FC_SIMESH_MATPLOTLIB package, User's Guide *

François Cuvelier†

May 11, 2017

Abstract

The FC_SIMESH Python package allows to use simplices meshes generated from **gmsh** (in dimension 2 or 3) or an hypercube triangulation (in any dimension). The FC_SIMESH_MAYAVI Python package presented in this report is an add-on to the FC_SIMESH Python package. A particular care was taken to the graphics representations of meshes and datas on meshes by using mayavi package.

Contents

*Compiled with Python 3.6.0, packages FC_HYPERMESH-0.0.4, FC_OOGMSH-0.0.4, FC_TOOLS-0.0.10, FC_SIMESH-0.0.5and the plotting libraries MAYAVI-4.5.0, FC_SIMESH_MAYAVI-4.5.0

†Université Paris 13, Sorbonne Paris Cité, LAGA, CNRS UMR 7539, 99 Avenue J-B Clément, F-93430 Villetaneuse, France, cuvelier@math.univ-paris13.fr.

This work was partially supported by ANR Dedales.

1 Functions of the fc_simesh_matplotlib package

1.1 function PLOTMESH

The PLOTMESH function displays the mesh or parts of the mesh defined by an SIMESH object.

Syntaxe

```
simlab.plotmesh(Th,)
simlab.plotmesh(Th,Key=Value, ...)
```

Description

simlab.plotmesh(Th,) displays all the Th.d-dimensional simplices elements.

simlab.plotmesh(Th,Key=Value, ...) specifies function options using one or more Key,Value pair arguments. Options of first level are

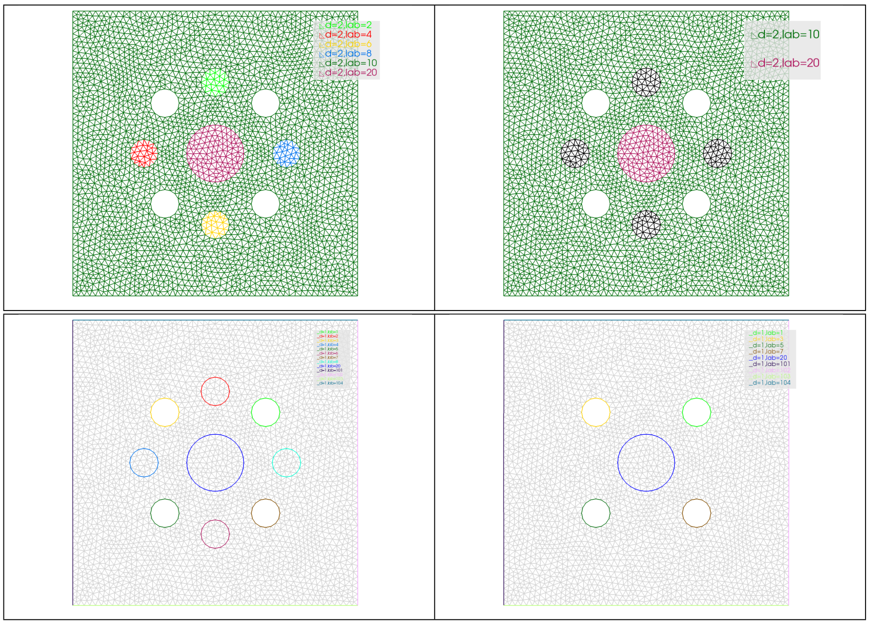
- `d` : to specify the dimension of the simplices elements (default : Th.d)
- `labels` : to select the labels of the elements to display,
- `color` : to specify the color of the displayed mesh elements. (default : use one color by displayed mesh elements),
- `legend` : add a legend to graph if True (default : False)
- `cute_planes'` : cut mesh by n plans given by a list of Plane objects (only in dimension 3). The Plane constructor is Plane(origin=[x,y,z], normal=[nx,ny,z]) which defined the plane coming through point origin and orthogonal to the vector normal. The normal vector pointed to the part of the mesh not displayed. (only in dimension 3) default : [] (no cut).

The options of second level depend on the type of elementaries mesh elements to represent.
One can use any option of the following functions according to the type of d-simplex to be represented.

- In dimension 3,
  - if $d == 3$, mlab.pipeline.surface function is used with tvtk.UnstructuredGrid and tvtk.Tetra().cell_type
  - if $d == 2$, mlab.pipeline.surface function is used with tvtk.UnstructuredGrid and tvtk.Triangle().cell_type
  - if $d == 1$, mlab.pipeline.surface function is used with tvtk.UnstructuredGrid and tvtk.Line().cell_type
  - if $d == 0$, not yet implemented
- In dimension 2,
  - if $d == 2$, mlab.triangular_mesh function is used,
  - if $d == 1$, if $d == 1$, mlab.pipeline.surface function is used with tvtk.UnstructuredGrid and tvtk.Line().cell_type
  - if $d == 0$, not yet implemented
- dimension 1, not yet implemented

1.1.1 2D example

The following example use the *.geo* file `condenser11.geo` which is in the directory `geodir` of the toolbox ....

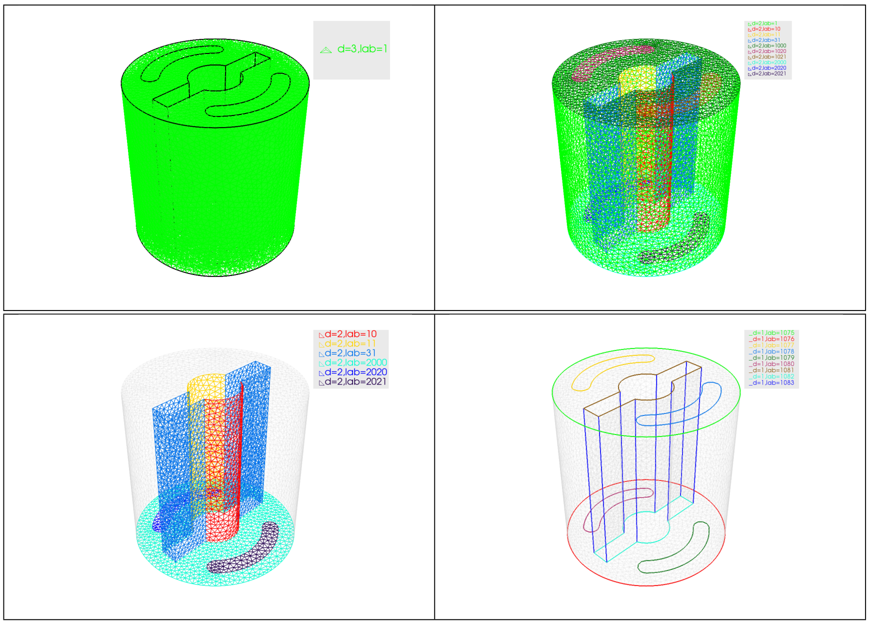



```
meshfile=gmsh.buildmesh2d('condenser11',25)
Th=siMesh(meshfile)
mlab.figure(1)
simlab.plotmesh(Th,legend=True)
mlab.view(0,0)
mlab.figure(2)
simlab.plotmesh(Th,labels=[10,20],legend=True)
simlab.plotmesh(Th,labels=[2,4,6,8],color='black') # ,'Linestyle',':')
mlab.view(0,0)
mlab.figure(3)
simlab.plotmesh(Th,color='LightGray')
simlab.plotmesh(Th,d=1,legend=True,line_width=2)
mlab.view(0,0)
mlab.figure(4)
simlab.plotmesh(Th,color='LightGray')
simlab.plotmesh(Th,d=1,legend=True,labels=[1,3,5,7,20,101,102,103,104])
mlab.view(0,0)
```

Listing 1: 2D plot mesh

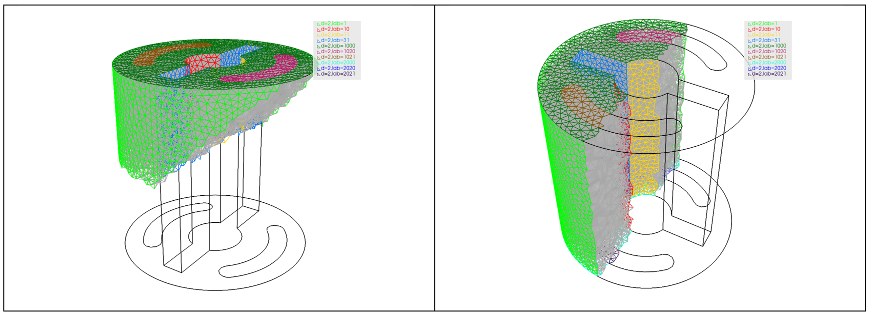
1.1.2 3D example

The following example use the *.geo* file `cylinderkey.geo` which is in the directory `geodir` of the toolbox. This file contains description of a 3D mesh with simplices of dimensions 1, 2 and 3.




```
meshfile=gmsh.buildmesh3d('cylinderkey',15)
Th=siMesh(meshfile)
mlab.figure(1)
simlab.plotmesh(Th,legend=True)
simlab.plotmesh(Th,d=1,color='black',line_width=3)
mlab.figure(2)
simlab.plotmesh(Th,d=2,legend=True)
mlab.figure(3)
simlab.plotmesh(Th,d=2,labels=[1,1000,1020,1021], color='LightGray', opacity=0.1)
simlab.plotmesh(Th,d=2,labels=[10,11,31,2000,2020,2021],legend=True)
mlab.figure(4)
simlab.plotmesh(Th,d=2,  color='LightGray', opacity=0.1)
simlab.plotmesh(Th,d=1,legend=True,line_width=3)
```

Listing 2: 3D plot mesh

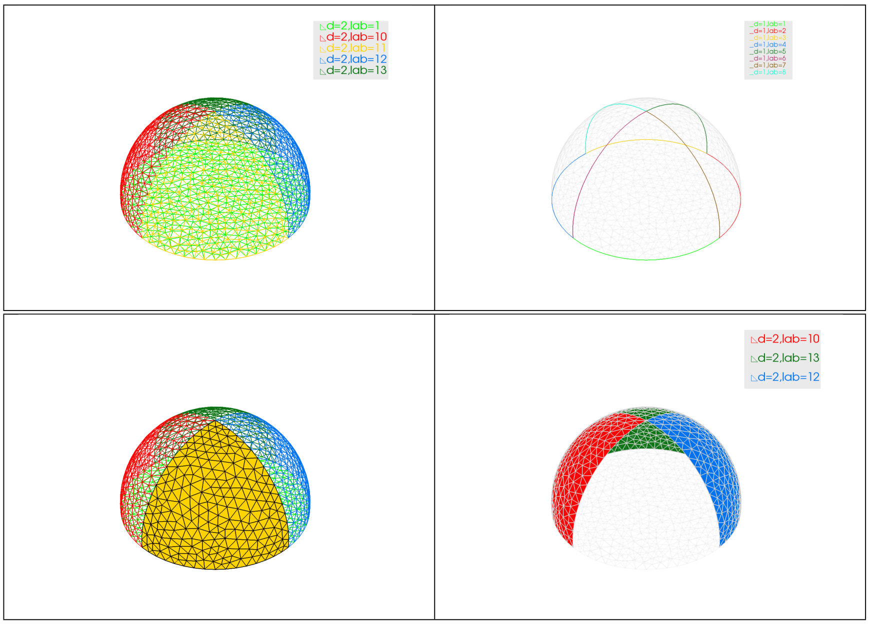


```
meshfile=gmsh.buildmesh3d('cylinderkey',15)
Th=siMesh(meshfile)
mlab.figure(5)
from fc_tools.graphics import Plane
P=[Plane(origin=[0,0,1],normal=[0,0,-1]), Plane(origin=[0,0,1],normal=[0,-1,-1])]
simlab.plotmesh(Th,d=1,color='black')
simlab.plotmesh(Th,cut_planes=P,color='DarkGrey')
simlab.plotmesh(Th,d=2,cut_planes=P,legend=True)
mlab.view(-146,67,6)
mlab.figure(6)
P=Plane(origin=[0,0,1],normal=[-1,0,0])
simlab.plotmesh(Th,d=1,color='black')
simlab.plotmesh(Th,cut_planes=[P],color='DarkGrey')
simlab.plotmesh(Th,d=2,cut_planes=[P],legend=True)
mlab.view(118,45,6)
```

Listing 3: 3D plot mesh,

1.1.3 3D surface example

The following example use the *.geo* file `demisphere5.geo` which is in the directory `geodir` of the toolbox. This file contains description of a 3D surface mesh with simplices of dimensions 1 and 2.




```
from fc_tools.colors import check_color
meshfile=gmsh.buildmesh3ds('demisphere5',20)
Th=siMesh(meshfile)
mlab.figure(1)
simlab.plotmesh(Th,legend=True)
mlab.figure(2)
simlab.plotmesh(Th,color='LightGray',opacity=0.1)
simlab.plotmesh(Th,d=1,legend=True,line_width=2)
mlab.figure(3)
simlab.plotmesh(Th,labels=[1,10,13,12])
vp=simlab.plotmesh(Th,labels=[11],representation='surface')
vp[0].actor.property.edge_visibility=True
mlab.figure(4)
vp=simlab.plotmesh(Th,legend=True,labels=[10,13,12], representation='surface')
for i in range(len(vp)):
  vp[i].actor.property.edge_visibility=True
  vp[i].actor.property.edge_color=check_color('LightGray')
simlab.plotmesh(Th,labels=[1,11],color='LightGray',opacity=0.1)
```

Listing 4: 3D surface mesh : plot function

1.2 function PLOT

The PLOT function displays scalar datas on the mesh or parts of the mesh defined by an SIMESH object.

Syntaxe

```
simlab.plot(Th,u)
simlab.plot(Th,u,Key=Value, ...)
```

Description

simlab.plot(Th,u) displays data u on all the Th.d-dimensional simplices elements. The data u is an 1D-array of size Th.nq or Th.nqGlobal or Th.nqParent.

$\boxed{\text{simlab.plot(Th,u,Key=Value, ...)}}$ specifies function options using one or more Key,Value pair arguments. Options of first level are

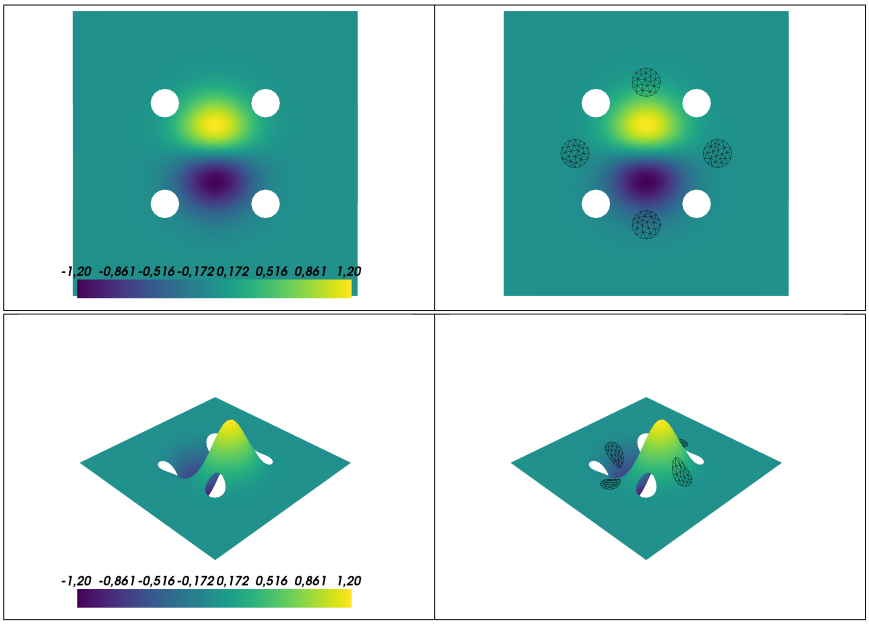
- `d` : to specify the dimension of the simplices elements (default : Th.d)
- `labels` : to select the labels of the elements to display data,
- `plane` : if True, (default : False)

The second level options depend on the type of elementaries mesh elements on which we want to represent datas.
One can use any option of the following functions according to the type of d-simplex.

- In dimension 3,
  - if $d == 3$, mlab.pipeline.surface function is used with tvtk.UnstructuredGrid and tvtk.Tetra().cell_type
  - if $d == 2$, mlab.triangular_mesh function is used.
  - if $d == 1$, mlab.pipeline.surface function is used with tvtk.UnstructuredGrid and tvtk.Line().cell_type
- In dimension 2,
  - if $d == 2$, mlab.triangular_mesh function is used.
  - if $d == 1$, mlab.pipeline.surface function is used with tvtk.UnstructuredGrid and tvtk.Line().cell_type.
- Dimension 1 : not implemented.

1.2.1 2D example

The following example use the *.geo* file `condenser11.geo` which is in the directory `geodir` of the package.

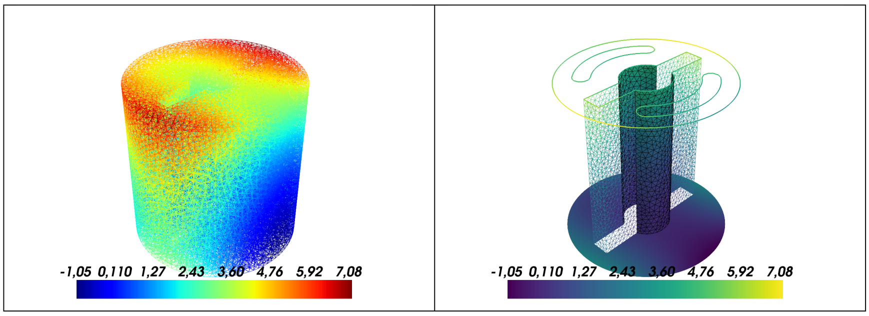



```
import numpy as np
meshfile=gmsh.buildmesh2d('condenser11',25)
Th=siMesh(meshfile)
u=Th.feval(lambda x,y: 5*np.exp(-3*(x**2+y**2))*np.cos(x)*np.sin(y))
mlab.figure(1)
simlab.plot(Th,u,colormap='viridis')
mlab.view(0,0)
mlab.colorbar()
mlab.figure(2)
simlab.plot(Th,u,labels=[10,20],colormap='viridis')
vp=simlab.plot(Th,u,labels=[2,4,6,8],colormap='viridis')
mlab.view(0,0)
vp.actor.property.edge_visibility=True
vp.actor.property.line_width=0.5
mlab.figure(3)
simlab.plot(Th,u,colormap='viridis',plane=False)
mlab.colorbar()
mlab.figure(4)
simlab.plot(Th,u,labels=[10,20],colormap='viridis',plane=False)
vp=simlab.plot(Th,u,labels=[2,4,6,8],colormap='viridis',plane=False,merge=False)
for i in range(len(vp)):
  vp[i].actor.property.edge_visibility=True
  vp[i].actor.property.line_width=0.5
mlab.figure(5)
simlab.plot(Th,u,d=1,colormap='viridis',line_width=2)
simlab.plotmesh(Th,color='LightGray',opacity=0.05,merge=True)
mlab.colorbar()
mlab.figure(6)
simlab.plot(Th,u,d=1,colormap='viridis',line_width=2,plane=False)
simlab.plotmesh(Th,color='LightGray',opacity=0.05,merge=True,z=u)
mlab.colorbar()
```

Listing 5: 2D mesh : plot function

1.2.2 3D example

The following example use the *.geo* file `cylinderkey.geo` which is in the directory `geodir` of the toolbox. This file contains description of a 3D mesh with simplices of dimensions 1, 2 and 3.

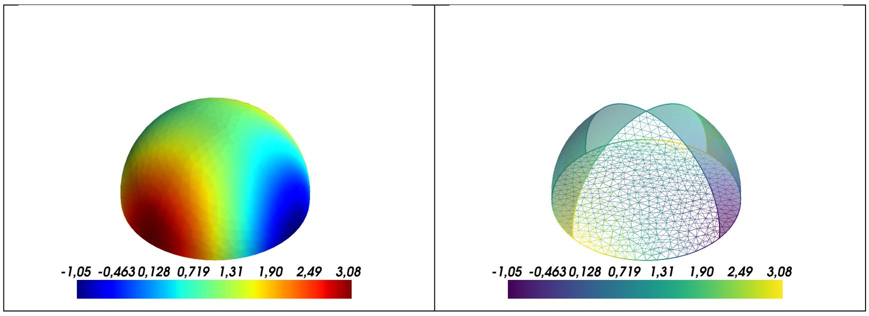



```python
meshfile=gmsh.buildmesh3d('cylinderkey',15)
Th=siMesh(meshfile)
u=Th.feval(lambda x,y,z: 3*x**2-y**3+z**2+x*y)
mlab.figure(1)
simlab.plot(Th,u,colormap='jet',line_width=0.5)
mlab.colorbar()
mlab.figure(2)
simlab.plot(Th,u,d=2,labels=[2000,2020,2021])
vp=simlab.plot(Th,u,d=2,labels=[31],representation='wireframe')
vp.actor.property.lighting=False
vp.actor.property.line_width=0.5
vp=simlab.plot(Th,u,d=2,labels=[10,11],merge=False)
for i in range(len(vp)):
  vp[i].actor.property.edge_visibility=True
  vp[i].actor.property.line_width=0.5
simlab.plot(Th,u,d=1,labels=[1075,1077,1078,1081],line_width=3)
mlab.colorbar()
```

Listing 6: 3D mesh : plot function

1.2.3 3D surface example

The following example use the *.geo* file `demisphere5.geo` which is in the directory `geodir` of the toolbox. This file contains description of a 3D surface mesh with simplices of dimensions 1 and 2.




```python
meshfile=gmsh.buildmesh3ds('demisphere5',30)
Th=siMesh(meshfile)
u=Th.feval(lambda x,y,z: 3*x**2-y**3+z**2+x*y)
mlab.figure(1)
simlab.plot(Th,u,colormap='jet',line_width=0.5)
mlab.colorbar()
mlab.figure(2)
simlab.plot(Th,u,labels=[10,12],opacity=0.5)
vp=simlab.plot(Th,u,labels=[1,11],representation='wireframe',merge=False)
for i in range(len(vp)):
  vp[i].actor.property.lighting=False
  vp[i].actor.property.line_width=0.5
simlab.plot(Th,u,d=1,line_width=3)
mlab.colorbar()
```

Listing 7: 3D surface mesh : plot function

1.3 function PLOTISO

The PLOTISO function displays isolines from datas on the mesh or parts of the mesh defined by an SIMESH object. This function only works with 2-simplices in space dimension 2 or 3.

Syntaxe

```
simlab.plotiso(Th,u)
simlab.plotiso(Th,u,Key=Value, ...)
```

Description

simlab.plotiso(Th,u) displays data u on all the 2-dimensional simplices elements as colored isovalues. The data u is an 1D-array of size Th.nq or Th.nqGlobal or Th.nqParent.

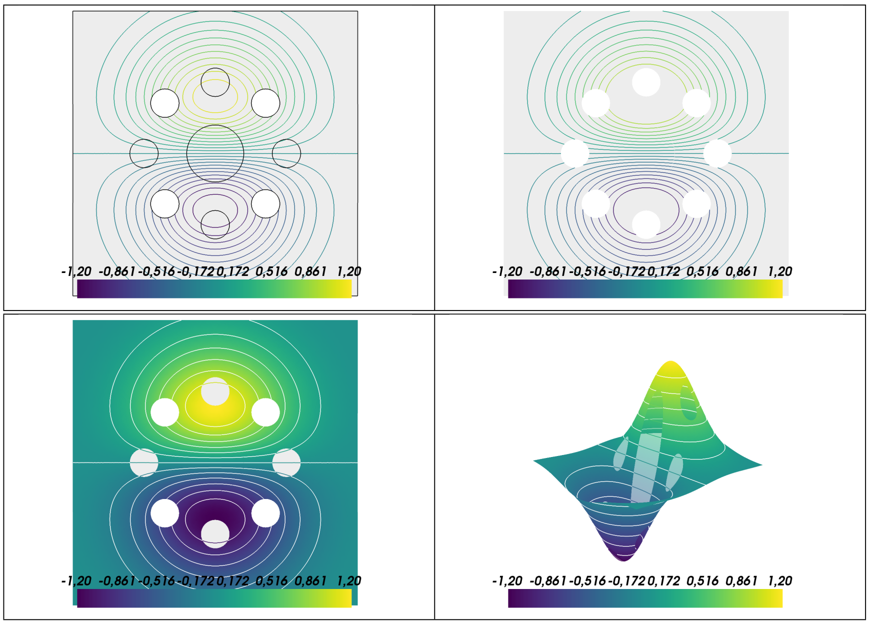
simlab.plotiso(Th,u,key=value, ...) specifies function options using one or more key,value pair arguments. Options of first level are

- `contours` : to specify the number of isolines (default : 10) or a list/numpy array of isovalues (default : empty)
- `labels` : to select the labels of the elements to display data,
- `plane` : if False draw 3D isovalues with data u as z values. (default : True)
- `color` : to specify one color for all isolines (default : None))

The second level options are the options of the mlab.pipeline.iso_surface which we use to draw the isovalues.

1.3.1 2D example

The following example use the *.geo* file `condenser11.geo` which is in the directory `geodir` of the toolbox.

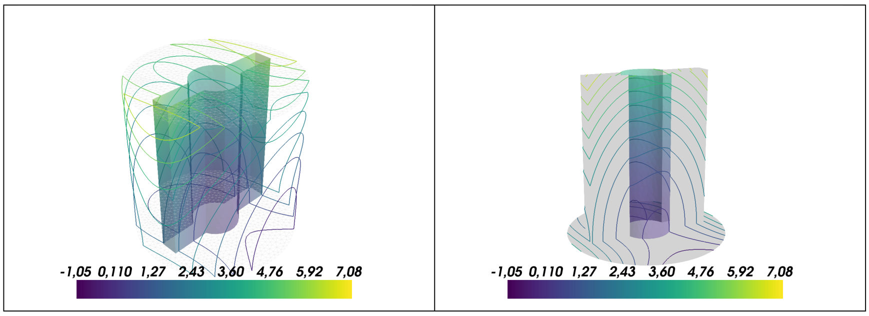



```python
import numpy as np
meshfile=gmsh.buildmesh2d('condenser11',50)
Th=siMesh(meshfile)
u=Th.feval(lambda x,y: 5*np.exp(-3*(x**2+y**2))*np.cos(x)*np.sin(y))
mlab.figure(1)
simlab.plotiso(Th,u,contours=25,colormap='viridis')
simlab.plotmesh(Th,d=1,color='black')
simlab.plotmesh(Th,d=2,color='LightGray',representation='surface', opacity=0.4)
mlab.view(0,0)
mlab.colorbar()
mlab.figure(2)
simlab.plotiso(Th,u,contours=np.arange(-1,1,0.1),labels=[10,20],
    colormap='viridis')
simlab.plotmesh(Th,d=2,color='LightGray',representation='surface', opacity=0.4,
    labels=[10,20] )
mlab.view(0,0)
mlab.colorbar()
mlab.figure(3)
simlab.plotiso(Th,u,contours=15,colormap='viridis',labels=[2,4,6,8])
simlab.plotmesh(Th,d=2,color='LightGray',representation='surface',
    opacity=0.4,labels=[2,4,6,8] )
simlab.plotiso(Th,u,contours=15,color='white',labels=[10,20])
simlab.plot(Th,u,labels=[10,20],colormap='viridis')
mlab.view(0,0)
mlab.colorbar()
mlab.figure(4)
simlab.plotiso(Th,u,contours=15,colormap='viridis',labels=[2,4,6,8,20],
    plane=False)
simlab.plot(Th,u,labels=[2,4,6,8,20],colormap='viridis',opacity=0.5, plane=False)
simlab.plotiso(Th,u,contours=15,color='white',labels=[10], plane=False)
simlab.plot(Th,u,labels=[10],colormap='viridis', plane=False)
mlab.colorbar()
mlab.view(-41,71,7)
```

Listing 8: 2D mesh : plotiso function

1.3.2 3D example

The following example use the *.geo* file `cylinderkey.geo` which is in the directory `geodir` of the toolbox. This file contains description of a 3D mesh with simplices of dimensions 1, 2 and 3.

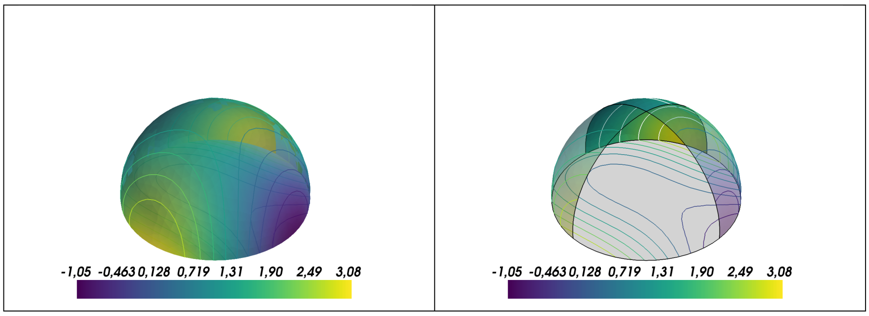



```python
meshfile=gmsh.buildmesh3d('cylinderkey',15)
Th=siMesh(meshfile)
u=Th.feval(lambda x,y,z: 3*x**2-y**3+z**2+x*y)
mlab.figure(1)
simlab.plot(Th,u,d=2,labels=[10,11,31],opacity=0.5)
simlab.plotiso(Th,u,d=2,line_width=2)
simlab.plotmesh(Th,d=2,labels=[1000,1020,1021,2000,2020,2021], color='LightGray',
    opacity=0.1)
mlab.colorbar()
mlab.figure(2)
simlab.plot(Th,u,d=2,labels=[10,11],opacity=0.5)
simlab.plotmesh(Th,d=2,labels=[31,2000,2020,2021], color='LightGray',
    representation='surface')
simlab.plotiso(Th,u,d=2,labels=[10,11,31,2000,2020,2021], contours=15,
    line_width=2)
mlab.colorbar()
mlab.view(65,74,7)
```

Listing 9: 3D mesh : plotiso function

1.3.3 3D surface example

The following example use the *.geo* file `demisphere5.geo` which is in the directory `geodir` of the toolbox. This file contains description of a 3D surface mesh with simplices of dimensions 1 and 2.




```python
meshfile=gmsh.buildmesh3ds('demisphere5',30)
Th=siMesh(meshfile)
u=Th.feval(lambda x,y,z: 3*x**2-y**3+z**2+x*y)
mlab.figure(1)
simlab.plot(Th,u, opacity=0.7)
simlab.plotiso(Th,u, line_width=1.5)
mlab.colorbar()
mlab.figure(2)
#simlab.plotmesh(Th,d=2,labels=[1,1000,1020,1021], color='LightGray', opacity=0.1)
simlab.plot(Th,u,labels=[13])
simlab.plotiso(Th,u, labels=[13], color='white', line_width=1.5)
simlab.plot(Th,u,labels=[10,12],opacity=0.4)
simlab.plotmesh(Th,labels=[1], representation='surface', color='LightGray')
simlab.plotiso(Th,u, labels=[1,10,12], line_width=1.5)
simlab.plotmesh(Th,d=1, color='black', line_width=1.5)
mlab.colorbar()
```

Listing 10: 3D surface mesh : plotiso function

1.3.4 function QUIVER

The QUIVER function displays vector field datas on the mesh or parts of the mesh defined by an SIMESH object.

Syntaxe

```python
simlab.quiver(Th,V)
simlab.quiver(Th,V,Key=Value, ...)
```

Description

simlab.quiver(Th,V) displays vector field V on each vertices of the d-dimensional simplices elements in dimension $d = 2$ or $d = 3$. The data V is an 2D-array numpy array of size dim-by-Th.nq .

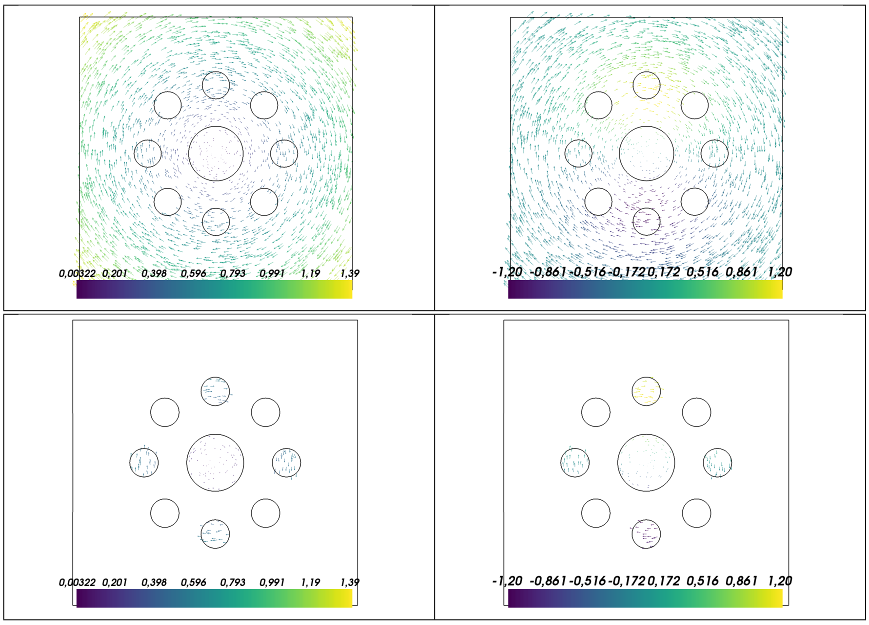
simlab.quiver(Th,V,Key=Value, ...) specifies function options using one or more Key,Value pair arguments. Options of first level are

- `labels` : to select the labels of the elements to display data,
- `scalars` : to set quivers color to a numpy array of size Th.nq (default : empty and use colors of the mesh elements).
- `color` : to specify one color for all quivers.

For key/value pairs, one could also used those of the mlab.quiver3d function.

1.3.5 2D example

The following example use the *.geo* file `condenser11.geo` which is in the directory `geodir` of the toolbox.

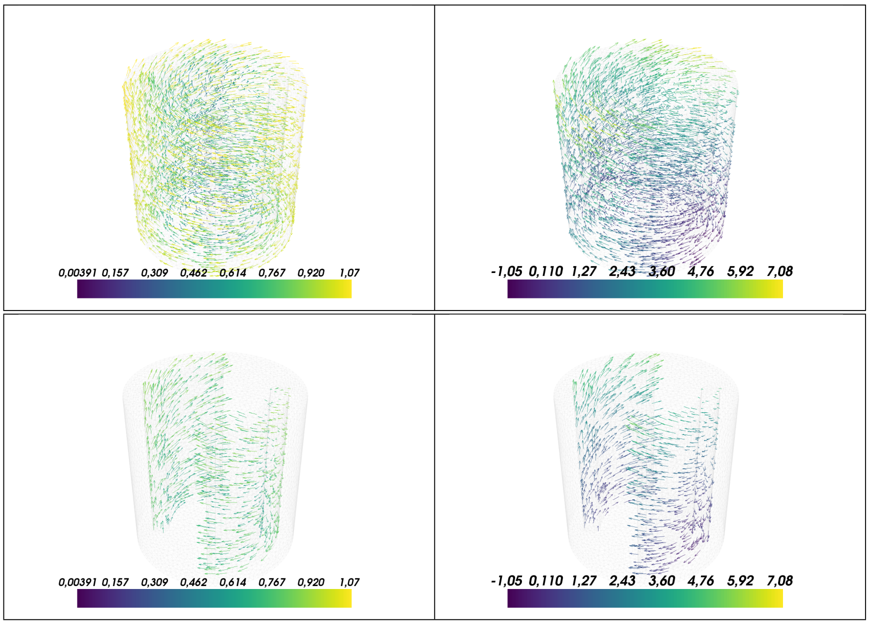



```python
import numpy as np
meshfile=gmsh.buildmesh2d('condenser11',50)
Th=siMesh(meshfile)
u=Th.feval(lambda x,y: 5*np.exp(-3*(x**2+y**2))*np.cos(x)*np.sin(y))
w=[lambda x,y: y*np.cos(-(x**2+y**2)/10), lambda x,y: -x*np.cos(-(x**2+y**2)/10)]
W=np.array([Th.feval(w[0]),Th.feval(w[1])])
mlab.figure(1)
simlab.quiver(Th,W, scale_factor=0.05 ,line_width=1, mask_points=Th.nq//2000)
simlab.plotmesh(Th,d=1,color='black')
mlab.vectorbar()
mlab.view(0,0)
mlab.figure(2)
simlab.quiver(Th,W, scalars=u, scale_factor=0.05,line_width=1,
    mask_points=Th.nq//2000)
simlab.plotmesh(Th,d=1,color='black')
mlab.scalarbar()
mlab.view(0,0)
mlab.figure(3)
simlab.quiver(Th,W,labels=[2,4,6,8,20] , scale_factor=0.05 ,line_width=1,
    mask_points=Th.nq//2000)
simlab.plotmesh(Th,d=1,color='black')
mlab.vectorbar()
mlab.view(0,0)
mlab.figure(4)
simlab.quiver(Th,W,labels=[2,4,6,8,20], scalars=u,
    scale_factor=0.05,line_width=1, mask_points=Th.nq//2000)
simlab.plotmesh(Th,d=1,color='black')
mlab.scalarbar()
mlab.view(0,0)
```

Listing 11: 2D mesh : quiver function

1.3.6 3D example

The following example use the *.geo* file `cylinderkey03.geo` which is in the directory `geodir/3d` of the toolbox. This file contains description of a 3D mesh with simplices of dimensions 1, 2 and 3.

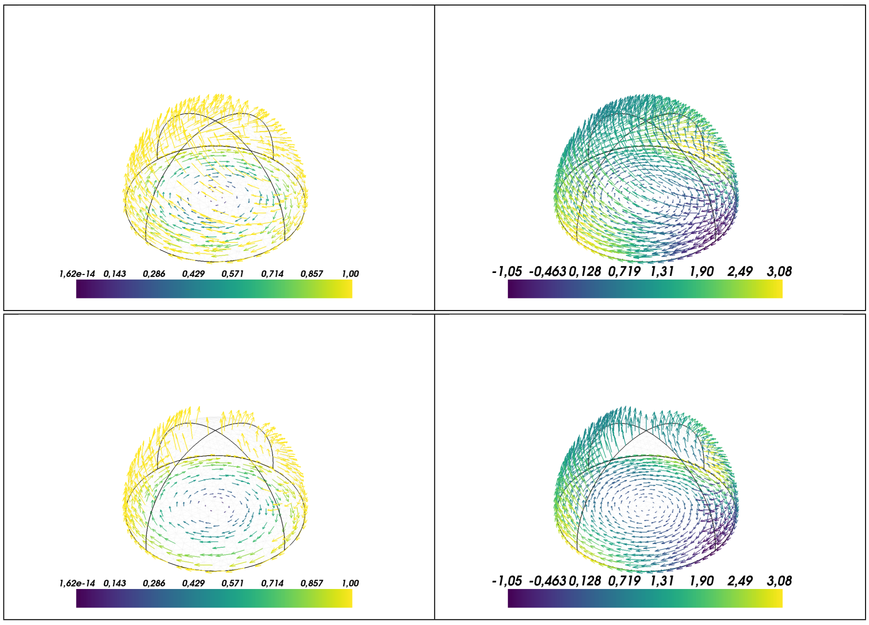



```python
import numpy as np
meshfile=gmsh.buildmesh3d('cylinder3dom',15)
mlab.close(all=True)
Th=siMesh(meshfile);
u=Th.feval(lambda x,y,z: 3*x**2-y**3+z**2+x*y)
w=[lambda x,y,z: y*np.cos(-(x**2+y**2)/10), lambda x,y,z:
    -x*np.cos(-(x**2+y**2)/10), lambda x,y,z: z/5]
W=np.array([Th.feval(w[0]),Th.feval(w[1]),Th.feval(w[2])])
mlab.figure(1)
simlab.quiver(Th,W,line_width=1, mask_points=Th.nq//3000)
simlab.plotmesh(Th,d=2,color='LightGray',opacity=0.05)
mlab.vectorbar()
mlab.figure(2)
simlab.quiver(Th,W,scalars=u, line_width=1, mask_points=Th.nq//3000)
simlab.plotmesh(Th,d=2,color='LightGray',opacity=0.05)
mlab.scalarbar()
mlab.figure(3)
simlab.quiver(Th,W,labels=[10,11],line_width=1, mask_points=Th.nq//3000)
simlab.plotmesh(Th,d=2,color='LightGray',opacity=0.05)
mlab.vectorbar()
mlab.figure(4)
simlab.quiver(Th,W,scalars=u, labels=[10,11],line_width=1,
    mask_points=Th.nq//3000)
simlab.plotmesh(Th,d=2,color='LightGray',opacity=0.05)
mlab.scalarbar()
```

Listing 12: 3D mesh : quiver function

1.3.7 3D surface example

The following example use the *.geo* file `demisphere5.geo` which is in the directory `geodir` of the toolbox. This file contains description of a 3D surface mesh with simplices of dimensions 1 and 2.




```python
import numpy as np
meshfile=gmsh.buildmesh3ds('demisphere5',30)
Th=siMesh(meshfile)
u=Th.feval(lambda x,y,z: 3*x**2-y**3+z**2+x*y)
w=[lambda x,y,z: y*np.cos(-(x**2+y**2)/10), lambda x,y,z:
    -x*np.cos(-(x**2+y**2)/10), lambda x,y,z: z]
W=np.array([Th.feval(w[0]),Th.feval(w[1]),Th.feval(w[2])])
mlab.figure(1)
simlab.quiver(Th,W)
simlab.plotmesh(Th,color='LightGray',opacity=0.05)
simlab.plotmesh(Th,d=1,color='black',line_width=2)
mlab.vectorbar()
mlab.figure(2)
simlab.quiver(Th,W,scalars=u)
simlab.plotmesh(Th,color='LightGray',opacity=0.05)
simlab.plotmesh(Th,d=1,color='black',line_width=2)
mlab.scalarbar()
mlab.figure(3)
simlab.quiver(Th,W,labels=[1,10,12])
simlab.plotmesh(Th,color='LightGray',opacity=0.05)
simlab.plotmesh(Th,d=1,color='black',line_width=2)
mlab.vectorbar()
mlab.figure(4)
simlab.quiver(Th,W,scalars=u,labels=[1,10,12])
simlab.plotmesh(Th,color='LightGray',opacity=0.05)
simlab.plotmesh(Th,d=1,color='black',line_width=2)
mlab.scalarbar()
```

Listing 13: 3D surface mesh : quiver function

1.4 function SLICEMESH

The SLICEMESH function displays intersection of a plane and a 3D mesh or parts of a 3D mesh defined by an SIMESH object.

Syntaxe

```python
simlab.slicemesh(Th,)
simlab.slicemesh(Th,Key=Value, ...)
```

Description

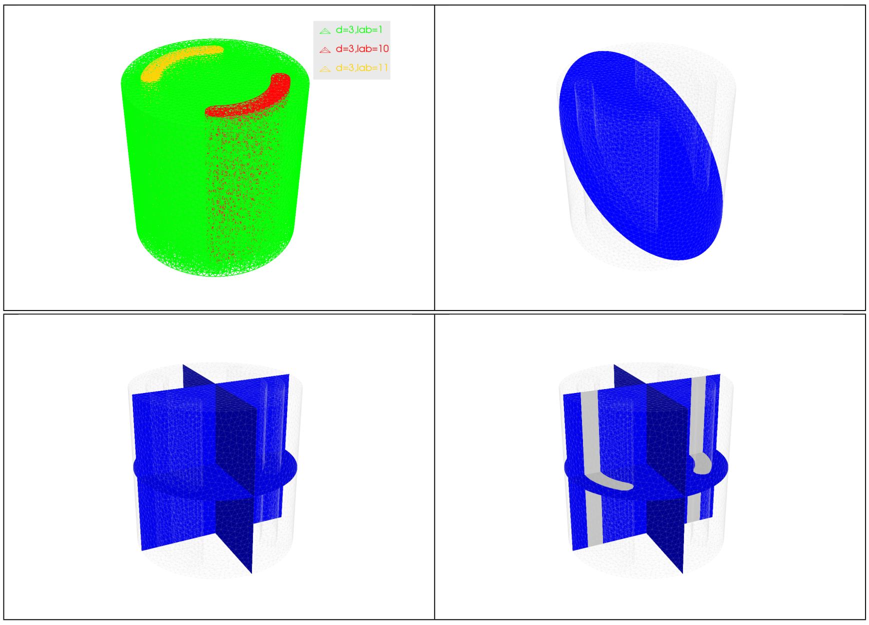
$\boxed{\text{simlab.slicemesh(Th,)}}$ displays intersection of a plane and all the 3-dimensional simplices elements. By default the plane is given by an *origin* point $(0, 0, 0)$ which lies on it and a *normal* vector $(0, 0, 1)$ which is orthogonal to the plane.

$\boxed{\text{simlab.slicemesh(Th,Key=Value, ...)}}$ specifies function options using one or more Key,Value pair arguments. Key could be:

- `origin` : to specify a point lying on the plane (default is $(0, 0, 0)$)
- `normal` : to specify a vector orthogonal to the plane (default is $(0, 0, 1)$)
- `labels` : to select the labels of the elements to intersect,

The other Key/Value pair options are those of scalar_cut_plane function from mayavi.mlab.pipeline.

The following example use the *.geo* file `cylinder3dom.geo` which is in the directory `geodir` of the toolbox. This file contains description of a 3D mesh with simplices of dimensions 1, 2 and 3.




```
meshfile=gmsh.buildmesh3d('cylinder3dom',15);
mlab.close(all=True)
Th=siMesh(meshfile)
mlab.figure(1)
simlab.plotmesh(Th,legend=True)
simlab.plotmesh(Th,d=1,color='black',line_width=3)
mlab.figure(2)
simlab.plotmesh(Th,d=2,color='LightGray',opacity=0.05)
simlab.slicemesh(Th,origin=(0,0,1),normal=(-1,0,1))
mlab.view(132,53,7)
mlab.figure(3)
simlab.plotmesh(Th,d=2,color='LightGray',opacity=0.05)
for normal in [(1,0,0),(0,1,0),(0,0,1)]:
  simlab.slicemesh(Th,origin=(0,0,1),normal=normal)
mlab.view(155,66,7)
mlab.figure(4)
simlab.plotmesh(Th,d=2,color='LightGray',opacity=0.05)
for normal in [(1,0,0),(0,1,0),(0,0,1)]:
  simlab.slicemesh(Th,labels=[10,11],origin=(0,0,1),normal=normal,
      color='LightGray')
  simlab.slicemesh(Th,labels=[1],origin=(0,0,1),normal=normal)
mlab.view(155,66,7)
```

Listing 14: 3D mesh : slicemesh function

1.5 function SLICE

The SLICE function displays datas on the intersection of a plane and a 3D mesh or parts of a 3D mesh defined by an SIMESH object.

Syntaxe

```
simlab.slice(Th,u)
simlab.slice(Th,u,Key=Value, ...)
```

Description

$\boxed{\text{simlab.}\textbf{slice}\text{(Th,u)}}$ displays u data on the intersection of a plane and all the 3-dimensional simplices elements. By default the plane is given by an *origin* point $(0, 0, 0)$ which lies on it and a *normal* vector $(0, 0, 1)$ which is orthogonal to the plane. The data u is an 1D-array of size Th.nq or Th.nqGlobal or Th.nqParent.

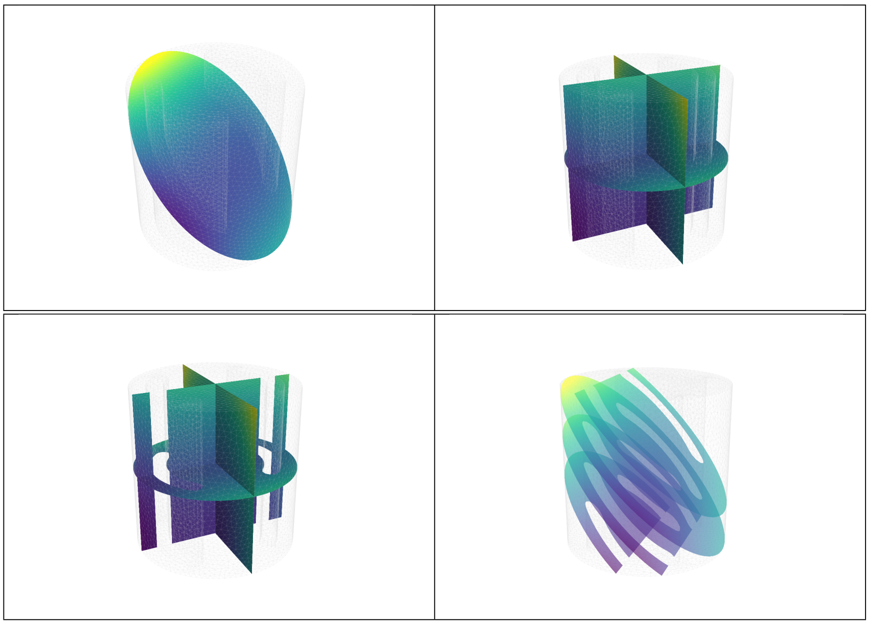
$\boxed{\text{simlab.}\textbf{slice}\text{(Th,u,Key,Value, ...)}}$ specifies function options using one or more Key,Value pair arguments. Key could be:

- `origin` : to specify a point lying on the plane (default is $(0, 0, 0)$)
- `normal` : to specify a vector orthogonal to the plane (default is $(0, 0, 1)$)
- `labels` : to select the labels of the elements to intersect,

The other Key/Value pair options are those of scalar_cut_plane function from mayavi.mlab.pipeline.

1.5.1 3D example

The following example use the *.geo* file `cylinder3dom.geo` which is in the directory `geodir/3d` of the toolbox. This file contains description of a 3D mesh with simplices of dimensions 1, 2 and 3.



```
meshfile=gmsh.buildmesh3d('cylinder3dom',15)
mlab.close(all=True)
Th=siMesh(meshfile)
u=Th.feval(lambda x,y,z: 3*x**2-y**3+z**2+x*y)
mlab.figure(1)
simlab.plotmesh(Th,d=2,color='LightGray',opacity=0.05)
simlab.slice(Th,u,origin=(0,0,1),normal=(-1,0,1))
mlab.view(132,53,7)
mlab.figure(2)
simlab.plotmesh(Th,d=2,color='LightGray',opacity=0.05)
for normal in [(1,0,0),(0,1,0),(0,0,1)]:
  simlab.slice(Th,u,origin=(0,0,1),normal=normal)
mlab.view(155,66,7)
mlab.figure(3)
simlab.plotmesh(Th,d=2,color='LightGray',opacity=0.05)
for normal in [(1,0,0),(0,1,0),(0,0,1)]:
  simlab.slice(Th,u,labels=[1],origin=(0,0,1),normal=normal)
mlab.view(155,66,7)
mlab.figure(4)
simlab.plotmesh(Th,d=2,color='LightGray',opacity=0.05)
for z in [0,0.5,1,1.5]:
  simlab.slice(Th,u,labels=[1],origin=(0,0,z),normal=(-1,0,1), opacity=0.6)
mlab.view(110,71,7)
```

Listing 15: 3D mesh : **slice** function

1.6 function SLICEISO

The VTK_SLICEISO function displays isolines of data on the intersection of a plane and a 3D mesh or parts of a 3D mesh defined by an SIMESH object.

Syntaxe

```
simlab.sliceiso(Th,u,P)
simlab.sliceiso(Th,u,Key=Value, ...)
```

Description

$\boxed{\text{simlab. sliceiso (Th,u)}}$ displays u data as isolines on the intersection of a plane and all the 3-dimensional simplices elements. By default the plane is given by an *origin* point $(0, 0, 0)$ which lies on it and a *normal* vector $(0, 0, 1)$ which is orthogonal to the plane.

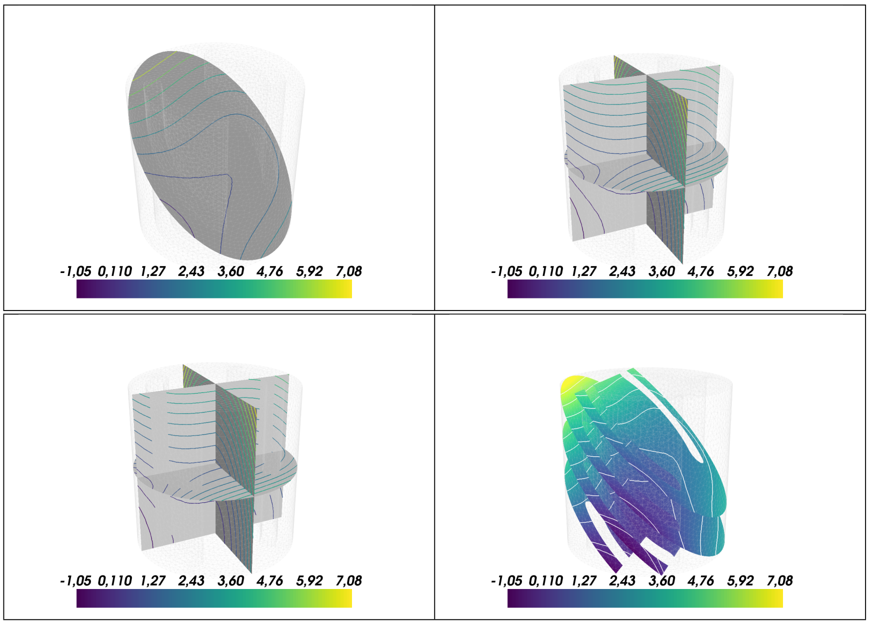
$\boxed{\text{simlab. sliceiso (Th,u,key=value, ...)}}$ allows additional key/value pairs to be used when displaying u. The key could be

- `labels` : to select the labels of the elements to intersect,
- `contours` : to specify the number of isolines (default : 10) or a list/numpy array of isovalues (default : empty)
- `color` : to specify one color for all isolines (default : empty)

For key/value pairs, one could also used those of the iso_surface function from mayavi.mlab.pipeline.

1.6.1 3D example

The following example use the *.geo* file `ball8.geo` which is in the directory `geodir` of the toolbox. This file contains description of a 3D mesh with simplices of dimensions 1, 2 and 3.




```python
meshfile=gmsh.buildmesh3d('cylinder3dom',15)
Th=siMesh(meshfile)
u=Th.feval(lambda x,y,z: 3*x**2-y**3+z**2+x*y)
mlab.figure(1)
simlab.sliceiso(Th,u,origin=(0,0,1),normal=(-1,0,1), colormap='viridis')
simlab.plotmesh(Th,d=2,color='LightGray',opacity=0.05)
simlab.slicemesh(Th,origin=(0,0,1),normal=(-1,0,1), color='Gray')
mlab.colorbar()
mlab.view(132,53,7)
mlab.figure(2)
simlab.plotmesh(Th,d=2,color='LightGray',opacity=0.05)
for normal in [(1,0,0),(0,1,0),(0,0,1)]:
  simlab.slicemesh(Th,origin=(0,0,1),normal=normal, color='LightGray')
  simlab.sliceiso(Th,u,contours=20,origin=(0,0,1),normal=normal, line_width=2)#,
      w_enabled=True)
mlab.colorbar()
mlab.view(155,66,7)
mlab.figure(3)
simlab.plotmesh(Th,d=2,color='LightGray',opacity=0.05)
for normal in [(1,0,0),(0,1,0),(0,0,1)]:
  simlab.slicemesh(Th,origin=(0,0,1),normal=normal, color='LightGray')
  simlab.sliceiso(Th,u, contours=20, labels=[1],origin=(0,0,1),normal=normal)
mlab.view(155,66,7)
mlab.colorbar()
mlab.figure(4)
simlab.plotmesh(Th,d=2,color='LightGray',opacity=0.05)
for z in [0,0.5,1,1.5]:
  simlab.slice(Th,u,labels=[1],origin=(0,0,z),normal=(-1,0,1) )
  simlab.sliceiso(Th,u,labels=[1],origin=(0,0,z),normal=(-1,0,1), color='white')
mlab.colorbar()
mlab.view(110,71,7)
```

Listing 16: 3D mesh : sliceiso function

1.7 function STREAMLINE

The STREAMLINE function allows to draw streamlines for given vector data and colorized by scalar data on a 3D mesh or parts of a 3D mesh defined by an SIMESH object. This supports various types of seed objects (line, sphere, plane and point seeds). It also allows to draw ribbons or tubes and further supports different types of interactive modes of calculating the streamlines.

Syntaxe

```
simlab.streamline(Th,u,V)
simlab.streamline(Th,u,V,Key=Value, ...)
```

Description

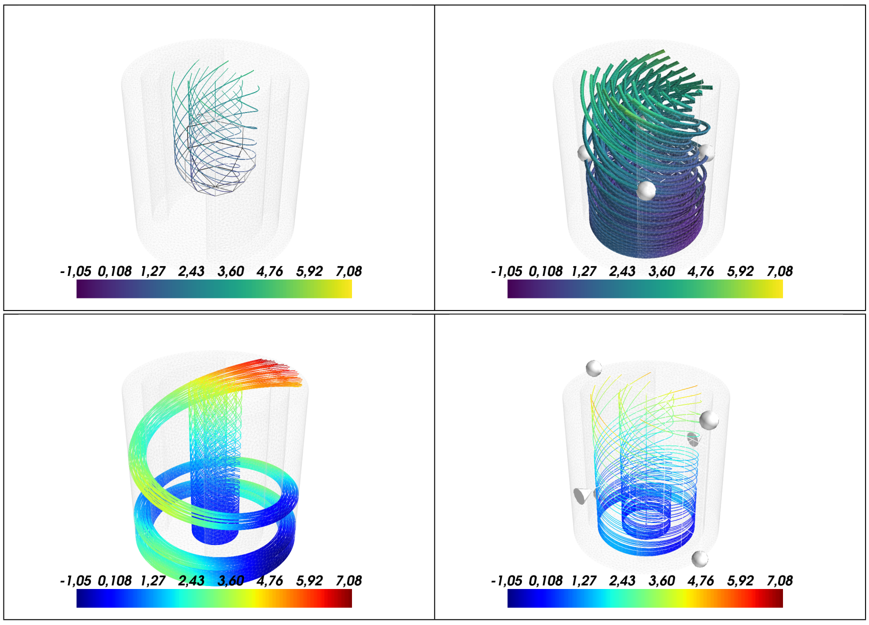
simlab.streamline(Th,u,V) displays streamlines computed from the vector data V and colorized by the scalar data u

simlab.streamline(Th,u,V,Key=Value, ...) specifies function options using one or more Key,Value pair arguments. Key could be:

- `labels` : to select the labels of the siMeshElt on which to draw the streamlines.
- `seed_options` :
- `seed_widget_options` :
- `streamtracer_options` :

The other Key/Value pair options are those of streamline function from mayavi.mlab.pipeline.

The following example use the *.geo* file `cylinder3dom.geo` which is in the directory `geodir` of the toolbox. This file contains description of a 3D mesh with simplices of dimensions 1, 2 and 3.




```
import numpy as np
meshfile=gmsh.buildmesh3d('cylinder3dom',20)
mlab.close(all=True)
Th=siMesh(meshfile)
U=Th.feval(lambda x,y,z: 3*x**2-y**3+z**2+x*y)
w=[lambda x,y,z: y*np.cos(-(x**2+y**2)/10), lambda x,y,z:
    -x*np.cos(-(x**2+y**2)/10), lambda x,y,z: z/5]
W=np.array([Th.feval(w[0]),Th.feval(w[1]),Th.feval(w[2])])
mlab.figure(1)
simlab.streamline(Th,U,W)
simlab.plotmesh(Th,d=2,color='LightGray',opacity=0.05)
mlab.colorbar()
mlab.figure(2)
s_options={'visible':True}
sw_options={'normal':(0,0,1),'resolution':6}
st_options={'integration_direction':'both'}
sl=simlab.streamline(Th,U,W,seedtype='plane',linetype='tube',
                     seed_options=s_options,
                     seed_widget_options=sw_options,
                     streamtracer_options=st_options)
simlab.plotmesh(Th,d=2,color='LightGray',opacity=0.05)
mlab.colorbar()

mlab.figure(3)
sw_options={'center':(0.9,0,1), 'radius':0.1,'phi_resolution':8,
            'theta_resolution':12,'enabled':False}
st_options={'integration_direction':'both'}
sl1=simlab.streamline(Th,U,W,seed_widget_options=sw_options,
                      streamtracer_options=st_options,colormap='jet')
sw_options['center']=(0,0,1)
sw_options['radius']=0.3
sl2=simlab.streamline(Th,U,W,seed_widget_options=sw_options,
                      streamtracer_options=st_options,colormap='jet')
mlab.scalarbar()
mlab.view(46.6,58,6.7)
simlab.plotmesh(Th,d=2,color='LightGray',opacity=0.05)

mlab.figure(4)
sw_options={'origin':(0,-1,0),'point1':(0,-1,2),'point2':(0,1,0),
            'enabled':True,'resolution':6}
st_options={'integration_direction':'both'}
sl1=simlab.streamline(Th,U,W,seedtype='plane',
                      seed_widget_options=sw_options,
                      streamtracer_options=st_options,colormap='jet')
mlab.scalarbar()
mlab.view(46.6,58,6.7)
simlab.plotmesh(Th,d=2,color='LightGray',opacity=0.05)
```

Listing 17: 3D mesh : streamline function