yt: Introduction and Jargon

May 17, 2012

Welcome!

What kinds of things can yt do?

- Handle raw data
- Hide data handling
- Visualize in 2D
- Visualze in non-spatial dimensions
- Volume render
- ...

Example Workflow

- Define new derived field
- Select topologically connected clumps in that field
- Evaluate mass flux over those clumps
- Evaluate gravitational boundedness of those clumps
- Volume render each clump along angular momentum vector

http://yt-project.org/workshop2012/

Does everyone have `yt` installed?

Things to Remember

- Once you have installed, you have to activate yt:

```
$ source yt-x86_64/bin/activate
$ yt help
```

Some other helpful bits:

- Side-channel chat is in IRC at `irc.freenode.net` in channel `#yt`. You can use Adium, irssi, or go to `http://yt-project.org/irc.html`
- The mailing lists are usually quite helpful! For after-hours stuff, email `yt-users` or `yt-dev`.

Stuff on the Website

(demo time!)

- Documentation
- Development info
- Cookbook
- Communication channels
- Hub

Definition of Terms

"Parameter File"

This is an object that stores non-mesh, non-fluid, non-particle data about the simulation. It provides the basis for *finding* the fluid data, for describing the current time, the redshift, and so on. This is a lightweight object that doesn't require reading much from disk.

“Hierarchy”

The usage of the word “hierarchy” is a holdover from the Enzo days. It’s much better to think about this in terms of the mesh and the geometry of the simulation. This contains information about coarse-grained locations of data, such as grid patches, as well as information about how to locate data. All subsequent geometric selection hangs off this object.

“Grid Patch”

FLASH users will know this as a “block,” but for `yt`’s purposes it’s a contiguous block of fluid data, defined by dimensions, edges and contents. In `yt` they are also defined to have children and parents, and you can inspect them manually.

“Profile”

We use profiles to refer to accumulation of values along non-spatial axes. This can take the form of histograms, weighted averages, and the like. For instance, total mass as a function of Density, or the average Temperature as a function of velocity.

“Slice”

A slice is an infinitesimally thin, variable resolution object that describes all the values at a given coordinate.

“Cutting Plane”

Just like a slice, except not aligned with an axis. A lot slower, too.

"Projection"

An on-axis traversal, such that each resultant point is composed of five elements: (x, y, dx, dy, v), calculated such that $v_i = \int f(z)dl$, where z is along the axis of integration and dl is the local path length. They can also be calculated to return a weighted-average along sight lines. In `yt` projections are variable resolution and must be ...

“Pixelized”

The process of depositing values from a variable-resolution object (such as a projection or a slice) into a 2D buffer, from which an image can be made.

"Volume Rendering"

Usually "volume rendering" refers to some off-axis accumulation of colors along a plane of rays, for the purpose of visualization. In `yt` we use it to refer to any traversal of the grid by a set of vectors, including off-axis projections, averages, calculations of covering fractions and so on. This includes as a side effect visualization!

Helpful Hints

Units

Wherever possible, units in `yt` are in CGS. You can look at units by inspecting a parameter file.

Length

By default, `yt` uses the code-specific length units. In many places you can use a tuple of the form `(val, unit)` like `(2.0, 'mpc')`.

Okay, that wraps up the intro.