Analysator

What is Analysator

1. An analysis tool developed for vlasiator
2. Originally developed as a pointn'click tool to get a velocity distributions
3. Functions optimized with Scipy and Numpy libraries (close to C performance)

Functions

1. Reading and writing *numpy* arrays with *VLSV* files
2. Various *analysis* tools
   - 2.1 time-evolution of cells
   - 2.2 cut-throughs
   - 2.3 pitch angle distributions
   - 2.4 Can be combined with *every* python module and function
   - 2.5 ..
3. Plotting using *matplotlib*
4. Plotting Vlasiator *grids* using MayaVi library
5. An interface to Urs' superb *particle pusher*
6. Full easy-to-use *documentation* for every function

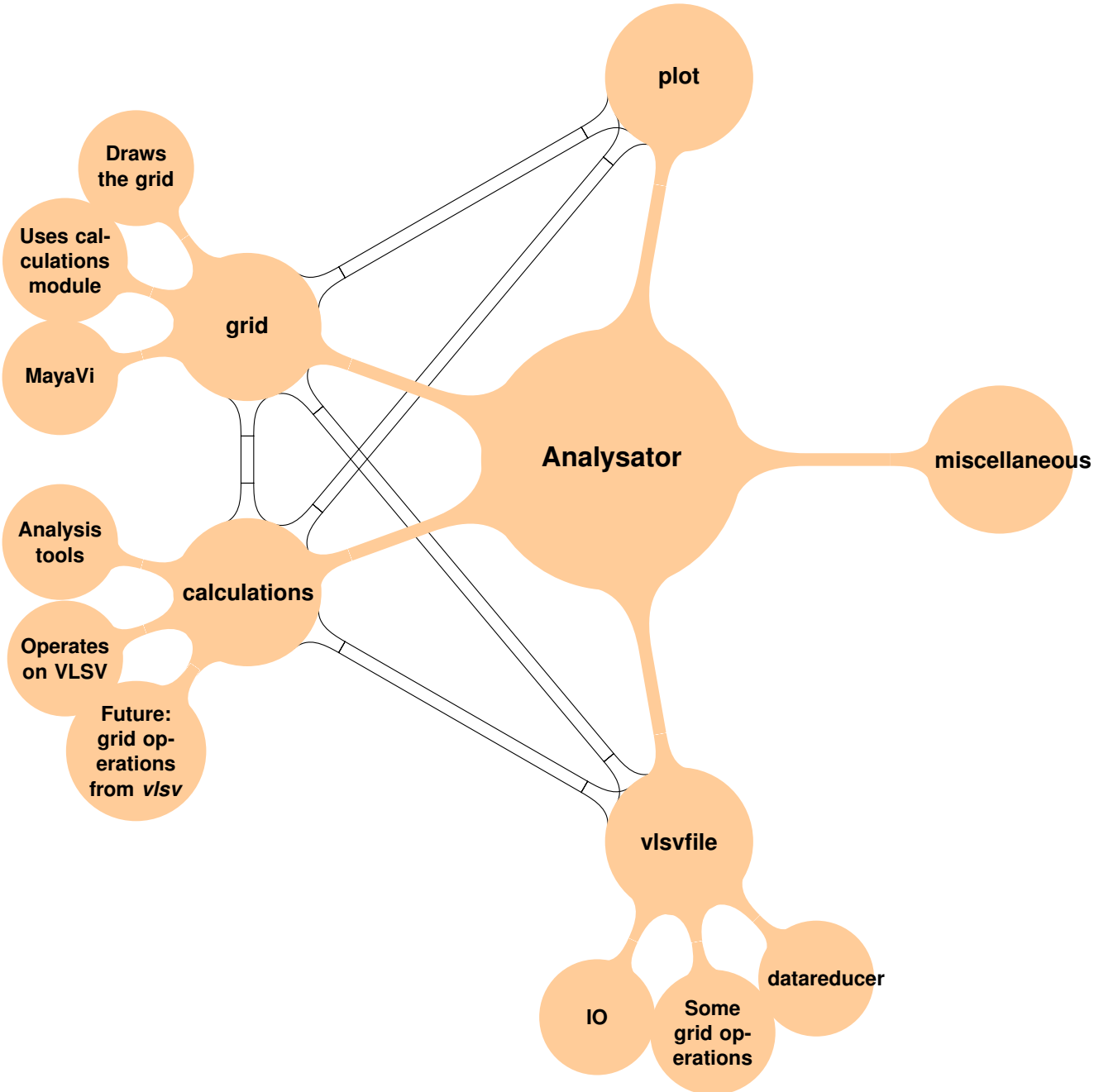
Example

```
# We are interested in reading data from the cell whose ID is 75

import pytools as pt # Import Analysator

# Open a vlsv file for reading
vlsvReader = pt.vlsvfile.VlsvReader('bulk.0003710.vlsv')

# Read a variable at cell id 75
rho = vlsvReader.read_variable( name='rho', cellids=75 )
```

Important notes

Documentation can be found at
`https://github.com/fmihpc/analysator/wiki`

Modules in Analysator



Navigation

Navigating through the code is the most important thing

```
# Import analysator
import pytools as pt

vlsvReader = pt.vlsvfile.VlsvReader('test.vlsv') #What?

cut_through = pt.calculations.cut_through(vlsvReader, point1=[0,0,0], point2=[2,5e6,0])
```

1. Otherwise you need to remember
   - 1.1 The name of the cut_through function
   - 1.2 The first function argument: *vlsvReader*
   - 1.3 The second argument: *point1*
   - 1.4 The third argument: *point2*
   - 1.5 The second and third argument's format and dimensions *(3d list)*

Navigation (the *most* important part)

```
# Import analysator
import pytools as pt

pt.
```

tab

```
In [2]: pt.
pt.calculations      pt.grid              pt.plot
pt.filemanagement    pt.miscellaneous     pt.vlsvfile
```

Navigation (the *most* important part)

```
# Import analysator
import pytools as pt

pt.calculations.
```

tab

```
pt.calculations.cell_time_evolution
pt.calculations.cut3d
pt.calculations.cut_through
pt.calculations.fit
pt.calculations.fourier
pt.calculations.gyrophase_angles_from_file
pt.calculations.lineout
pt.calculations.pitch_angles
pt.calculations.VariableInfo
```

Navigation (the *most* important part)

```
# Import analysator
import pytools as pt

pt.calculations.cut_through?
```

Enter

```
In [2]: pt.calculations.cut_through?
Definition: pt.calculations.cut_through(vlsvReader, point1, point2)
Docstring:
Returns cell ids and distances from point 1 for every cell in a line between
given point1 and point2

:param vlsvReader:      Some open VlsvReader
:type vlsvReader:       :class:`vlsvfile.VlsvReader`
:param point1:          The starting point of a cut-through line
:param point2:          The ending point of a cut-through line
:returns: an array containing cell ids, coordinates and distances in the following
format: [cell ids, distances, coordinates]

.. code-block:: python

   Example:
   vlsvReader = VlsvReader("testfile.vlsv")
   cut_through = cut_through(vlsvReader, [0,0,0], [2,5e6,0])
   cellids = cut_through[0]
   distances = cut_through[1]
   print "Cell ids: " + str(cellids)
   print "Distance from point 1 for every cell: " + str(distances)
```

Navigation summary

```
pt.calculations.
```

tab Autocompletion

```
pt.calculations?
```

Enter Getting documentation

Basic usage

1. Open a *vlsv* file
2. Read from the *vlsv* file
3. Operate on the data

Basic usage

```
# Import analysator
import pytools as pt

# Open a vlsv file
vlsvReader = pt.vlsvfile.VlsvReader('test.vlsv')

# Read vlsv data
rho = vlsvReader.read_variable('rho')
```

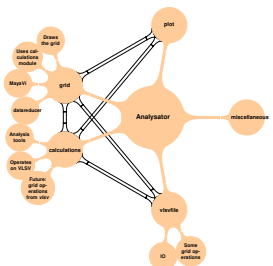
Hands-on

Summary of the basics:

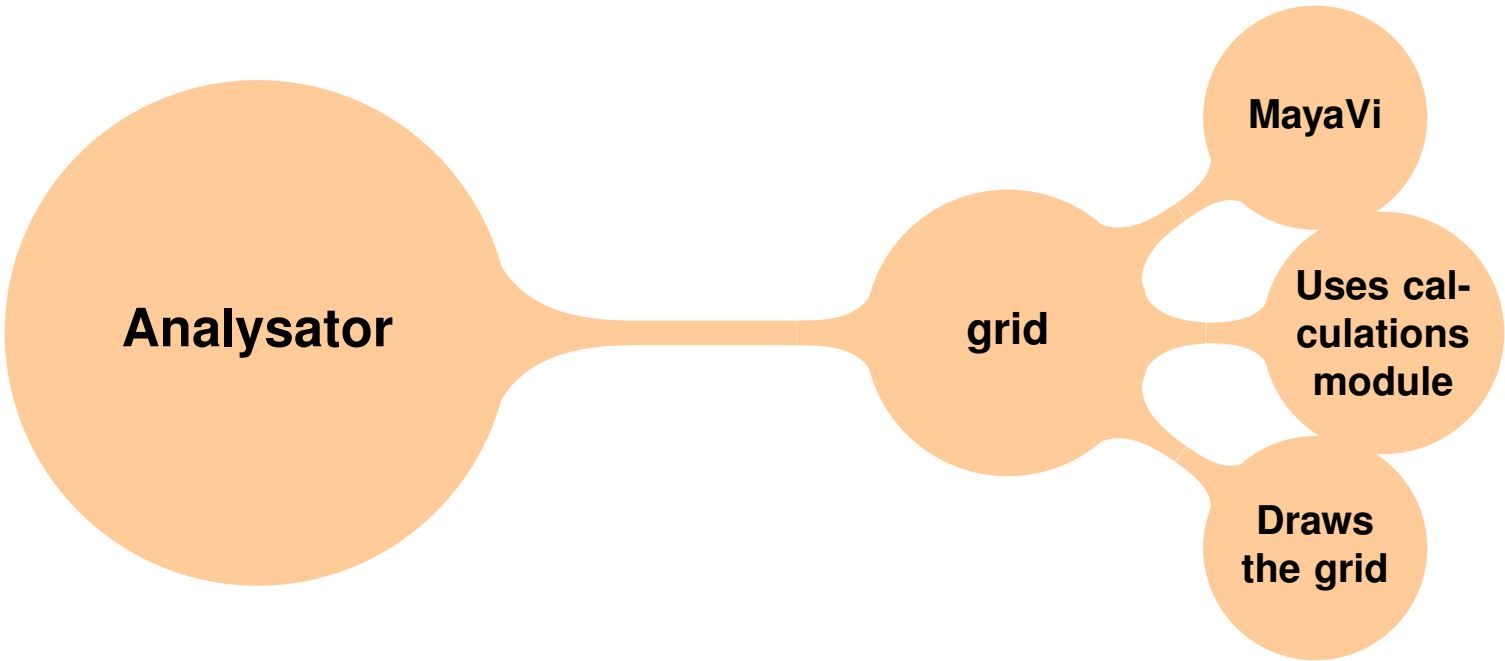
1. Remember to *import analysator*
2. Nearly every script starts by opening a *vlsv* file
3. tab to navigate
4. ? + Enter to get documentation

Note: if you have virtualbox you have to set up a shared folder:

```
sudo mount -t vboxsf -o rw,uid=1000,gid=1000 \
vlsv_data $HOME/share
```



Plotting


Analysator
grid
MayaVi
Uses cal-
culations
module
Draws
the grid

Plotting

The most important concepts

1. How to plot
2. How to use MayaVi
3. How to use the Analysator's *pointnclick* interface

Plotting

1. How to plot

Plotting can be done simply by feeding a vlsv file to the grid class as follows:

```
# Import Analysator
import pytools as pt

# Open a vlsv file
vlsvReader = pt.vlsvfile.VlsvReader('test.vlsv')

# Plot rho
grid = pt.grid.MayaviGrid(vlsvReader, variable='rho')
```

Plotting

1. How to plot
2. How to use MayaVi

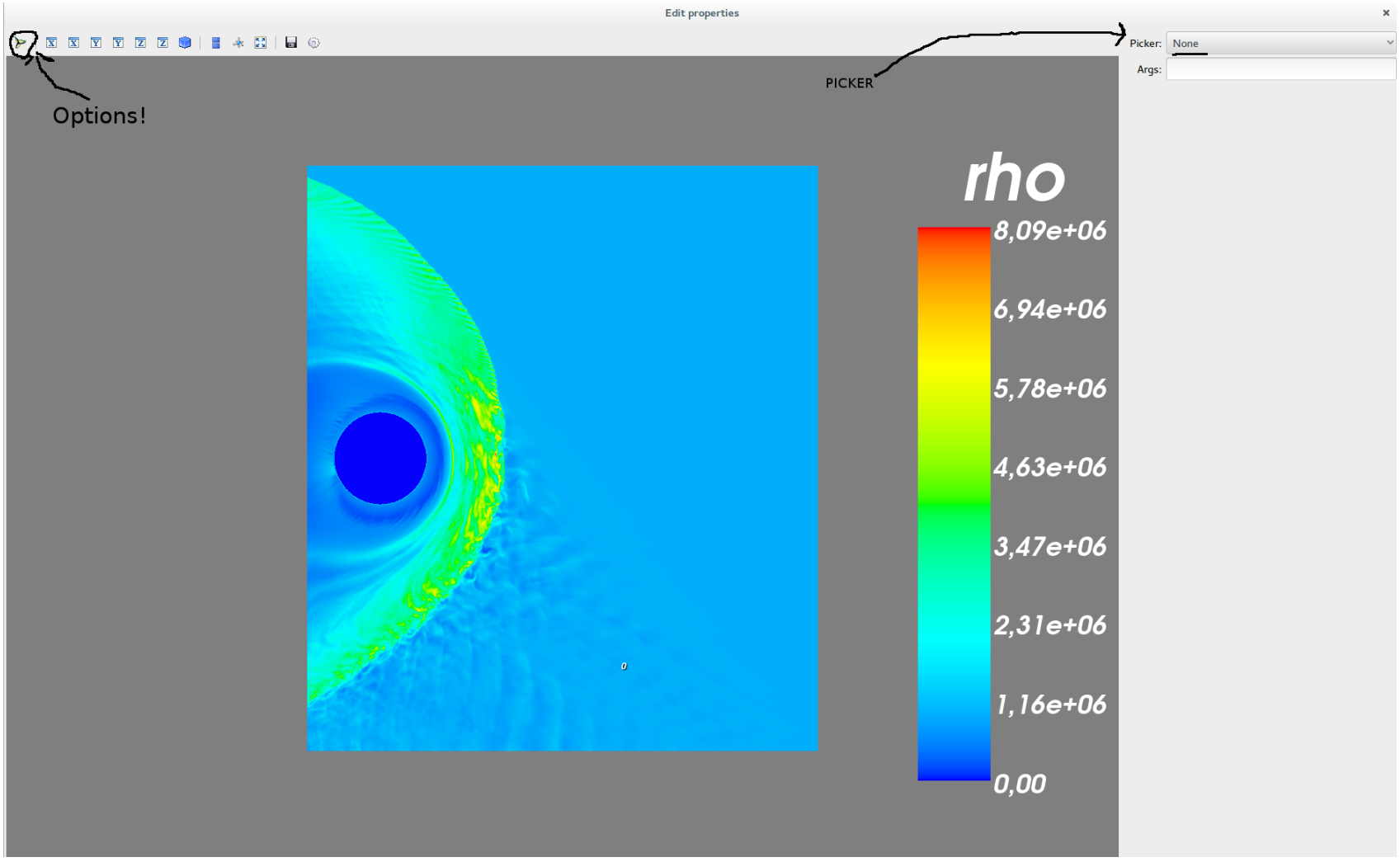


Figure : Example plot with Mayavi, highlighting options and pointnclick tool.

Plotting

1. How to plot
2. How to use MayaVi

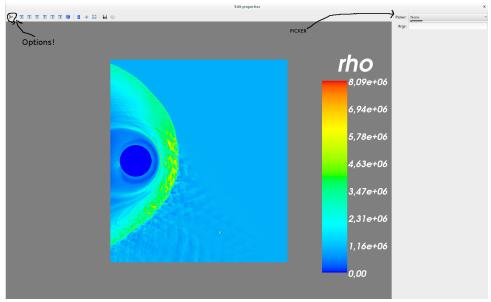


Figure : Example plot with Mayavi, highlighting options and pointnclick tool.

1. Scroll to zoom
2. Mouse 3 to move the image
3. Mouse 1 + Hold to pan the image
4. Mouse 1 to click

Plotting

1. How to plot
2. How to use MayaVi
3. How to use the Analysator's *pointnclick* interface

Pointnclick options:

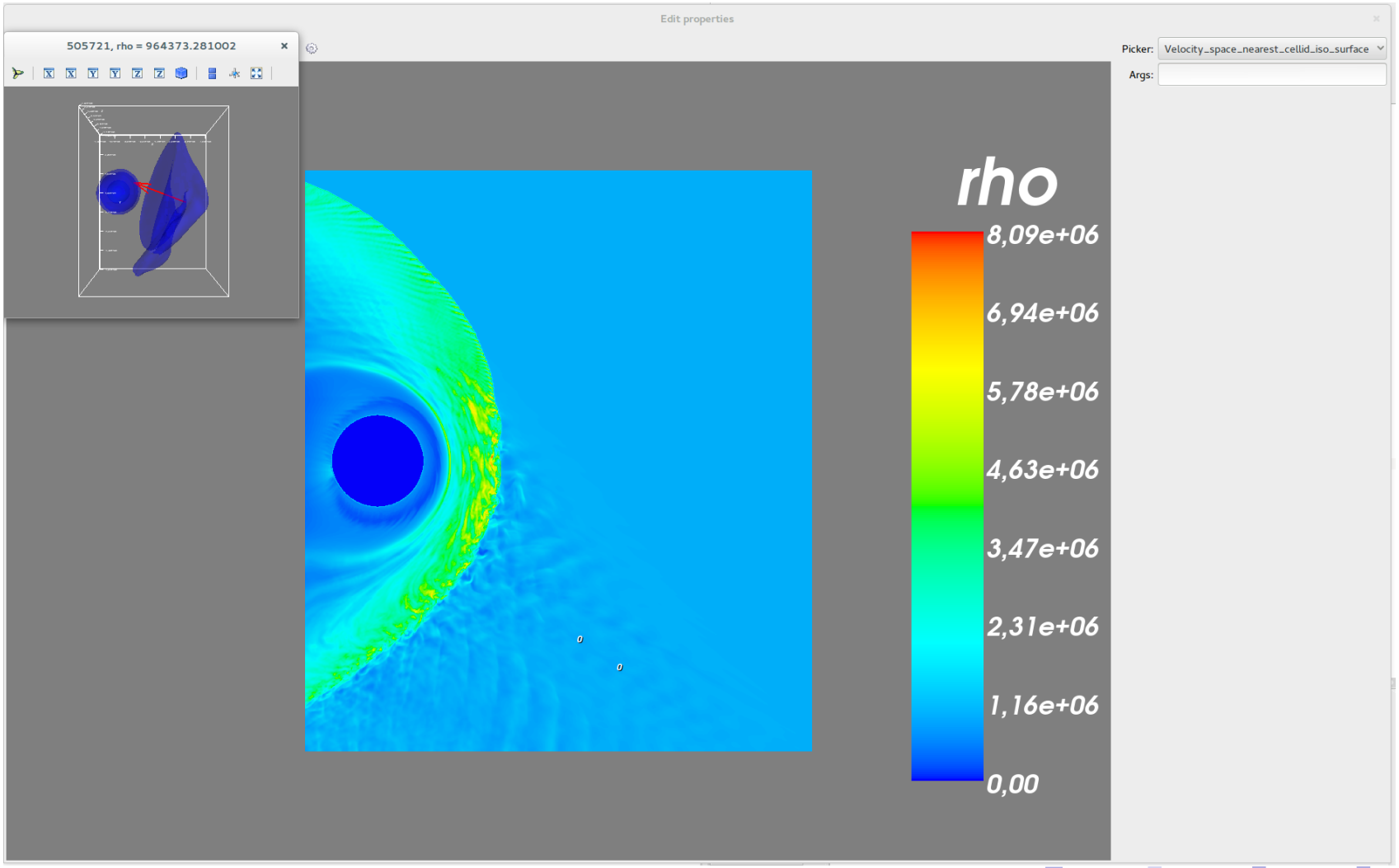
1. None
2. Velocity_space_nearest_cellid
3. Velocity_space_nearest_cellid_iso_surface
4. Pitch_angle
5. Gyrophase_angle (Made by *Yann*, yay)
6. Cut_through

Click Mouse 1 somewhere on the *grid*

Plotting

1. How to plot
2. How to use MayaVi
3. How to use the Analysator's *pointnclick* interface

Sorry about the horribly small text..



Hands-on

To get started:

```
grid = pt.grid.MayaviGrid(vlsvReader, 'rho')
```

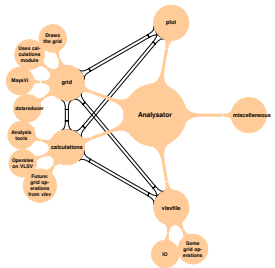
Summary:

1. Mouse scroll to zoom
2. Mouse 3 to move the image
3. Mouse 1 + Hold to pan
4. Mouse 1 to use the Analysator *pointnclick*

Note on *pointnclick* **cut-through option**:
Press Mouse 1 in two places, and there is a *field* where you have to type:

```
plot rho
```

Example from *calculations*




```
# Import analysator
import pytools as pt

# Open a vlsv file
vlsvReader = pt.vlsvfile.vlsvReader('testfile.vlsv')

# Get a cut-through starting at (x=0, y=0; z=0) to (x=2, y=5e6, z=0) :
cut_through = pt.calculations.cut_through(vlsvReader, point1=[0,0,0], point2=[2,5e6,0])

# We now have the cut_through, so now we want to print cellids:
cellids = cut_through[0]
print cellids
```