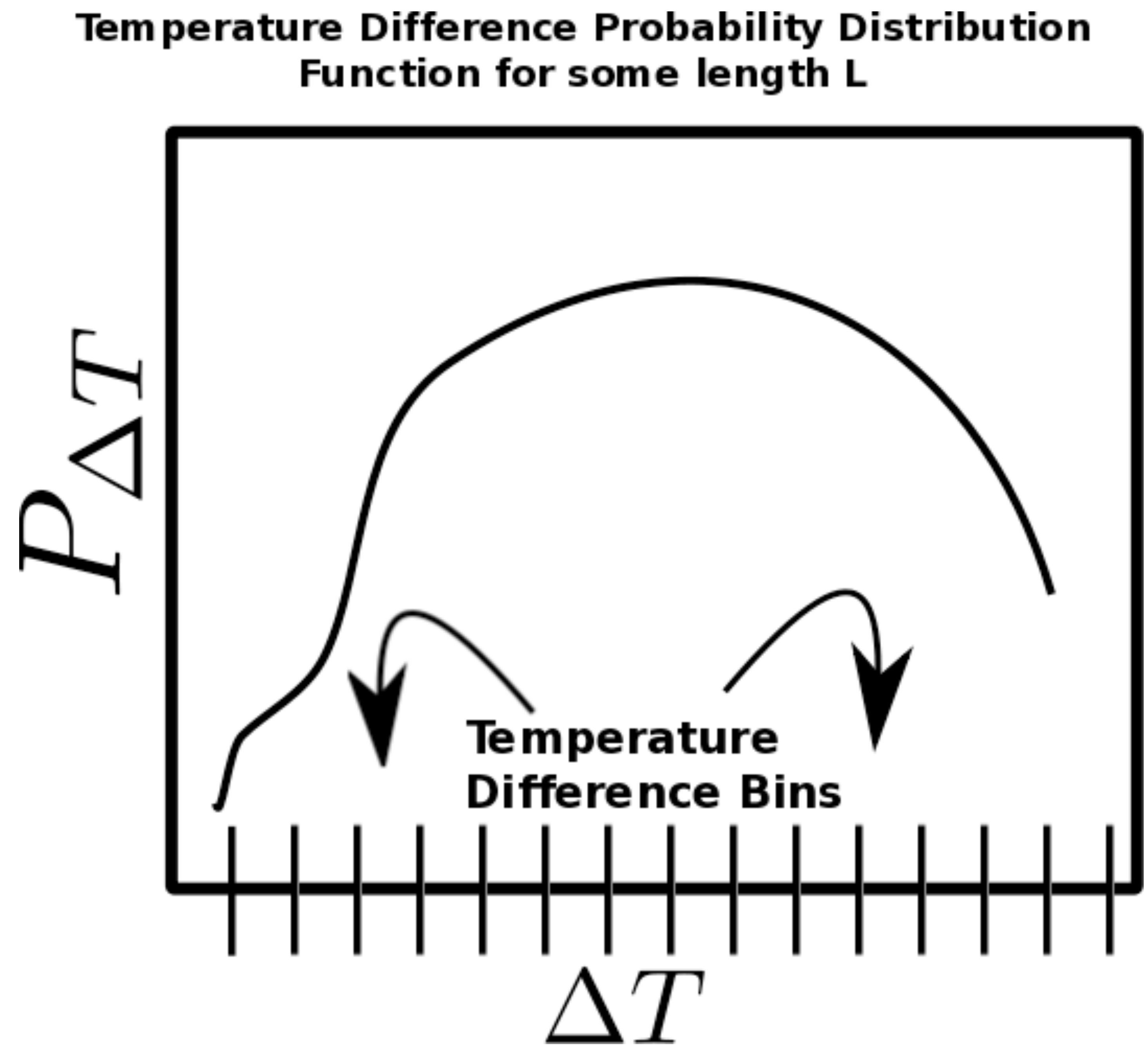

800
960
1120
1280
1440
Temperature [K]
1e-02
1e-01
$H_2$ mass fraction
1e-15
1e-14
1e-13
1e-12
Density [g cm$^{-3}$]

# Synthetic Spectra





# Two-Point Functions


Temperature Difference Probability Distribution
Function for some length L
$P_{\Delta T}$
Temperature
Difference Bins
$\Delta T$

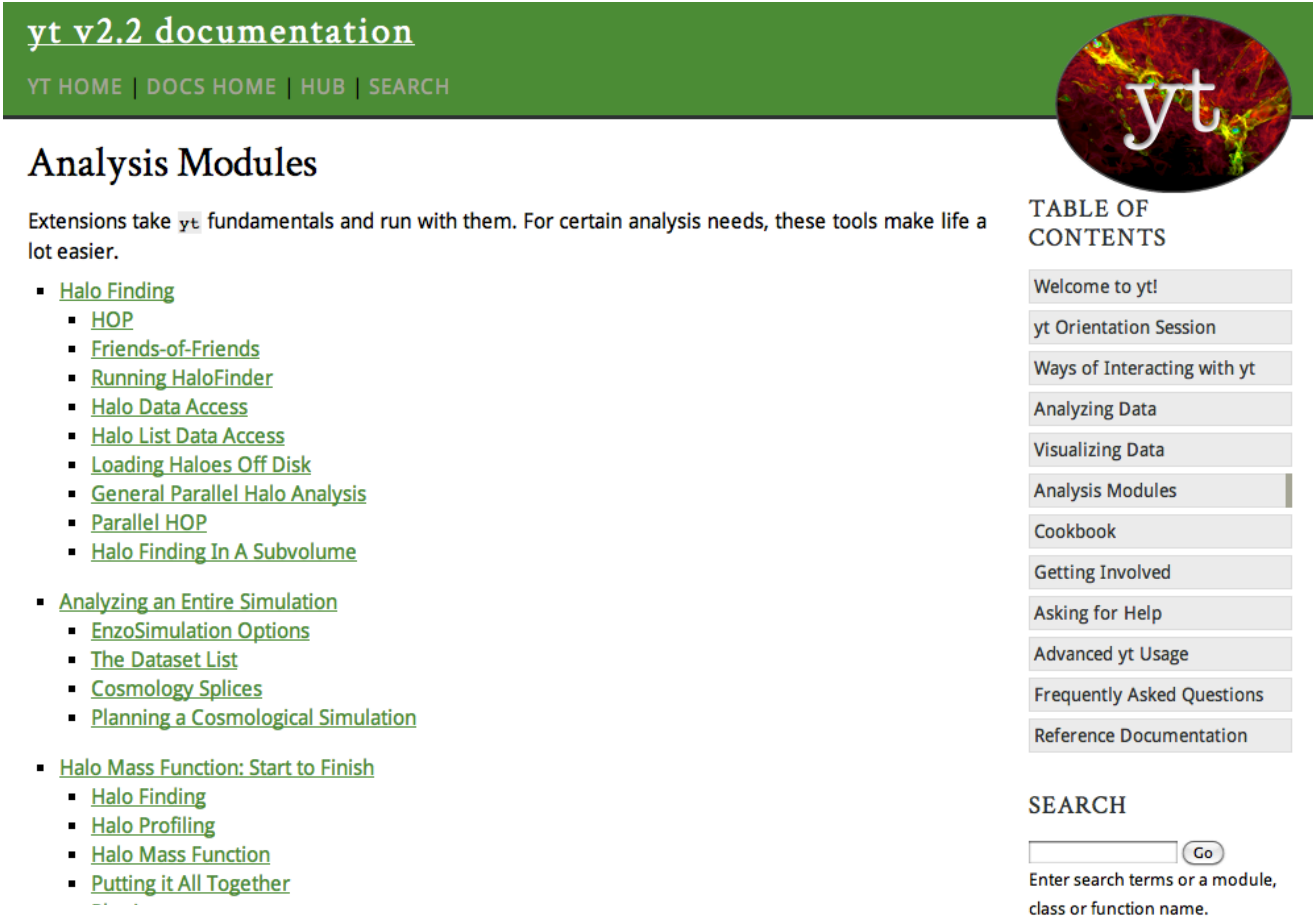
# Three Halo Finders:

Standard HOP

Friends of Friends

Parallel HOP

Rockstar (beta)



yt v2.2 documentation

YT HOME | DOCS HOME | HUB | SEARCH

# Analysis Modules

Extensions take `yt` fundamentals and run with them. For certain analysis needs, these tools make life a lot easier.

- Halo Finding
  - HOP
  - Friends-of-Friends
  - Running HaloFinder
  - Halo Data Access
  - Halo List Data Access
  - Loading Haloes Off Disk
  - General Parallel Halo Analysis
  - Parallel HOP
  - Halo Finding In A Subvolume
- Analyzing an Entire Simulation
  - EnzoSimulation Options
  - The Dataset List
  - Cosmology Splices
  - Planning a Cosmological Simulation
- Halo Mass Function: Start to Finish
  - Halo Finding
  - Halo Profiling
  - Halo Mass Function
  - Putting it All Together

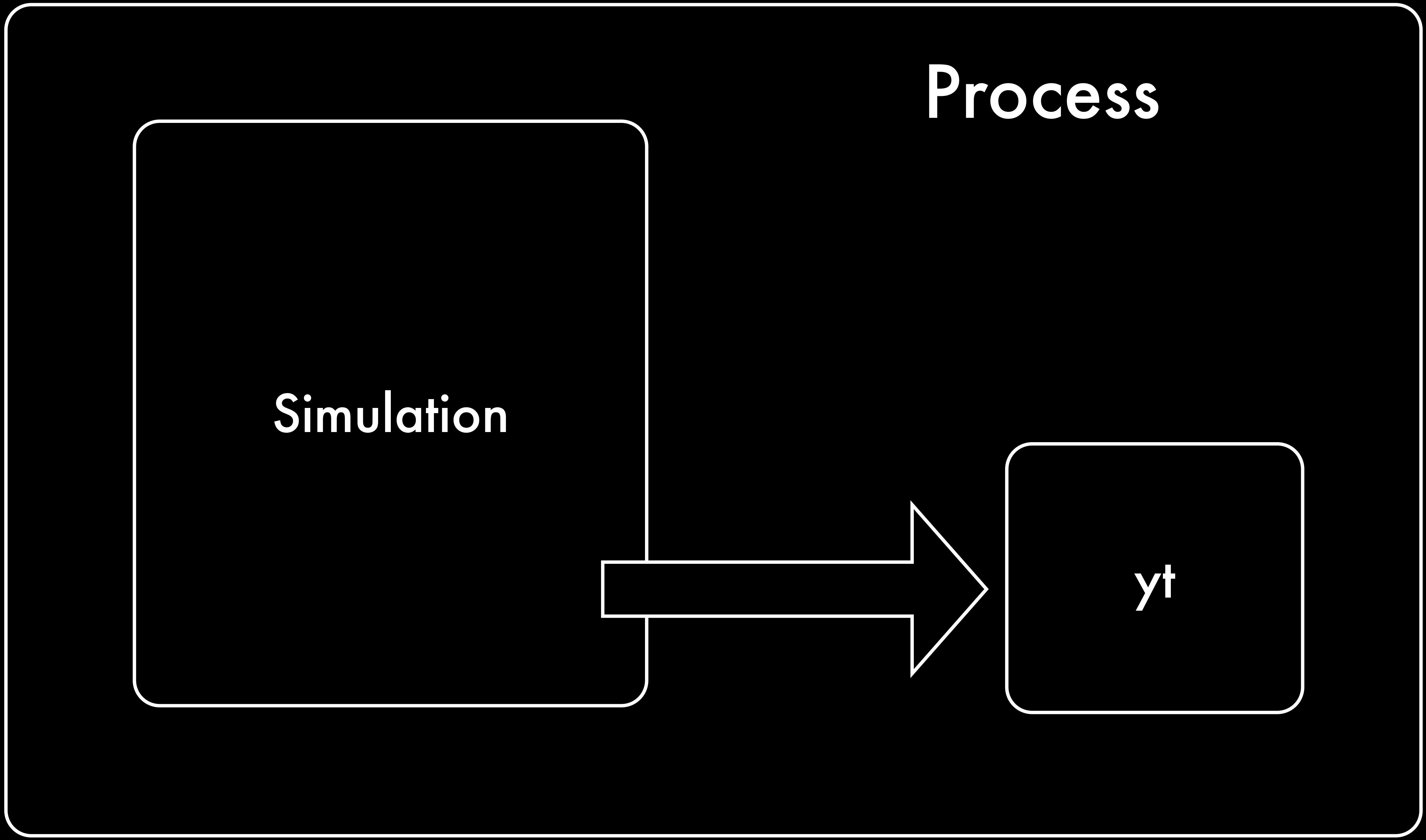
TABLE OF CONTENTS

SEARCH

Go

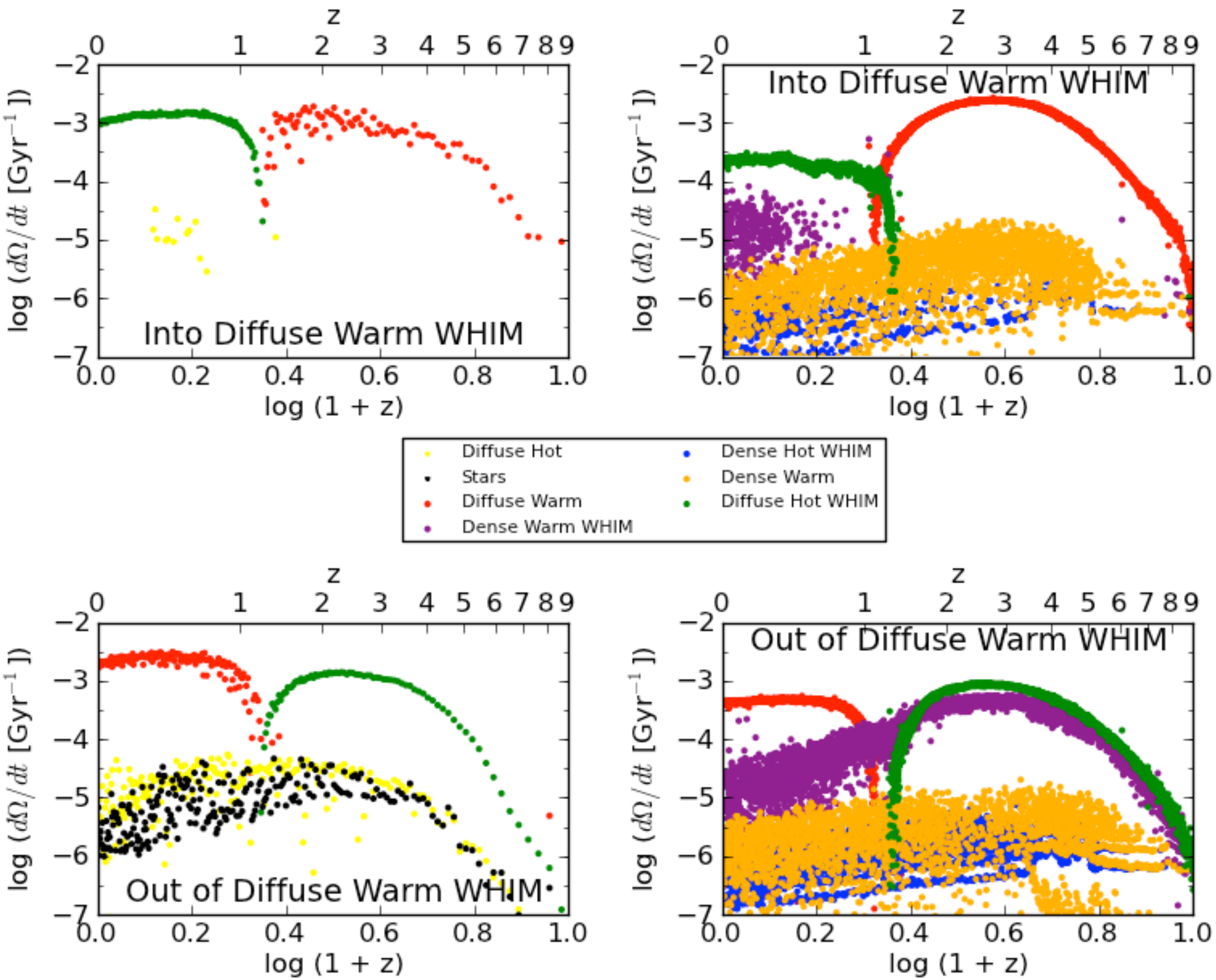
Enter search terms or a module, class or function name.

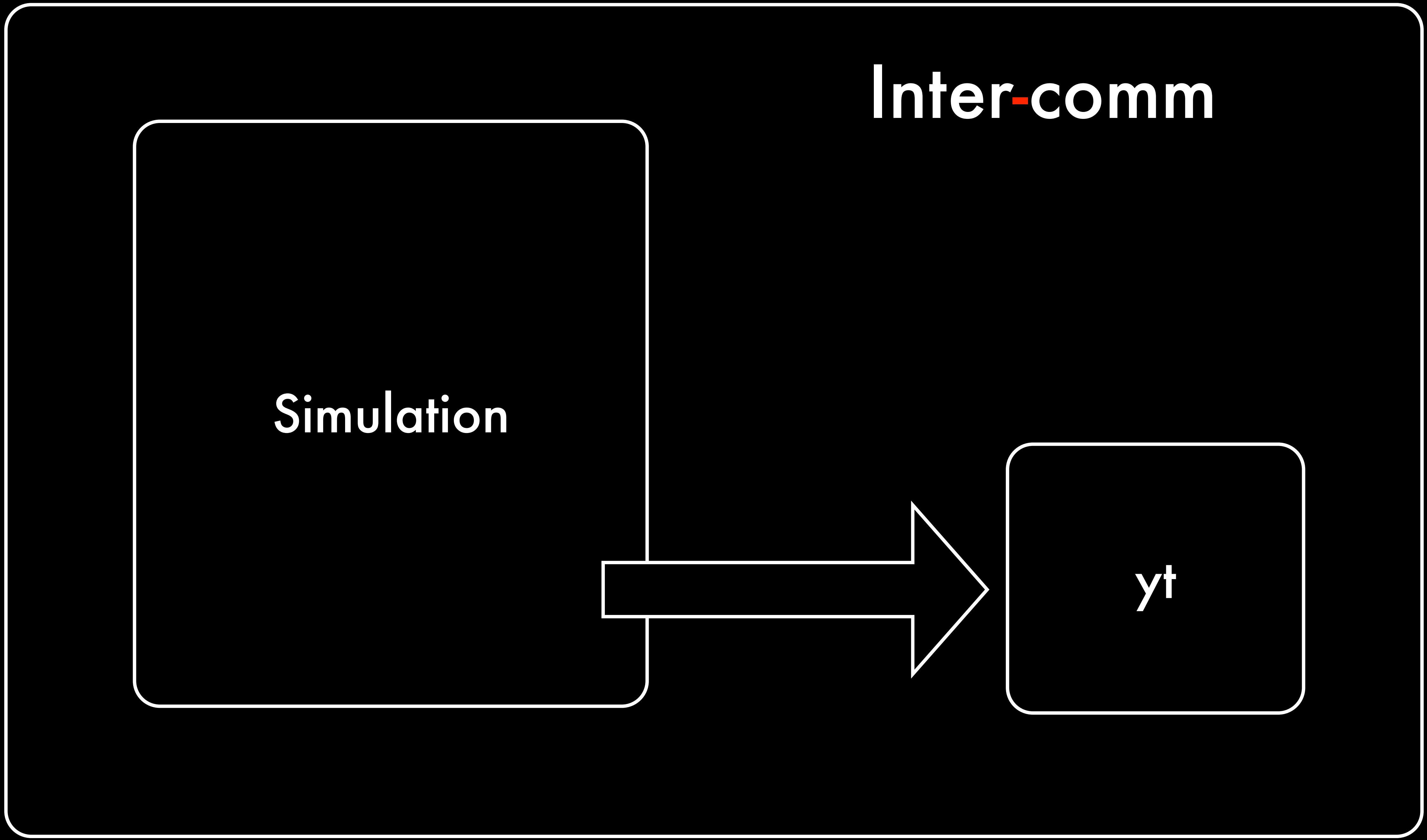
# co-scheduled & in situ viz



Thin NumPy wrappers and stop-n-go

# On Disc → In-situ





Fire and forget, no embedded interpreter

# Developing as a Team

## Code review:

eyes on (nearly) every changeset

Forky development:

very low barrier to entry;
everything comes in the box.

# Testing:

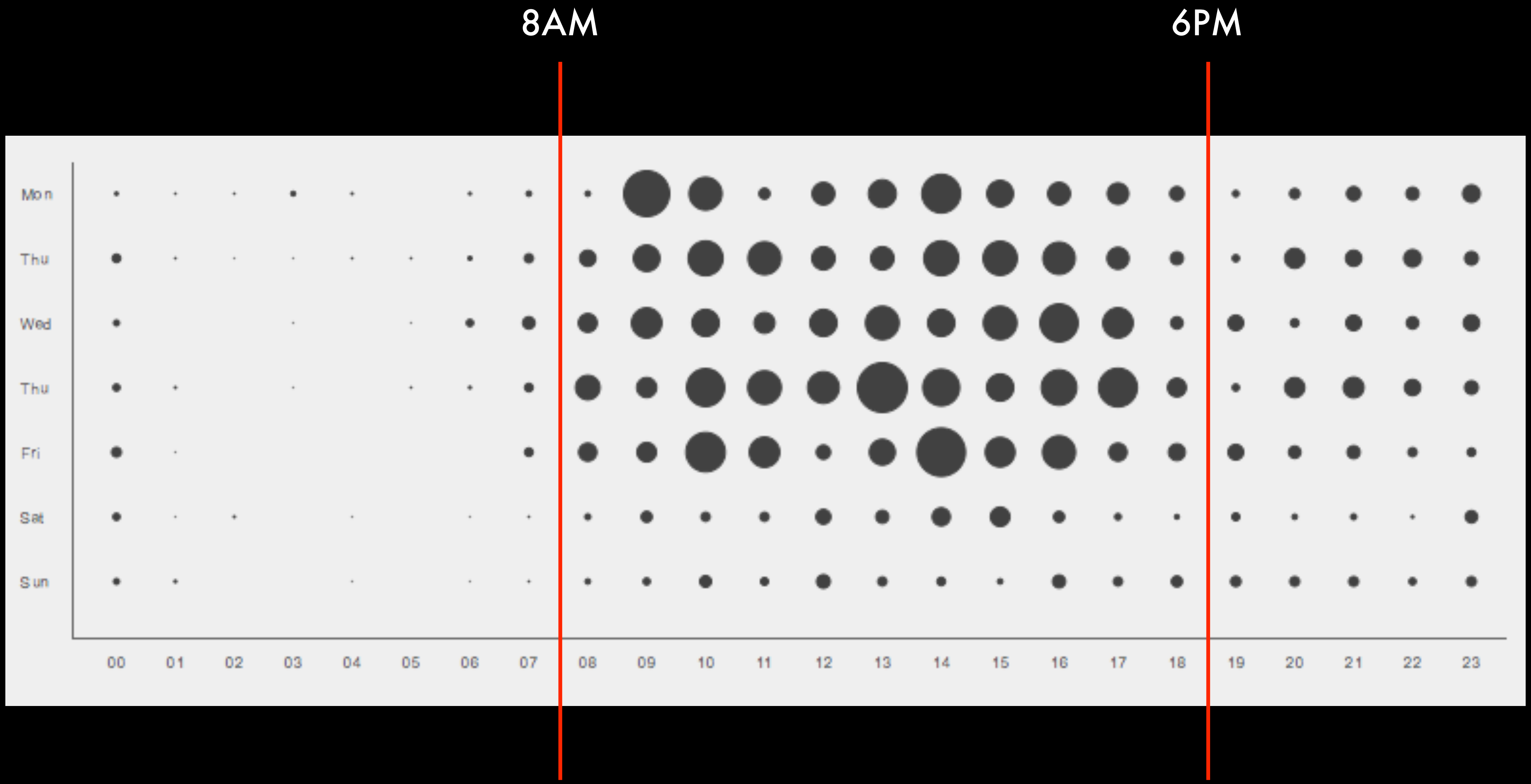
answer as well as integration tests get run every 30 minutes.

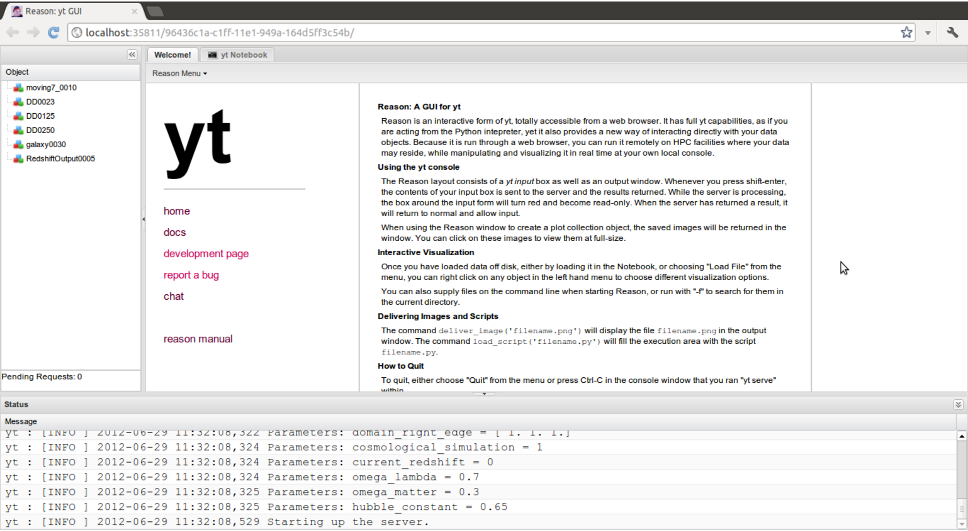
80,000 lines of code

Python, Cython, C

20 contributors (60+ users)

Contributors from 10+ institutions





**Reason: A GUI for yt**

Reason is an interactive form of yt, totally accessible from a web browser. It has full yt capabilities, as if you are acting from the Python intepreter, yet it also provides a new way of interacting directly with your data objects. Because it is run through a web browser, you can run it remotely on HPC facilities where your data may reside, while manipulating and visualizing it in real time at your own local console.

**Using the yt console**

The Reason layout consists of a *yt input* box as well as an output window. Whenever you press shift-enter, the contents of your input box is sent to the server and the results returned. While the server is processing, the box around the input form will turn red and become read-only. When the server has returned a result, it will return to normal and allow input.

When using the Reason window to create a plot collection object, the saved images will be returned in the window. You can click on these images to view them at full-size.

**Interactive Visualization**

Once you have loaded data off disk, either by loading it in the Notebook, or choosing "Load File" from the menu, you can right click on any object in the left hand menu to choose different visualization options.

You can also supply files on the command line when starting Reason, or run with "-f" to search for them in the current directory.

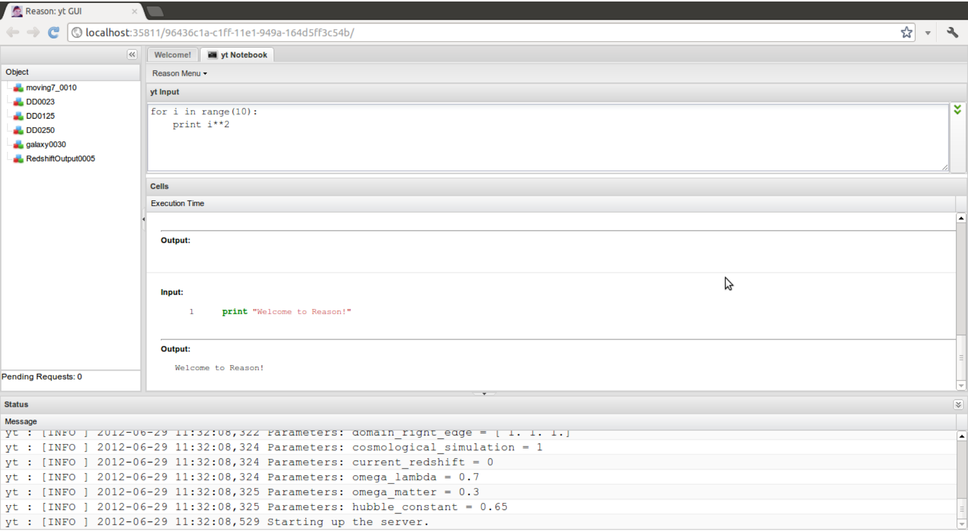
**Delivering Images and Scripts**

The command `deliver_image('filename.png')` will display the file `filename.png` in the output window. The command `load_script('filename.py')` will fill the execution area with the script `filename.py`.

**How to Quit**

To quit, either choose "Quit" from the menu or press Ctrl-C in the console window that you ran "yt serve" within

```
yt : [INFO ] 2012-06-29 11:32:08,322 Parameters: domain_right_edge = [ 1. 1. 1.]
yt : [INFO ] 2012-06-29 11:32:08,324 Parameters: cosmological_simulation = 1
yt : [INFO ] 2012-06-29 11:32:08,324 Parameters: current_redshift = 0
yt : [INFO ] 2012-06-29 11:32:08,324 Parameters: omega_lambda = 0.7
yt : [INFO ] 2012-06-29 11:32:08,325 Parameters: omega_matter = 0.3
yt : [INFO ] 2012-06-29 11:32:08,325 Parameters: hubble_constant = 0.65
yt : [INFO ] 2012-06-29 11:32:08,529 Starting up the server.
```

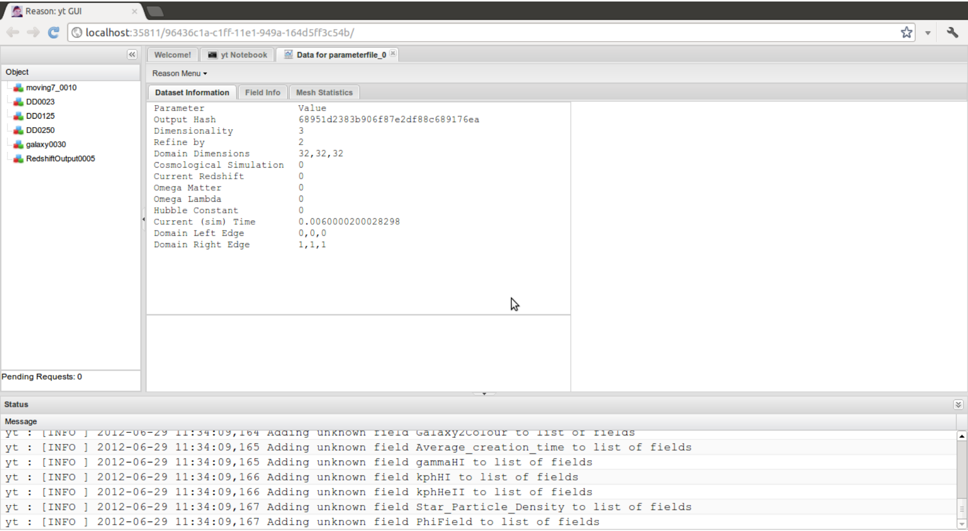


```
for i in range(10):
    print i**2
```

```
print "Welcome to Reason!"
```

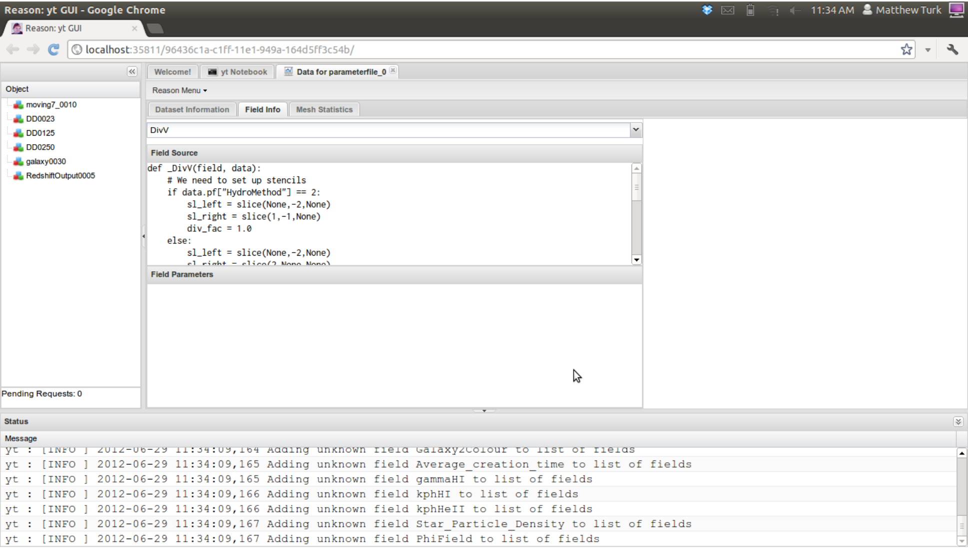
```
Welcome to Reason!
```

```
yt : [INFO ] 2012-06-29 11:32:08,322 Parameters: domain_right_edge = [ 1. 1. 1.]
yt : [INFO ] 2012-06-29 11:32:08,324 Parameters: cosmological_simulation = 1
yt : [INFO ] 2012-06-29 11:32:08,324 Parameters: current_redshift = 0
yt : [INFO ] 2012-06-29 11:32:08,324 Parameters: omega_lambda = 0.7
yt : [INFO ] 2012-06-29 11:32:08,325 Parameters: omega_matter = 0.3
yt : [INFO ] 2012-06-29 11:32:08,325 Parameters: hubble_constant = 0.65
yt : [INFO ] 2012-06-29 11:32:08,529 Starting up the server.
```



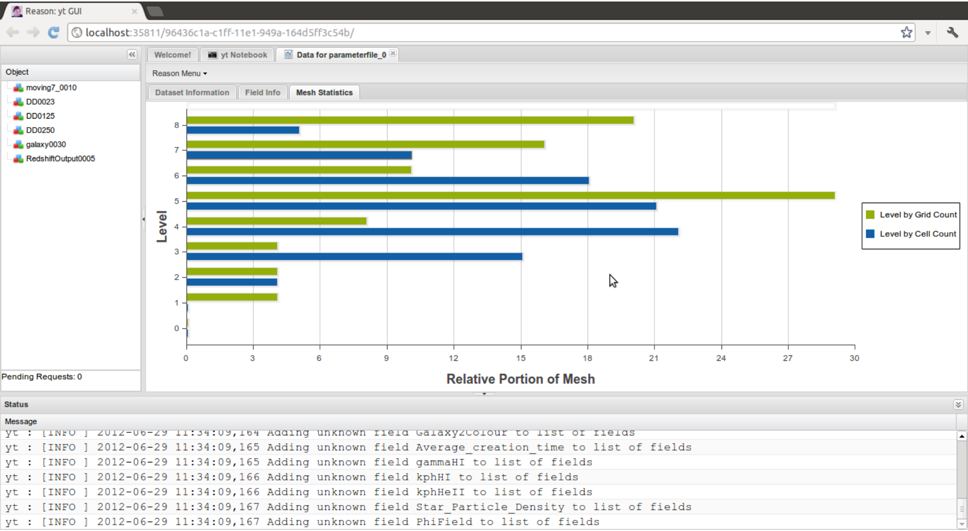
```
Parameter                Value
Output Hash              68951d2383b906f87e2df88c689176ea
Dimensionality           3
Refine by                2
Domain Dimensions        32,32,32
Cosmological Simulation  0
Current Redshift         0
Omega Matter             0
Omega Lambda             0
Hubble Constant          0
Current (sim) Time       0.0060000200028298
Domain Left Edge         0,0,0
Domain Right Edge        1,1,1
```

```
yt : [INFO ] 2012-06-29 11:34:09,164 Adding unknown field Galaxy2Colour to list of fields
yt : [INFO ] 2012-06-29 11:34:09,165 Adding unknown field Average_creation_time to list of fields
yt : [INFO ] 2012-06-29 11:34:09,165 Adding unknown field gammaHI to list of fields
yt : [INFO ] 2012-06-29 11:34:09,166 Adding unknown field kphHI to list of fields
yt : [INFO ] 2012-06-29 11:34:09,166 Adding unknown field kphHeII to list of fields
yt : [INFO ] 2012-06-29 11:34:09,167 Adding unknown field Star_Particle_Density to list of fields
yt : [INFO ] 2012-06-29 11:34:09,167 Adding unknown field PhiField to list of fields
```

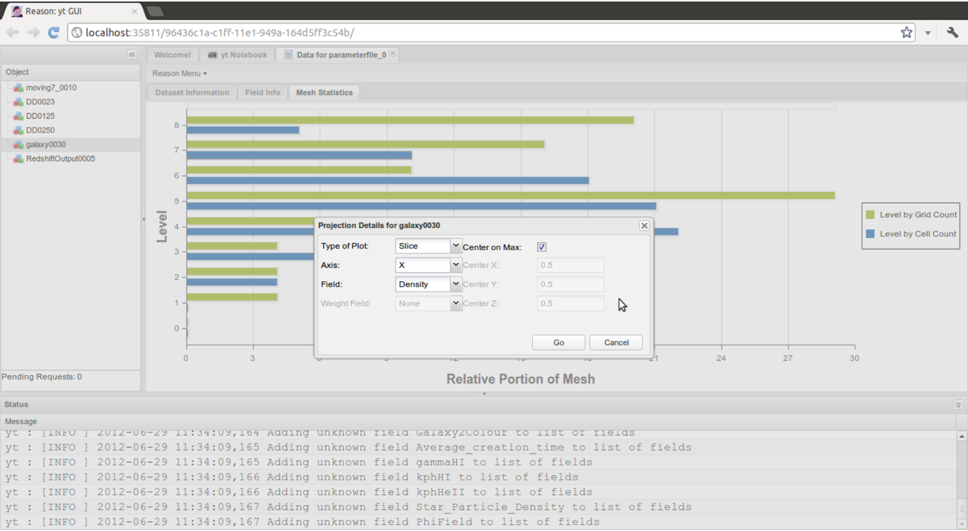


```
def _DivV(field, data):
    # We need to set up stencils
    if data.pf["HydroMethod"] == 2:
        sl_left = slice(None,-2,None)
        sl_right = slice(1,-1,None)
        div_fac = 1.0
    else:
        sl_left = slice(None,-2,None)
        sl_right = slice(2,None,None)
```

```
yt : [INFO ] 2012-06-29 11:34:09,164 Adding unknown field Galaxy2Colour to list of fields
yt : [INFO ] 2012-06-29 11:34:09,165 Adding unknown field Average_creation_time to list of fields
yt : [INFO ] 2012-06-29 11:34:09,165 Adding unknown field gammaHI to list of fields
yt : [INFO ] 2012-06-29 11:34:09,166 Adding unknown field kphHI to list of fields
yt : [INFO ] 2012-06-29 11:34:09,166 Adding unknown field kphHeII to list of fields
yt : [INFO ] 2012-06-29 11:34:09,167 Adding unknown field Star_Particle_Density to list of fields
yt : [INFO ] 2012-06-29 11:34:09,167 Adding unknown field PhiField to list of fields
```



```
yt : [INFO ] 2012-06-29 11:34:09,164 Adding unknown field Galaxy2Colour to list of fields
yt : [INFO ] 2012-06-29 11:34:09,165 Adding unknown field Average_creation_time to list of fields
yt : [INFO ] 2012-06-29 11:34:09,165 Adding unknown field gammaHI to list of fields
yt : [INFO ] 2012-06-29 11:34:09,166 Adding unknown field kphHI to list of fields
yt : [INFO ] 2012-06-29 11:34:09,166 Adding unknown field kphHeII to list of fields
yt : [INFO ] 2012-06-29 11:34:09,167 Adding unknown field Star_Particle_Density to list of fields
yt : [INFO ] 2012-06-29 11:34:09,167 Adding unknown field PhiField to list of fields
```



Projection Details for galaxy0030

Type of Plot: Slice

Center on Max: ☑

Axis: X

Center X: 0.5

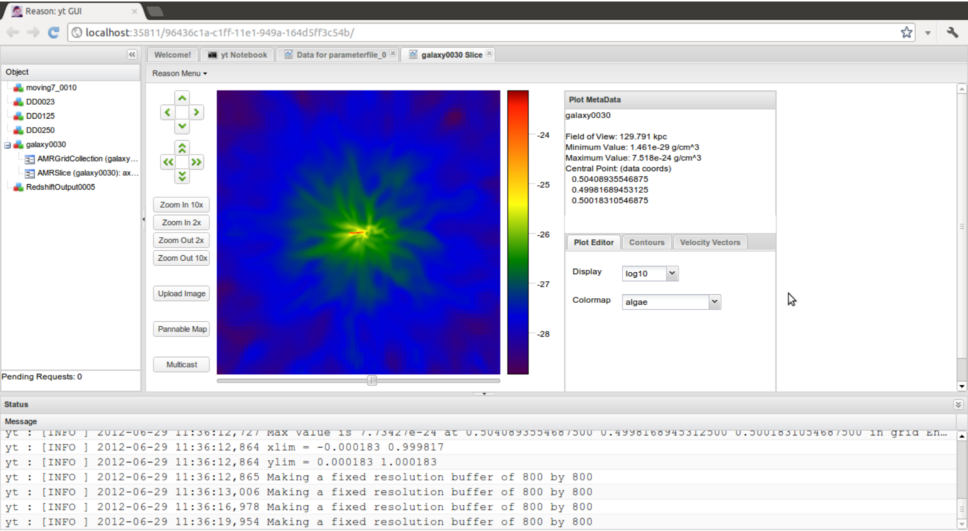
Field: Density

Center Y: 0.5

Weight Field: None

Center Z: 0.5

```
yt : [INFO ] 2012-06-29 11:34:09,164 Adding unknown field Galaxy2Colour to list of fields
yt : [INFO ] 2012-06-29 11:34:09,165 Adding unknown field Average_creation_time to list of fields
yt : [INFO ] 2012-06-29 11:34:09,165 Adding unknown field gammaHI to list of fields
yt : [INFO ] 2012-06-29 11:34:09,166 Adding unknown field kphHI to list of fields
yt : [INFO ] 2012-06-29 11:34:09,166 Adding unknown field kphHeII to list of fields
yt : [INFO ] 2012-06-29 11:34:09,167 Adding unknown field Star_Particle_Density to list of fields
yt : [INFO ] 2012-06-29 11:34:09,167 Adding unknown field PhiField to list of fields
```



Plot MetaData

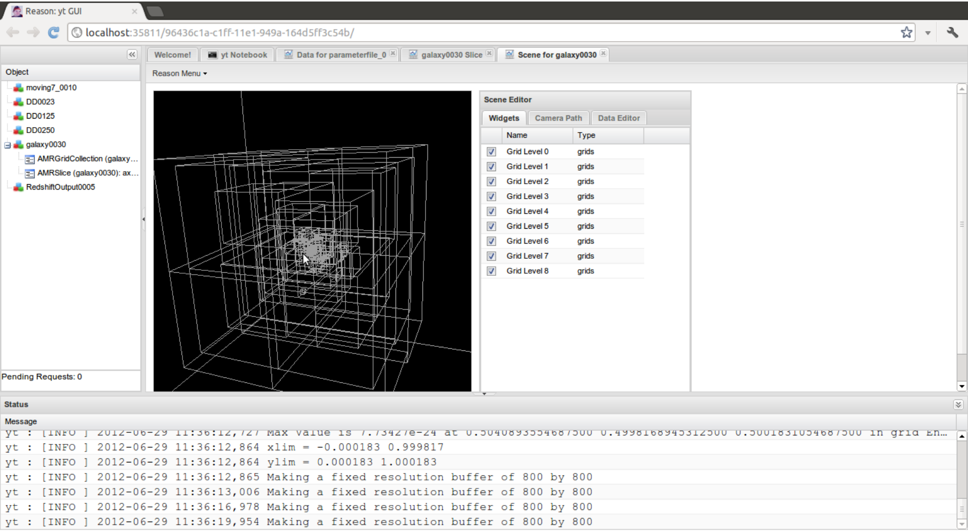
galaxy0030

Field of View: 129.791 kpc
Minimum Value: 1.461e-29 g/cm^3
Maximum Value: 7.518e-24 g/cm^3
Central Point: (data coords)
  0.50408935546875
  0.49981689453125
  0.50018310546875

Display: log10

Colormap: algae

```
yt : [INFO ] 2012-06-29 11:36:12,727 Max Value is 7.73427e-24 at 0.5040893554687500 0.4998168945312500 0.5001831054687500 in grid En…
yt : [INFO ] 2012-06-29 11:36:12,864 xlim = -0.000183 0.999817
yt : [INFO ] 2012-06-29 11:36:12,864 ylim = 0.000183 1.000183
yt : [INFO ] 2012-06-29 11:36:12,865 Making a fixed resolution buffer of 800 by 800
yt : [INFO ] 2012-06-29 11:36:13,006 Making a fixed resolution buffer of 800 by 800
yt : [INFO ] 2012-06-29 11:36:16,978 Making a fixed resolution buffer of 800 by 800
yt : [INFO ] 2012-06-29 11:36:19,954 Making a fixed resolution buffer of 800 by 800
```



| | Name | Type |
|---|---|---|
| ☑ | Grid Level 0 | grids |
| ☑ | Grid Level 1 | grids |
| ☑ | Grid Level 2 | grids |
| ☑ | Grid Level 3 | grids |
| ☑ | Grid Level 4 | grids |
| ☑ | Grid Level 5 | grids |
| ☑ | Grid Level 6 | grids |
| ☑ | Grid Level 7 | grids |
| ☑ | Grid Level 8 | grids |

```
yt : [INFO ] 2012-06-29 11:36:12,727 Max value is 7.73427e-24 at 0.5040893554687500 0.4998168945312500 0.5001831054687500 in grid En…
yt : [INFO ] 2012-06-29 11:36:12,864 xlim = -0.000183 0.999817
yt : [INFO ] 2012-06-29 11:36:12,864 ylim = 0.000183 1.000183
yt : [INFO ] 2012-06-29 11:36:12,865 Making a fixed resolution buffer of 800 by 800
yt : [INFO ] 2012-06-29 11:36:13,006 Making a fixed resolution buffer of 800 by 800
yt : [INFO ] 2012-06-29 11:36:16,978 Making a fixed resolution buffer of 800 by 800
yt : [INFO ] 2012-06-29 11:36:19,954 Making a fixed resolution buffer of 800 by 800
```

# Contributors

Tom Abel
David Collins
Andrew Cunningham
**Nathan Goldbaum**
**Cameron Hummels**
Ji-hoon Kim
Steffen Klemer
Kacper Kowalik
Michael Kuhlen
Eve Lee
Chris Malone
**Chris Moody**
Andrew Myers
**Jeff Oishi**

Jean-Claude Passy
Thomass Robitaille
Anna Rosen
Anthony Scopatz
Devin Silvia
**Sam Skillman**
**Stephen Skory**
**Britton Smith**
Geoffrey So
Casey Stark
Elizabeth Tasker
Rick Wagner
**John Wise**
**John ZuHone**

# How to get started

http://yt-project.org/

http://yt-project.org/issac2012/