SCIRun Visualization Basics

SCIRun 4.3 Documentation

Center for Integrative Biomedical Computing
Scientific Computing & Imaging Institute
University of Utah

SCIRun software download:

http://software.sci.utah.edu

Center for Integrative Biomedical Computing:

http://www.sci.utah.edu/cibc

Supported by:

NIH grant P41- RR12553-07

Author(s):
Ayla Khan, David Weinstein

Contents

Chapter 1

SCIRun Overview

This tutorial demonstrates how to build a simple SCIRun dataflow network.

1.1 Software requirements

1.1.1 SCIRun 4.3

Download SCIRun version 4.3 and the SCIRunData archive from the SCI software portal. Make sure to update to the most up-to-date build available, which will include the latest bug fixes.

The easiest way to get started with SCIRun is to download and install a binary version for Windows or Mac OS X. Sources are available for Linux, however this option is recommended only for advanced Linux users.

Unpack the SCIRunData archive in a convenient location. Recall from the User Guide that the path to data can be set using the `SCIRUN_DATA` environment variable or by setting `SCIRUN_DATA` in the *.scirunrc* file.

Chapter 2

Simple Dataflow Network

Scope: Read Data File - Slice - Visualize - Bounding Box - Isosurface

2.1 Slice Field

2.1.1 Read Data File

Read and manipulate a structured mesh dataset. Create a **ReadField** module by going to the Modules menu and selecting the module from the **SCIRun DataIO** submenu. Recall from the User Guide that a module can be located in the Modules menu tree by entering all or part of it's name in the **Show Module** text input.

Navigate to the SCIRunData directory and select the dataset *volume/engine.nhdr*.

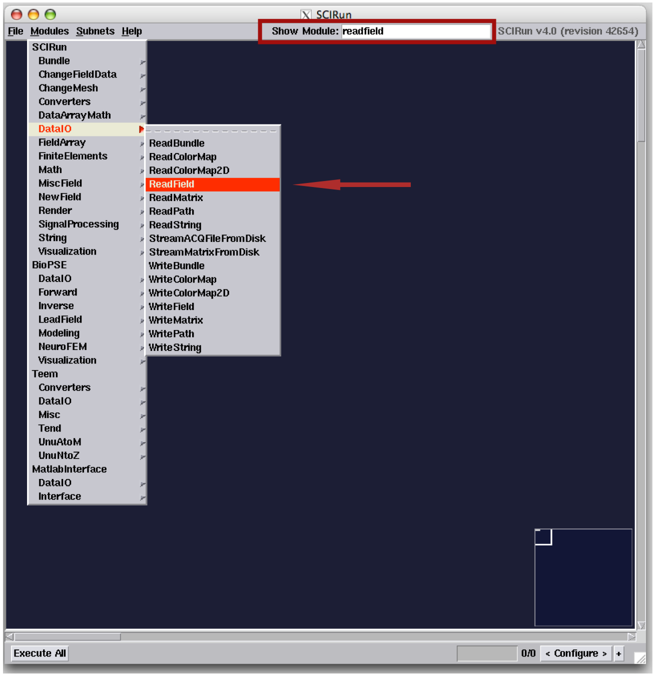
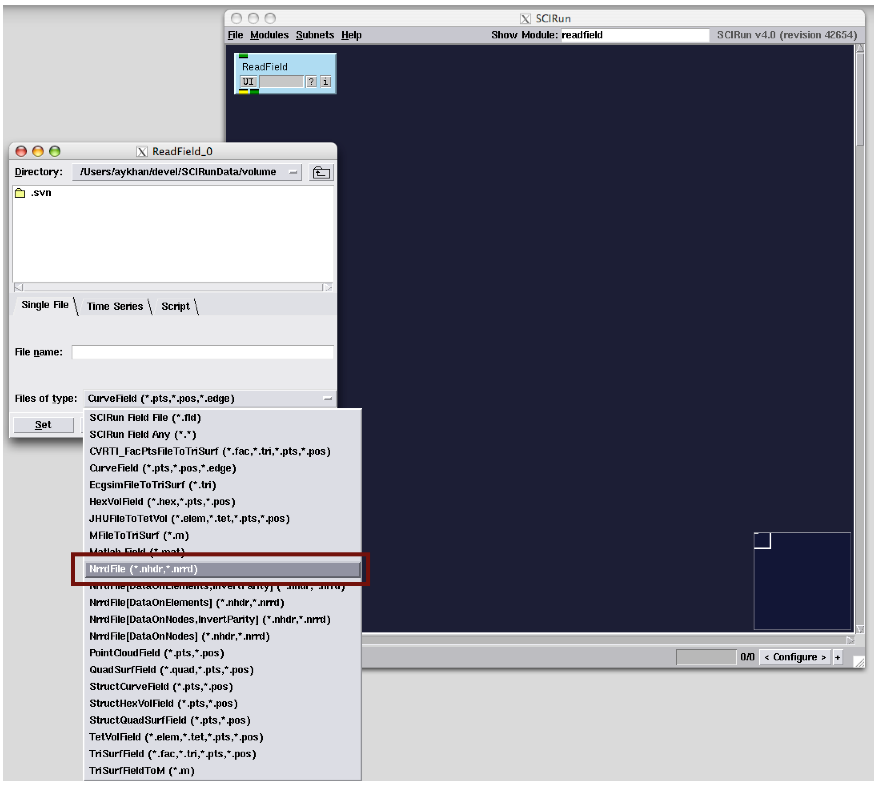

Show Module: readfield

Figure 2.1. Locate ReadField module.



Show Module: readfield

Directory: /Users/aykhan/devel/SCIRunData/volume

File name:

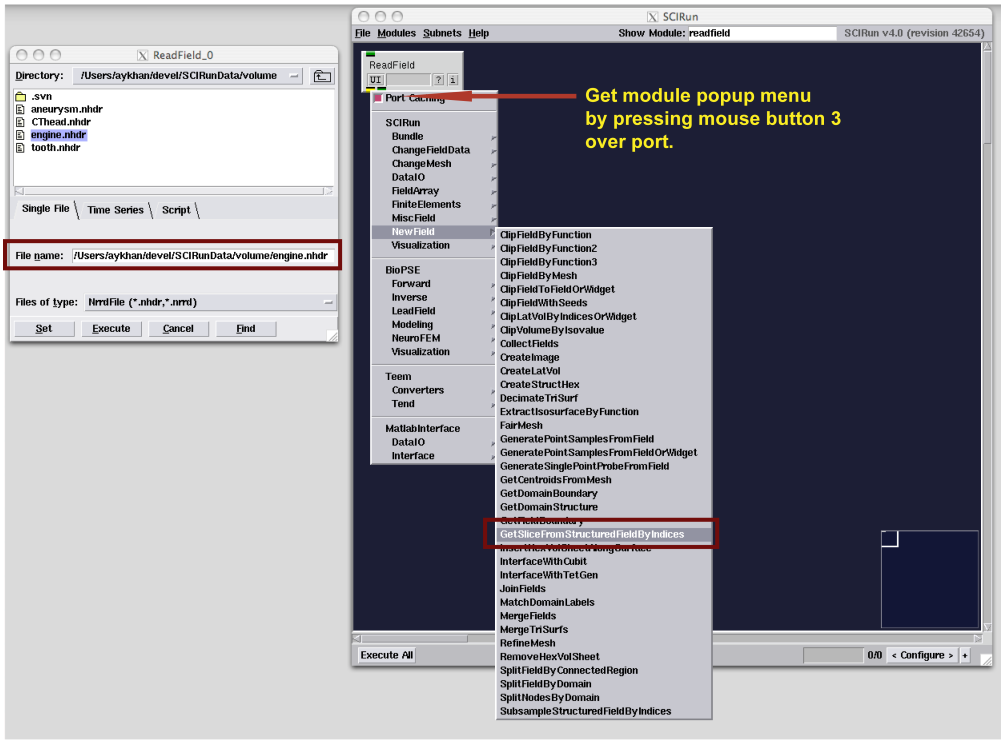
Files of type: CurveField (*.pts,*.pos,*.edge)

SCIRun
File
Modules
Subnets
Help
SCIRun v4.0 (revision 42654)
ReadField
UI
?
i
ReadField_0
.svn
Single File
Time Series
Script
Set
SCIRun Field File (*.fld)
SCIRun Field Any (*.*)
CVRTI_FacPtsFileToTriSurf (*.fac,*.tri,*.pts,*.pos)
CurveField (*.pts,*.pos,*.edge)
EcgsimFileToTriSurf (*.tri)
HexVolField (*.hex,*.pts,*.pos)
JHUFileToTetVol (*.elem,*.tet,*.pts,*.pos)
MFileToTriSurf (*.m)
Matlab Field (*.mat)
NrrdFile (*.nhdr,*.nrrd)
NrrdFile[DataOnElements,InvertParity] (*.nhdr,*.nrrd)
NrrdFile[DataOnElements] (*.nhdr,*.nrrd)
NrrdFile[DataOnNodes,InvertParity] (*.nhdr,*.nrrd)
NrrdFile[DataOnNodes] (*.nhdr,*.nrrd)
PointCloudField (*.pts,*.pos)
QuadSurfField (*.quad,*.pts,*.pos)
StructCurveField (*.pts,*.pos)
StructHexVolField (*.pts,*.pos)
StructQuadSurfField (*.pts,*.pos)
TetVolField (*.elem,*.tet,*.pts,*.pos)
TriSurfField (*.fac,*.tri,*.pts,*.pos)
TriSurfFieldToM (*.m)
0/0
< Configure >
+

Figure 2.2. ReadField can be used to read many data file formats.

2.1.2 Slice Field

Slice the engine field by node index along a given axis by instantiating the module **GetSlicesFromStructuredFieldByIndices** and connecting it to **ReadField**.



Directory: /Users/aykhan/devel/SCIRunData/volume

File name: /Users/aykhan/devel/SCIRunData/volume/engine.nhdr

Files of type: NrrdFile (*.nhdr,*.nrrd)

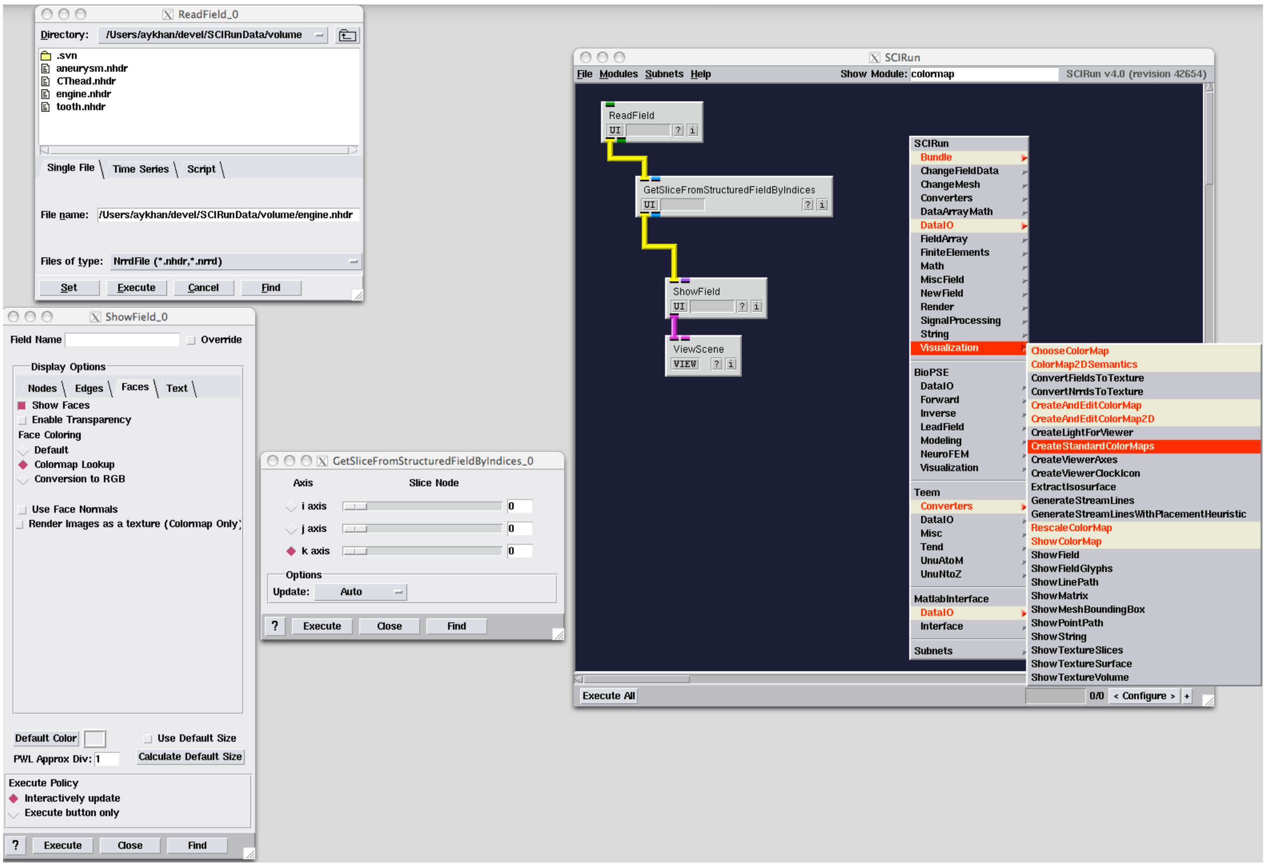
Show Module: readfield

Get module popup menu by pressing mouse button 3 over port.

Figure 2.3. Using the ReadField Field port's popup menu to instantiate GetSliceFromStructuredFieldByIndices.

2.1.3 Visualize Field

To visualize the field geometry, instantiate module **ShowField** in the **Visualization** category and module **ViewScene** in the **Render** category. ShowField takes a field as input, and outputs scene-graph geometry. ViewScene displays the geometry and allows a user to interact with the scene. Choose colors to apply to the geometry using **CreateStandardColorMaps** and **RescaleColorMaps** modules in textbfVisualization. Execute network and use the sliders in the GetSlicesFromStructuredFieldByIndices UI to change slice position (see figure 2.6).



Directory: /Users/aykhan/devel/SCIRunData/volume

File name: /Users/aykhan/devel/SCIRunData/volume/engine.nhdr

Files of type: NrrdFile (*.nhdr,*.nrrd)

Field Name:

☑ Show Faces

● Colormap Lookup

PWL Approx Div: 1

● Interactively update

i axis 0

j axis 0

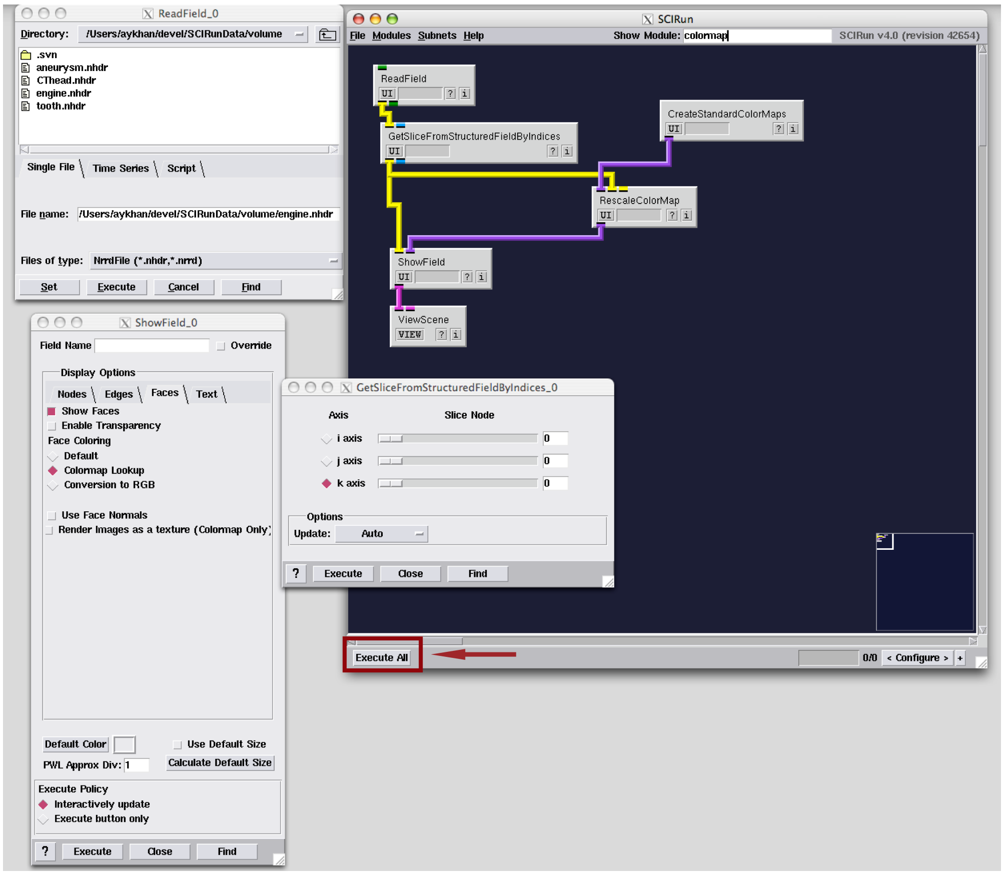
● k axis 0

Update: Auto

Show Module: colormap

SCIRun v4.0 (revision 42654)

Figure 2.4. SCIRun provides a useful set of default colormaps.


ReadField_0

Directory: /Users/aykhan/devel/SCIRunData/volume

- .svn
- aneurysm.nhdr
- CThead.nhdr
- engine.nhdr
- tooth.nhdr

File name: /Users/aykhan/devel/SCIRunData/volume/engine.nhdr

Files of type: NrrdFile (*.nhdr,*.nrrd)

SCIRun

Show Module: colormap

SCIRun v4.0 (revision 42654)

ShowField_0

Display Options: Faces

- ■ Show Faces
- ☐ Enable Transparency

Face Coloring:

- ○ Default
- ● Colormap Lookup
- ○ Conversion to RGB

- ☐ Use Face Normals
- ☐ Render Images as a texture (Colormap Only)

☐ Use Default Size

PWL Approx Div: 1

Execute Policy:

- ● Interactively update
- ○ Execute button only

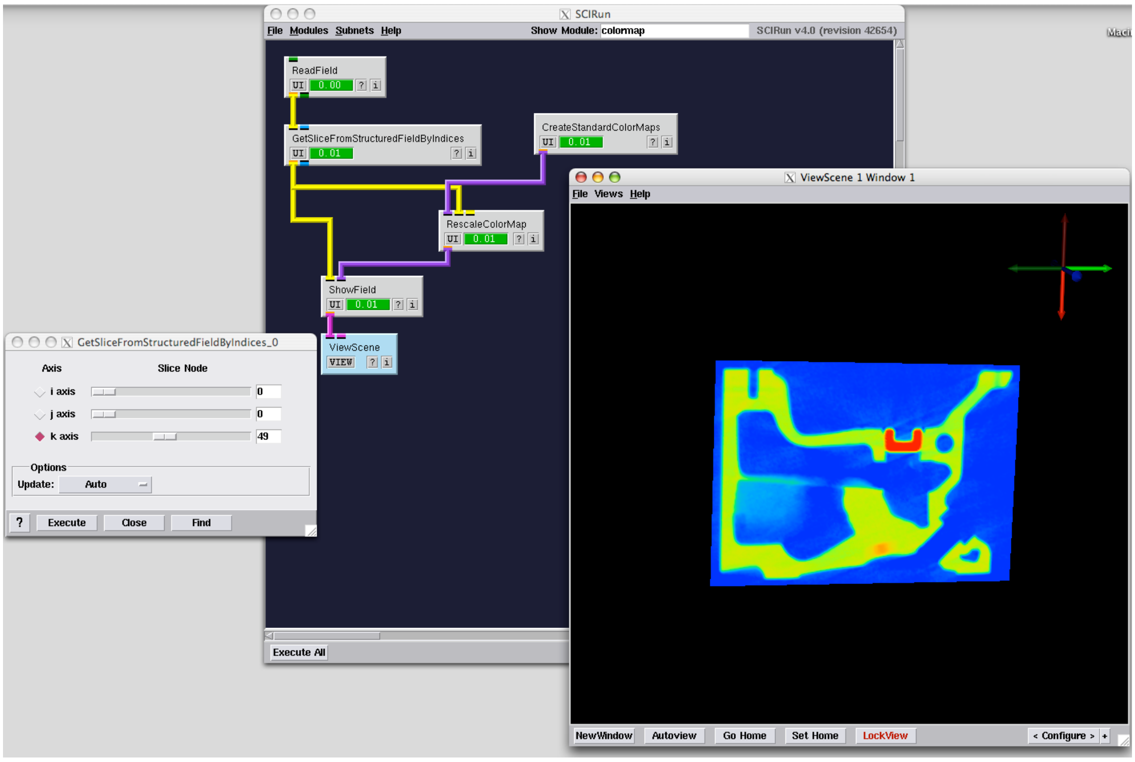
GetSliceFromStructuredFieldByIndices_0

Axis	Slice Node
○ i axis	0
○ j axis	0
● k axis	0

Options

Update: Auto

Figure 2.5. Rescale colormap to data values.



Show Module: colormap

i axis 0

j axis 0

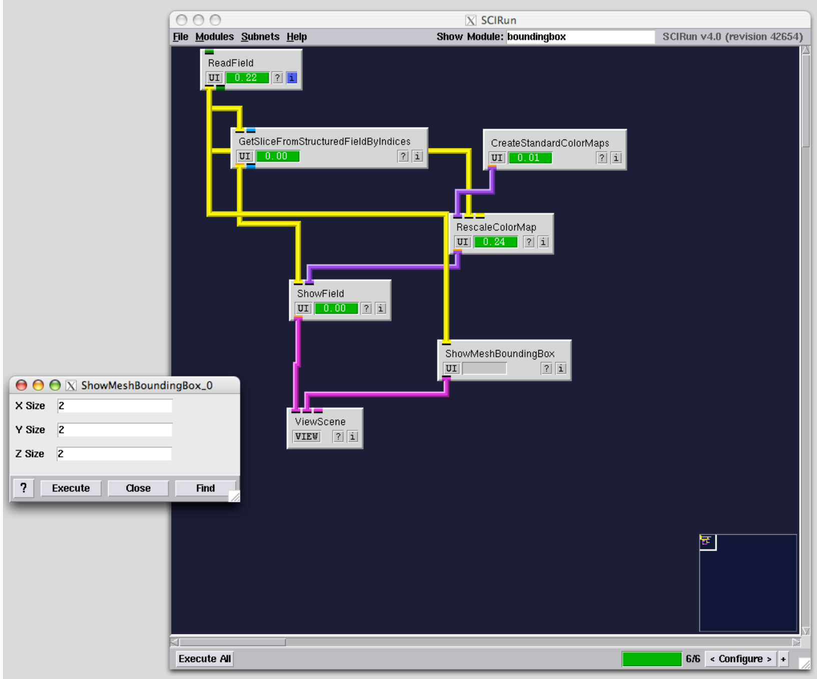
k axis 49

Update: Auto

Figure 2.6. Visualize the slice field’s geometry.

2.2 Show Bounding Box

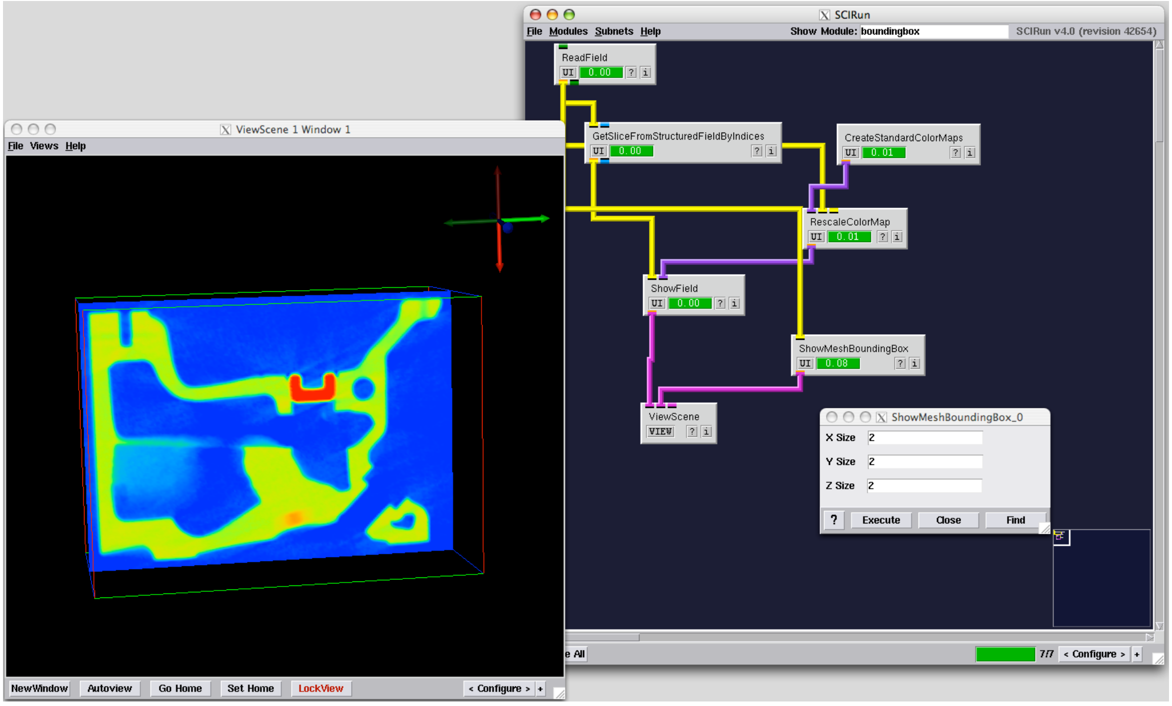
Add the **ShowMeshBoundingBox** module and change the number of lines in cage for each dimension. Execute the network.



Show Module: boundingbox

X Size 2
Y Size 2
Z Size 2

Figure 2.7. Configure bounding box cage.



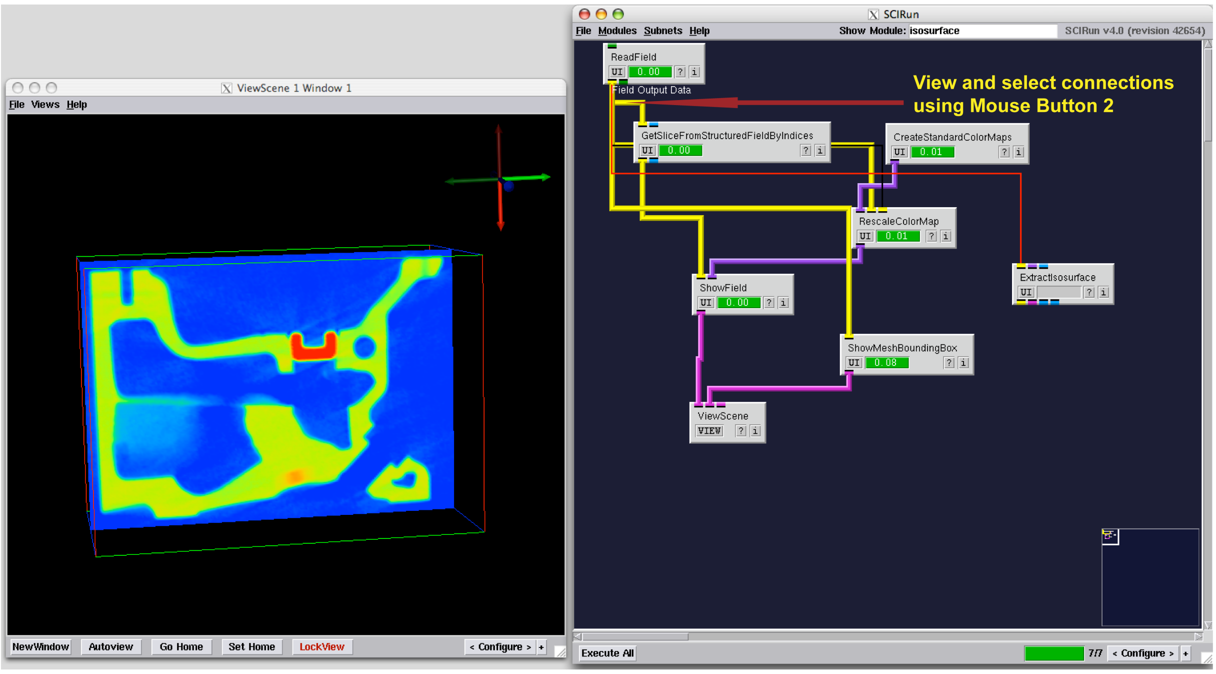
Show Module: boundingbox

X Size 2
Y Size 2
Z Size 2

Figure 2.8. View the mesh's bounding box.

2.3 Isosurface

Construct an isosurface from the field by instantiating and connecting an **Visualization** module **ExtractIsosurface** to ReadField (figure 2.9). Color isosurface output geometry by connecting the RescaleColorMap module. Execute the network and use the slider in the ExtractIsosurface UI to change the isovalue (see figure 2.11).



Show Module: isosurface

View and select connections using Mouse Button 2

Figure 2.9. Extract an isosurface from field.

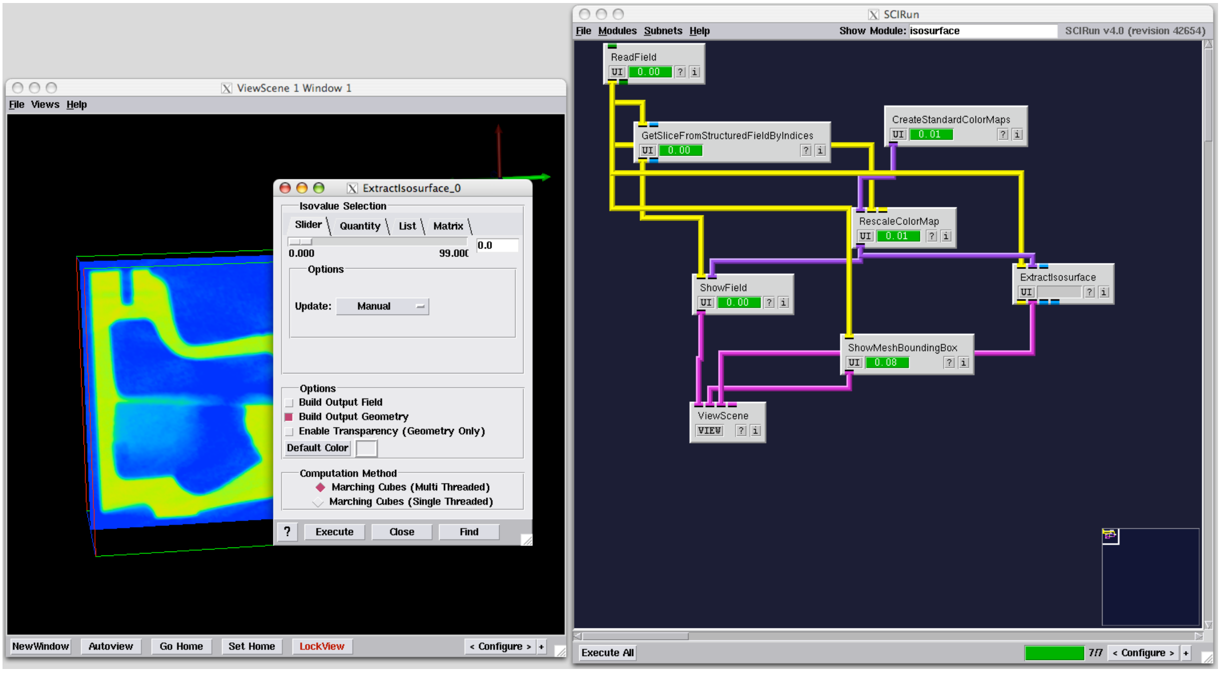
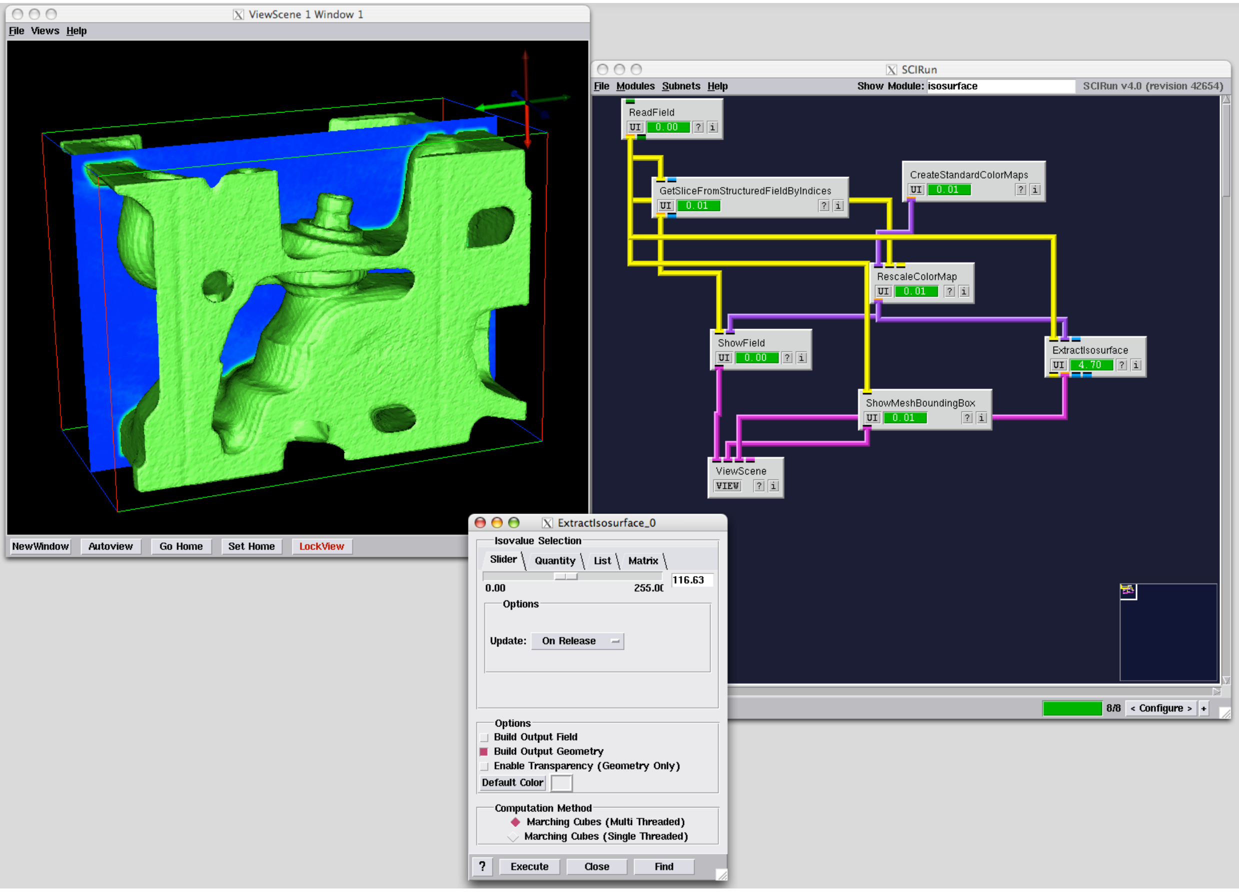


Figure 2.10. The ExtractIsosurface UI.



Show Module: isosurface

116.63

Update: On Release

☐ Build Output Field
☑ Build Output Geometry
☐ Enable Transparency (Geometry Only)

● Marching Cubes (Multi Threaded)
○ Marching Cubes (Single Threaded)

Figure 2.11. Visualize the isosurface.

Chapter 3

Create, Manipulate and Visualize Field

Scope: Generate Lattice Volume - Isosurface - Visualize Geometry

3.1 Create Field

Create and manipulate a structured mesh type in this exercise. Start by creating a lattice volume using CreateLatVol module. Assign data at nodes using CalculateFieldData module. Connect CalculateFieldData to CreateLatVol and use the expression $RESULT = sqrt(X * X + Y * Y + Z * Z)$ to compute data for each node.

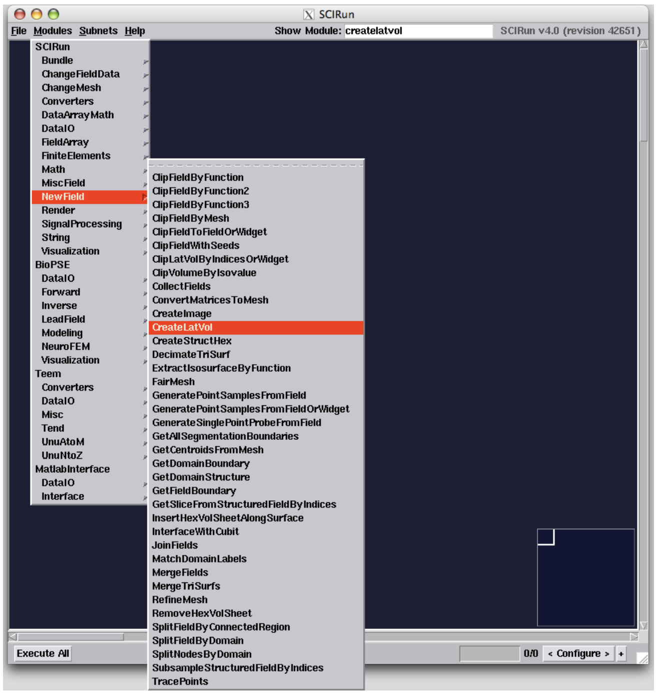
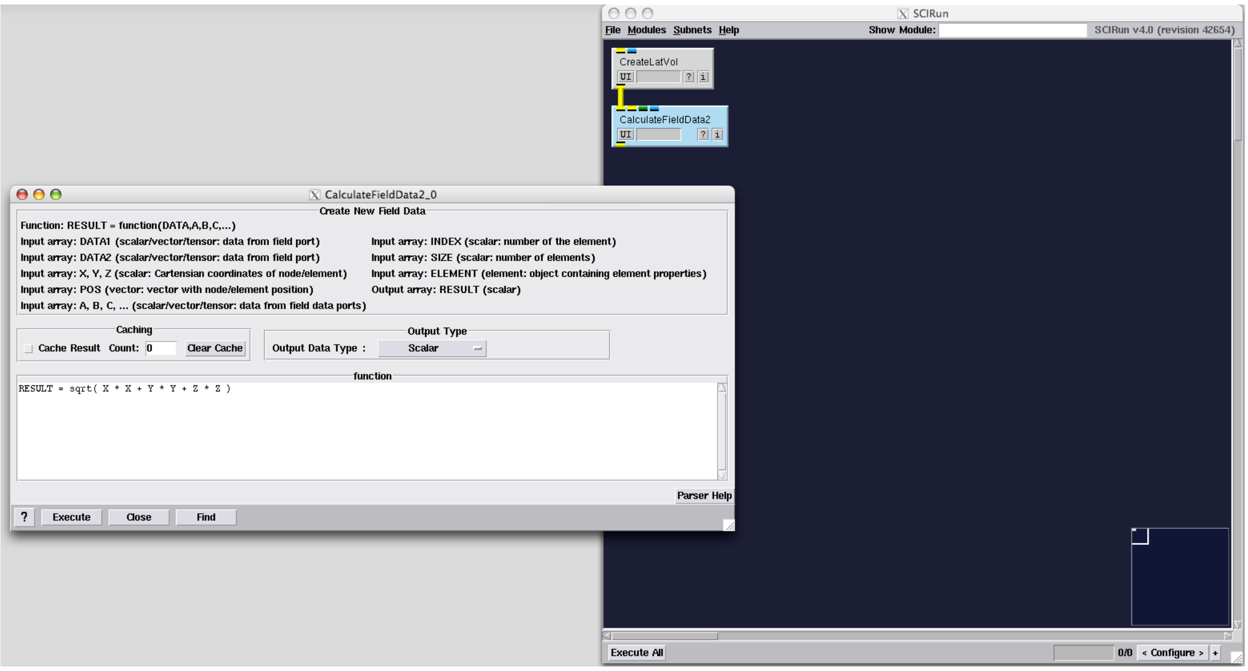


Figure 3.1. Create lattice volume field.



Create New Field Data

Function: RESULT = function(DATA,A,B,C,...)
Input array: DATA1 (scalar/vector/tensor: data from field port)
Input array: DATA2 (scalar/vector/tensor: data from field port)
Input array: X, Y, Z (scalar: Cartensian coordinates of node/element)
Input array: POS (vector: vector with node/element position)
Input array: A, B, C, ... (scalar/vector/tensor: data from field data ports)
Input array: INDEX (scalar: number of the element)
Input array: SIZE (scalar: number of elements)
Input array: ELEMENT (element: object containing element properties)
Output array: RESULT (scalar)

Caching
☐ Cache Result Count: 0

Output Type
Output Data Type : Scalar

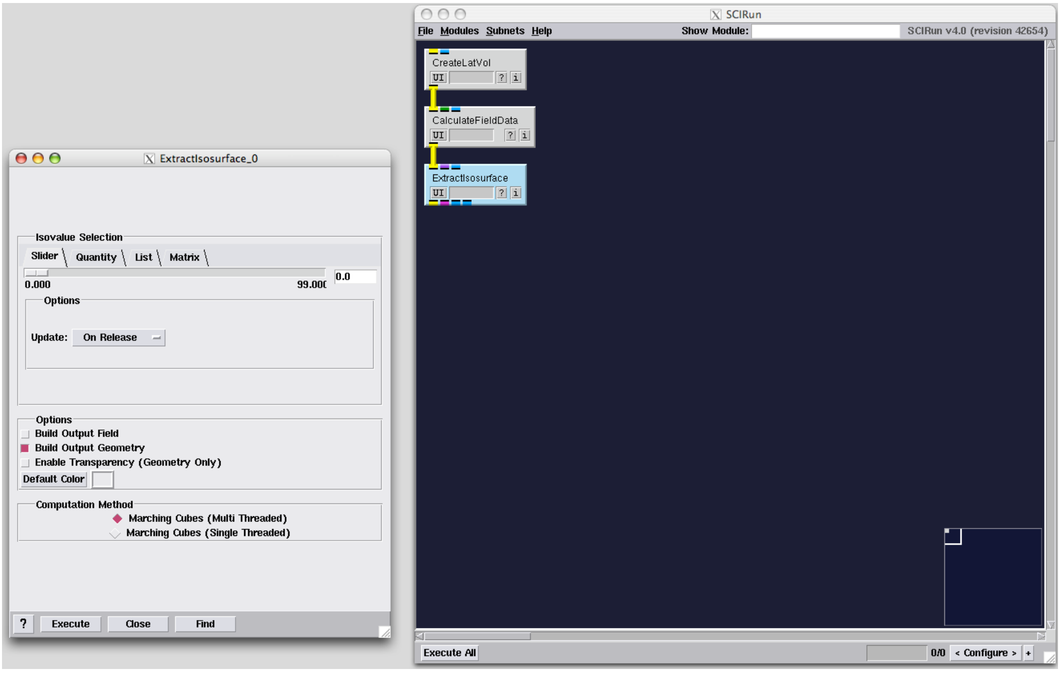
function

```
RESULT = sqrt( X * X + Y * Y + Z * Z )
```

Figure 3.2. Calculate node data.

3.2 Isosurface

Generate the isosurface by instantiating and connecting an ExtractIsosurface module to CalculateFieldData. Complete the network by adding a colormap. Execute the network and show 4x4x4 mesh bounding box as in section 2.2.



0.0

Update: On Release

☐ Build Output Field
☑ Build Output Geometry
☐ Enable Transparency (Geometry Only)

● Marching Cubes (Multi Threaded)
○ Marching Cubes (Single Threaded)

Figure 3.3. Extract an isosurface from the field data.

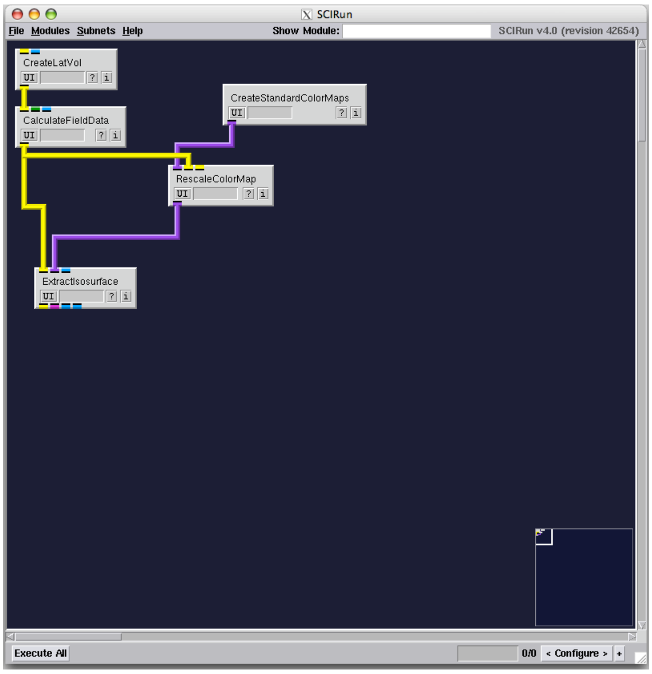
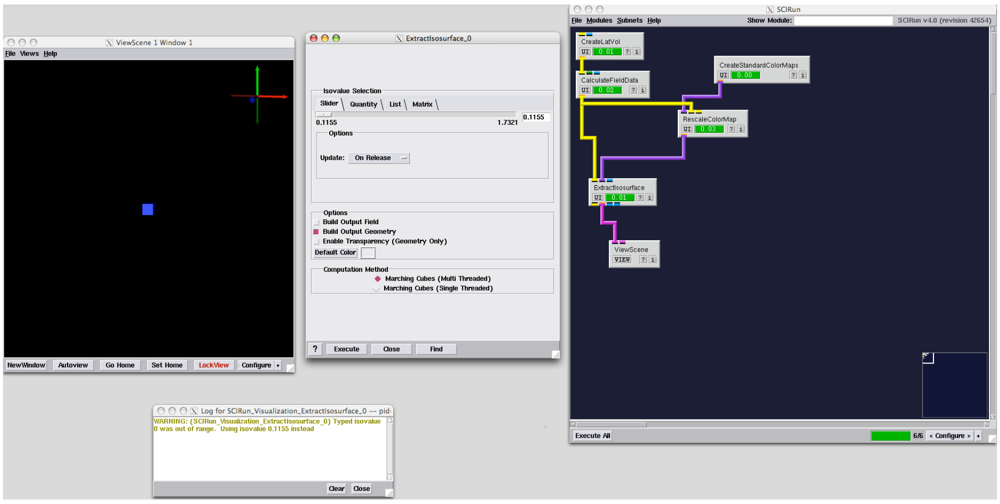


Figure 3.4. Map color to the isovalue.

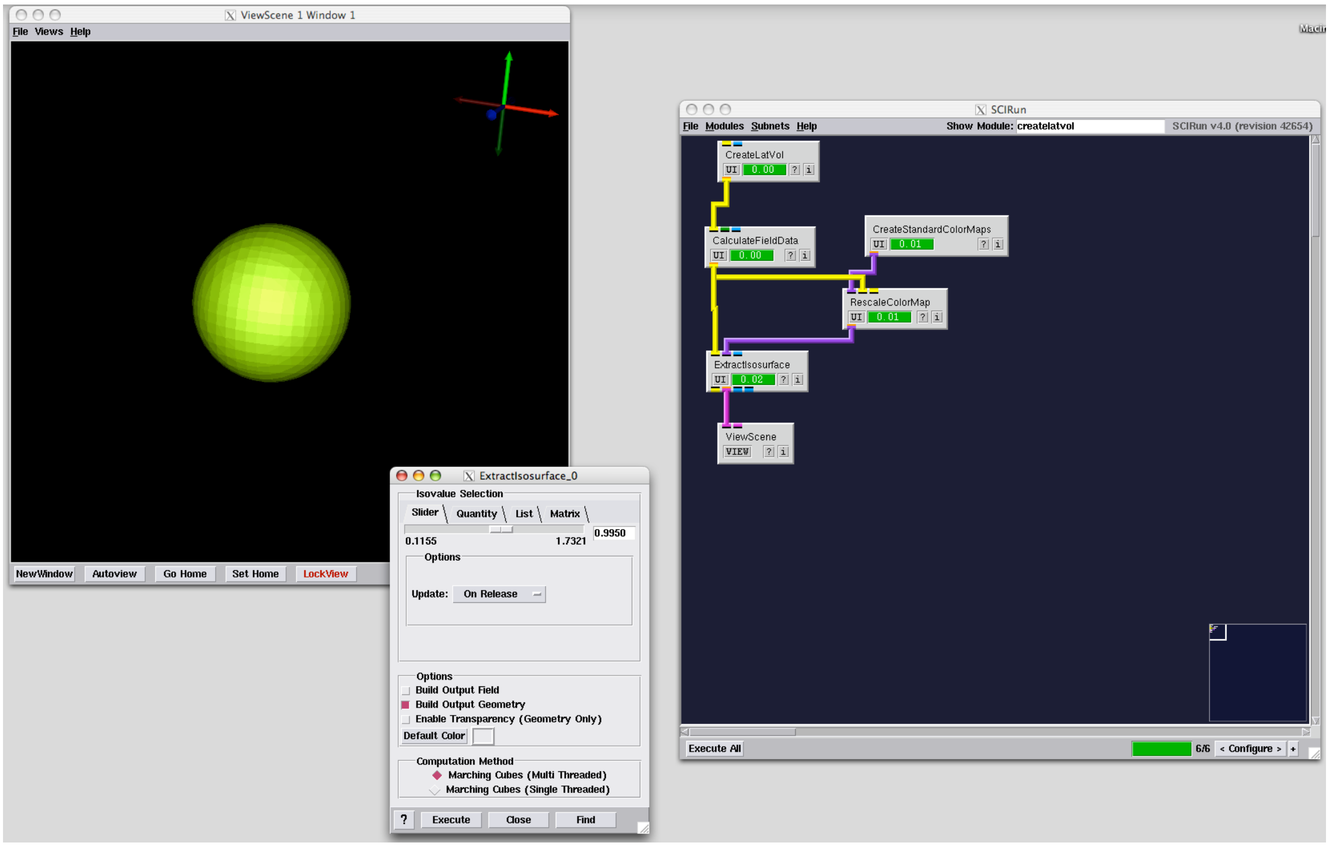


Isovalue: 0.1155

Update: On Release

```
WARNING: (SCIRun_Visualization_ExtractIsosurface_0) Typed isovalue
0 was out of range.  Using isovalue 0.1155 instead
```

Figure 3.5. Change the default isovalue.

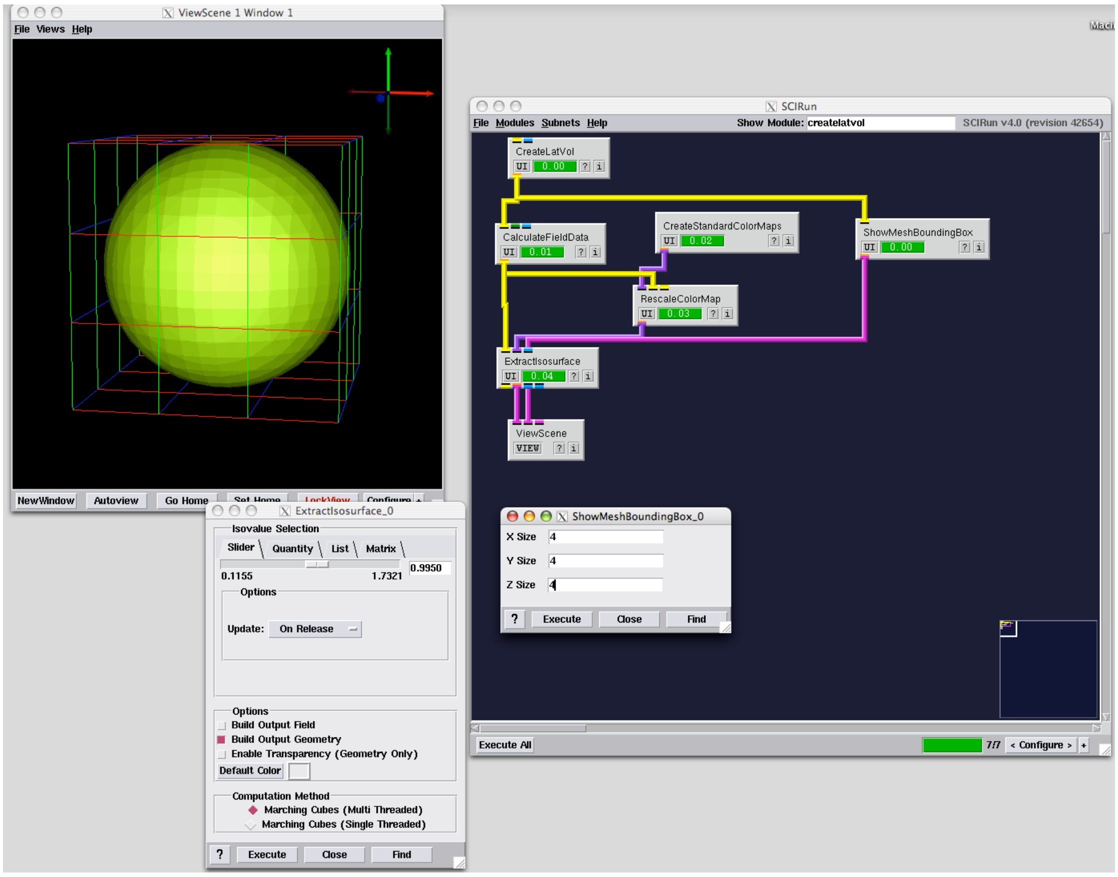


Isovalue: 0.9950

Update: On Release

Show Module: createlatvol

Figure 3.6. Visualize the isosurface.



X Size 4

Y Size 4

Z Size 4

Isovalue: 0.9950

Update: On Release

☐ Build Output Field

☑ Build Output Geometry

☐ Enable Transparency (Geometry Only)

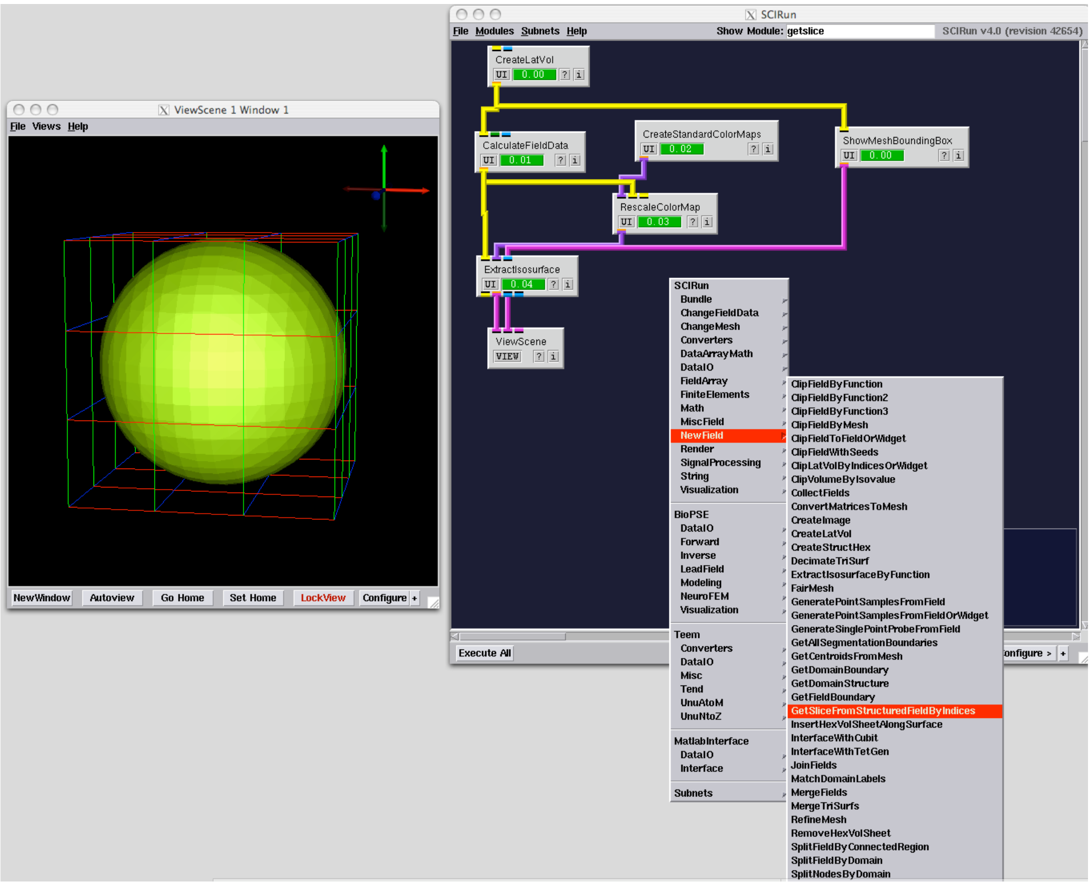
● Marching Cubes (Multi Threaded)

○ Marching Cubes (Single Threaded)

Figure 3.7. Show the field's bounding box.

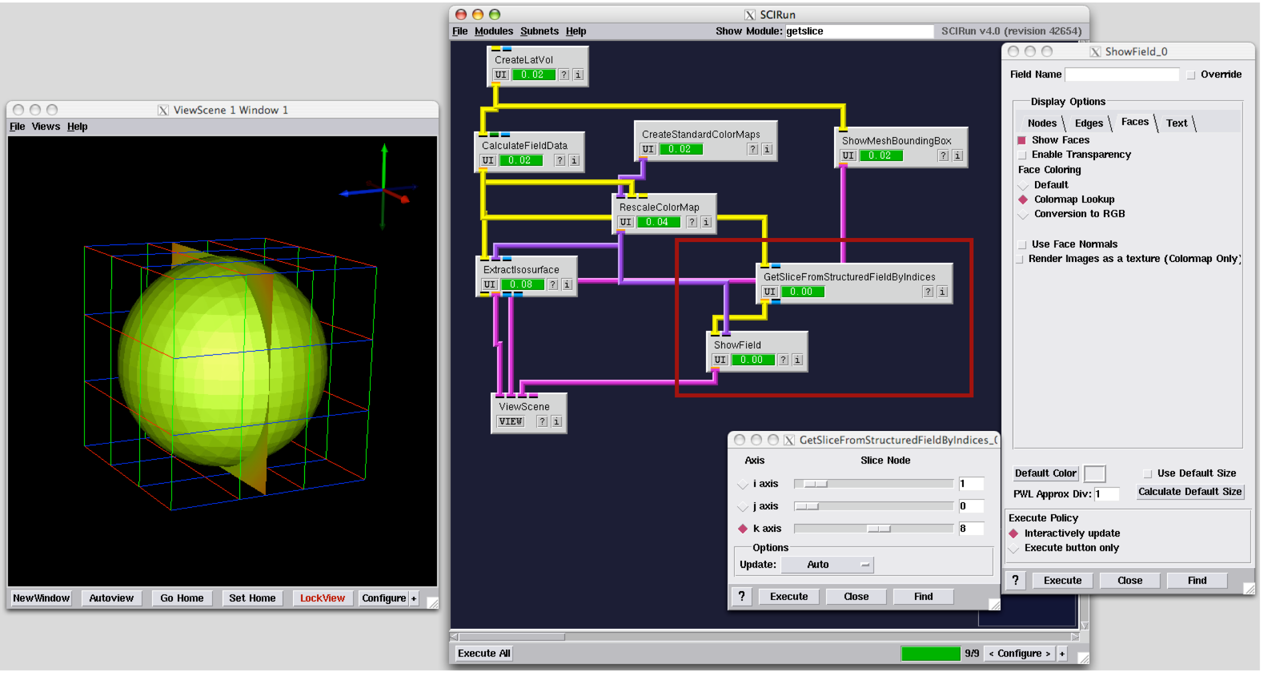
3.3 Slice Field

Extend the functionality of this network by slicing the field using GetSliceFromStructuredFieldByIndices as in section 2.1.2.



Show Module: getslice

Figure 3.8. Using the SCIRun Network Editor's popup menu to instantiate GetSliceFromStructuredFieldByIndices.



○ i axis 1
○ j axis 0
● k axis 8
Update: Auto

● Show Faces
● Colormap Lookup
PWL Approx Div: 1
● Interactively update

Figure 3.9. Use ShowField to visualize the slice field.

3.4 Clip Field

Clip out a subset of the original field by converting the lattice volume to an unstructured mesh using **ConvertMeshToUnstructuredMesh** (figure 3.10) and adding **ClipFieldByFunction** (figure 3.11) to the network. Set the clipping location setting in ClipFieldByFunction to *all nodes*. Use the expression $DATA > 1\&\&X < 0$ to clip the field.

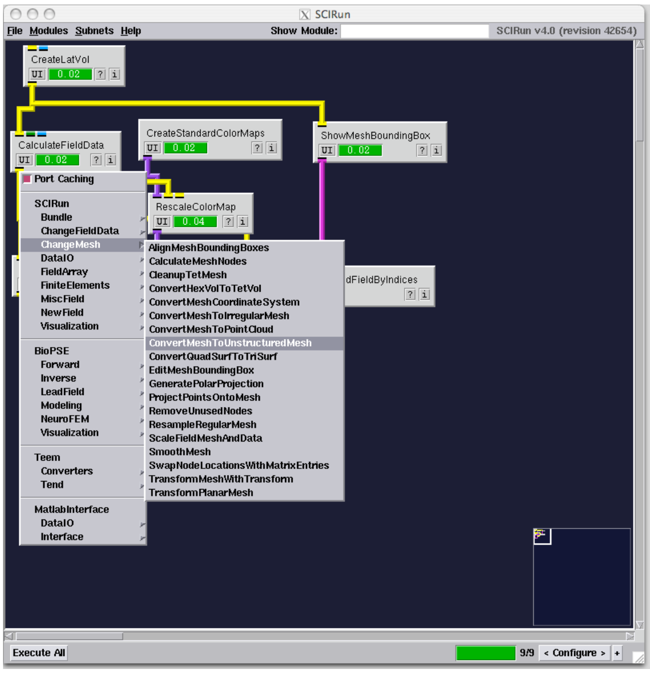


Figure 3.10. Convert the original field to an unstructured mesh.

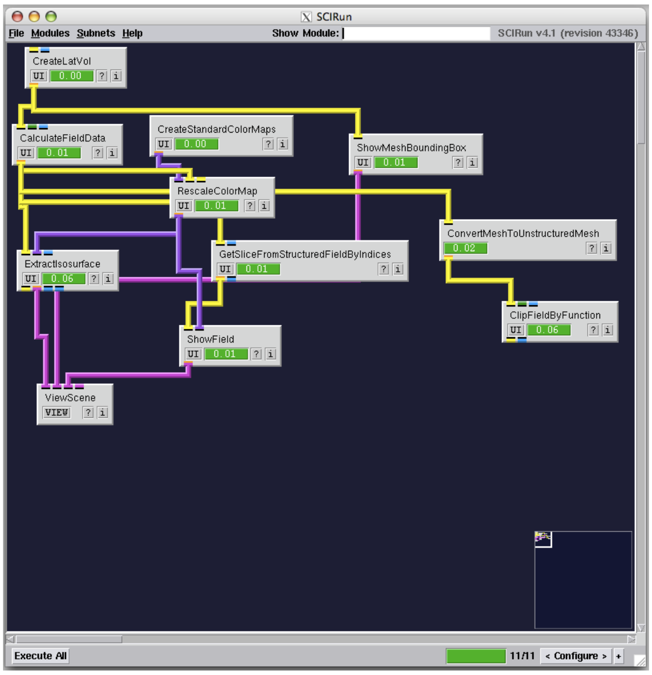
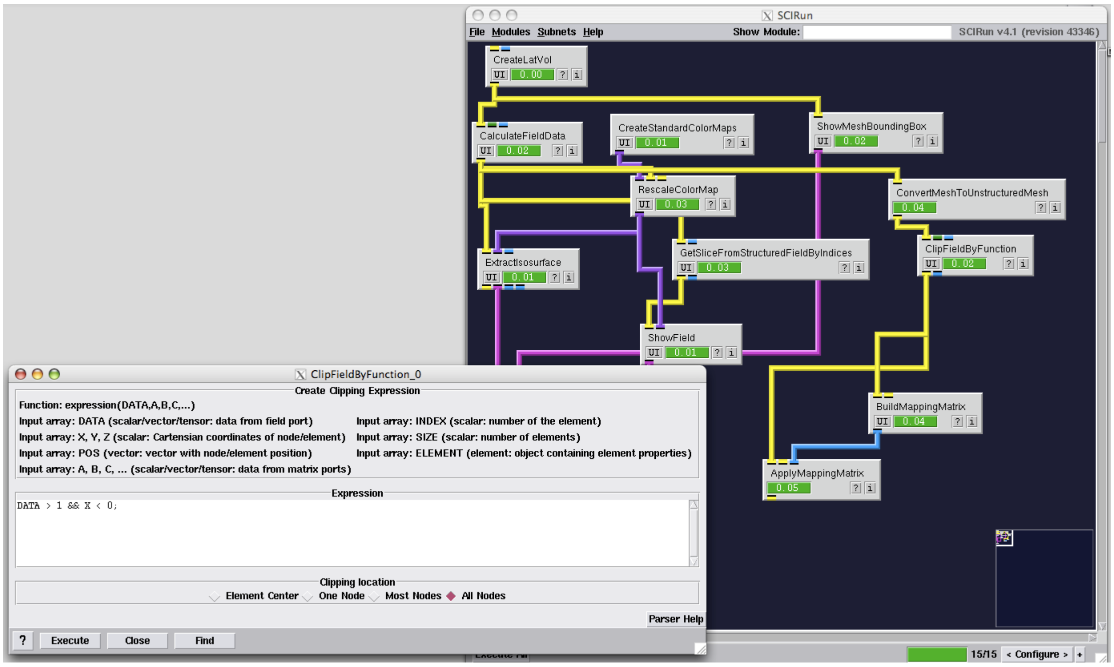


Figure 3.11. Clip field.

3.4.1 Extract Boundary

Use GetFieldBoundary to extract the boundary surface from the lattice volume. At this point, it will be necessary to map the fields by interpolating the the boundary surface field to the clipping field. First, use **BuildMappingMatrix** to build a matrix that maps a linear combination of data values in the clipping field to a value in the boundary field, then use **ApplyMappingMatrix** to multiply the data vector of the clipping field with the mapping matrix to obtain the data vector for the boundary surface field.



Function: expression(DATA,A,B,C,...)

Input array: DATA (scalar/vector/tensor: data from field port)

Input array: X, Y, Z (scalar: Cartensian coordinates of node/element)

Input array: POS (vector: vector with node/element position)

Input array: A, B, C, ... (scalar/vector/tensor: data from matrix ports)

Input array: INDEX (scalar: number of the element)

Input array: SIZE (scalar: number of elements)

Input array: ELEMENT (element: object containing element properties)

```
DATA > 1 && X < 0;
```

Clipping location: ◇ Element Center ◇ One Node ◇ Most Nodes ◆ All Nodes

Figure 3.12. Build and apply mapping.

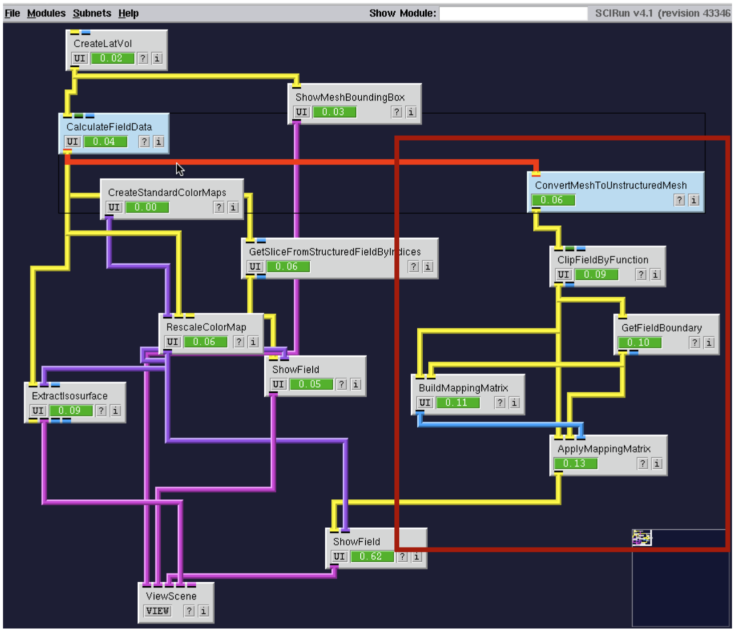
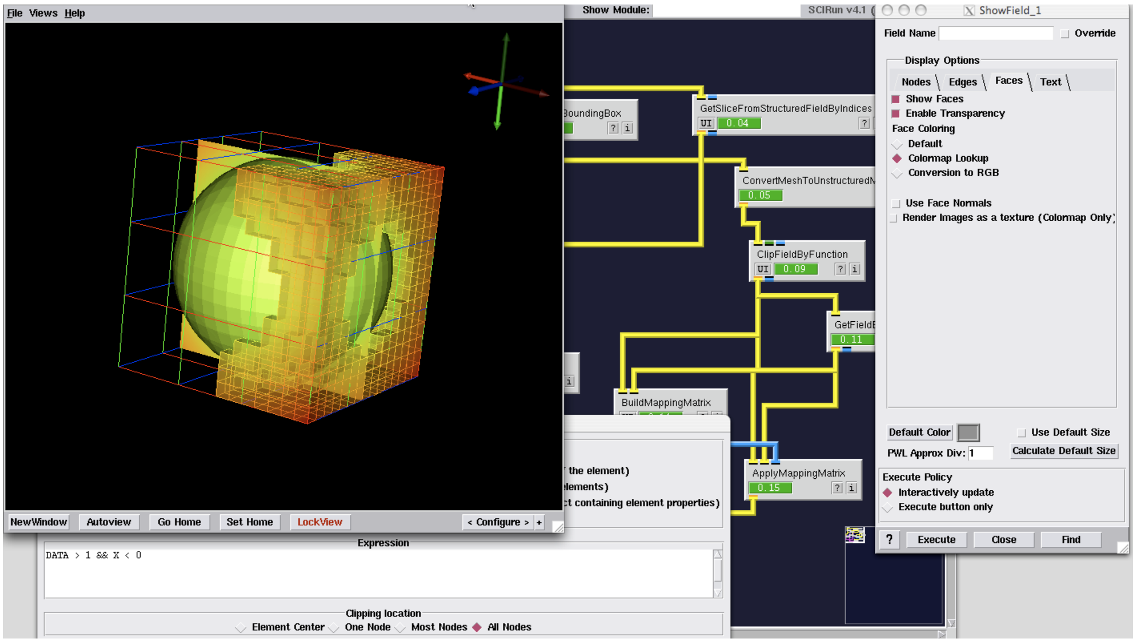


Figure 3.13. Add mapping matrix to network.



Expression

```
DATA > 1 && X < 0
```

Clipping location: ○ Element Center ○ One Node ○ Most Nodes ● All Nodes

Field Name ☐ Override

■ Show Faces
■ Enable Transparency
Face Coloring
○ Default
● Colormap Lookup
○ Conversion to RGB
☐ Use Face Normals
☐ Render Images as a texture (Colormap Only)

☐ Use Default Size
PWL Approx Div: 1

Execute Policy
● Interactively update
○ Execute button only

Figure 3.14. Visualize field and boundary.

Finally, it is not strictly necessary to explicitly convert the original mesh to an unstructured mesh using ConvertMeshToUnstructuredMesh because ClipFieldByFunction can implicitly convert structured mesh types to unstructured mesh types before clipping the field. As a final exercise, delete ConvertMeshToUnstructuredMesh from the network and try to obtain the same result.