



fc siplt Octave package, User's Guide*

version 0.2.7

François Cuvelier[†]

January 30, 2025

Abstract

This Octave package uses a **siMesh** object, coming from the **fc-simesh** package, to display simplicial meshes or datas on simplicial meshes. Its kernel uses the **fc-graphics4mesh** package.

0 Contents

1	Introduction	2
2	Installation	3
3	fc_siplt.plotmesh function	5
3.1	2D example	6
3.2	3D example	6
3.3	3Ds example	8
4	fc_siplt.plot function	8
4.1	2D example	9
4.2	3D example	10
4.3	3Ds example	11
5	fc_siplt.plotiso function	12
5.1	2D example	13
5.2	3D example	14
5.3	3Ds example	15
6	fc_siplt.slicemesh function	16
6.1	3D example	16
7	fc_siplt.slice function	17
7.1	3D example	17

* $\text{\LaTeX}$ manual, revision 0.2.7.a, compiled with Octave 9.3.0, and packages **fc-siplt**[0.2.7], **fc-tools**[0.0.36], **fc-bench**[0.1.4], **fc-amat**[0.1.4], **fc-hypermesh**[1.0.5], **fc-meshtools**[0.1.5], **fc-graphics4mesh**[0.1.7], **fc-oogmsh**[0.3.1], **fc-simesh**[0.4.7]

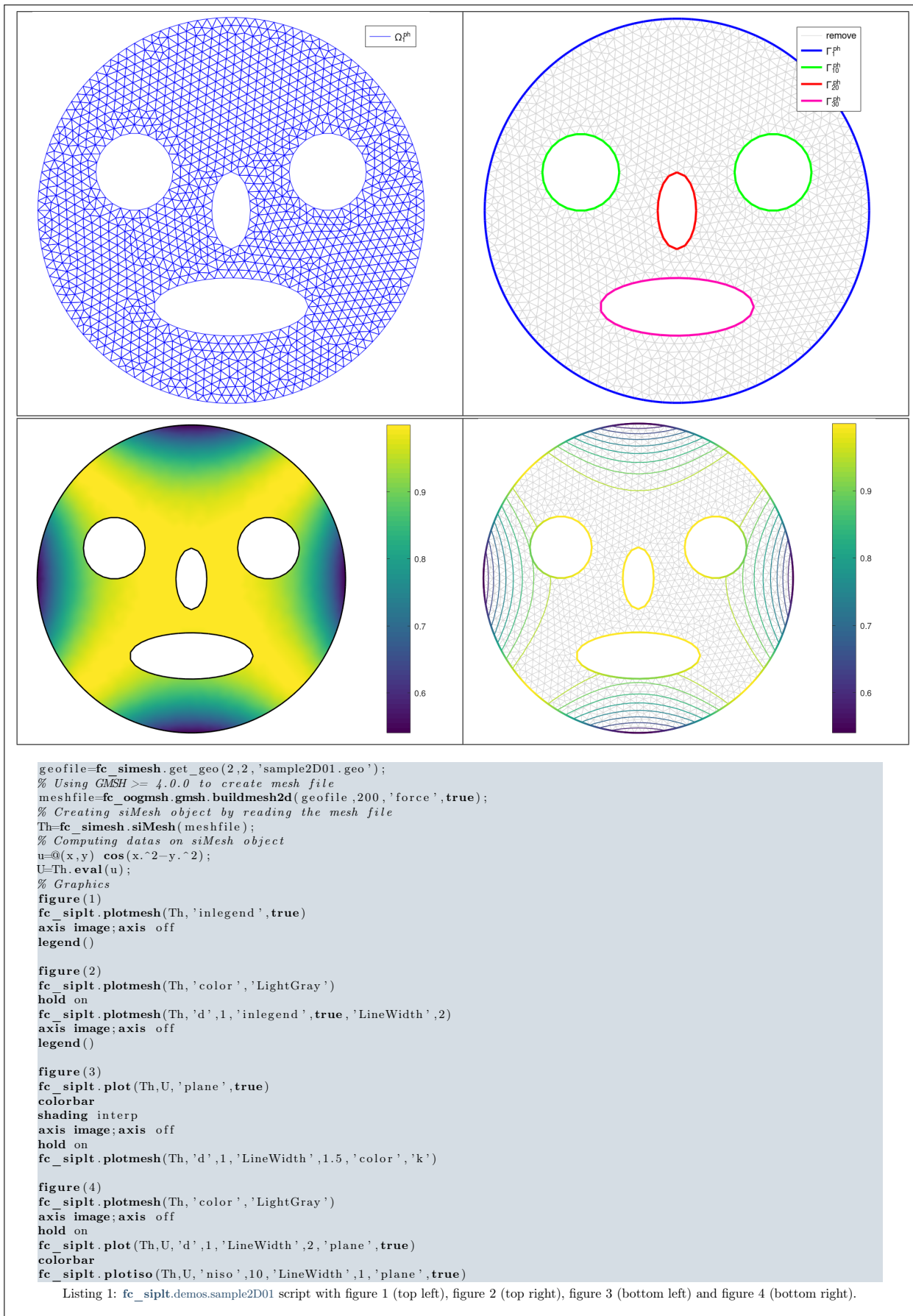
[†]Université Sorbonne Paris Nord, LAGA, CNRS, UMR 7539, F-93430, Villetaneuse, France, cuvelier@math.univ-paris13.fr.
This work was supported by the ANR project DEDALES under grant ANR-14-CE23-0005.

8	<code>fc_siplt.sliceiso</code> function	18
8.1	3D example	18
9	<code>fc_siplt.plotquiver</code> function	20
9.1	2D example	20
9.2	3D example	21
9.3	3Ds example	22
10	<code>fc_siplt.scatter</code> function	23
10.1	2D example	24
10.2	3Ds example	25
10.3	3D example	26
Appendices		28

1 Introduction

This **experimental** Octave package uses the [graphics4mesh](#) package[1] to do some graphic representations on a **siMesh** object of the [simesh](#) package[2].

In Listing 1, a 2D example is provided with the 4 generated figures. For graphic representations, one can also used `Th.plotmesh(...)` instead of `fc_siplt.plotmesh(Th,...)`, `Th.plot(...)` instead of `fc_siplt.plot(Th,...)` and so on.



2 Installation

This package was only tested on Ubuntu 24.04.1 with Octave 9.3.0.

One just has to get/download the install file

ofc_siplt_install.m

or get it on the dedicated web page. Thereafter, one run it under Octave. This command download, extract and configure the *fc-siplt* package and all the required packages in the current directory.

For example, to install this package in `~/Octave/packages` directory, one have to copy the file `mfc_siplt_install.m` in the `~/Octave/packages` directory. Then in a Octave terminal run the following commands

```
>> cd ~/Octave/packages
>> mfc_siplt_install
```

There is the output of the `mfc_siplt_install` command on a Linux computer:

```
Parts of the <fc-siplt> Octave package.
Copyright (C) 2017-2025 F. Cuvelier

1- Downloading and extracting the packages
2- Setting the <fc-siplt> package
Write in ~/Octave/fc-siplt-full/fc_siplt-0.2.7/configure_loc.m ...
3- Using packages :
->          fc-tools : 0.0.36
->          fc-bench : 0.1.4
->          fc-amat : 0.1.4
->          fc-hypermesh : 1.0.5
->          fc-meshtools : 0.1.5
->          fc-graphics4mesh : 0.1.7
->          fc-oogmsh : 0.3.1
->          fc-simesh : 0.4.7
with          fc-siplt : 0.2.7
*** Using instructions
To use the <fc-siplt> package:
addpath('~/Octave/fc-siplt-full/fc_siplt-0.2.7')
fc_siplt.init()

See ~/Octave/ofc_siplt_set.m
```

The complete package (i.e. with all the other needed packages) is stored in the directory

`~/Octave/packages/fc-siplt-full`

and, for each Octave session, one have to set the package by:

```
>> addpath('~/Octave/packages/fc-siplt-full/fc_siplt-0.2.7')
>> fc_siplt.init()
```

If it's the first time the `fc_siplt.init()` function is used, then its output is

```
Try to use default parameters!
Use fc_tools.configure to configure.
Write in ~/Octave/fc-siplt-full/fc_tools-0.0.36/configure_loc.m ...
Try to use default parameters!
Use fc_bench.configure to configure.
Write in ~/Octave/fc-siplt-full/fc_bench-0.1.4/configure_loc.m ...
Try to use default parameters!
Use fc_amat.configure to configure.
Write in ~/Octave/fc-siplt-full/fc_amat-0.1.4/configure_loc.m ...
Try to use default parameters!
Use fc_hypermesh.configure to configure.
Write in ~/Octave/fc-siplt-full/fc_hypermesh-1.0.5/configure_loc.m ...
Try to use default parameters!
Use fc_meshtools.configure to configure.
Write in ~/Octave/fc-siplt-full/fc_meshtools-0.1.5/configure_loc.m ...
Try to use default parameters!
Use fc_graphics4mesh.configure to configure.
Write in ~/Octave/fc-siplt-full/fc_graphics4mesh-0.1.7/configure_loc.m ...
Try to use default parameters!
Use fc_oogmsh.configure to configure.
Write in ~/Octave/fc-siplt-full/fc_oogmsh-0.3.1/configure_loc.m ...
Try to use default parameters!
Use fc_simesh.configure to configure.
Write in ~/Octave/fc-siplt-full/fc_simesh-0.4.7/configure_loc.m ...
Using fc_siplt[0.2.7] with fc_tools[0.0.36], fc_bench[0.1.4], fc_amat[0.1.4],
          fc_hypermesh[1.0.5], fc_meshtools[0.1.5], fc_graphics4mesh[0.1.7],
          fc_oogmsh[0.3.1], fc_simesh[0.4.7].
[fc-oogmsh] Configured to use gmsh 4.13.1 with default MSH file format version 4.1
```

Otherwise, the output of the `fc_siplt.init()` function is

```
Using fc_siplt[0.2.7] with fc_tools[0.0.36], fc_bench[0.1.4], fc_amat[0.1.4],
      fc_hypermesh[1.0.5], fc_meshtools[0.1.5], fc_graphics4mesh[0.1.7],
      fc_oogmsh[0.3.1], fc_simesh[0.4.7].
[fc-oogmsh] Configured to use gmsh 4.13.1 with default MSH file format version 4.1
```

For **uninstalling**, one just have to delete directory

~/Octave/packages/fc-siplt-full

3 `fc_siplt.plotmesh` function

The `fc_siplt.plotmesh` function displays the mesh or parts of the mesh defined by an `siMesh` object.

Syntaxe

```
fc_siplt.plotmesh(Th)
fc_siplt.plotmesh(Th,Name,Value, ...)
```

Description

`fc_siplt.plotmesh(Th)` displays all the (`Th.d`)-dimensional simplices elements of `Th`, a `siMesh` object.

`fc_siplt.plotmesh(Th,Name,Value, ...)` specifies function options using one or more `Name,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),
- 'inlegend' : add a legend name to graph if true (default : `false`)
- 'bounds' : If `true`, draw the borders of the selected elementaries mesh elements (only for 2-dimensional simplices). (default : `false`)
- 'cutPlane' : cut mesh by n plans given by n -by-4 array P where the equation of the i -th cut plane is given by

$$P(i,1)x + P(i,2)y + P(i,3)z + P(i,4) = 0.$$

The normal vector $P(i,1 : 3)$ pointed to the part of the mesh not displayed. (only for simplices in dimension 3) default : `[]` (no cut).

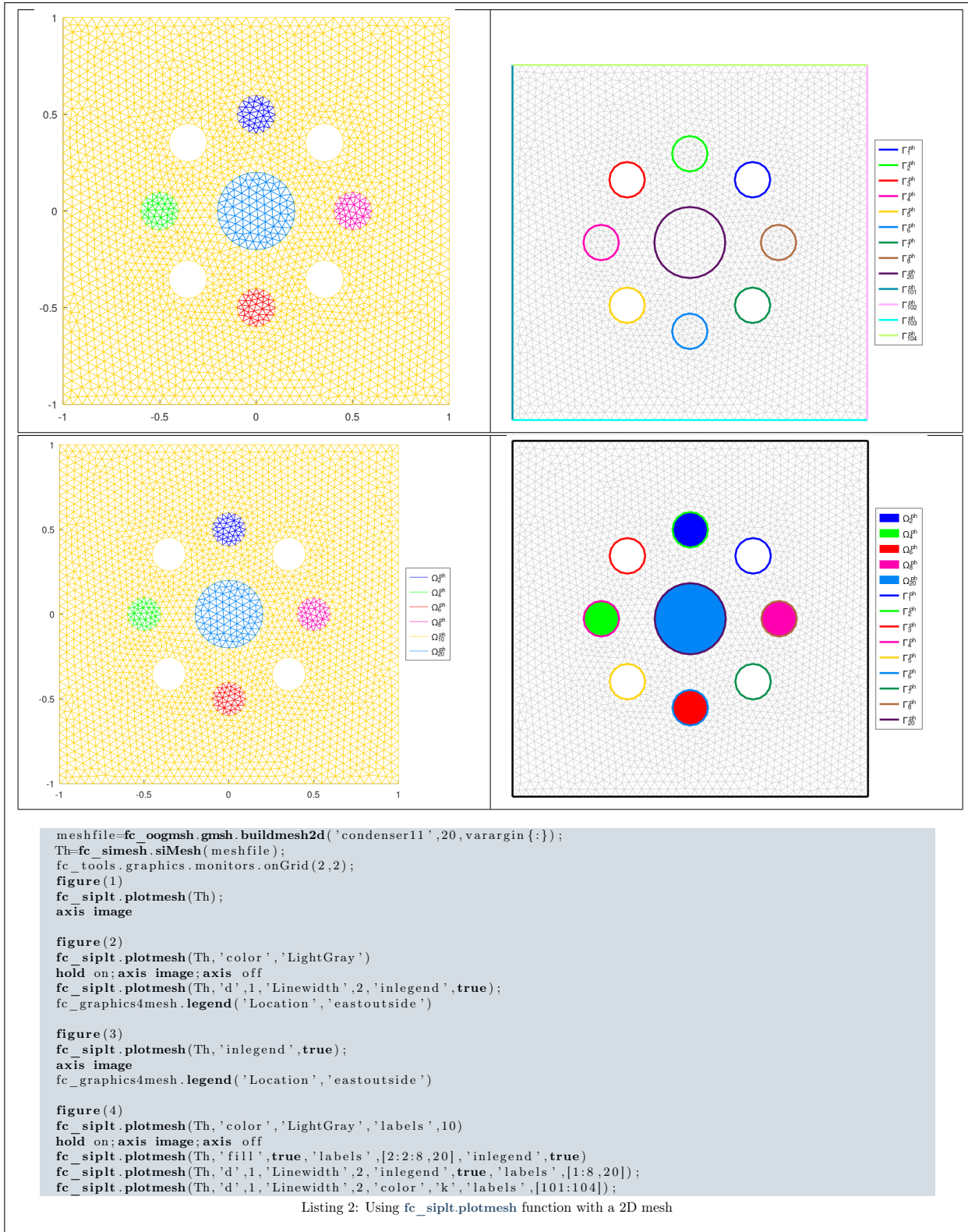
The options of second level depend on the type of elementaries mesh elements to represent.

One can use any option of the following functions according to the type of d -simplex to be represented.

- In dimension 3,
 - if $d == 3$, `patch` function is used,
 - if $d == 2$, `trimesh` function is used,
 - if $d == 1$, `plot3` function is used,
 - if $d == 0$, `plot3` function is used,
- In dimension 2,
 - if $d == 2$, `trimesh` function is used,
 - if $d == 1$, `plot` function is used,
 - if $d == 0$, `plot` function is used,
- In dimension 1,
 - if $d == 1$, `line` function is used,
 - if $d == 0$, `plot` function is used,

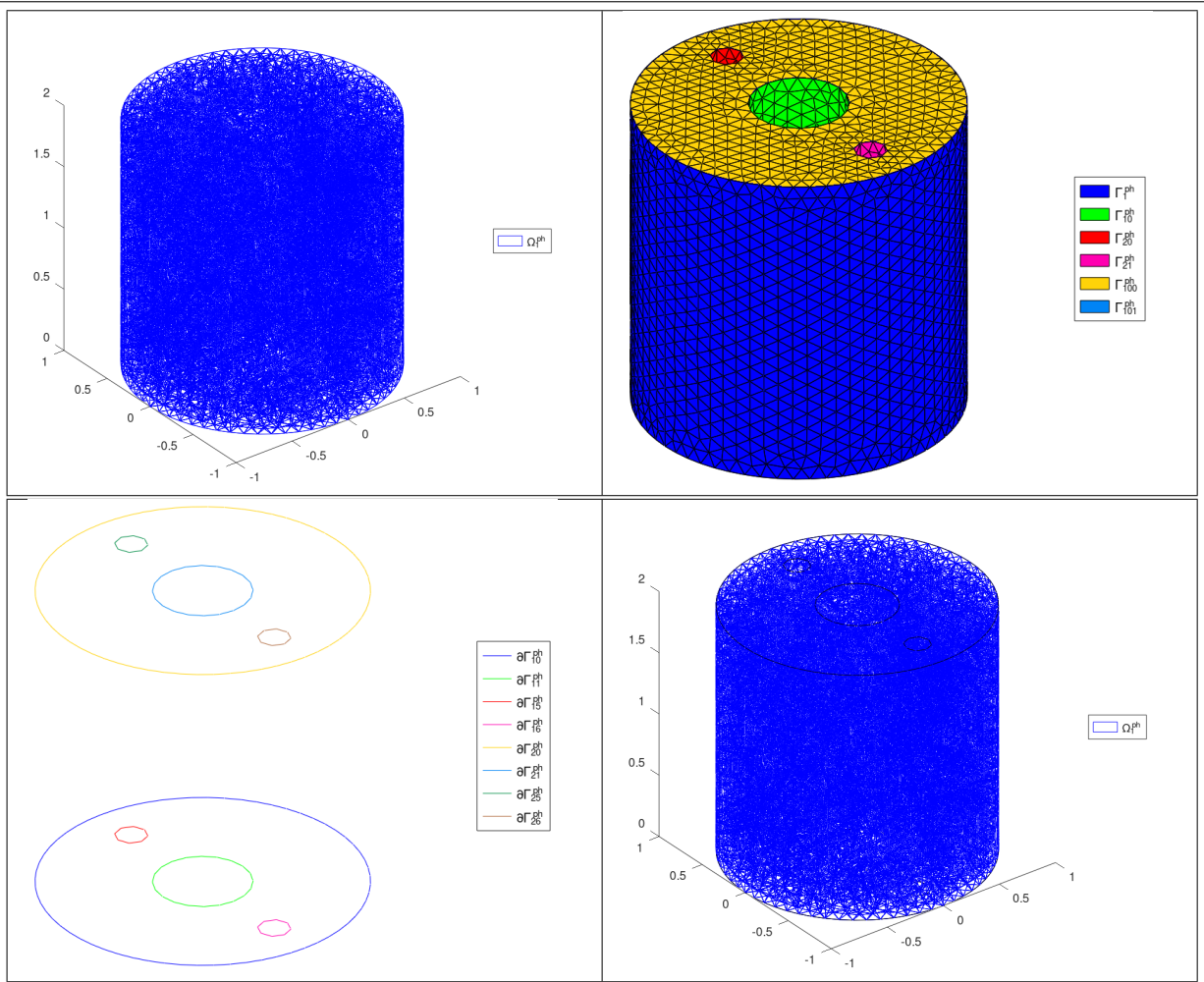
3.1 2D example

the following code is part of the `fc_siplt.demos.plotmesh2D` function.



3.2 3D example

the following code is part of the `fc_siplt.demos.plotmesh3D` function.



```

meshfile=fc_oogmsh.gmsh.buildmesh3d('cylinder3holes',10,varargin{:});
Th=fc_simesh.siMesh(meshfile);
fc_tools.graphics.monitors.onGrid(3,3,'figures',1:7);
figure(1)
fc_siplt.plotmesh(Th,'inlegend',true)
axis image
fc_graphics4mesh.legend('Location','eastoutside')

figure(2)
fc_siplt.plotmesh(Th,'d',2,'inlegend',true);
axis image;axis off
fc_graphics4mesh.legend('Location','eastoutside')

figure(3)
fc_siplt.plotmesh(Th,'d',1,'inlegend',true);
axis image;axis off
fc_graphics4mesh.legend('Location','eastoutside')

figure(4)
fc_siplt.plotmesh(Th,'inlegend',true)
hold on
fc_siplt.plotmesh(Th,'d',1,'color','k');
axis image
fc_graphics4mesh.legend('Location','eastoutside')

figure(5)
fc_siplt.plotmesh(Th,'d',2,'inlegend',true);
axis image
fc_graphics4mesh.legend('Location','eastoutside')

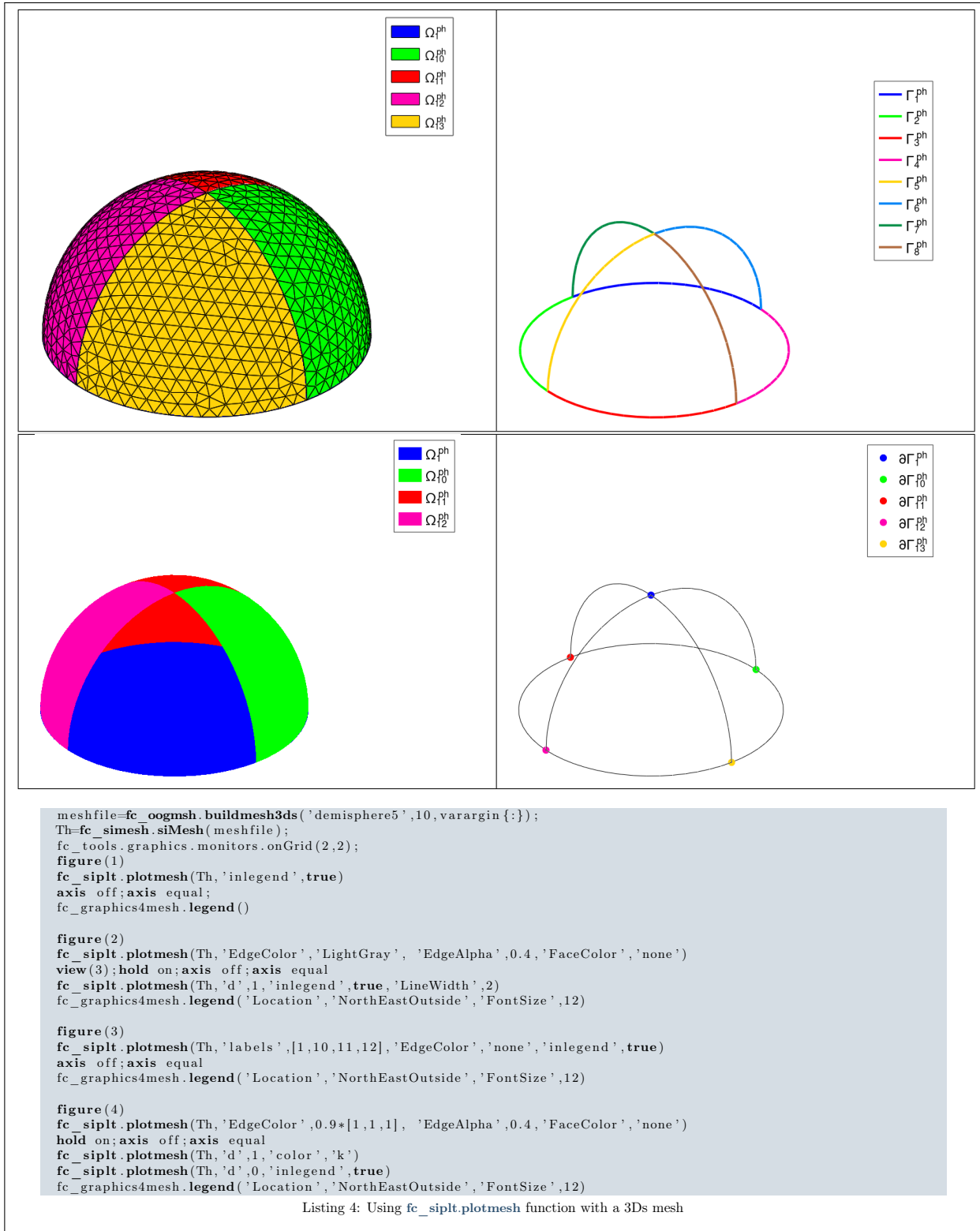
figure(6)
fc_siplt.plotmesh(Th,'d',2,'edgecolor',0.8*[1 1 1],'facecolor','None','edgealpha',0.5)
hold on;axis image
fc_siplt.plotmesh(Th,'d',1,'inlegend',true);
fc_graphics4mesh.legend('Location','eastoutside')

figure(7)
P=[fc_tools.graphics.PlaneCoefs([0 0 1],[0 0 1]); fc_tools.graphics.PlaneCoefs([0 0 1],[-1 0 0])];
fc_siplt.plotmesh(Th,'cutPlane',P,'Color','DarkGrey')
hold on;axis image
fc_siplt.plotmesh(Th,'d',2,'cutPlane',P,'inlegend',true);

```

Listing 3: Using `fc_siplt.plotmesh` function with a 3D mesh

the following code is part of the `fc_siplt.demos.plotmesh3Ds` function.



4 `fc_siplt.plot` function

The `fc_siplt.plot` function displays scalar datas on the mesh or parts of the mesh defined by an `siMesh` object.

Syntaxe

```
fc_siplt.plot(Th,u)
fc_siplt.plot(Th,u,Name,Value, ...)
```

Description

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

`fc_siplt.plot(Th,u,Name,Value, ...)` specifies function options using one or more `Name,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, made a 2D representation in the xy -plane, otherwise made a 3D representation with z -value set to `u` (default : `false`)

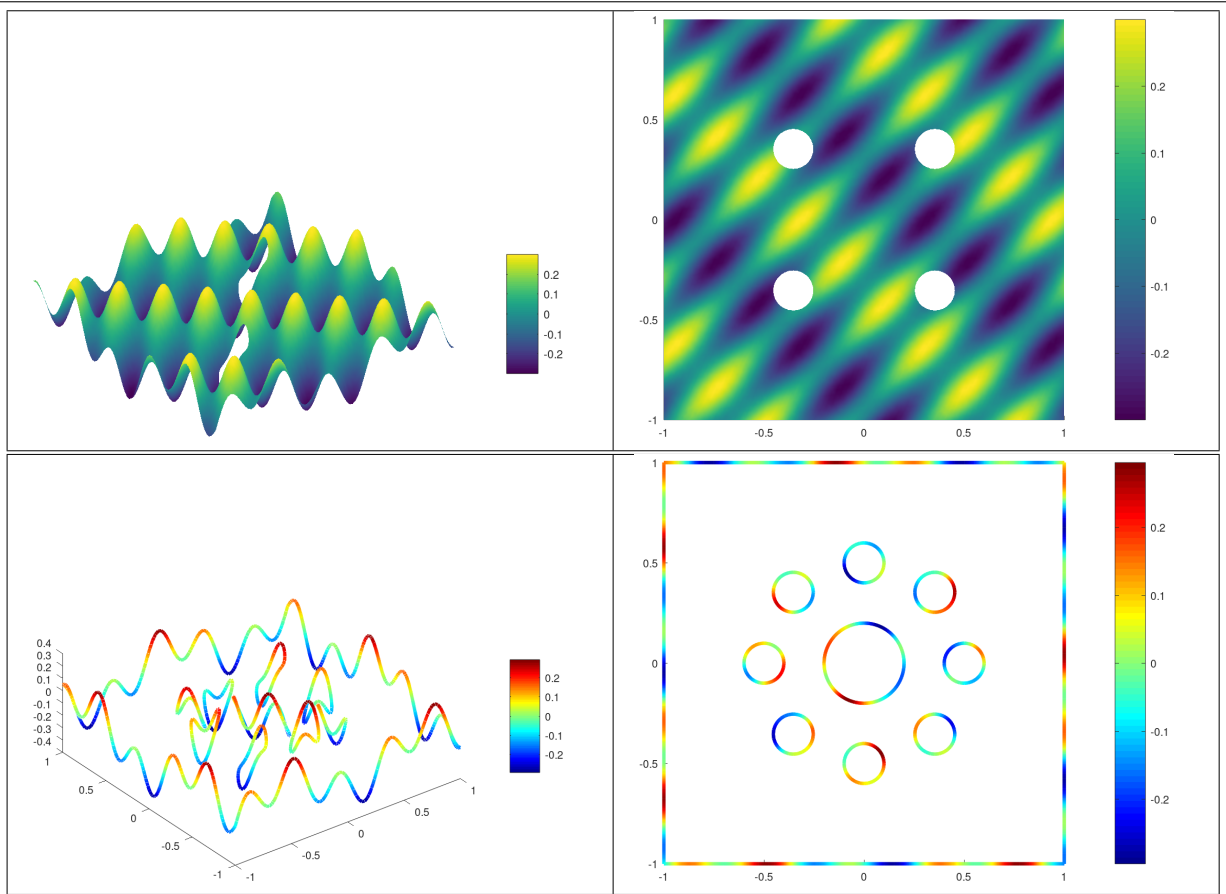
The options of second level depend on the type of elementaries mesh elements on which we want to represent datas.

One can use any option of the following functions according to the type of d -simplex.

- In dimension 3, `patch` function is used for $d \in \llbracket 1, 3 \rrbracket$.
- In dimension 2,
 - for $d == 2$, if `'plane'` option is true, `patch` function is used, otherwise it's `trisurf` function,
 - for $d == 1$, `patch` function is used.
- In dimension 1 and $d == 1$, `plot` function is used

4.1 2D example

the following code is part of the `fc_siplt.demos.plot2D` function.



```
meshfile=fc_oogmsh.gmsh.buildmesh2d('condenser11',40,varargin{:});
Th=fc_simesh.siMesh(meshfile);
u=@(x,y) 0.3*cos(5*(2*x-y)).*sin(5*(x-2*y));
U=Th.eval(u);
```

```
fc_tools.graphics.monitors.onGrid(2,2);
figure(1)
fc_siplt.plot(Th,U);
axis image;axis off;view(3)
colorbar
shading interp
```

```
figure(2)
fc_siplt.plot(Th,U,'plane',true);
axis image
colorbar
shading interp
```

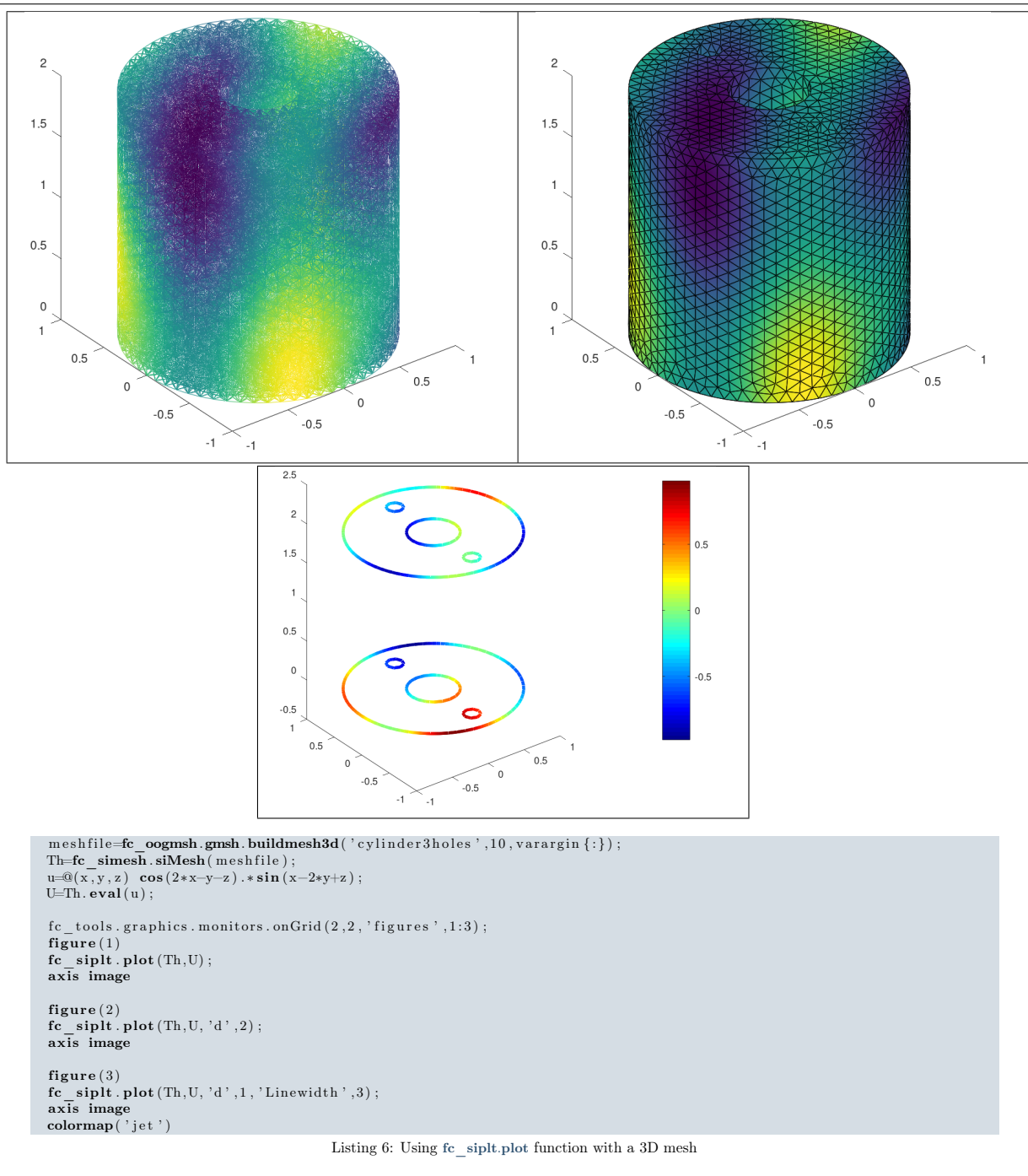
```
figure(3)
fc_siplt.plot(Th,U,'d',1,'Linewidth',3);
axis image
colormap('jet')
view(3)
colorbar
```

```
figure(4)
fc_siplt.plot(Th,U,'d',1,'Linewidth',3,'plane',true);
axis image
colormap('jet')
```

Listing 5: Using `fc_siplt.plot` function with a 2D mesh

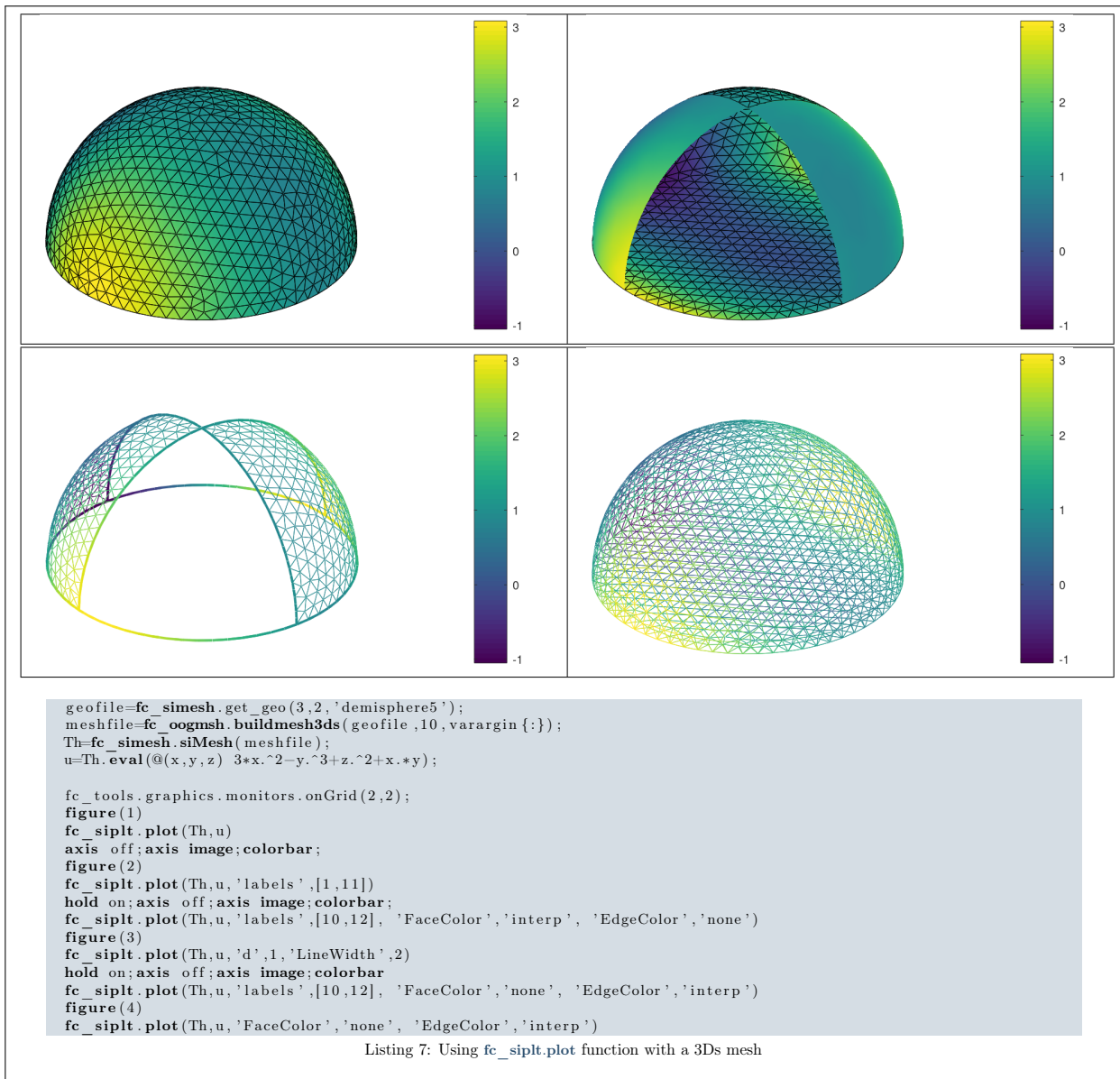
4.2 3D example

the following code is part of the `fc_siplt.demos.plot3D` function.



4.3 3Ds example

the following code is part of the `fc_siplt.demos.plot3Ds` function.



5 `fc_siplt.plotiso` function

The `fc_siplt.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

```

fc_siplt.plotiso(Th,u)
fc_siplt.plotiso(Th,u,Name,Value,...)

```

Description

`fc_siplt.plotiso(Th,u)` displays data `u` on all the 2-dimensional simplices elements of `Th`, a `siMesh` object.. The data `u` is an 1D-array of size `Th.nq` or `Th.nqGlobal` or `Th.nqParent`.

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

- `'niso'` : to specify the number of isolines (default : 10)
- `'isorange'` : to specify the list of isovalues (default : empty)
- `'isocolorbar'` : if `true`, colorbar with isovalues is drawn (default : `false`)

- `'format'` : to specify the format of the isovalues on the colorbar (default : `'%g'`)
- `'labels'` : to select the labels of the elements to display data,
- `'plane'` : if `true`, isolines are in the xy -plane, otherwise isolines are in 3D with z -value set to `u` (default : `false`)
- `'color'` : to specify one color for all isolines (default : empty)
- `'mouse'` : if `true`, display information on clicked isoline (default : `false`)

The options of second level are all options of

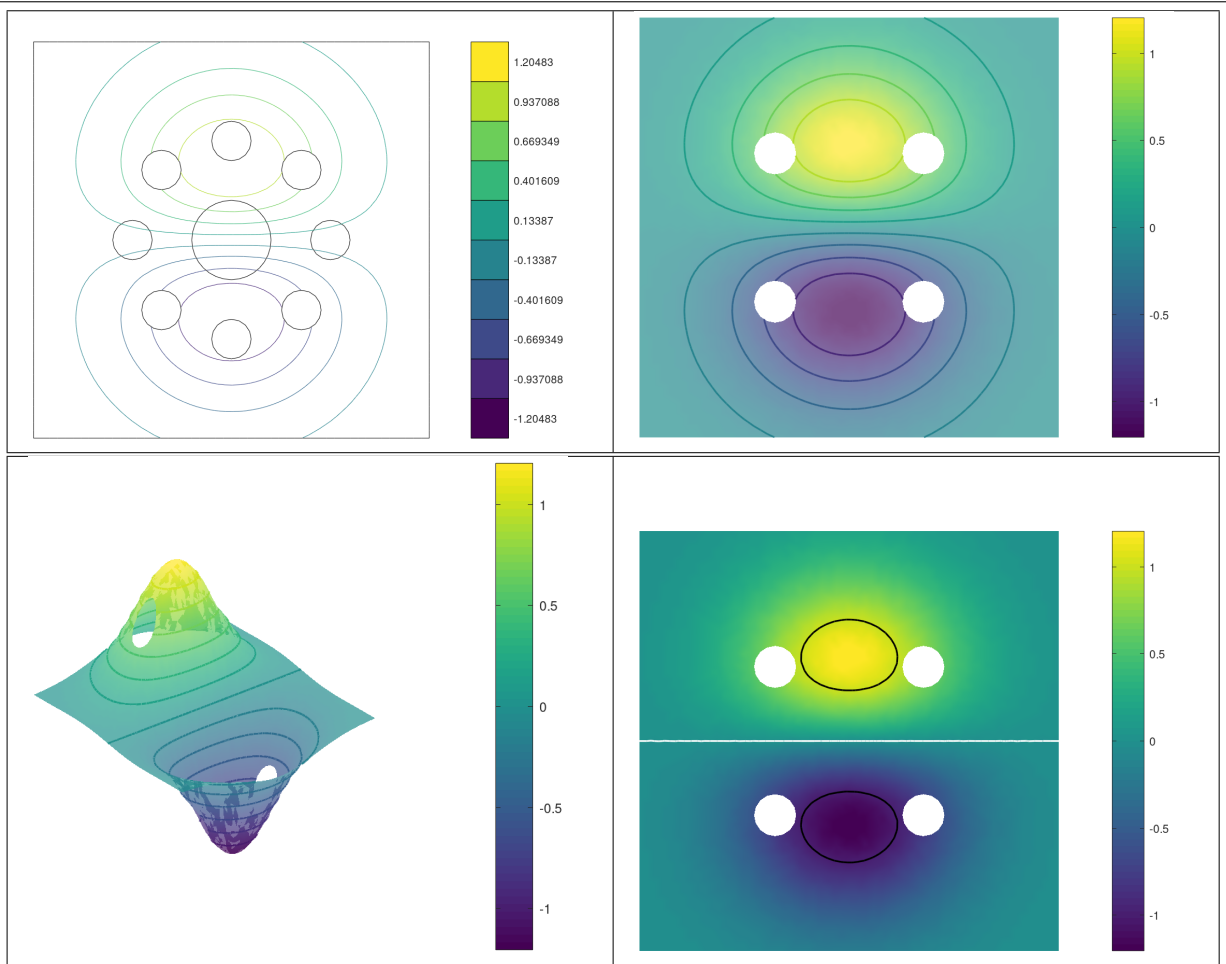
- `plot3` function in dimension 3 or in dimension 2 with `'plane'` option set to `false`
- `plot` function in 2 with `'plane'` option set to `true`

This function accepts until 4 output arguments :

- 1st output is the colors of the isolines
- 2nd output is the isovalues of the isolines
- 3th output is the handle of the colobar iso.
- 4th output is all the handles of the isolines as an 2D-array of dimension N-by-niso, where N is the number of 2-simplex elementary meshes where isolines are drawn.

5.1 2D example

the following code is part of the `fc_siplt.demos.plotiso2D` function.



```

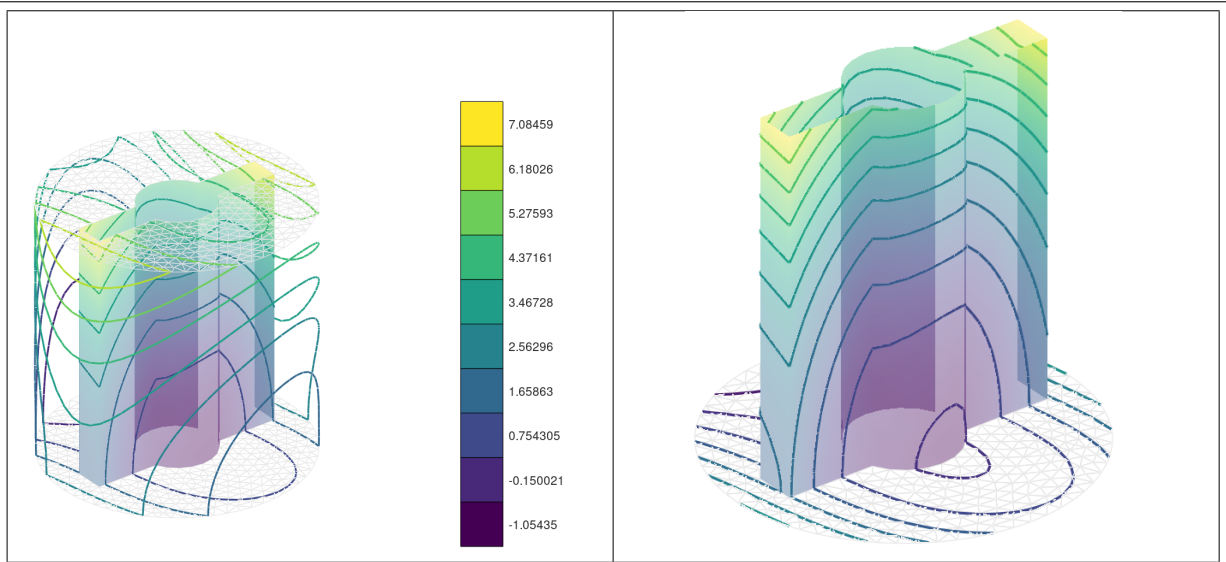
meshfile=fc_oogmsh.gmsh.buildmesh2d('condenser11',25,varargin{:});
Th=fc_simesh.siMesh(meshfile);
u=Th.eval(@(x,y) 5*exp(-3*(x.^2+y.^2)).*cos(x).*sin(y));
fc_tools.graphics.monitors.onGrid(2,2);
figure(1)
fc_siplt.plotmesh(Th,'d',1,'color','k')
hold on;axis off;axis image;
fc_siplt.plotiso(Th,u,'isocolorbar',true)
figure(2)
fc_siplt.plot(Th,u,'plane',true,'FaceAlpha',0.7)
hold on;axis off;axis image;shading interp;
fc_siplt.plotiso(Th,u,'plane',true,'LineWidth',1.5)
colorbar
figure(3)
fc_siplt.plot(Th,u,'FaceAlpha',0.7)
view(3)
shading interp;hold on;axis off;axis image;
fc_siplt.plotiso(Th,u,'niso',15,'LineWidth',1.5)
colorbar
figure(4)
fc_siplt.plot(Th,u,'plane',true)
shading interp;hold on;axis off;axis image;
fc_siplt.plotiso(Th,u,'isorange',0,'LineWidth',1.5,'color','w')
fc_siplt.plotiso(Th,u,'isorange',[-1,1],'LineWidth',1.5,'color','k','plane',true)

```

Listing 8: Using `fc_siplt.plotiso` function with a 2D mesh

5.2 3D example

the following code is part of the `fc_siplt.demos.plotiso3D` function.



```

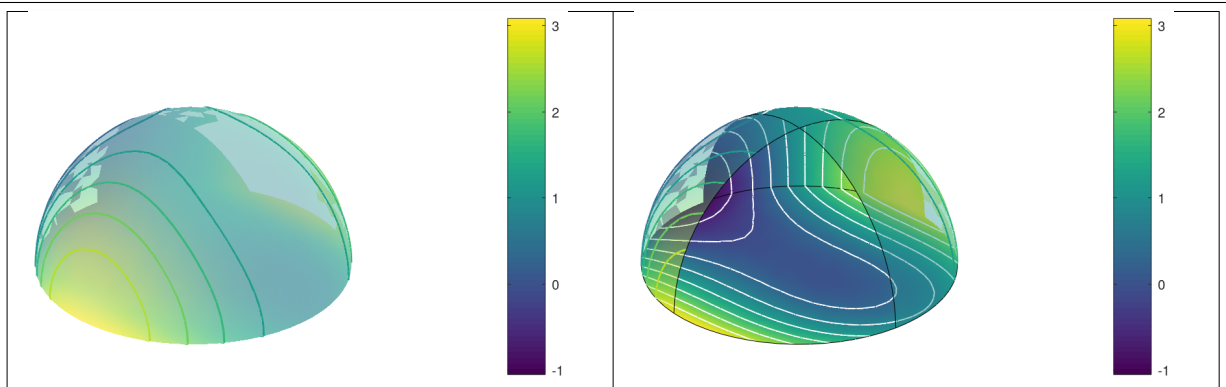
geofile=fc_simesh.get_geo(3,3,'cylinderkey');
meshfile=fc_oogmsh.gmsh.buildmesh3d(geofile,10,varargin{:});
Th=fc_simesh.siMesh(meshfile);
u=Th.eval(@(x,y,z) 3*x.^2-y.^3+z.^2+x.*y);
fc_tools.graphics.monitors.onGrid(1,2);
figure(1)
fc_siplt.plot(Th,u,'d',2,'labels',[10,11,31],'FaceColor','interp','EdgeColor','none','FaceAlpha',0.4)
hold on;view(3);axis off;axis equal;
fc_siplt.plotmesh(Th,'d',2,'labels',[1000,1020,1021,2000,2020,2021],'FaceColor','none','EdgeColor',0.9*[1,1,1])
fc_siplt.plotiso(Th,u,'isocolorbar',true,'LineWidth',1.5)
figure(2)
fc_siplt.plot(Th,u,'d',2,'labels',[10,11,31],'FaceColor','interp','EdgeColor','none','FaceAlpha',0.4)
hold on;axis off;axis equal;
fc_siplt.plotmesh(Th,'d',2,'labels',[2000,2020,2021],'FaceColor','none','EdgeColor',0.9*[1,1,1])
fc_siplt.plotiso(Th,u,'labels',[10,11,31,2000,2020,2021],'LineWidth',1.5,'niso',15)

```

Listing 9: Using `fc_siplt.plotiso` function with a 3D mesh

5.3 3Ds example

the following code is part of the `fc_siplt.demos.plotiso3Ds` function.



```

meshfile=fc_oogmsh.gmsh.buildmesh3ds('demisphere5',10,varargin{:});
Th=fc_simesh.siMesh(meshfile);

u=Th.eval(@(x,y,z) 3*x.^2-y.^3+z.^2+x.*y);
fc_tools.graphics.monitors.onGrid(1,2);
figure(1)
fc_siplt.plot(Th,u,'FaceColor','interp','EdgeColor','none','FaceAlpha',0.4)
hold on;axis off;axis equal;colorbar
fc_siplt.plotiso(Th,u,'LineWidth',1.5)

figure(2)
fc_siplt.plot(Th,u,'labels',[1,11],'FaceColor','interp','EdgeColor','none')
hold on;axis off;axis equal;colorbar;
fc_siplt.plotiso(Th,u,'labels',[1,11],'LineWidth',1,'color','w')
fc_siplt.plot(Th,u,'labels',[10,12],'FaceColor','interp','EdgeColor','none','FaceAlpha',0.4)
fc_siplt.plotiso(Th,u,'labels',[10,12],'LineWidth',1.5)

```

Listing 10: Using `fc_siplt.plotiso` function with a 3Ds mesh

6 `fc_siplt.slicemesh` function

The `fc_siplt.slicemesh` function displays intersection of a plane and a 3D mesh or parts of a 3D mesh defined by an `siMesh` object.

Syntaxe

```
fc_siplt.slicemesh(Th,P)
fc_siplt.slicemesh(Th,P,Name,Value,...)
```

Description

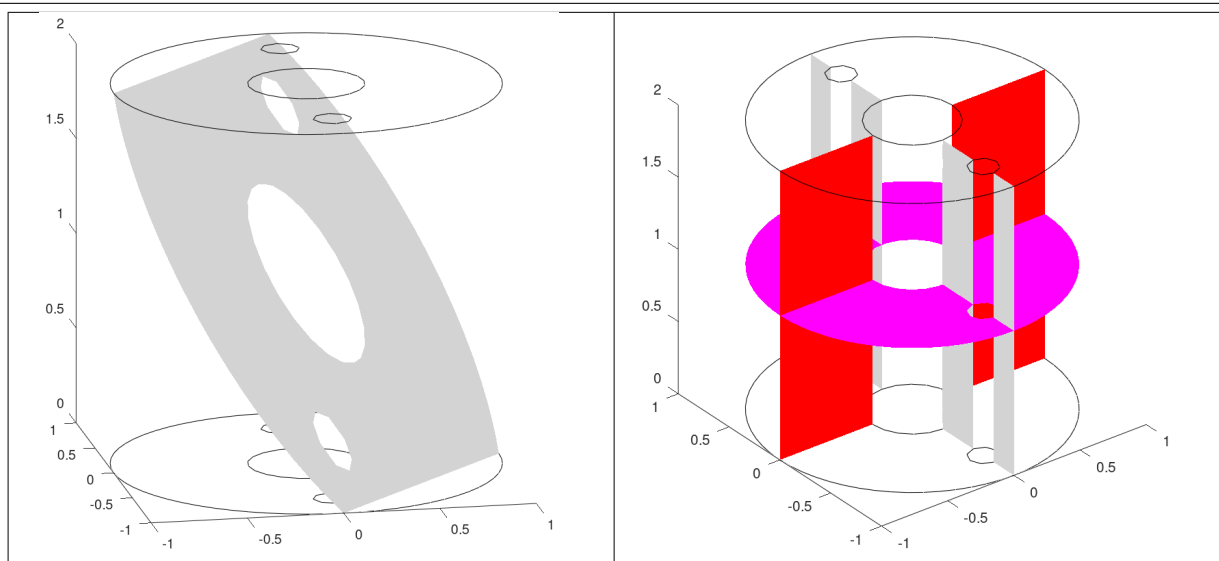
`fc_siplt.slicemesh(Th,P)` displays intersection of the plane defined by $P(1)x + P(2)y + P(3)z + P(4) = 0$ and all the 3-dimensional simplices elements of `Th`, a `siMesh` object. To compute P one can use the function `fc_tools.graphics.PlaneCoefs` of the `fc_tools` package. With this function, the array P , is obtained with $P = \text{fc_tools.graphics.PlaneCoefs}(Q,V)$ where Q is a point in the plane and V is a vector orthogonal to it.

`fc_siplt.slicemesh(Th,P,Name,Value,...)` specifies function options using one or more `Name,Value` pair arguments. Options of first level are

- 'color' : to specify the slice color (default : 'lightgrey', `rgb=[0.9,0.9,0.9]`)
- 'labels' : to select the labels of the elements to intersect,

6.1 3D example

the following code is part of the `fc_siplt.demos.slicemesh3D` function.



```
meshfile=fc_oogmsh.buildmesh3d('cylinder3holes',10,varargin{:});
Th=fc_simesh.siMesh(meshfile);
fc_tools.graphics.monitors.onGrid(1,2);
figure(1)
P=fc_tools.graphics.PlaneCoefs([0 0 1],[1 -1 1]);
fc_siplt.slicemesh(Th,P)
hold on;axis equal;
fc_siplt.plotmesh(Th,'d',1,'color','k')
view(-11,15)

figure(2)
P=[fc_tools.graphics.PlaneCoefs([0 0 1],[1 0 0]); ...
   fc_tools.graphics.PlaneCoefs([0 0 1],[0 1 0]); ...
   fc_tools.graphics.PlaneCoefs([0 0 1],[0 0 1])];
fc_siplt.slicemesh(Th,P,'color',{'LightGray',[1,0,0],'m'}) % [1,0,0] -> RGB => red
hold on;axis equal;
fc_siplt.plotmesh(Th,'d',1,'color','k')
```

Listing 11: Using `fc_siplt.slicemesh` function with a 3D mesh

7 `fc_siplt.slice` function

The method `fc_siplt.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

```
fc_siplt.slice(Th,u,P)
fc_siplt.slice(Th,u,P,Name,Value, ...)
```

Description

`fc_siplt.slice(Th,u,P)` displays `u` data on the intersection of the plane defined by $P(1)x + P(2)y + P(3)z + P(4) = 0$ and all the 3-dimensional simplices elements of `Th`, a `siMesh` object. The data `u` is an 1D-array of size `Th.nq` or `Th.nqGlobal` or `Th.nqParent`. To compute `P` one can use the function `fc_tools.graphics.PlaneCoefs` of the `fc_tools` package. With this function, the array `P`, is obtained with `P=fc_tools.graphics.PlaneCoefs(Q,V)` where `Q` is a point in the plane and `V` is a vector orthogonal to it.

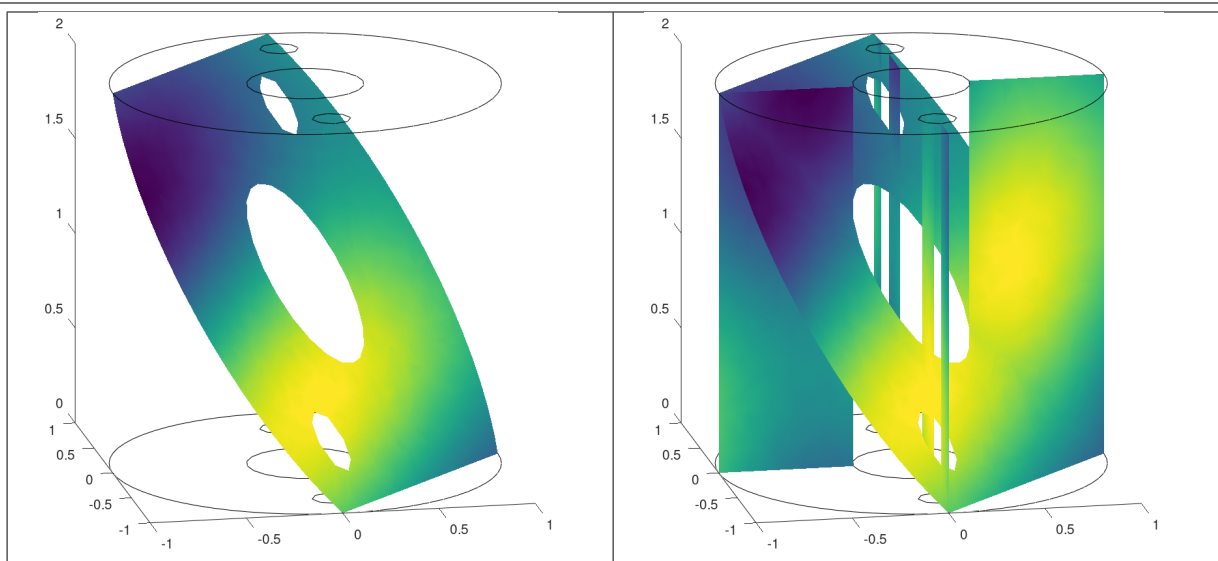
`fc_siplt.slice(Th,u,P,Name,Value, ...)` specifies function options using one or more `Name,Value` pair arguments.

Options of first level are

- `'labels'` : to select the labels of the elements to intersect,

7.1 3D example

the following code is part of the `fc_siplt.demos.slice3D` function.



```
meshfile=fc_oogmsh.buildmesh3d('cylinder3holes',10,varargin{:});
Th=fc_simesh.siMesh(meshfile);

u=@(x,y,z) cos(2*x-y-z).*sin(x-2*y+z);
U=Th.eval(u);

figure(1)
P=fc_tools.graphics.PlaneCoefs([0 0 1],[1 -1 1]);
fc_siplt.slice(Th,U,P)
hold on
fc_siplt.plotmesh(Th,'d',1,'color','k')
axis equal
view(-11,15)

P=[fc_tools.graphics.PlaneCoefs([0 0 1/2],[1 0 0]); ...
   fc_tools.graphics.PlaneCoefs([0 0 1/2],[0 1 0]); ...
   fc_tools.graphics.PlaneCoefs([0 0 1],[1 -1 1])];
figure(2)
fc_siplt.slice(Th,U,P)
hold on
fc_siplt.plotmesh(Th,'d',1,'color','k')
axis equal
```

Listing 12: Using `fc_siplt.slice` function with a 3D mesh

8 `fc_siplt.sliceiso` function

The `fc_siplt.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

```
fc_siplt.sliceiso(Th,u,P)
fc_siplt.sliceiso(Th,u,P,Name,Value, ...)
```

Description

`fc_siplt.sliceiso(Th,u,P)` displays `u` data as isolines on the intersection of the plane defined by $P(1)x + P(2)y + P(3)z + P(4) = 0$ and all the 3-dimensional simplices elements of `Th`, a `siMesh` object. The data `u` is an 1D-array of size `Th.nq` or `Th.nqGlobal` or `Th.nqParent`. To compute P one can use the function `fc_tools.graphics.PlaneCoefs` of the `fc_tools` package. With this function, the array `P`, is obtained with $P = \text{fc_tools.graphics.PlaneCoefs}(Q,V)$ where `Q` is a point in the plane and `V` is a vector orthogonal to the plane.

`fc_siplt.sliceiso(Th,u,P,key,value, ...)` allows additional key/value pairs to be used when displaying `u`. The key strings could be

- `'labels'` : to select the labels of the elements to intersect,
- `'niso'` : to specify the number of isolines (default : 10)
- `'isorange'` : to specify the list of isovalues (default : empty)
- `'color'` : to specify one color for all isolines (default : empty)
- `'isocolorbar'` : if true display a colorbar. Default is false.
- `'format'` : to specify the format of the isovalues print in the colorbar. Default is `'%g'`.

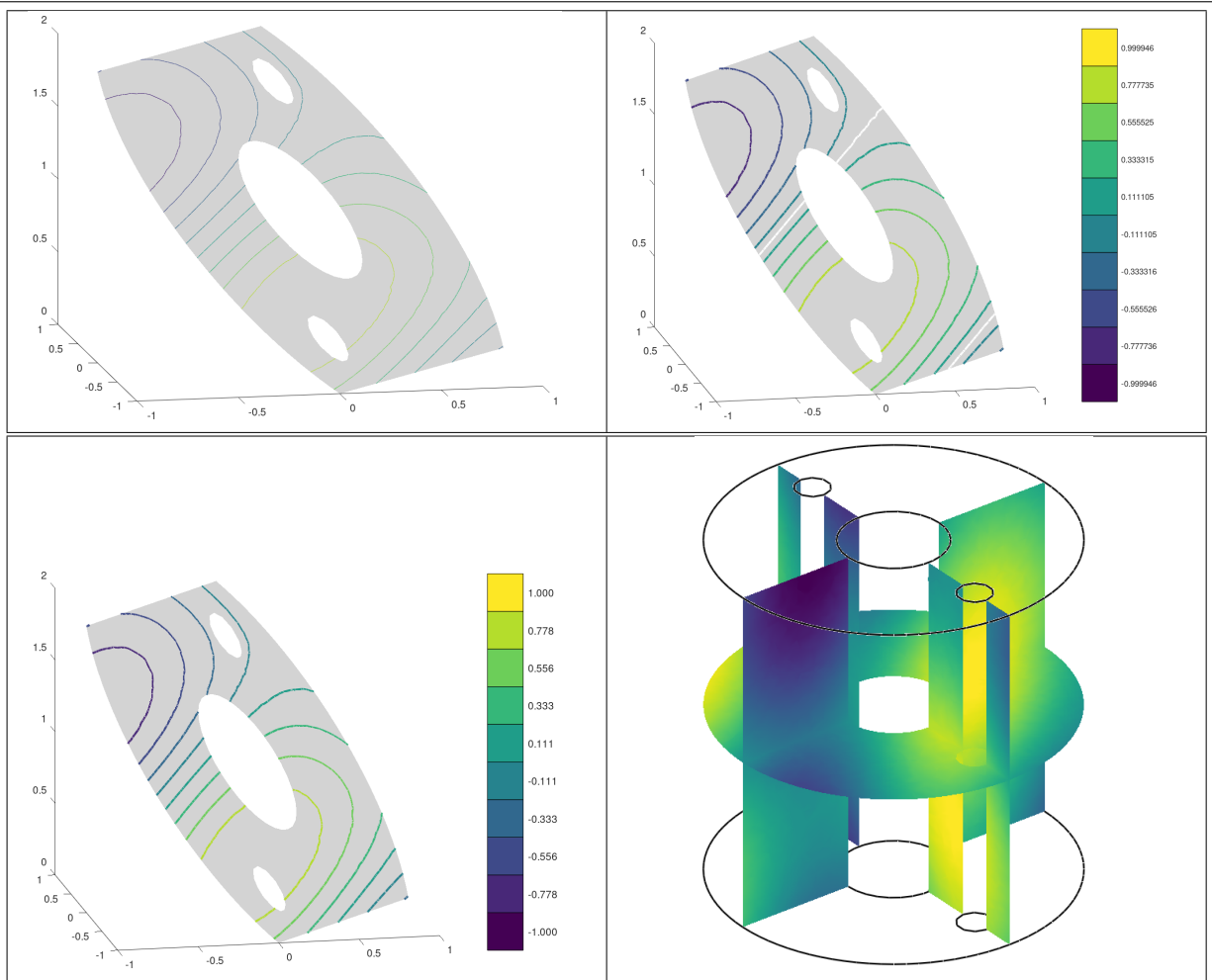
For key strings, one could also used any options of the `plot3` function.

This function accepts until 4 output arguments :

- 1st output is the colors of the isolines
- 2nd output is the isovalues of the isolines
- 3th output is the handle of the colobar iso.
- 4th output is all the handles of the isolines as an 2D-array of dimension N-by-niso, where N is the number of elementary meshes where isolines are drawn.

8.1 3D example

the following code is part of the `fc_siplt.demos.sliceiso3D` function.



```

meshfile=fc_oogmsh.buildmesh3d('cylinder3holes',15,varargin{:});
Th=fc_simesh.siMesh(meshfile);
u=@(x,y,z) cos(2*x-y-z).*sin(x-2*y+z);
U=Th.eval(u);

fc_tools.graphics.monitors.onGrid(2,3,'figures',1:5);
figure(1)
P=fc_tools.graphics.PlaneCoefs([0 0 1],[1 -1 1]);
fc_siplt.slicemesh(Th,P)
hold on
fc_siplt.sliceiso(Th,U,P)
view(-11,15)

figure(2)
P=fc_tools.graphics.PlaneCoefs([0 0 1],[1 -1 1]);
fc_siplt.slicemesh(Th,P)
hold on
[~,~,cax,h]=fc_siplt.sliceiso(Th,U,P,'isocolorbar',true);
fc_siplt.sliceiso(Th,U,P,'isorange',0,'color','w','LineWidth',2);
view(-11,15)
I=~isnan(h);
set(h(I),'LineWidth',2)
set(cax,'FontSize',8)

figure(3)
P=fc_tools.graphics.PlaneCoefs([0 0 1],[1 -1 1]);
fc_siplt.slicemesh(Th,P)
hold on
fc_siplt.sliceiso(Th,U,P,'isocolorbar',true,'LineWidth',2,'format','%3f');
view(-11,15)

figure(4)
P=[fc_tools.graphics.PlaneCoefs([0 0 1],[1 0 0]); ...
fc_tools.graphics.PlaneCoefs([0 0 1],[0 1 0]); ...
fc_tools.graphics.PlaneCoefs([0 0 1],[0 0 1])];
fc_siplt.plotmesh(Th,'d',1,'LineWidth',1,'color','k');
hold on;axis off;axis image;
fc_siplt.slice(Th,u,P,'Facecolor','interp')

figure(5)
fc_siplt.plotmesh(Th,'d',1,'LineWidth',2,'color','k');
hold on;axis off;axis image;
fc_siplt.slicemesh(Th,P(1:2,:))
fc_siplt.sliceiso(Th,U,P(1:2,:))
fc_siplt.slice(Th,U,P(3,:), 'Facecolor','interp')

```

Listing 13: Using `fc_siplt.sliceiso` function with a 3D mesh

9 `fc_siplt.plotquiver` function

The `fc_siplt.plotquiver` function displays vector field datas on the mesh or parts of the mesh defined by an `siMesh` object.

Syntaxe

```
fc_siplt.plotquiver(Th,V)
fc_siplt.plotquiver(Th,V,Key,Value,...)
```

Description

`fc_siplt.plotquiver(Th,V)` displays vector field `U` on all the `d`-dimensional simplices elements in dimension $d = 2$ or $d = 3$. The data `V` is an 2D-array of size `Th.nq`-by- d or 2-by-`Th.nq`.

`fc_siplt.plotquiver(Th,V,Key,Value,...)` specifies function options using one or more `Key,Value` pair arguments. Options of first level are

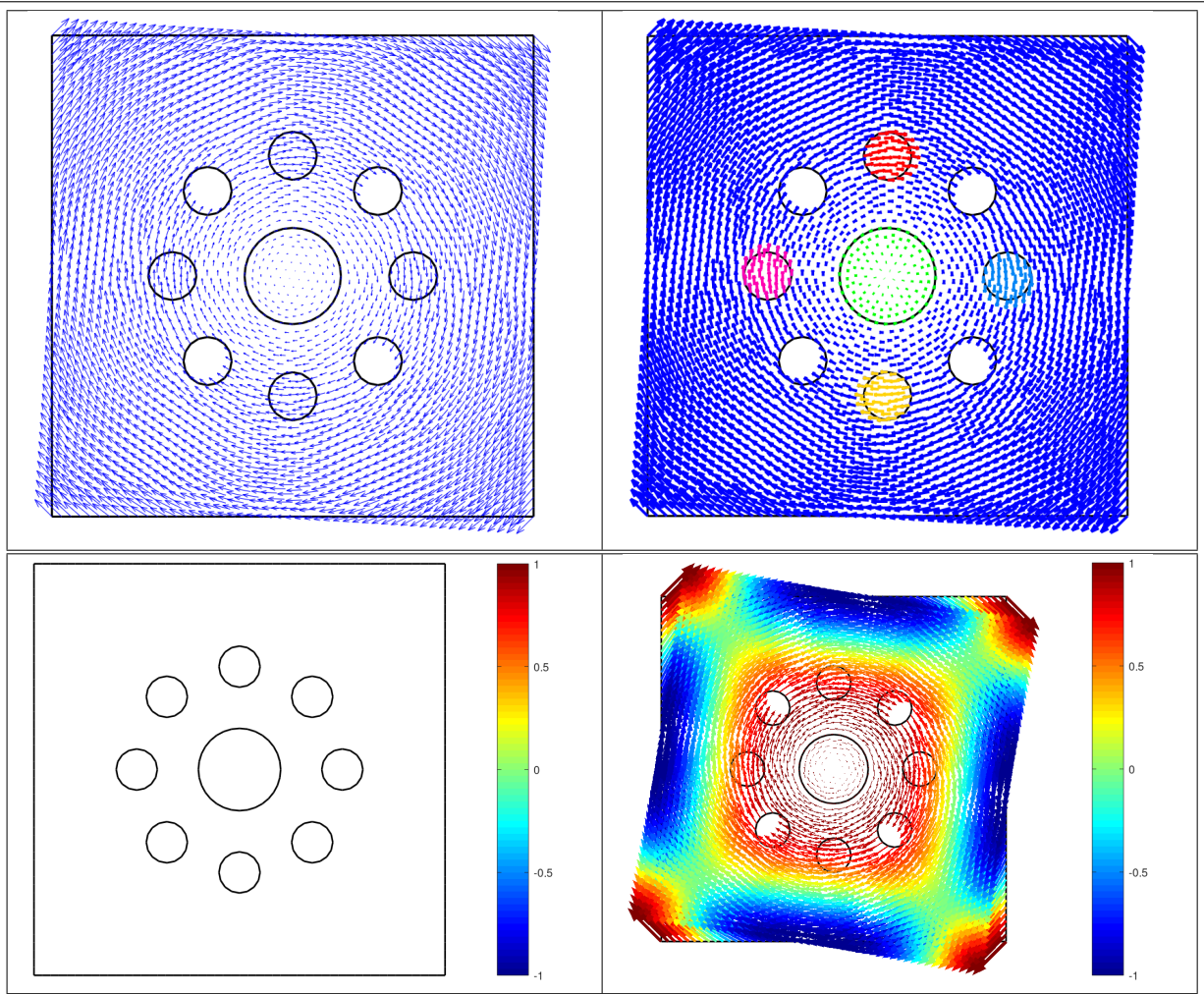
- `'labels'` : to select the labels of the elements to display data,
- `'freq'` : quiver frequencie, (default : 1)
- `'scale'` : quiver scale, (default : ...)
- `'colordata'` : set colors on each quiver (default : empty).

The options of second level depend on space dimension and `'colordata'` option. One can use any option of the following functions

- `quiver` function in dimension 2 with an empty `'colordata'`
- `quiver3` function in dimension 3 with an empty `'colordata'`
- `vfield3` function in dimension 2 or 3 with `'colordata'` set to an 1D-array of length `Th.nq`.

9.1 2D example

the following code is part of the `fc_siplt.demos.plotquiver2D` function.



```

geofile=fc_simesh.get_geo(2,2,'condenser11');
meshfile=fc_oogmsh.buildmesh2d(geofile,25,varargin{:});
Th=fc_simesh.siMesh(meshfile);

u=@(x,y) cos(pi*x.^2).*cos(pi*y.^2);
U=Th.eval(u);
w=@(x,y) y.*cos(-(x.^2+y.^2)/10),@(x,y) -x.*cos(-(x.^2+y.^2)/10);
W=Th.eval(w);
fc_tools.graphics.monitors.onGrid(2,2);
figure(1)
fc_siplt.plotmesh(Th,'d',1,'color','k','LineWidth',1.5)
hold on;axis off;axis image
fc_siplt.plotquiver(Th,W)

figure(2)
fc_siplt.plotmesh(Th,'d',1,'color','k','LineWidth',1.5)
hold on;axis off;axis image
fc_siplt.plotquiver(Th,W,'LineWidth',2,'merge',false)

figure(3)
fc_siplt.plotmesh(Th,'d',1,'color','k','LineWidth',1.5)
hold on;axis off;axis image
fc_siplt.plotquiver(Th,W,'colordata',U,'labels',[2:2:8,20])
caxis([min(U) max(U)])
colormap('jet');colorbar

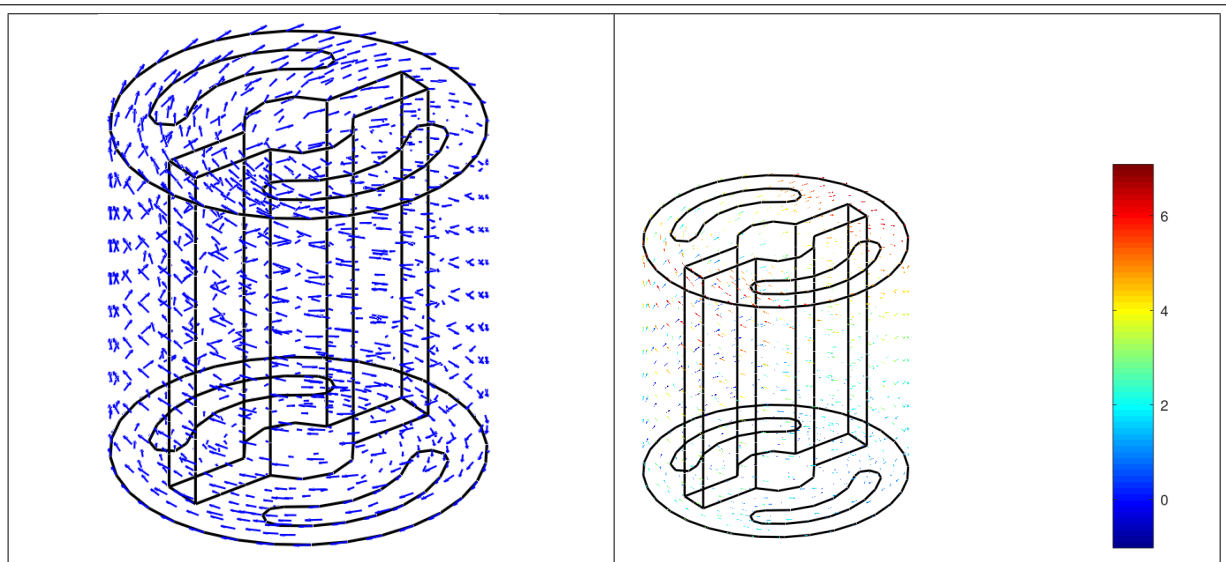
figure(4)
fc_siplt.plotmesh(Th,'d',1,'color','k','LineWidth',1.5)
hold on;axis off;axis image
fc_siplt.plotquiver(Th,W,'colordata',U,'scale',0.2)

```

Listing 14: Using `fc_siplt.plotquiver` function with a 2D mesh

9.2 3D example

the following code is part of the `fc_siplt.demos.plotquiver3D` function.



```

geofile=fc_oogmsh.get_geo(3,3,'cylinderkey');
meshfile=fc_oogmsh.buildmesh3d(geofile,5,varargin{:});
Th=fc_simesh.siMesh(meshfile);

w=@(x,y,z) y.*cos(-(x.^2+y.^2)/10),@(x,y,z) -x.*cos(-(x.^2+y.^2)/10),@(x,y,z) z/5 };
W=Th.eval(w);
u=Th.eval(@(x,y,z) 3*x.^2-y.^3+z.^2+x.*y);
fc_tools.graphics.monitors.onGrid(1,2);
figure(1)
fc_siplt.plotmesh(Th,'d',1,'color','k','LineWidth',1.5)
hold on
fc_siplt.plotquiver(Th,W,'LineWidth',1)
axis off;axis image

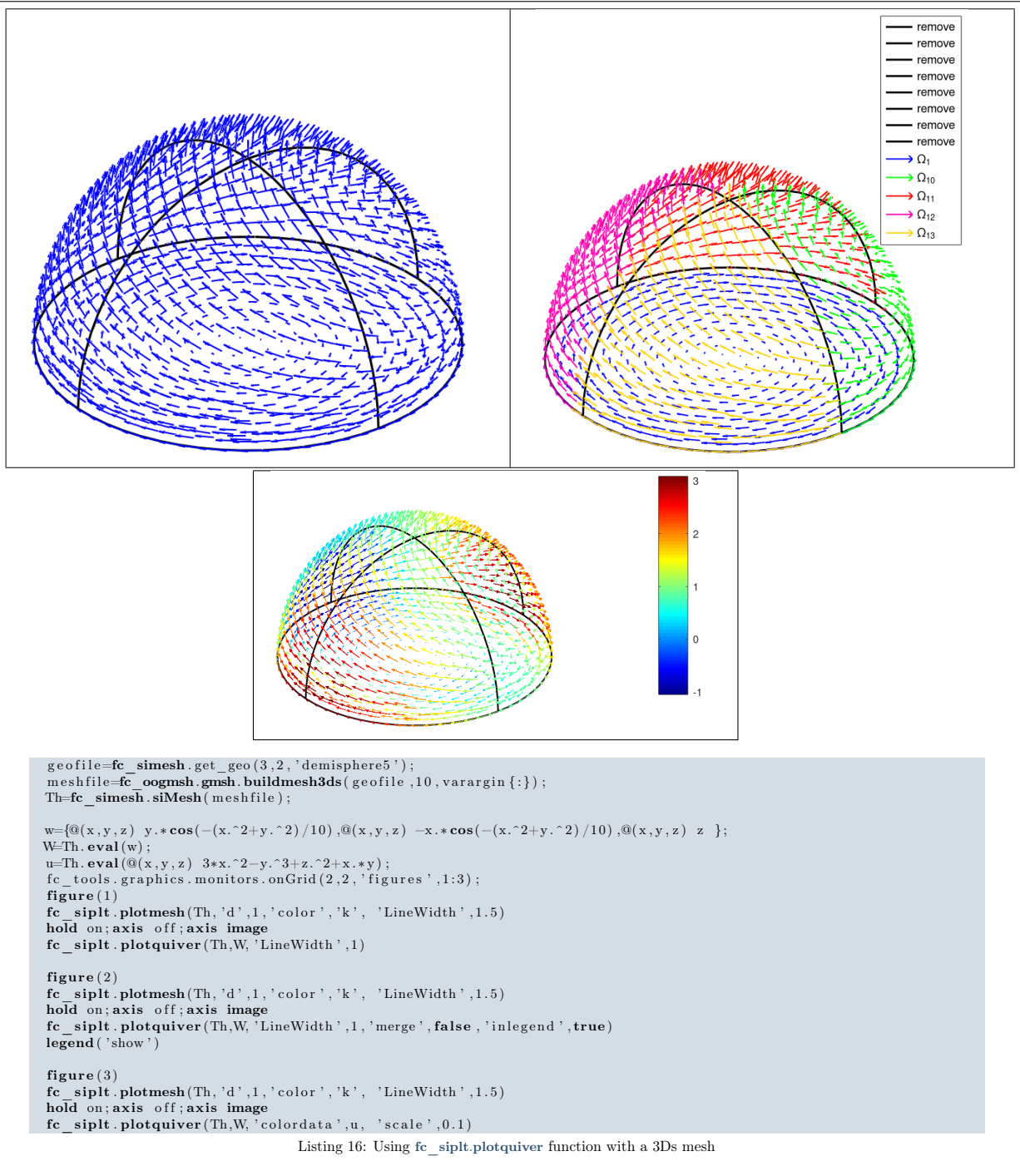
figure(2)
fc_siplt.plotmesh(Th,'d',1,'color','k','LineWidth',1.5)
hold on
fc_siplt.plotquiver(Th,W,'colordata',u,'scale',0.05)
axis off;axis image

```

Listing 15: Using `fc_siplt.plotquiver` function with a 3D mesh

9.3 3Ds example

the following code is part of the `fc_siplt.demos.plotquiver3Ds` function.



10 `fc_siplt.scatter` function

The `fc_siplt.scatter` function displays scalar datas as colored points on the mesh or parts of the mesh defined by an `siMesh` object.

Syntaxe

```

fc_siplt.scatter(Th,u)
fc_siplt.scatter(Th,u,Name,Value,...)

```

Description

`fc_siplt.scatter(Th,u)` displays data `u` on all the $(Th.d)$ -dimensional simplices elements of `Th`, a `siMesh` object. The data `u` is an 1D-array of size `Th.nq` or `Th.nqGlobal` or `Th.nqParent`.

`fc_siplt.scatter(Th,u,Name,Value, ...)` specifies function options using one or more `Name,Value` pair arguments.

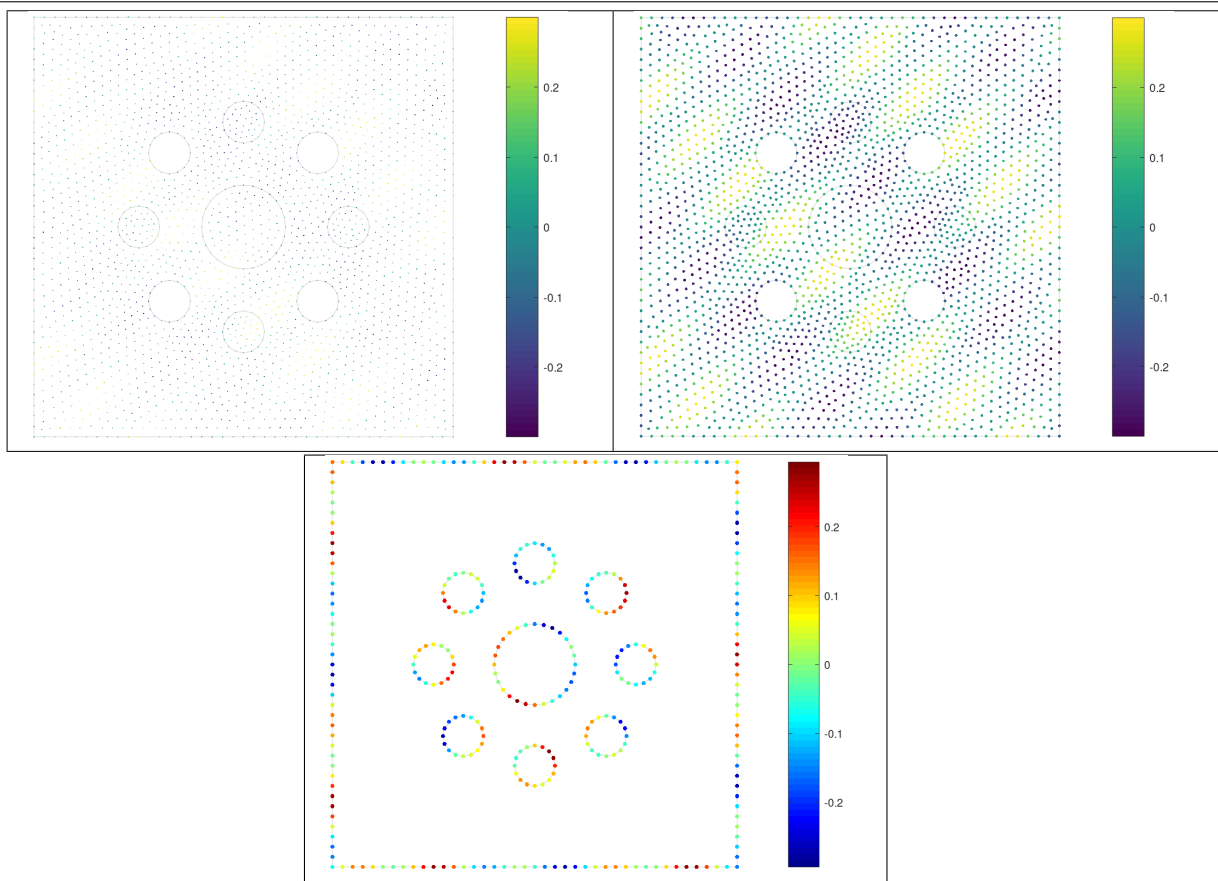
Options of first level are

- `'d'` : to specify the dimension of the simplices elements (default : `Th.d`)
- `'labels'` : to select the labels of the elements to display data,
- `'MarkerSize'` : size of the marker. Default is 1.
- `'ForcePatch'` : if `true`, uses `patch` function, otherwise uses `scatter` function in dimension 2 or `scatter3` function in dimension 3. Default is `true`.

The options of second level are those of the function used (see `'ForcePatch'` option).

10.1 2D example

the following code is part of the `fc_siplt.demos.scatter2D` function.



```

meshfile=fc_oogmsh.gmsh.buildmesh2d('condenser11',20,varargin{:});
Th=fc_simesh.siMesh(meshfile);
u=@(x,y) 0.3*cos(5*(2*x-y)).*sin(5*(x-2*y));
U=Th.eval(u);

fc_tools.graphics.monitors.onGrid(2,2,'figures',1:3);

figure(1)
fc_siplt.plotmesh(Th,'d',1,'color','LightGray')
hold on
fc_siplt.scatter(Th,U);
axis image;axis off;
colorbar

figure(2)
fc_siplt.plotmesh(Th,'d',1,'color','LightGray')
hold on
fc_siplt.scatter(Th,U,'MarkerSize',8);
axis off;axis image
colorbar

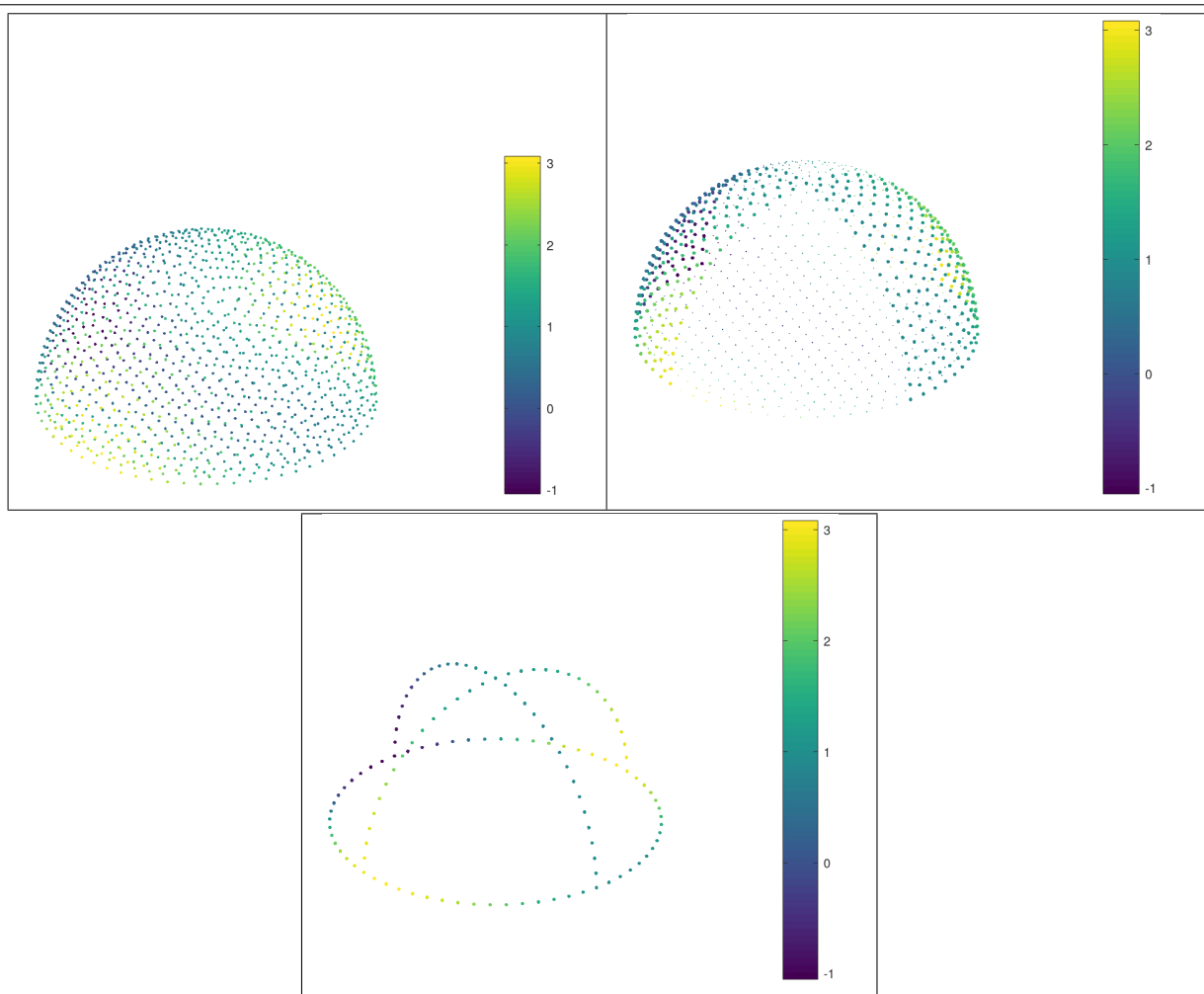
figure(3)
fc_siplt.plotmesh(Th,'d',1,'color','LightGray')
hold on % '*Alpha' options not yet implemented under Octave
fc_siplt.scatter(Th,U,'d',1,'MarkerSize',12,'MarkerEdgeAlpha',0,'MarkerFaceColor','flat','MarkerFaceAlpha',0.8)
axis off;axis image
colormap('jet')
colorbar

```

Listing 17: Using `fc_siplt.scatter` function with a 2D mesh

10.2 3Ds example

the following code is part of the `fc_siplt.demos.scatter3Ds` function.



```

geofile=fc_simesh.get_geo(3,2,'demisphere5');
meshfile=fc_oogmsh.buildmesh3ds(geofile,10,varargin{:});
Th=fc_simesh.siMesh(meshfile);
u=Th.eval(@(x,y,z) 3*x.^2-y.^3+z.^2+x.*y);

fc_tools.graphics.monitors.onGrid(2,2,'figures',1:3);

figure(1)
fc_siplt.scatter(Th,u,'MarkerSize',7)
view(3);axis off;axis image;colorbar;

figure(2)
fc_siplt.scatter(Th,u,'labels',[1,11])
hold on;axis off;axis image;colorbar;
fc_siplt.scatter(Th,u,'labels',[10,12], 'MarkerSize',9)
view(3)

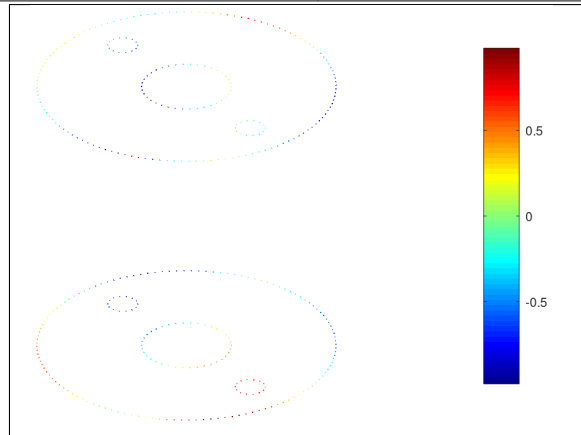
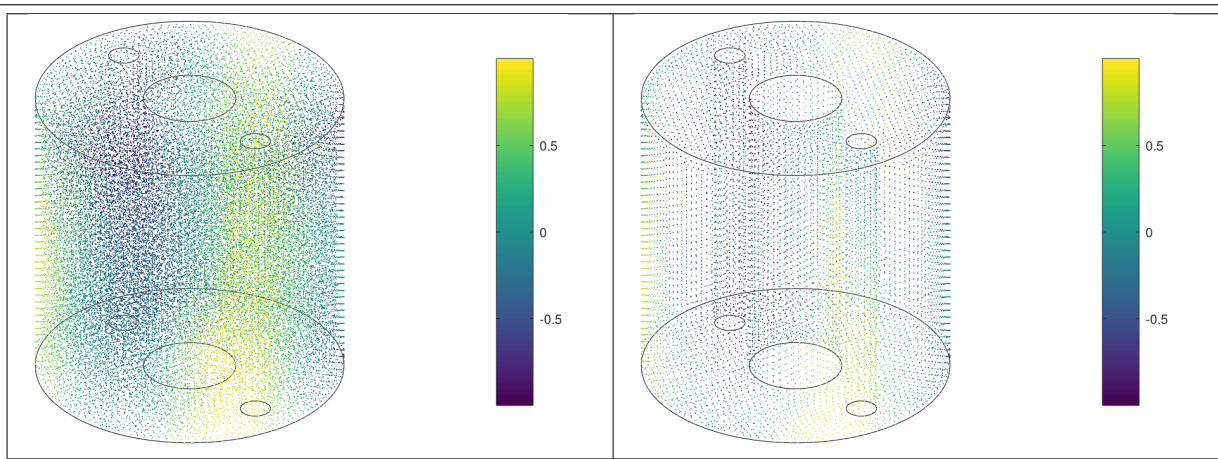
figure(3)
fc_siplt.scatter(Th,u,'d',1,'MarkerSize',9)
axis off;axis image;colorbar
view(3)

```

Listing 18: Using `fc_siplt.scatter` function with a 3Ds mesh

10.3 3D example

the following code is part of the `fc_siplt.demos.scatter3D` function.



```
meshfile=fc_oogmsh.gmsh.buildmesh3d('cylinder3holes',20,varargin{:});
Th=fc_simesh.siMesh(meshfile);
u=@(x,y,z) cos(2*x-y-z).*sin(x-2*y+z);
U=Th.eval(u);
```

```
fc_tools.graphics.monitors.onGrid(2,2,'figures',1:3);
```

```
figure(1)
fc_siplt.plotmesh(Th,'d',1,'color','k')
hold on
fc_siplt.scatter(Th,U,'MarkerEdgeAlpha',0.1);
axis image;axis off;view(3)
colorbar
```

```
figure(2)
fc_siplt.plotmesh(Th,'d',1,'color','k')
hold on
fc_siplt.scatter(Th,U,'d',2);
axis image;axis off;view(3)
colorbar
```

```
figure(3)
fc_siplt.scatter(Th,U,'d',1);
axis image;axis off;view(3)
colormap('jet')
colorbar
```

Listing 19: Using `fc_siplt.scatter` function with a 3D mesh

Appendices

A Listings

1	<code>fc_siplt.demos.sample2D01</code> script with figure 1 (top left), figure 2 (top right), figure 3 (bottom left) and figure 4 (bottom right).	3
2	Using <code>fc_siplt.plotmesh</code> function with a 2D mesh	6
3	Using <code>fc_siplt.plotmesh</code> function with a 3D mesh	7
4	Using <code>fc_siplt.plotmesh</code> function with a 3Ds mesh	8
5	Using <code>fc_siplt.plot</code> function with a 2D mesh	10
6	Using <code>fc_siplt.plot</code> function with a 3D mesh	11
7	Using <code>fc_siplt.plot</code> function with a 3Ds mesh	12
8	Using <code>fc_siplt.plotiso</code> function with a 2D mesh	14
9	Using <code>fc_siplt.plotiso</code> function with a 3D mesh	15
10	Using <code>fc_siplt.plotiso</code> function with a 3Ds mesh	15
11	Using <code>fc_siplt.slicemesh</code> function with a 3D mesh	16
12	Using <code>fc_siplt.slice</code> function with a 3D mesh	17
13	Using <code>fc_siplt.sliceiso</code> function with a 3D mesh	19
14	Using <code>fc_siplt.plotquiver</code> function with a 2D mesh	21
15	Using <code>fc_siplt.plotquiver</code> function with a 3D mesh	22
16	Using <code>fc_siplt.plotquiver</code> function with a 3Ds mesh	23
17	Using <code>fc_siplt.scatter</code> function with a 2D mesh	25
18	Using <code>fc_siplt.scatter</code> function with a 3Ds mesh	26
19	Using <code>fc_siplt.scatter</code> function with a 3D mesh	27

B References

- [1] F. Cuvelier. `fc_graphics4mesh`: an Octave package for displaying simplices meshes or datas on simplices meshes. <http://www.math.univ-paris13.fr/~cuvelier/software/>, 2017. User’s Guide.
- [2] F. Cuvelier. `fc_simesh`: an object-oriented Octave package for using simplices meshes generated from gmsh (in dimension 2 or 3) or an hypercube triangulation (in any dimension). <http://www.math.univ-paris13.fr/~cuvelier/software/>, 2017. User’s Guide.

Informations for git maintainers of the Octave package

git informations on the packages used to build this manual

```
-----
name : fc-siplt
tag : 0.2.7
commit : 2eb4295cf81bc69179ac0a5790fe1d724ea3f7bb
date : 2025-01-30
time : 11-30-29
status : 0
-----
name : fc-tools
tag : 0.0.36
commit : 00c110c58dff7e001ec8130802d1725abf991f33
date : 2025-01-26
time : 05-09-25
status : 0
-----
name : fc-bench
tag : 0.1.4
commit : b00bc133994a648d8909c4952f01659f8dbb5a8f
date : 2025-01-26
time : 05-13-31
status : 0
-----
name : fc-amat
tag : 0.1.4
commit : c8ac405135a3606b8c2809d9d2dd71ee0989b5b8
date : 2025-01-26
time : 05-19-40
status : 0
-----
name : fc-hypermesh
tag : 1.0.5
commit : 2c8ceb7f897520cecc80ea9b69f31071c29ce610
date : 2025-01-28
time : 09-53-41
status : 0
-----
name : fc-meshtools
tag : 0.1.5
commit : b7ff9340a05d6fb443c84cc27ca6b13339c2fd81
date : 2025-01-26
time : 07-33-01
status : 0
-----
name : fc-graphics4mesh
tag : 0.1.7
commit : 795429326c1c20a60c186f27df2fef24680a431
date : 2025-01-28
time : 08-12-35
status : 0
-----
name : fc-oogmsh
tag : 0.3.1
commit : 0b324a10b6d93dae7cb37fc9cf1d811584f1e18a
date : 2025-01-29
time : 08-35-39
status : 0
-----
name : fc-simesh
tag : 0.4.7
commit : f19b2f5678dc253ecb7ea822f1f9be9a094a497d
date : 2025-01-30
time : 11-29-09
status : 0
-----
[fc-tools] waiting 2(s) to finish saving figures
```

git informations on the L^AT_EX package used to build this manual

```
-----
name : fctools
tag :
commit : 03d38737a795cdf4e1a8754470e963cdf83316
date : 2025-01-24
time : 09:58:52
status : 1
-----
[fc-tools] waiting 2(s) to finish saving figures
```

Using the remote configuration repository:

```
url      ssh://lagagit/MCS/Cuvelier/Matlab/fc-config
commit   27c3da029668900f91e2968ecc7fc7e9e5287a32
```