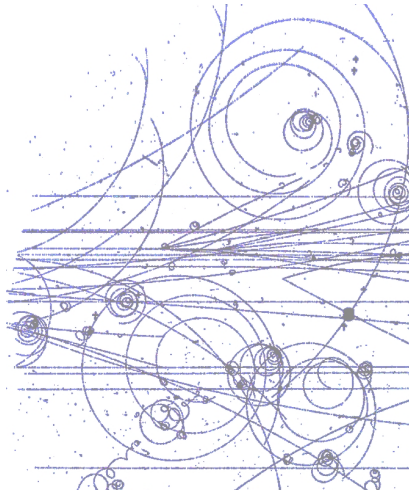
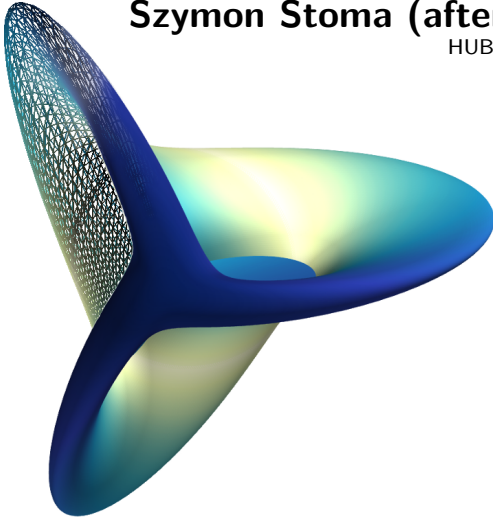


Introduction to Mayavi2

Szymon Stoma (after Gaël Varoquaux)

HUB



1 Data source, filters, and visualization modules

2 A component model

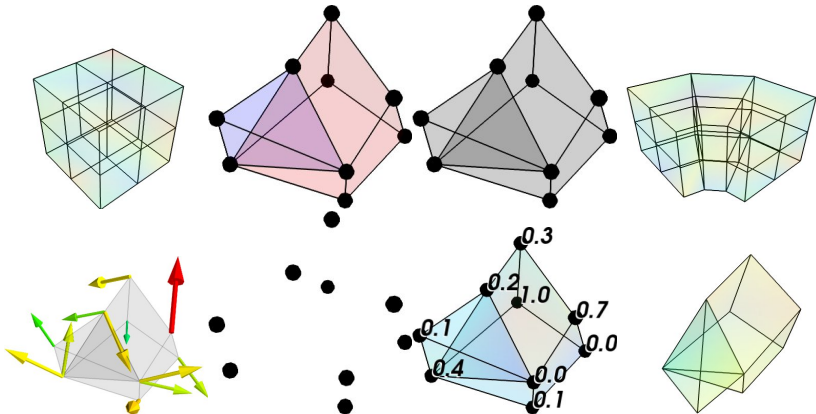
3 The `mlab_source` attribute

1 Data source, filters, and visualization modules

1 What: data sources

Tell me what your data looks like

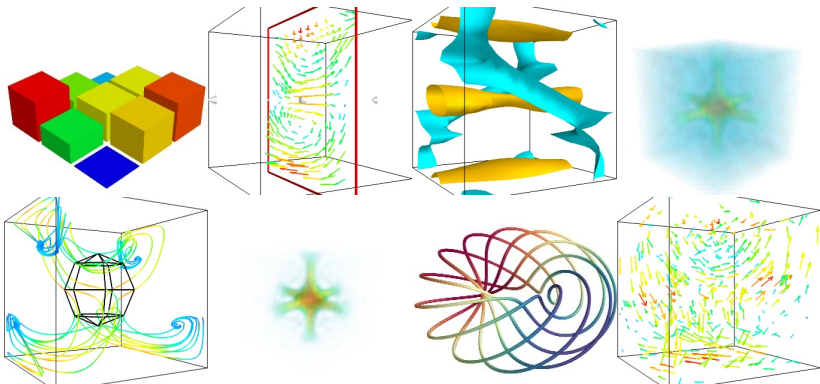
- Connected, or not?
- Regularly sampled or not?
- Vectors, scalars, both?



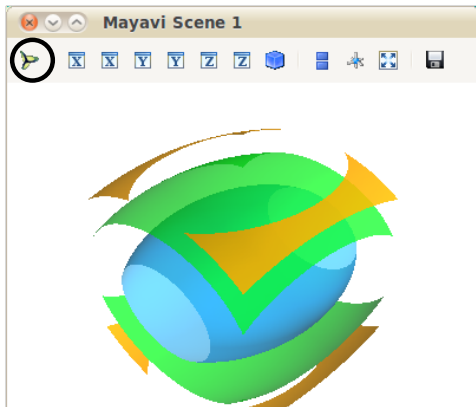
1 How: visualizations modules

Tell me how you want it represented

- On a plane?
- With arrows, or lines?
- Isosurfaces, or volumetric?



1 mlab: simple scripting



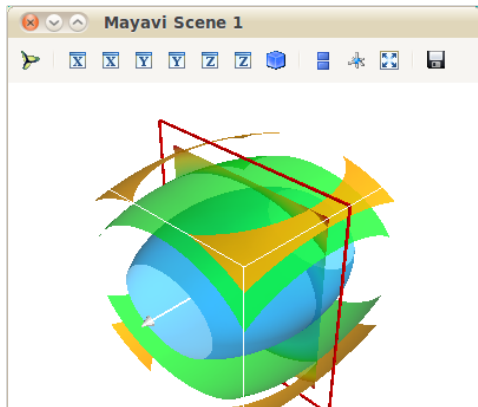
- Mayavi Scene 2
 - ScalarField
 - Colors and legends
 - IsoSurface

ipython -wthread

```
x, y, z = np.ogrid[-5:5:100j, -5:5:100j, -5:5:100j]
scalars = x*x*0.5 + y*y + z*z*2.0
from enthought.mayavi import mlab
iso = mlab.contour3d(scalars, transparent=True)
```

x

1 mlab.pipeline: scripting with power



Adding a cut plane

- ▼ Mayavi Scene 2
 - ▼ ScalarField
 - ▼ Colors and legends
 - ▲ IsoSurface
 - ▲ ScalarCutPlane

```
x, y, z = np.ogrid[-5:5:100j, -5:5:100j, -5:5:100j]
scalars = x*x*0.5 + y*y + z*z*2.0
from enthought.mayavi import mlab
iso = mlab.contour3d(scalars, transparent=True)

mlab.pipeline.scalar_cut_plane(iso)
```

x

1 mlab: hands on exercise

```
varoquau@resting: ~/Projects/Python_talks/EuroScipy
resting ~/Projects/Python_talks/EuroScipy $ ipython -wthread -no

In [1]: import numpy as np

In [2]: u, v, w = np.load('field.npy')

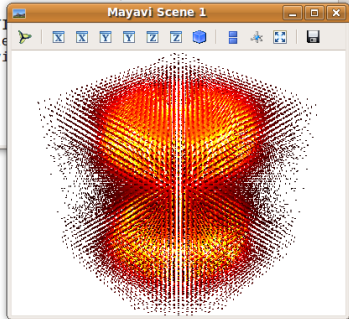
In [3]: u.shape, v.shape, w.shape
Out[3]: ((31, 31, 31), (31, 31, 31), (31, 31, 31))

In [4]: from enthought.mayavi import mlab

In [5]: mlab.quiver3d(u, v, w, colormap='hot')

(python:23053): Gtk-CRITICAL: Glib: g_object_unref: assertion 'G_OBJECT_IS_VALID (widget)' failed
Out[5]: <enthought.mayavi

In [6]: □
```

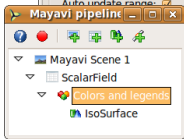
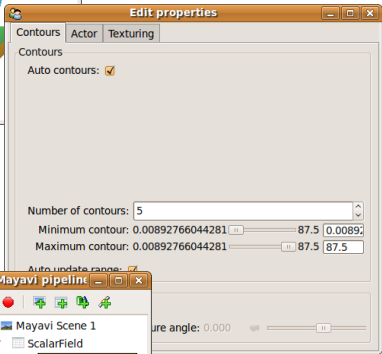
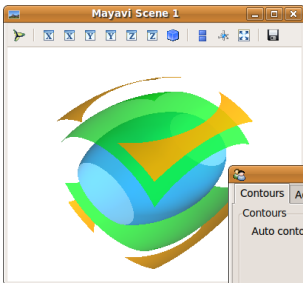


1. 'quiver' visualization
2. Isosurfaces of the intensity
3. Cut plane of the vector field

2 A component model

2 A set of live components

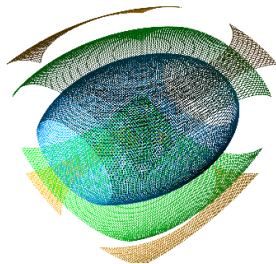
- `mlab.figure()`
- `iso.edit_traits()`
- `mlab.show_pipeline(rich_view=False)`



2 Modifying properties

`iso.actor.property.representation = 'points'`

How to find which attributes to modify?



The record button

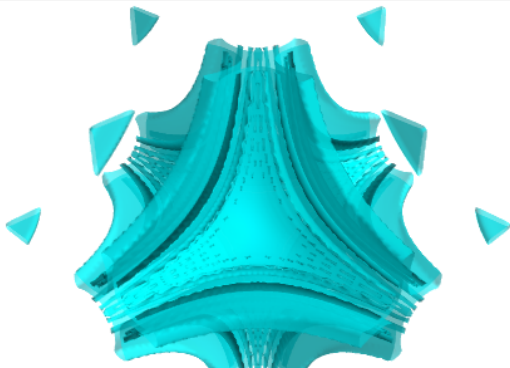
The image shows a screenshot of the Mayavi software interface. On the left, the 'Pipeline' view shows a tree structure with 'Mayavi Scene 2' containing 'ScalarField' and 'Colors and legends', which includes 'IsoSurface'. On the right, the 'Edit properties' dialog box is open, showing a 'Recording' checkbox and a code editor with the following Python code:

```
7 engine = Engine()
8 engine.start()
9 if len(engine.scenes) == 0:
10     engine.new_scene()
11 # -----
12 iso_surface = engine.scenes[0].children[0].children[0].c
13 iso_surface.actor.property.representation = 'wireframe'
14 iso_surface.actor.property.representation = 'points'
15 # -----
16 from enthought.mayavi.tools.show import show
17 show()
18
```

Below the code editor is a 'Save Script' button and an 'OK' button. At the bottom of the dialog, the 'Property' section shows 'Representation' set to 'points' and a 'Color' selection tool.

3 The `mlab_source` attribute

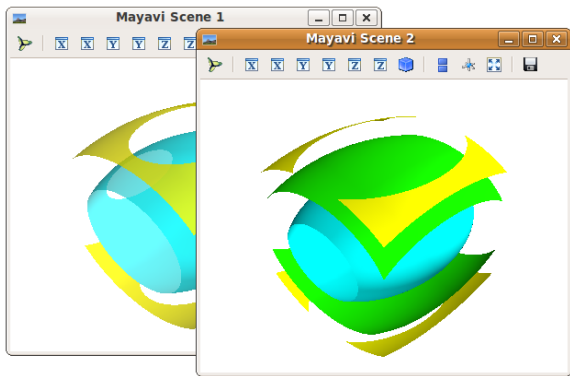
3 Modifying data in place



```
x, y, z = np.ogrid[-5:5:100j, -5:5:100j, -5:5:100j]
scalars = np.sin(x*y*z)/(x*y*z)
iso = mlab.contour3d(scalars, transparent=True,
                    contours=[0.5])

for i in range(1, 20):
    scalars = np.sin(i*x*y*z)/(x*y*z)
    iso.mlab_source.scalars = scalars
```

3 Trick: sharing data



```
iso = mlab.test_contour3d()  
  
mlab.figure()  
iso2 = mlab.pipeline.iso_surface(iso.mlab_source.  
    dataset)
```

x

Mayavi2 authors:

Prabhu Ramachandran Gaël Varoquaux

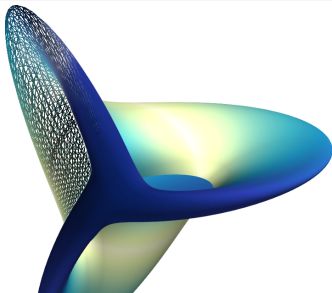
Presentation credits:

Gaël Varoquaux

I need help:

- Integrating mayavi2 into STSE
(<http://stse-software.org>)
- Adjusting mayavi2 to cell biology
- Constructing mayavi2 based GUI

Conclusion



<http://code.enthought.com/projects/mayavi>

Very complete documentation