Introduction to
SAMRAI VisIt Data Writer
&
VisIt

Peter L. Williams

Outline of Talk

- VisIt vs. Vizamrai.
- How to create VisIt dump files in SAMRAI.
- Overview of VisIt visualizations.

New SAMRAI VisIt Data Writer Capabilities

- **Node-centered data as well as cell-centered.**
- **Deformed AMR meshes (moving grids).**
- **Variables don’t need to exist on all patches.**
- **Subsets of processors in parallel runs can dump to one file.**

New SAMRAI VisIt Data Writer Capabilities (cont’d)

- Dumps from parallel runs need no assembly.
- Ghost data can be dumped.
- Material-related data can be dumped.
- 2nd order tensors can be dumped.

SAMRAI’s VisIt Data Writer Usage Schema

1. **Create Data Writer Object (DWO)**
2. **[Set default derived data writer]**
3. **Register variables to be dumped**
4. **DWO:Write registered items to dump file**

Normally, steps 1 - 3 are done once at the beginning of the simulation, and step 4 repeated as necessary. However, step 3 can also be repeated, allowing new variables to be added to the dump at future time steps. There is no provision for de-registering a data item.

SAMRAI VisItDataWriter

Public Methods

Constructor

- VisItDataWriter(

```
        string& object_name,
        string& dump_directory_name,
        int number_procs_per_file = 1);
```

object_name	*String name for object (for debugging purposes only).*
dump_directory_name	*Name for dump directory, may include a path.*
number_procs_per_file	*Optional. Number processors to share a common dump file.*

Registering Data

- **State Variables**
- **Optionally:**
  - **Derived Data**
  - **Coordinates of Deformed Grids**
  - **Material Data**

- **registerPlotScalar(**

```
registerPlotScalar(
        string& variable_name,
        int patch_data_array_index,
        int depth_index = 0,
        double scale_factor = 1.0,
        bool omit_ghost_data = false);
```

variable_name *String name of variable.*

patch_data_array_index *Integer patch data array index.*

depth_index *Optional integer parameter specifying the component of the data to be written as a scalar.*

scale_factor *Optional parameter specifying double precision scale factor with which to multiply all data values.*

omit_ghost_data *Optional. If this scalar field has ghost data, and you want the data writer* ***not*** *to write out the ghost data, set to true.*

- **registerPlotVector(**

```
        string& variable_name,
        int patch_data_array_index,
        double scale_factor = 1.0,
        int start_depth_index = 0,
        bool omit_ghost_data = false);
```

variable_name *String name of variable.*

patch_data_array_index *Integer patch data array index.*

scale_factor *Optional parameter specifying double precision scale factor with which to multiply all data values.*

start_depth_index *Optional integer parameter specifying depth index of first component of vector to be written.*

omit_ghost_data *Optional. If this scalar field has ghost data, and you want the data writer* ***not*** *to write out the ghost data, set to true.*

```
• registerPlotTensor(
        string& variable_name,
        int patch_data_array_index,
        double scale_factor = 1.0,
        int start_depth_index = 0,
        bool omit_ghost_data = false);
```

variable_name *String name of variable.*

patch_data_array_index *Integer patch data array index.*

scale_factor *Optional parameter specifying double precision scale factor with which to multiply all data values.*

start_depth_index *Optional integer parameter specifying depth index of first component of tensor to be written.*

omit_ghost_data *Optional. If this scalar field has ghost data, and you want the data writer* ***not*** *to write out the ghost data, set to true.*

- **resetLevelPlotScalar(**

```
        string variable_name,
        int level_number,
        int patch_data_array_index,
        int depth_index = 0);
```

variable_name *String name of variable.*

level_number *Level number on which data is being reset.*

patch_data_array_index *New patch data array index.*

depth_index *Optional. New depth index, (if one component of vector data being treated as a scalar.)*

Use this method when variable lives at different patch data slots on different hierarchy levels.

- **resetLevelPlotVector(**

```
        string variable_name,
        int level_number,
        int patch_data_array_index,
        int start_depth_index = -1);
```

variable_name	*String name of variable.*
level_number	*Level number on which data is being reset.*
patch_data_array_index	*New patch data array index.*
start_depth_index	*Optional. New start depth index.* *Default is to use original value.*

- **resetLevelPlotTensor(**

```
string variable_name,
int level_number,
int patch_data_array_index,
int start_depth_index = -1);
```

variable_name	*String name of variable.*
level_number	*Level number on which data is being reset.*
patch_data_array_index	*New patch data array index.*
start_depth_index	*Optional. New start depth index.* *Default is to use original value.*

Writing the Data

Writing the Data

- **writePlotData(**

  tbox_Pointer <   > hierarchy,
  int time_step,
  double plot_time = 0.0);

hierarchy *pointer to patch hierarchy on which data to be plotted is defined.*

time_step *integer value specifying current time step number.*

plot_time *Optional argument specifying the double precision plot time.*

Registering Data

- State Variables
- **Optionally:**
  - **Derived Data**
  - Coordinates of Deformed Grids
  - Material Data

Derived Variable

- Data that does not exist in the simulation, but which is derived from state variables in the simulation.

- For example,

  Momentum = Density * Velocity

```
registerDerivedPlotScalar(
    string& variable_name,
    appu_VisDerivedDataStrategyX* derived_writer =
                                   (appu_VisDerivedDataStrategyX*)NULL,
    const string& centering = “CELL_CENTERED”,
    double scale_factor = 1.0,
    const hier_IntVectorX& ghost_cell_width = hier_IntVectorX(0));
```

variable_name	*Name of derived scalar variable*
derived_writer	*Optional derived data strategy object to use to calculate the data..*
centering	*Optional. May specify “NODE_CENTERED”.*
scale_factor	*Optional. Scale factor.*
ghost_cell_width	*Optional. Integer vector of ghost cell widths. Default is no ghost data. If non-zero ghost cell width, VisIt expects ghost data to be dumped.*

```
registerDerivedPlotVector(
    string& variable_name,
    appu_VisDerivedDataStrategyX* derived_writer =
                                    (appu_VisDerivedDataStrategyX*)NULL,
    const string& centering = “CELL_CENTERED”,
    double scale_factor = 1.0,
    const hier_IntVectorX& ghost_cell_width = hier_IntVectorX(0));
```

variable_name	*Name of derived vector variable*
derived_writer	*Optional derived data strategy object to use to calculate the data..*
centering	*Optional. May specify “NODE_CENTERED”.*
scale_factor	*Optional. Scale factor.*
ghost_cell_width	*Optional. Integer vector of ghost cell widths. Default is no ghost data.*

```
registerDerivedPlotTensor(
   string& variable_name,
   appu_VisDerivedDataStrategyX* derived_writer =
                                 (appu_VisDerivedDataStrategyX*)NULL,
   const string& centering = “CELL_CENTERED”,
   double scale_factor = 1.0,
   const hier_IntVectorX& ghost_cell_width = hier_IntVectorX(0));
```

variable_name	*Name of derived tensor variable*
derived_writer	*Optional derived data strategy object to use to calculate the data..*
centering	*Optional. May specify “NODE_CENTERED”.*
scale_factor	*Optional. Scale factor.*
ghost_cell_width	*Optional. Integer vector of ghost cell widths. Default is no ghost data.*

- **SetDefaultDerivedDataWriter**(
  appu_VisDerivedDataStrategyX*
  default_derived_writer);

default_derived_writer *Pointer to default derived data strategy object.*

The default derived data writer will be used only if registerDerivedPlotScalar/Vector/Tensor() does not specify a derived data strategy object to use.

For Derived Data, Ap Class Inherits …

```
#include “VisDerivedDataStrategy.h”
Class Applic :
     public VisDerivedDataStrategy
```

---- *Applic needs to implement this method* ----

```
bool packDerivedDataIntoDoubleBuffer(
           double *buffer,
           const hier_PatchX& patch,
           const hier_BoxX& region,
           const string& variable_name,
           int depth_index);
```

Arguments described on next page.

```
bool packDerivedDataIntoDoubleBuffer(
          double *buffer,
          const hier_PatchX& patch,
          const hier_BoxX& region,
          const string& variable_name,
          int depth_index);
```

buffer — *Double precision buffer, already allocated to correct size.*

patch — *Patch on which data exists.*

region — *Box region over which to pack data.*

variable_name — *Name previously registered for this derived variable.*

depth_index — *Depth index of data to be packed. For scalar data, always 0. For vector data this index varies between 0 and NDIM-1, for tensor data index varies between 0 and (NDIM*NDIM)-1.*

Return Value — *Boolean indicating if derived data exists on this patch.*

Registering Data

- State Variables
- Optionally:
  - Derived Data
  - Coordinates of Deformed Grids
  - Material Data

Registering Coordinates of Deformed Structured AMR Grids

- registerNodeCoordinate(

```
        int coordinate_number,
        int patch_data_array_index,
        int depth_index = 0,
        double scale_factor = 1.0);
```

coordinate_number *Integer indicating which dimension of coordinate is being registered. 0 <= coordinate_number < NDIM.*

patch_data_array_index *Integer index of coordinate data.*

depth_index *If patch_data_array_index refers to a vector, this optional parameter specifies the component of that vector to be used..*

scale_factor *May be different for each component.*

This method must be called once for each of the NDIM dimensions.

Registering Data

- State Variables
- **Optionally:**
  - Derived Data
  - Coordinates of Deformed Grids
  - **Material Data**

Material-related Data

- **Applications with cells containing fractional amounts of material compounds, e.g. *copper, gold, gas, fluid*. Each of these is a material.**

- **A material may have subcomponents called species: e.g. *gas* may be composed of *O_2, N_2* & *methane*. We say: *O_2* is a species of *gas*.**

- **Each material may have own set of species.**

Materials vs Species

- Materials: heterogeneous mixture of substances (with distinct boundaries), e.g. concrete, granite, …
- Species: homogeneously mixed substances, e.g. seawater, Coke, air, …

- **Scalar/Vector data may be defined over set of materials – referred to as material state variables.**

- **e.g. a different temperature may be associated with each material on a cell by cell basis.**

- **All material fractions, species fractions and material state variables must be “cell-centered”.**
  - **(If necessary, can convert node- to cell- centered in packing routines to be described later.)**

- **Assumption: Every cell contains fractional amount *mf* (0 < *mf* <= 1.0) of every material called material fraction. Sum of *mf*'s over all materials for a cell must be 1.0.**

- **Similarly for species, called species fraction. Sum of *sf*'s for all species of material *m* must be 1.0 in every cell in which *m* appears.**

- **Every material state variable must have value for each cell, for each material *m*, if *m* has non-zero fraction in that cell.**

- **VisIt Data Writer allows user to dump: material fractions, species fractions and material state variables. In addition, species state variables (SSV) may be registered.**

- **VisIt treats SSV's in unique way. When displayed, each value of SSV for a cell is multiplied by sum of species fractions for that cell for currently selected species.**

- **(Species selected in VisIt's subset window.)**

- **E.g *pressure p* registered as SSV, if *p* = 100 for cell *c*, and one species selected, say N_2, and N_2's species fraction for *c* is 0.45, then the partial pressure for *c* will be 45.**

- **VisIt will automatically display partial pressure field for N_2 if pressure registered as SSV.**

- **Species fractions may also be treated as scalar field to show "concentrations".**

- **VisIt uses <u>material</u> fractions to reconstruct material boundaries within cells containing multiple materials.**

- **VisIt can display material(s) as multiple colored contiguous regions.**

- **Material state variables can be displayed over a material.**

- **Material fractions can be displayed as scalar field**

SAMRAI's VisIt Data Writer Usage Schema

1. Create Data Writer Object (DWO)
2. [Set default derived data writer]
3. Register variables to be dumped
4. **Register material-related data**
5. DWO:Write registered items to dump file

- **registerMaterialNames(**

```
    const tbox_Array<string>& material_names,
    const hier_IntVectorX& ghost_cell_width =
                                       hier_IntVectorX(0));
```

material_names *String array of the names of all the materials.*

ghost_cell_width *Optional integer vector of ghost cell widths. Default is no ghost data. If non-zero ghost cell width specified, VisIt expects ghost data to be dumped. This ghost cell width applies to all material-related data.*

- **registerSpeciesNames(**

```
    const string& material_name,
    const tbox_Array<string>& species_names);
```

material_name *Name of material whose species are being registered.*

species_names *String array of the names of all the species for this material.*

registerMaterialNames() must be called before this method is invoked.

- **registerMaterialStateVariable(**
  const string& state_variable_name,
  const int depth = 1,
  const double scale_factor = 1.0);

state_variable_name *name of cell-centered state variable*
depth *optional integer depth of state variable;*
*allowable values: 1, NDIM, NDIM*NDIM*
scale_factor *optional scale factor.*

registerMaterialNames() must be called before this method is invoked

- **registerSpeciesStateVariable(**
  const string& state_variable_name);

state_variable_name *name of state variable, can be node or cell centered.*

- **SetMaterialsDataWriter(
  appu_VisMaterialsDataStrategyX*
  materials_data_writer);**

materials_data_writer *Pointer to materials data writer object.*

For Material Data, Ap Class Inherits …

```
#include “VisMaterialsDataStrategy.h”
Class Applic :
    public VisMaterialsDataStrategy
```

---- Applic needs to implement this method ----

```
int  packMaterialFractionsIntoDoubleBuffer(
          double *buffer,
          const hier_PatchX& patch,
          const hier_BoxX& region,
          const string& material_name);
```

Arguments described on next page.

```
int  packMaterialFractionsIntoDoubleBuffer(
        double *buffer,
        const hier_PatchX& patch,
        const hier_BoxX& region,
        const string& material_name);
```

buffer — *Double precision buffer, already allocated to correct size.*

patch — *Patch on which data exists.*

region — *Box region over which to pack data.*

material_name — *Name of the material.*

Return Value — *Enumeration constant:*
VisMaterialsDataStrategy::ALL_ZEROS,
VisMaterialsDataStrategy::ALL_ONES, or
VisMaterialsDataStrategy::SOME.
(See documentation)

Material Fractions must be cell-centered.

If species are used, implement packSpeciesFractionsIntoDoubleBuffer() described next.

```
int  packSpeciesFractionsIntoDoubleBuffer(
            double *buffer,
           const hier_PatchX& patch,
           const hier_BoxX& region,
           const string& material_name,
           const string& species_name);
```

buffer *Double precision buffer, already allocated to correct size.*

patch *Patch on which data exists.*

region *Box region over which to pack data.*

material_name *Name of the material which has this species.*

species_name *Name of the species.*

Return Value *Enumeration constant:*
VisMaterialsDataStrategy::ALL_ZEROS,
VisMaterialsDataStrategy::ALL_ONES, or
VisMaterialsDataStrategy::SOME.
(See documentation)

Species Fractions must be cell-centered --- If necessary convert from node-centered to cell-centered in this packing routine.

If material state variables are used, implement packMaterialStateVariableIntoDoubleBuffer() described next.

```
void packMaterialStateVariableIntoDoubleBuffer(
          double *buffer,
          const hier_PatchX& patch,
          const hier_BoxX& region,
          const string& material_name,
          const string& state_variable_name ,
         const int depth_index);
```

buffer *Double precision buffer, already allocated to correct size.*

patch *Patch on which data exists.*

region *Box region over which to pack data.*

material_name *Name of the material.*

state_variable_name *Name of the state variable.*

depth_index *Depth index of data to be packed. For scalar data, always 0. For vector data, index varies from 0 to NDIM-1; for tensor data index varies from 0 to (NDIM*NDIM)-1.*

Material State Variables must be cell-centered --- If necessary convert from node-centered to cell-centered in this packing routine.

GOTCHA’s

- **Dumping data not in floating point range.**
  - Since VisIt only works with float data, data > FLT_MAX will be clamped to FLT_MAX.
  - Use scale_factor to keep data in range and avoid this.

- **Not initializing all ghost cells (nodes).**
  - Be sure all ghost cells (nodes) have a value, not just the ones your application uses.
  - Can use SAMRAI method fillAll(0.)

Documentation

“Generating VisIt Visualization Data Files in SAMRAI”

- **More details on what we covered today.**
- **Complete set of example VisIt Data Writer calls in an application code.**
- **Brief introduction to use of VisIt with SAMRAI data, and pointers to VisIt documentation.**
- **Available in SAMRAI distribution at:**

docs/userdocs/VisIt-writer.pdf

Brief Overview of VisIt

New Capabilities with VisIt

- Scalable rendering --- order of magnitude faster for large data sets if parallel compute engine available.

- VisIt can be extended with new plot & operator plugins.

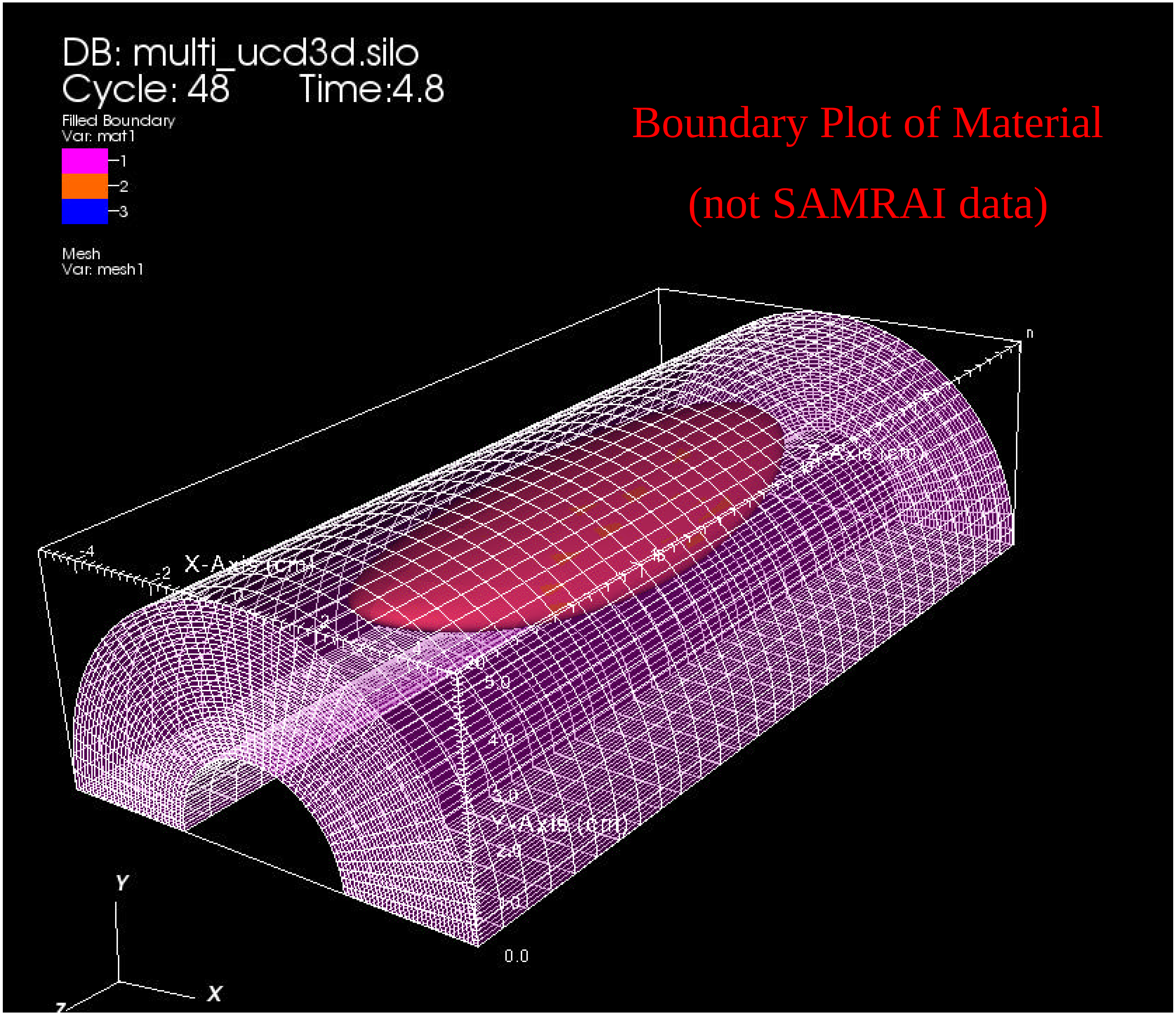
- VisIt allows mathematical expressions involving variables to be defined at vis time (thus offering a similar capability to SAMRAI's derived data).

- Special material-related viewing capabilities.

- Stereo viewing.

VisIt Plot Types

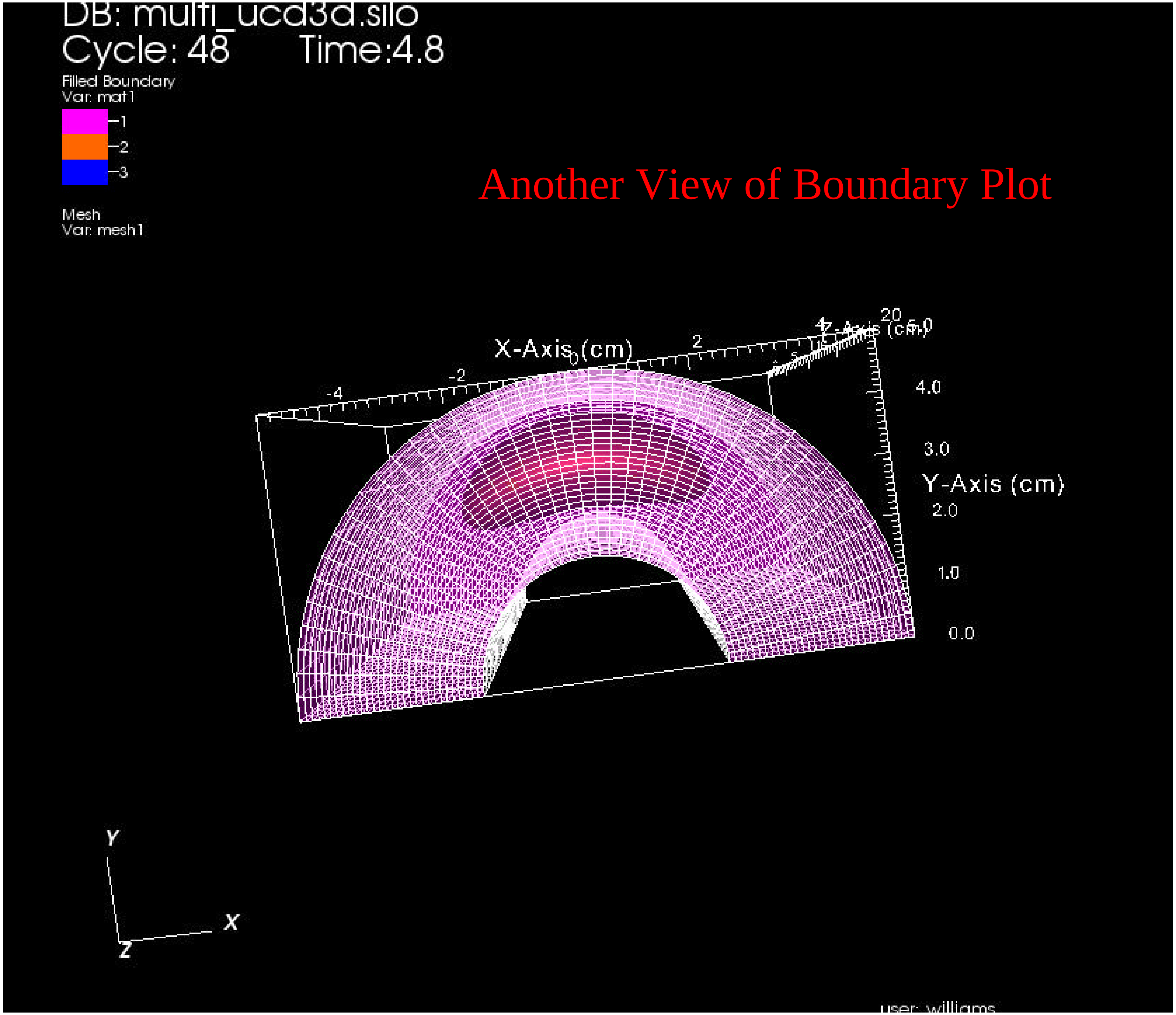
- Boundary: show bnds. between materials, patches, ..
  (see examples on next 3 slides)
- Contours: multiple semitransparent isosurfaces
- Mesh: line smoothing available
- Pseudocolor: paint variable value onto surface
- Streamlines: multiple sources - point, line, plane, ..
- Subset: select specific materials, levels, patches, etc.
- Surface: height field, for 2D only.
- Vector: glyphs indicating direction & magnitude.
- Volume visualization
- Roll your own plot: create a VisIt plot plugin.

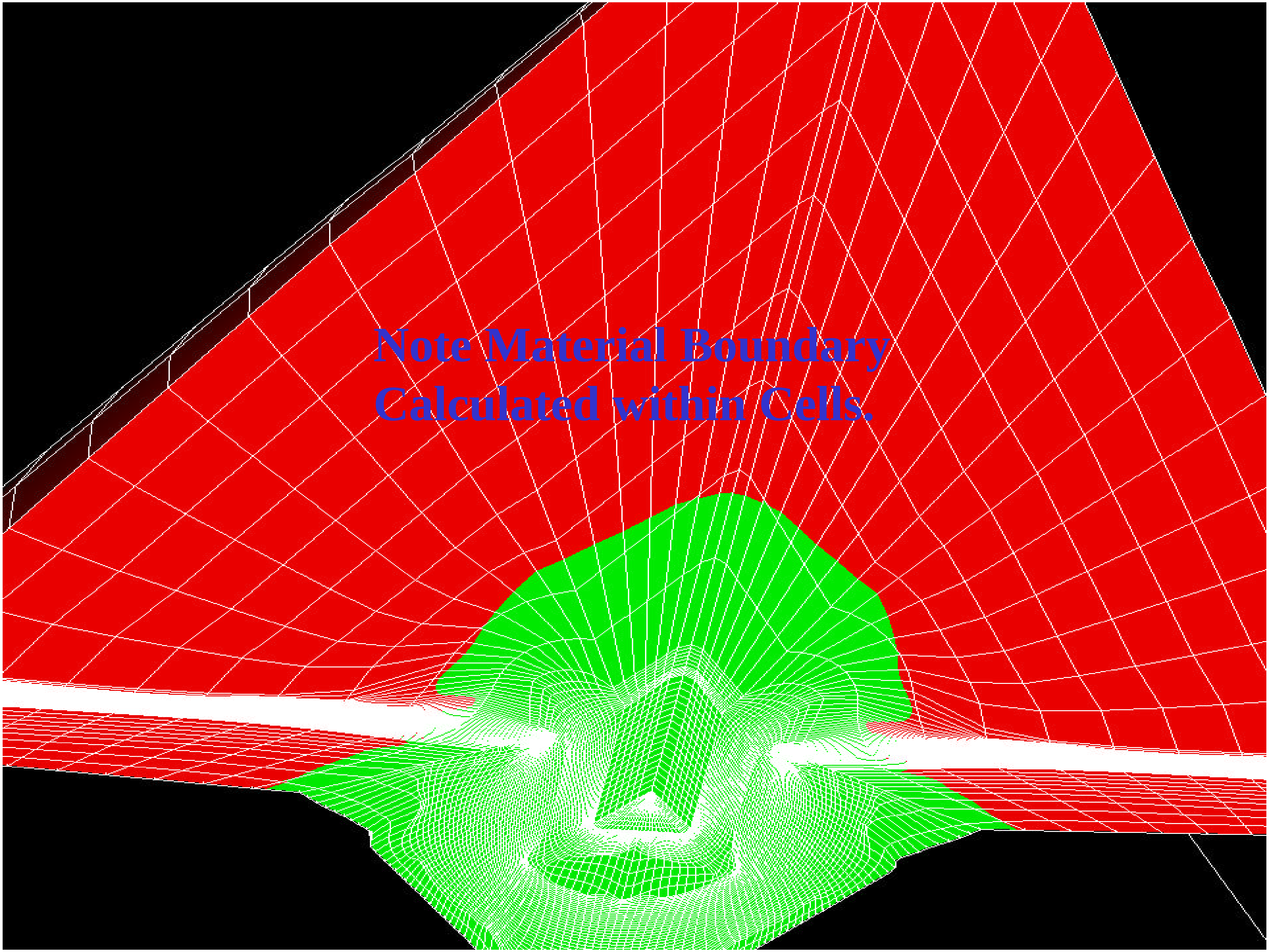


Boundary Plot of Material

(not SAMRAI data)

Another View of Boundary Plot

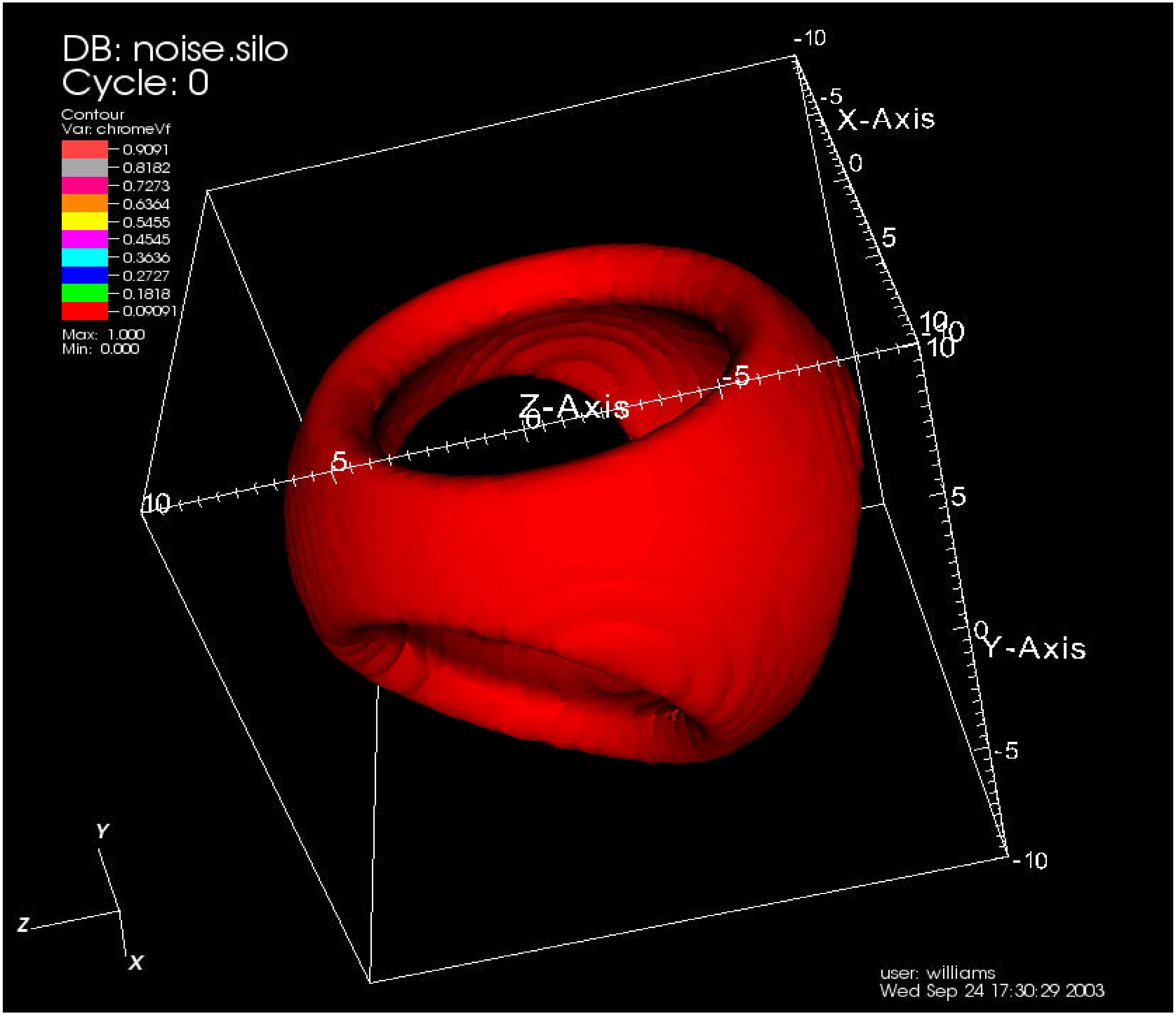


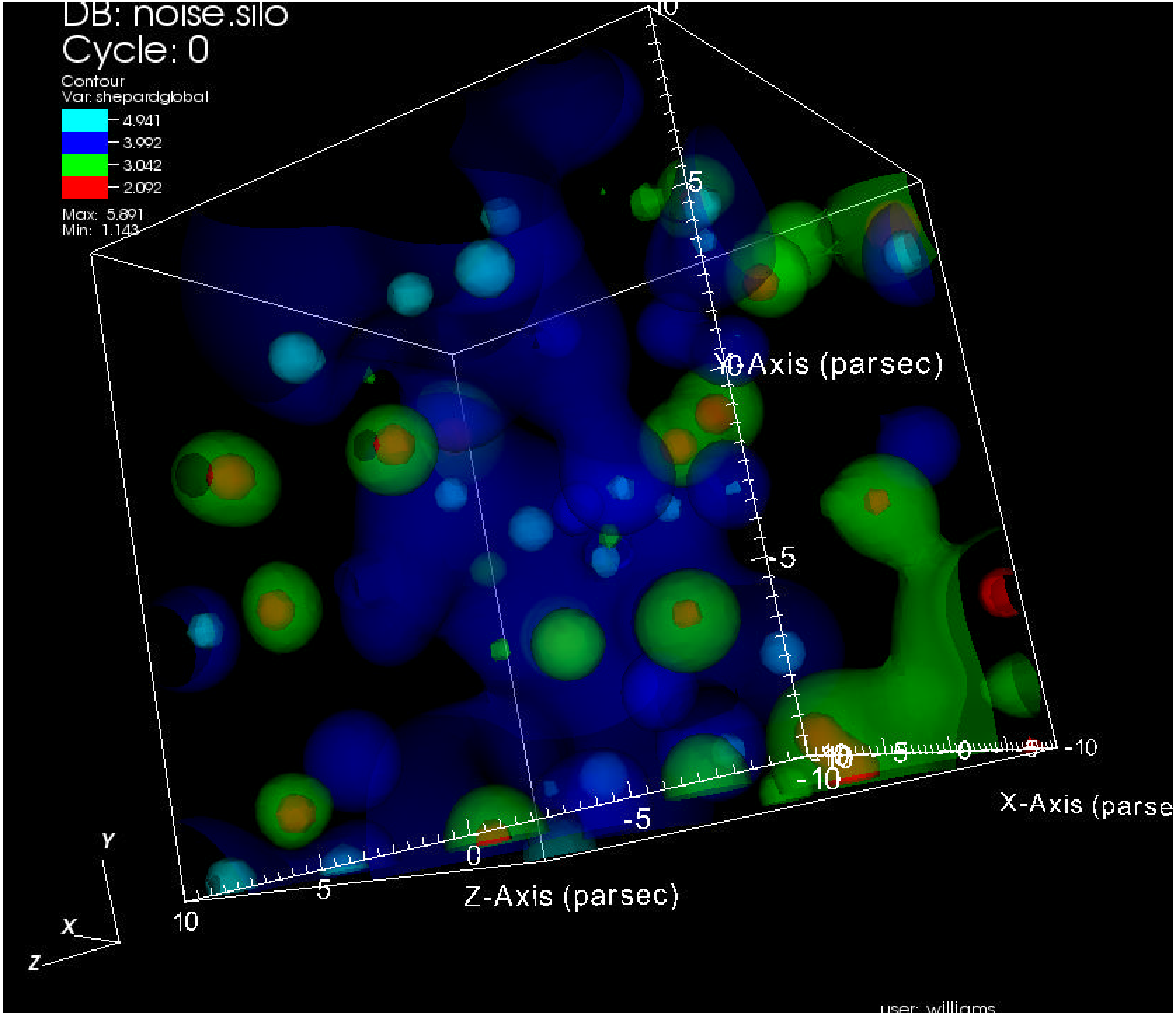


Note Material Boundary
Calculated within Cells.

VisIt Plot Types

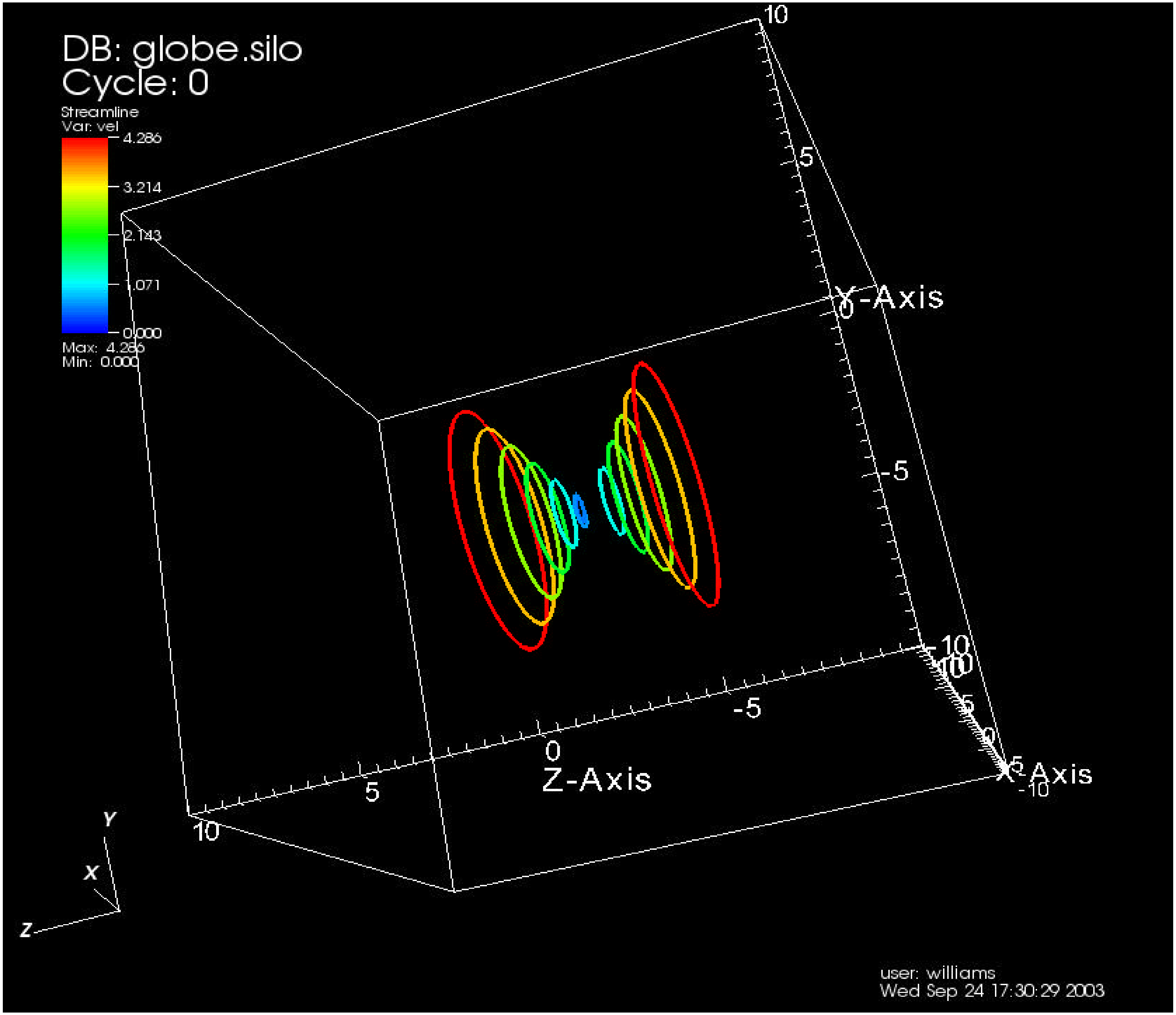
- Boundary: show bnds. between materials, patches, ..
- Contours: multiple semitransparent isosurfaces (see examples next 2 slides)
- Mesh: line smoothing available
- Pseudocolor: paint variable value onto surface
- Streamlines: multiple sources - point, line, plane, ..
- Subset: select specific materials, levels, patches, etc.
- Surface: height field, for 2D only.
- Vector: glyphs indicating direction & magnitude.
- Volume visualization
- Roll your own plot: create a VisIt plot plugin.


DB: noise.silo
Cycle: 0
Contour
Var: chromeVf
0.9091
0.8182
0.7273
0.6364
0.5455
0.4545
0.3636
0.2727
0.1818
0.09091
Max: 1.000
Min: 0.000
X-Axis
-10
-5
0
5
10
Z-Axis
10
5
0
-5
-10
Y-Axis
10
5
0
-5
-10
Y
Z
X
user: williams
Wed Sep 24 17:30:29 2003


DB: noise.silo
Cycle: 0
Contour
Var: shepardglobal
4.941
3.992
3.042
2.092
Max: 5.891
Min: 1.143
Y-Axis (parsec)
X-Axis (parse
Z-Axis (parsec)
10
5
0
-5
-10
Y
X
Z
user: williams

VisIt Plot Types

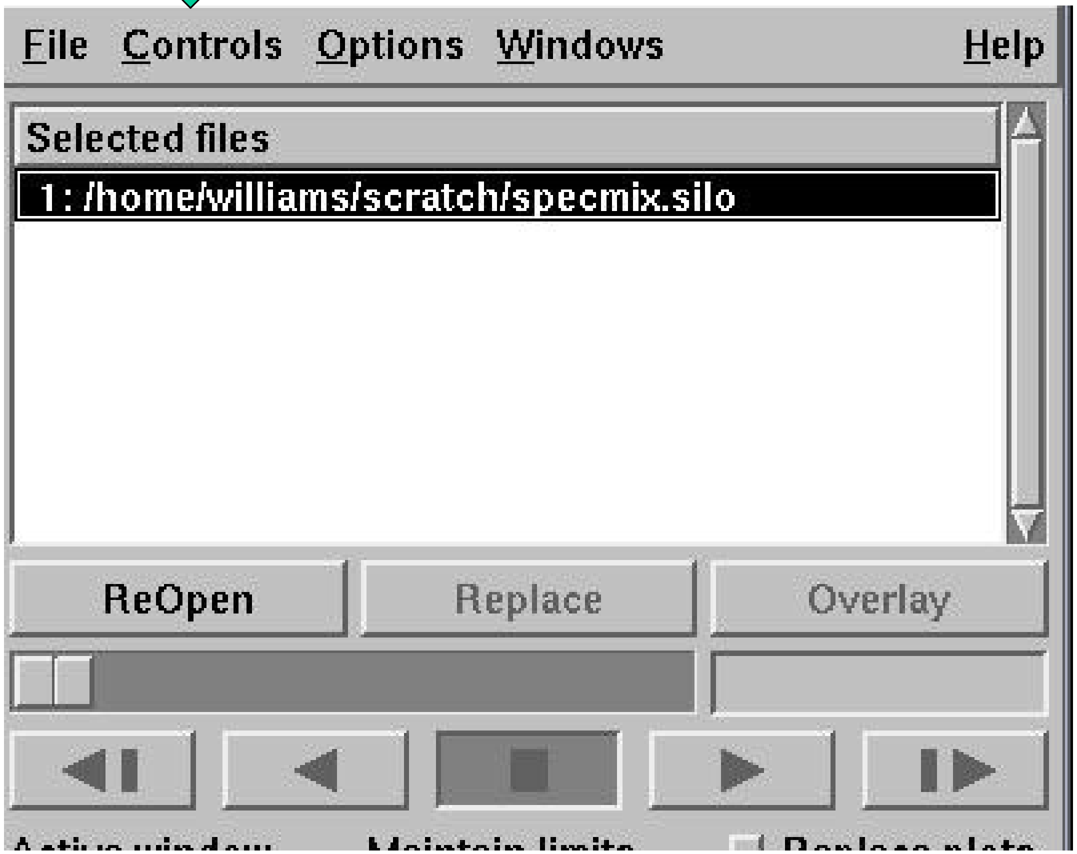
- **Boundary: show bnds. between materials, patches, ..**
- **Contours: multiple semitransparent isosurfaces**
- **Mesh: line smoothing available**
- **Pseudocolor: paint variable value onto surface**
- **Streamlines: multiple sources - point, line, plane, .. (see example on next slide)**
- **Subset: select specific materials, levels, patches, etc.**
- **Surface: height field, for 2D only.**
- **Vector: glyphs indicating direction & magnitude.**
- **Volume visualization**
- **Roll your own plot: create a VisIt plot plugin.**


DB: globe.silo
Cycle: 0
Streamline
Var: vel
4.286
3.214
2.143
1.071
0.000
Max: 4.286
Min: 0.000
10
5
0
-5
-10
Y-Axis
Z-Axis
X-Axis
Y
X
Z
user: williams
Wed Sep 24 17:30:29 2003

VisIt Plot Types

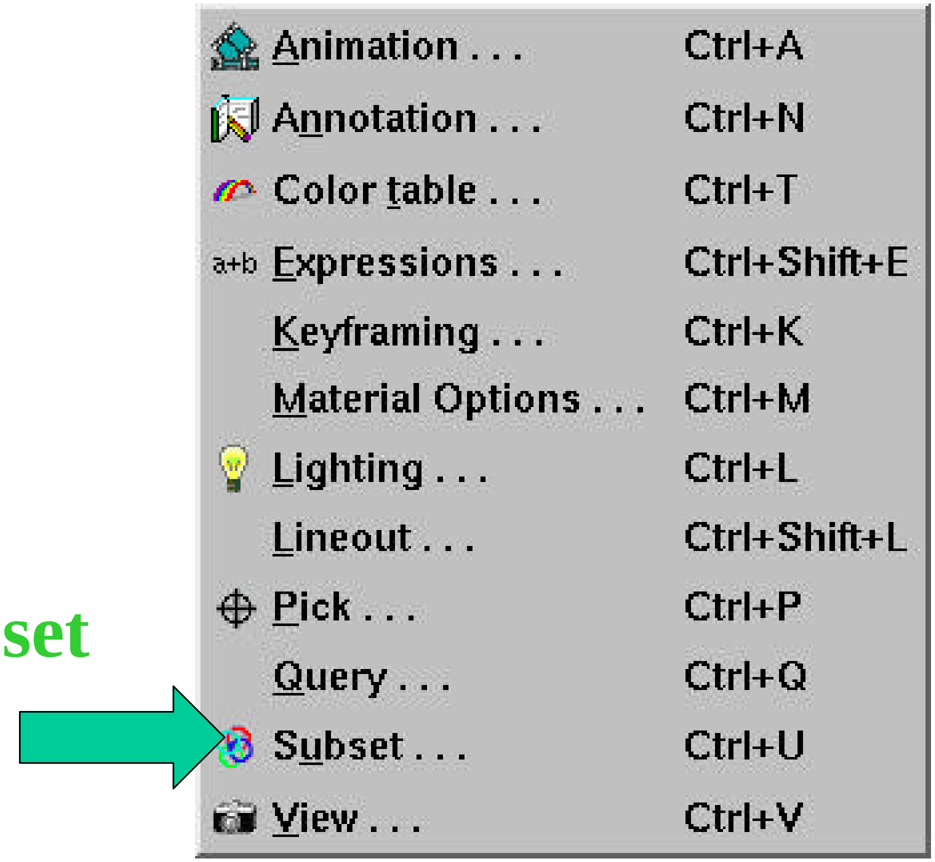
- **Boundary:** show bnds. between materials, patches, ..
- **Contours:** multiple semitransparent isosurfaces
- **Mesh:** line smoothing available
- **Pseudocolor:** paint variable value onto surface
- **Streamlines:** multiple sources - point, line, plane, ..
- **Subset:** select specific materials, levels, patches, etc. (This is very useful tool!! To access, use pull-down Controls menu at top – see next 3 slides)
- Surface: height field, for 2D only.
- Vector: glyphs indicating direction & magnitude.
- Volume visualization
- Roll your own plot: create a VisIt plot plugin.

Pull down Controls Menu



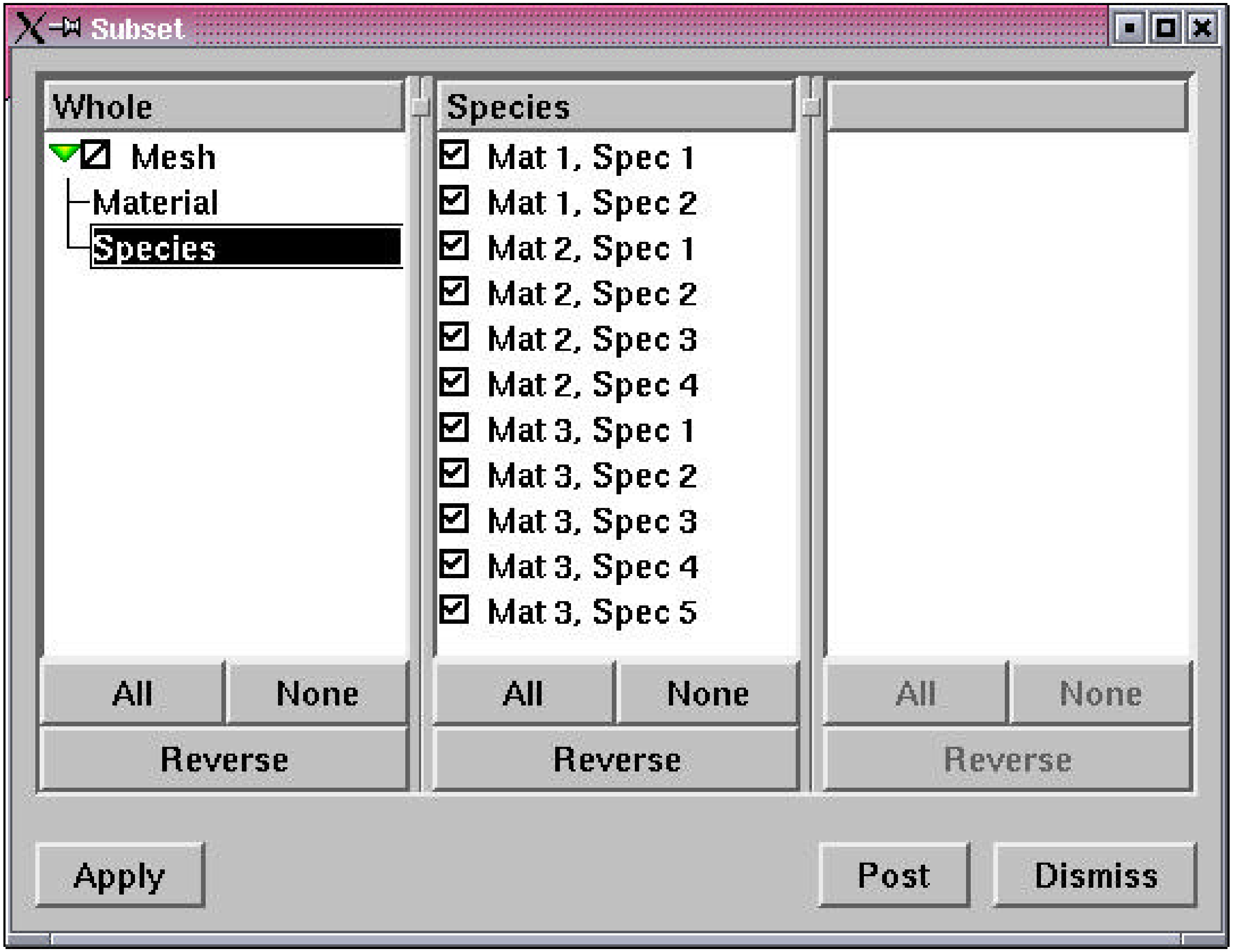
Selected files

1: /home/williams/scratch/specmix.silo



Select Subset

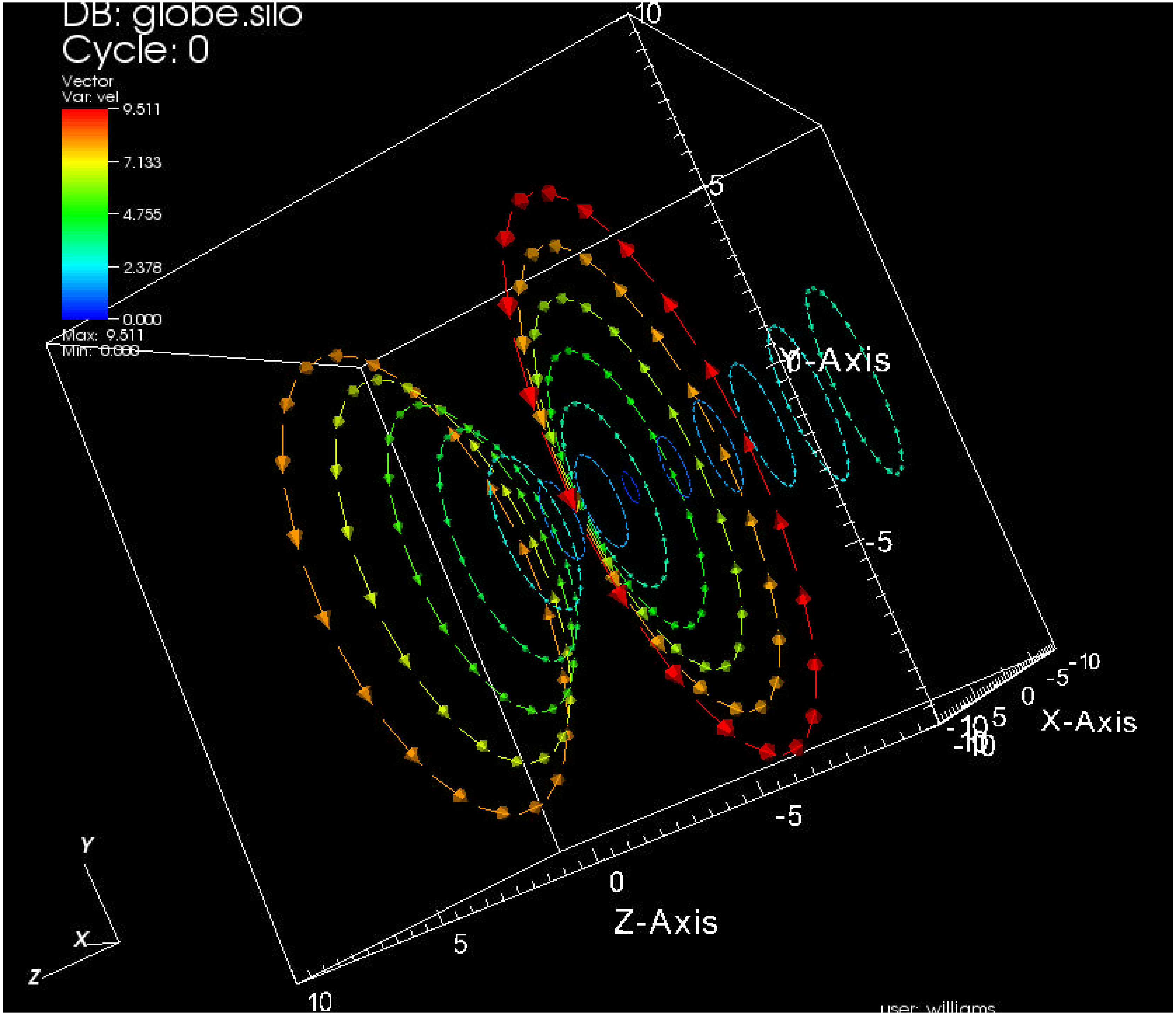
Select material / species you want.



☑ Mat 1, Spec 1
☑ Mat 1, Spec 2
☑ Mat 2, Spec 1
☑ Mat 2, Spec 2
☑ Mat 2, Spec 3
☑ Mat 2, Spec 4
☑ Mat 3, Spec 1
☑ Mat 3, Spec 2
☑ Mat 3, Spec 3
☑ Mat 3, Spec 4
☑ Mat 3, Spec 5

VisIt Plot Types

- **Boundary:** show bnds. between materials, patches, ..
- **Contours:** multiple semitransparent isosurfaces
- **Mesh:** line smoothing available
- **Pseudocolor:** paint variable value onto surface
- **Streamlines:** multiple sources - point, line, plane, ..
- **Subset:** select specific materials, levels, patches, etc.
- **Surface:** height field, for 2D only.
- **Vector:** glyphs indicating direction & magnitude. (see example next slide)
- Volume visualization
- Roll your own plot: create a VisIt plot plugin.


DB: globe.silo
Cycle: 0
Vector
Var: vel
9.511
7.133
4.755
2.378
0.000
Max: 9.511
Min: 0.000
Y-Axis
X-Axis
Z-Axis
10
5
0
-5
-10
Y
X
Z
user: williams

VisIt Plot Types

- **Boundary:** show bnds. between materials, patches, ..
- **Contours:** multiple semitransparent isosurfaces
- **Mesh:** line smoothing available
- **Pseudocolor:** paint variable value onto surface
- **Streamlines:** multiple sources - point, line, plane, ..
- **Subset:** select specific materials, levels, patches, etc.
- **Surface:** height field, for 2D only.
- **Vector:** glyphs indicating direction & magnitude.
- **Volume visualization**
- **Roll your own plot:** create a VisIt plot plugin.

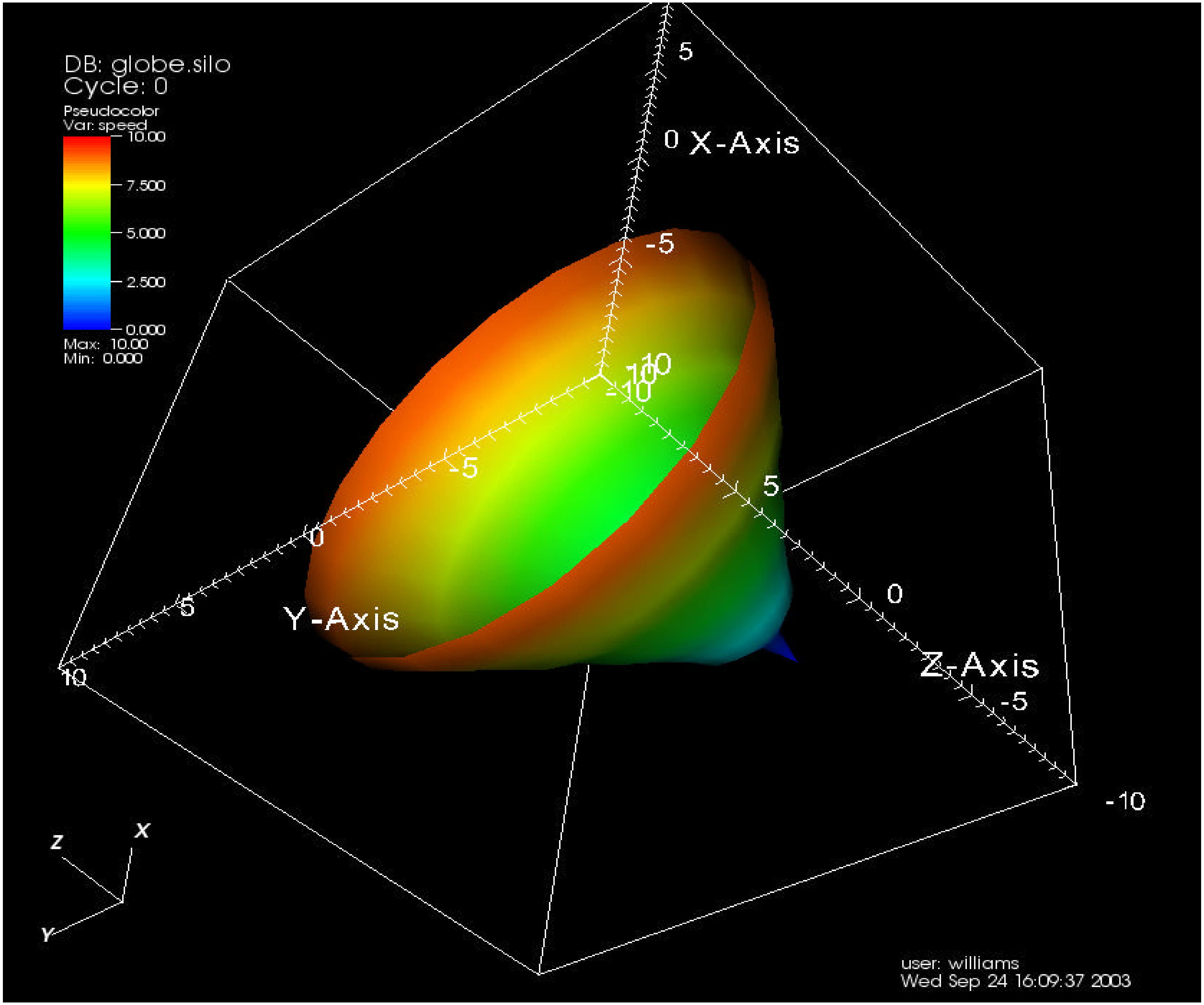
VisIt Operators: *filters applied to variable*

- **Box – clip cells outside *axis-aligned* box, individual cells not clipped.**

- **Clip – clip box / sphere shaped regions, individual cells clipped, arbitrary aligned boxes.**

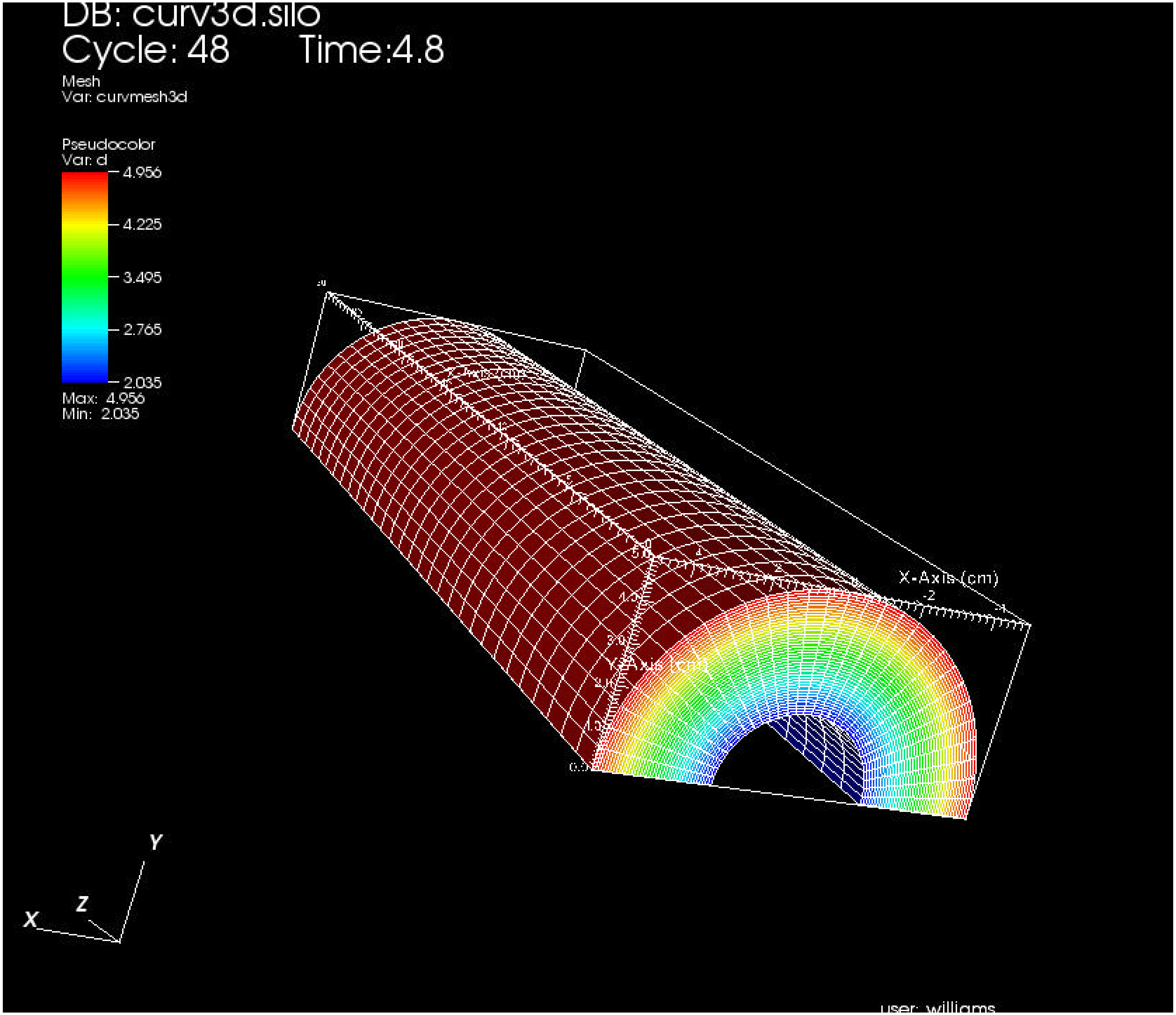
- **Index Select – select subset based on range of cell indices or patch numbers.**

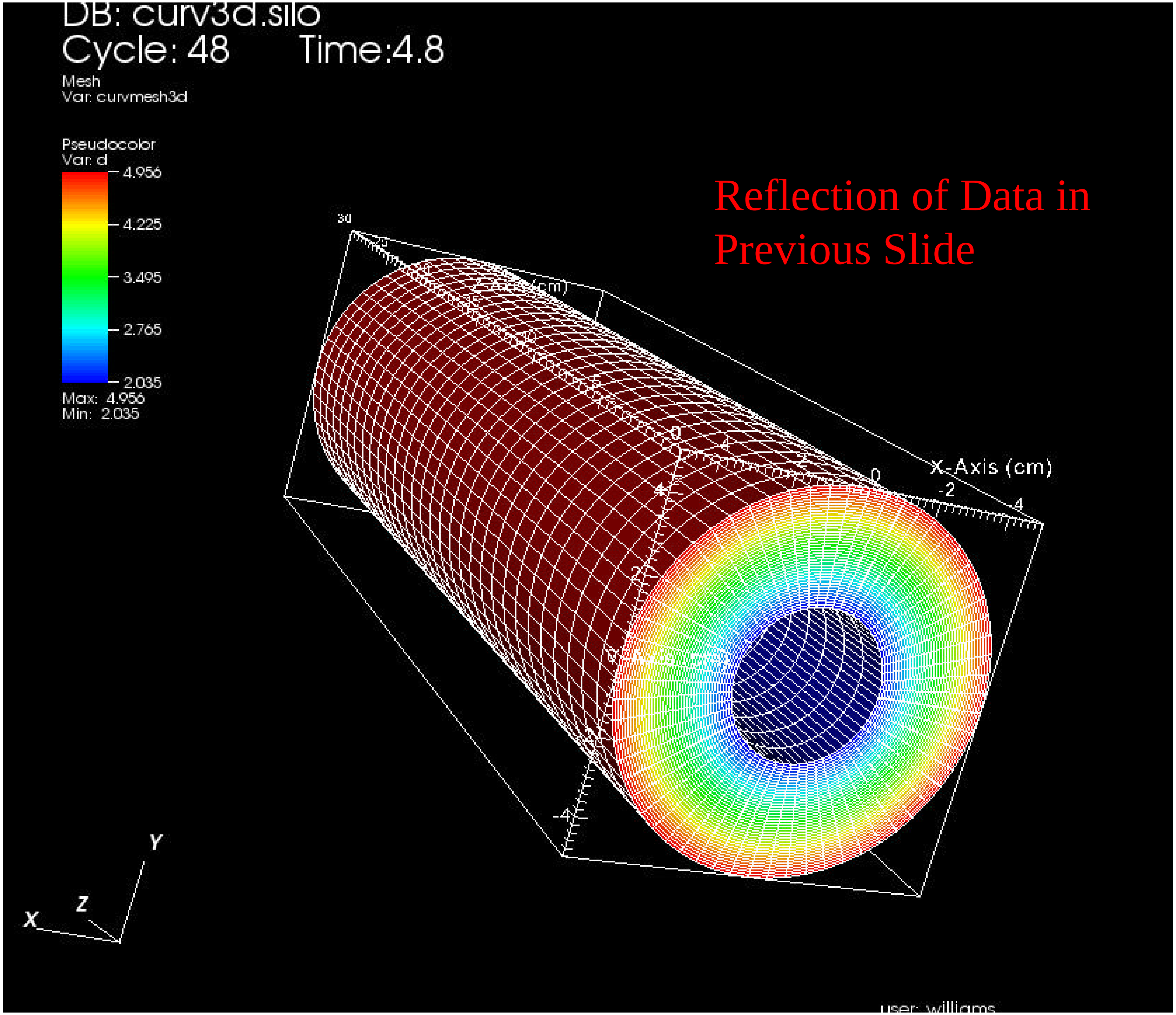
- **Cone – slice 3D data with cone. (see example next slide)**


DB: globe.silo
Cycle: 0
Pseudocolor
Var: speed
10.00
7.500
5.000
2.500
0.000
Max: 10.00
Min: 0.000
5
0 X-Axis
-5
-10
-10
Y-Axis
-5
0
5
10
Z-Axis
5
0
-5
-10
Z
X
Y
user: williams
Wed Sep 24 16:09:37 2003

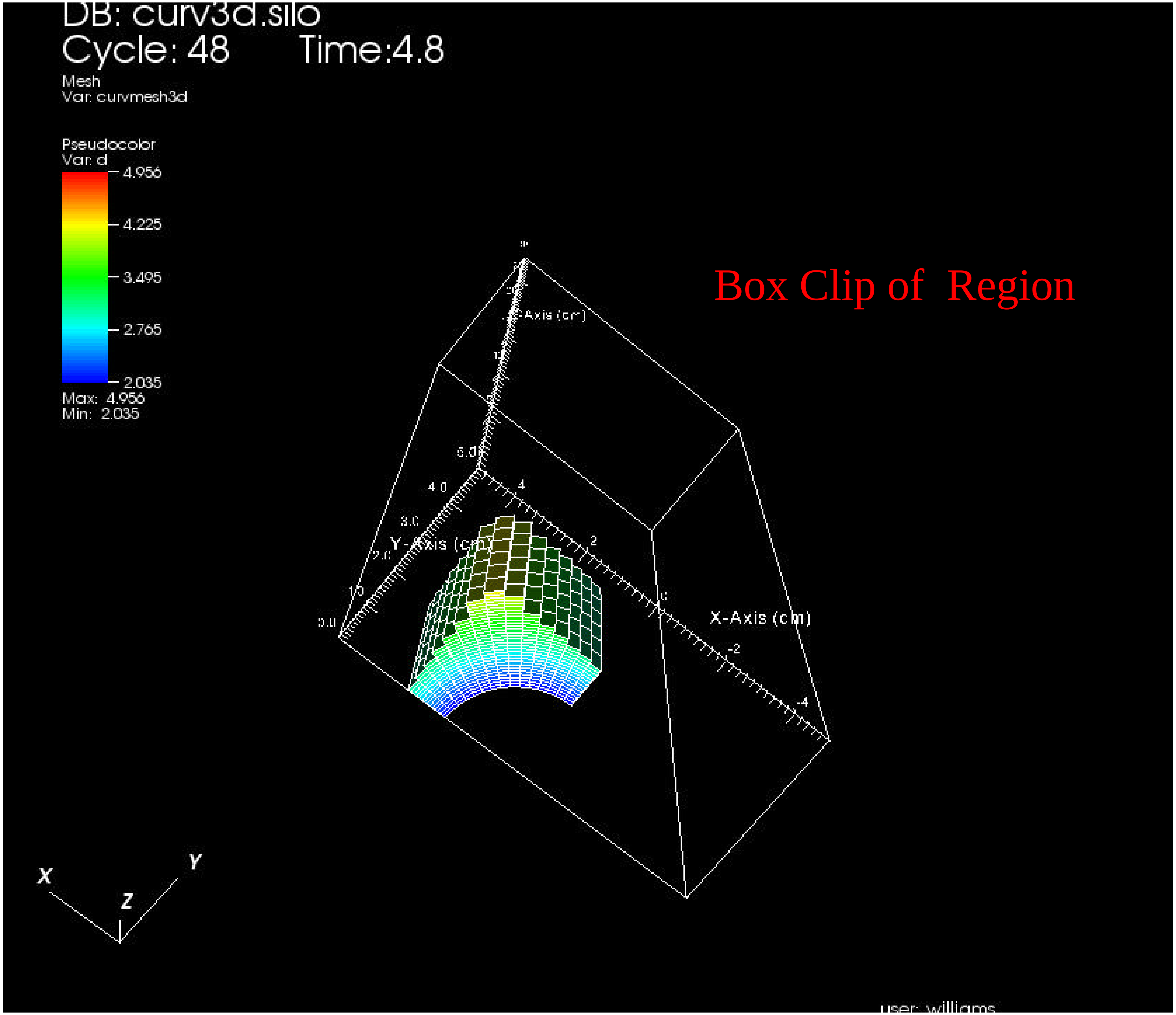
VisIt Operators: *filters applied to variable*

- **Isosurface:** **isosurface colored by different var.**
- **Lineout:** **extract 1D data from 2D or 3D plots.**
- **Onion Peel:** **grow image outwards in layers from seed cell.**
- **Reflect:** **reflect geometry across axes. (next three slides)**
- **Slice: 3D data mapped to 2D surface.**


DB: curv3d.silo
Cycle: 48 Time:4.8
Mesh
Var: curvmesh3d
Pseudocolor
Var: d
4.956
4.225
3.495
2.765
2.035
Max: 4.956
Min: 2.035
X-Axis (cm)
Y-Axis (cm)
Y
Z
X
user: williams



Reflection of Data in Previous Slide



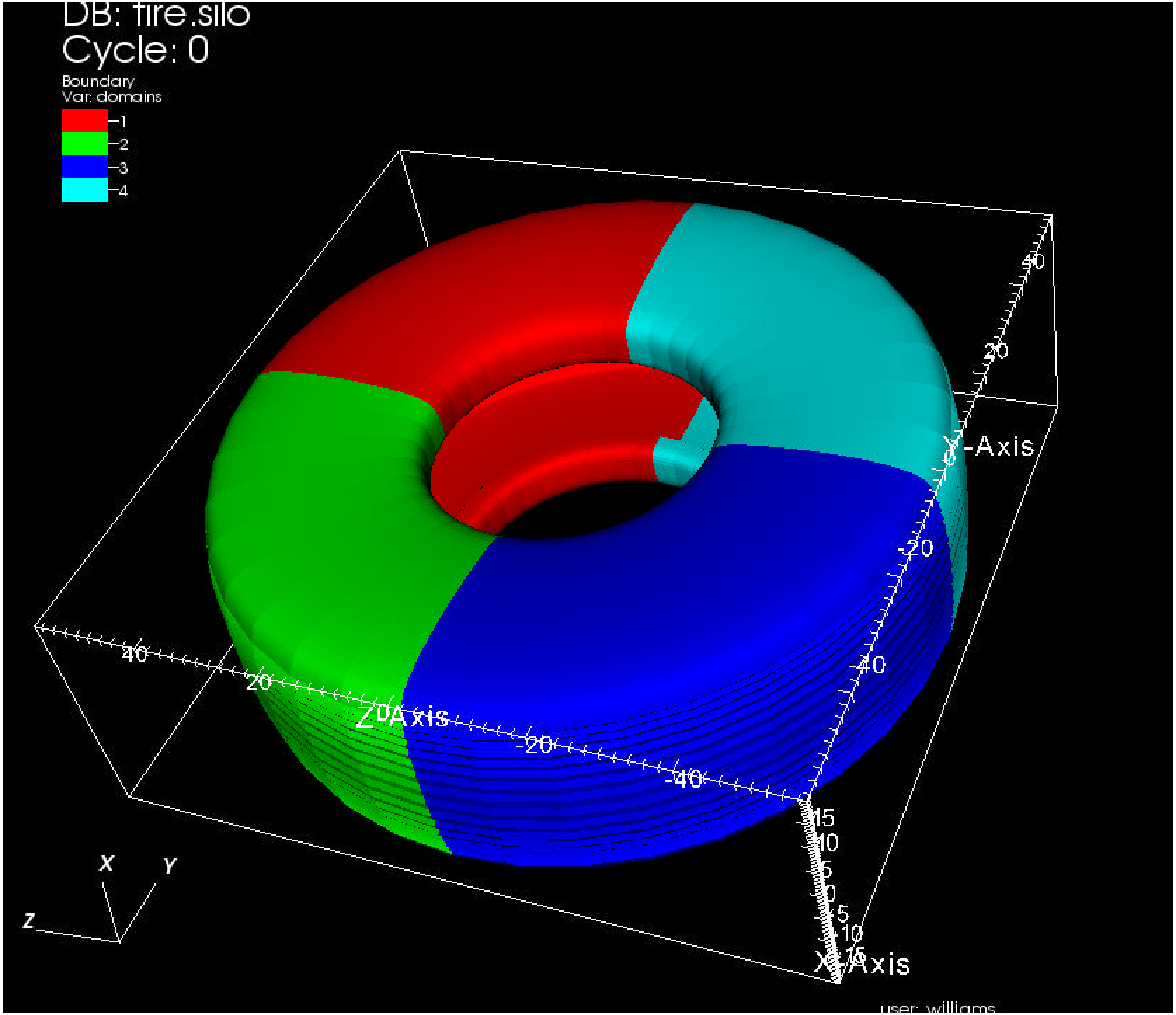
Box Clip of Region

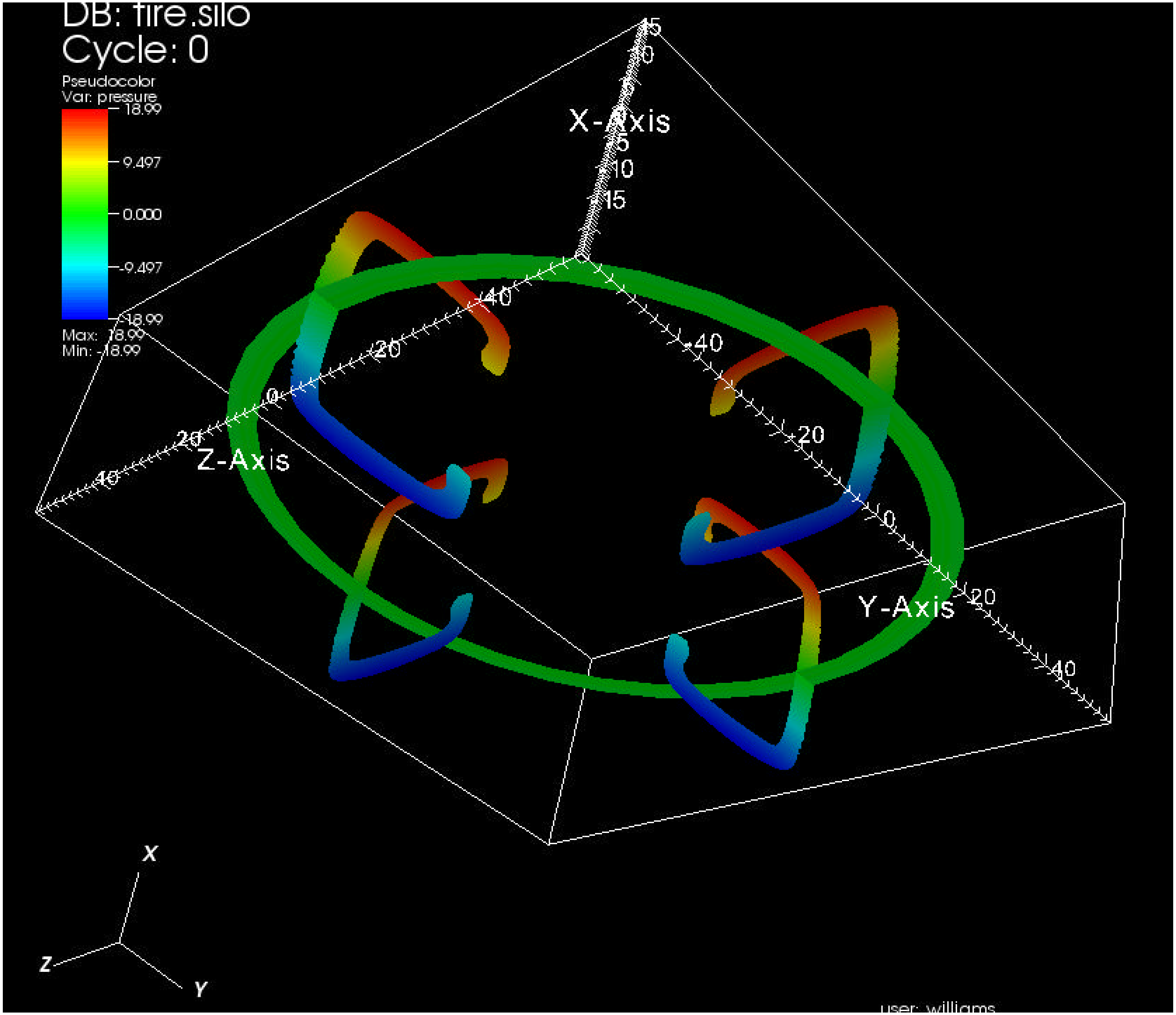
VisIt Operators: *filters applied to variable*

- **Isosurface: isosurface colored by different var.**
- **Lineout: extract 1D data from 2D or 3D plots.**
- **Onion Peel: grow image outwards in layers from seed cell.**
- **Reflect: reflect geometry across axes.**
- **Slice: 3D data mapped to 2D surface.**

VisIt Operators: *filters applied to variable*

- **Spherical Slice:** slice with sphere.
- **Three Slice:** 3 mutually perpendicular slices. (next 2 slides)
- Threshold: remove all cells not in specified data range.
- Roll Your Own: create your own operator plugin.


DB: fire.silo
Cycle: 0
Boundary
Var: domains
🟥 1
🟩 2
🟦 3
4
40
20
0
-20
-40
Z-Axis
40
20
0
-20
-40
Y-Axis
15
10
5
0
-5
-10
-15
X-Axis
X
Y
Z
user: williams


DB: fire.silo
Cycle: 0
Pseudocolor
Var: pressure
18.99
9.497
0.000
-9.497
-18.99
Max: 18.99
Min: -18.99
X-Axis
Y-Axis
Z-Axis
15
10
5
-5
-10
-15
40
20
0
-20
-40
X
Y
Z
user: williams

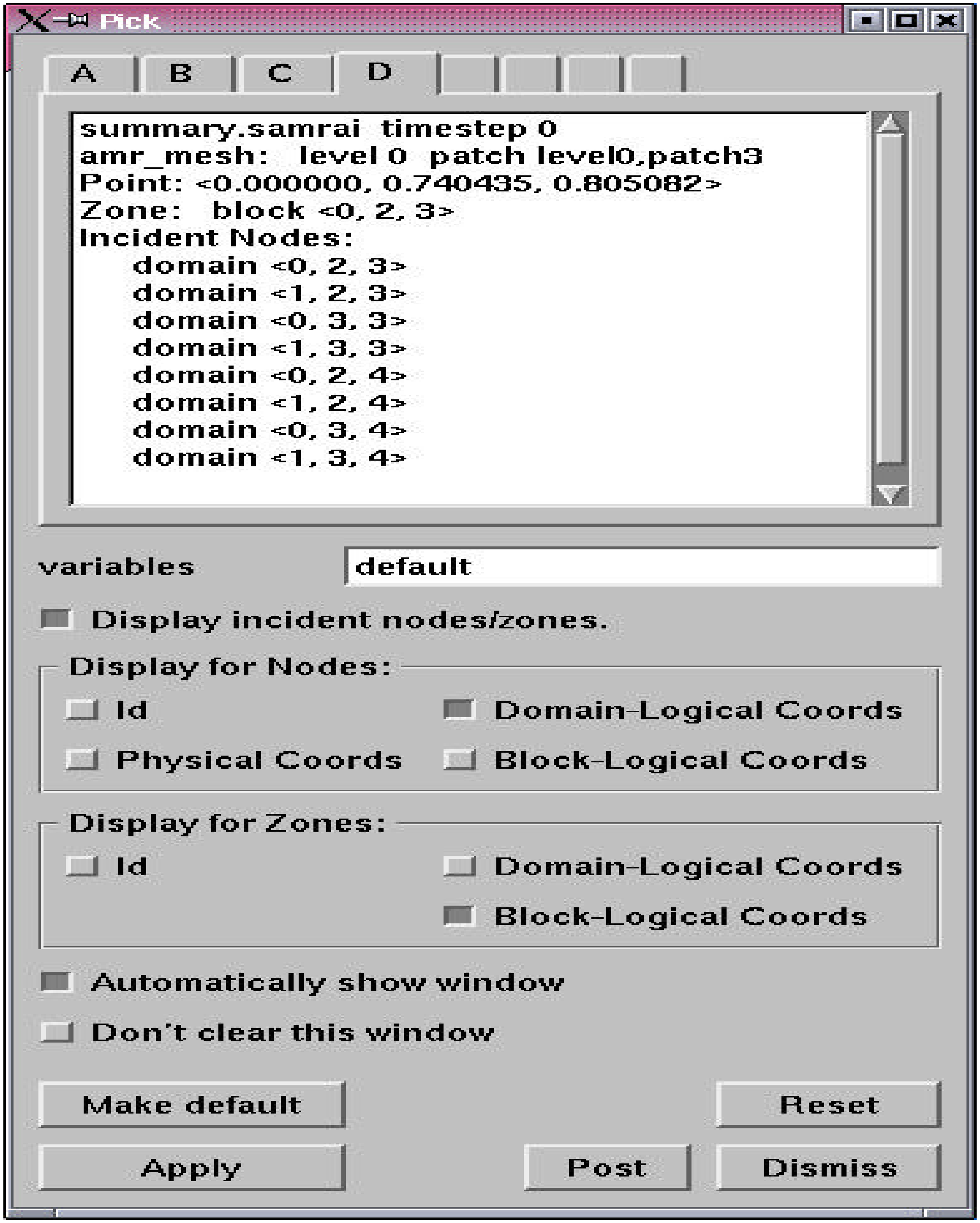
VisIt Operators: *filters applied to variable*

- **Spherical Slice:** slice with sphere.
- **Three Slice:** 3 mutually perpendicular slices.
- **Threshold:** remove all cells not in specified data range.
- **Roll Your Own:** create your own operator plugin.

Other VisIt Features

- **Animation**
- **Box, Sphere, Plane, Line & Point Tools**
- **Quantitative analysis**
- **Pick & Query – useful for finding numeric data value in specific cell. Gives cell number, coordinates, and data value.**

(see next slide)



```
summary.samrai  timestep 0
amr_mesh:  level 0  patch level0,patch3
Point: <0.000000, 0.740435, 0.805082>
Zone:  block <0, 2, 3>
Incident Nodes:
    domain <0, 2, 3>
    domain <1, 2, 3>
    domain <0, 3, 3>
    domain <1, 3, 3>
    domain <0, 2, 4>
    domain <1, 2, 4>
    domain <0, 3, 4>
    domain <1, 3, 4>
```

variables default

☑ Display incident nodes/zones.

Display for Nodes:

☐ Id ☑ Domain-Logical Coords

☐ Physical Coords ☐ Block-Logical Coords

Display for Zones:

☐ Id ☐ Domain-Logical Coords

☑ Block-Logical Coords

☑ Automatically show window

☐ Don't clear this window

Pick window

Shows (x,y,z) coords,
level & patch number
cell (i,j,k) indices
and indices of nodes.

If a variable is being
visualized, also shows
value of variable.

This work was performed under the auspices of the U.S. Department of Energy by University of California Lawrence Livermore National Laboratory under contract No. W-7405-Eng-48.